ibaPDA-Request-S7

Applications of the Request data interface for SIMATIC S7-300/S7-400

Manual part 2

Issue 1.0

Measurement Systems for Industry and Energy

www.iba-ag.com

Manufacturer

iba AG

Gebhardtstrasse 10-20

90762 Fuerth

Germany

Contacts

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

Version	Date	Revision	Author	Version SW
1.0	08-2026	First edition; combined licenses (ibaPDA-Request-S7-DP/PN/ibaNet-E and ibaPDA-Request-S7-UDP)	mm	8.14.0

Contents

1 About this documentation

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

The product *ibaPDA-Request-S7*is an extension of *ibaPDA* for the direct access to S7 symbols and S7 operands when recording data from SIMATIC S7-CPUs. This documentation describes only the extensions and deviations. You can find further information on all other functions and operating functions in the documentation of the devices, interfaces and *ibaPDA*.

Other documentation

This documentation provides supplementary information to the *ibaPDA* documentation.

1.1 Target group and previous knowledge

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

This documentation in particular addresses persons who are concerned with the configuration, test, commissioning or maintenance of Programmable Logic Controllers of the supported products. For the handling *ibaPDA-Request-S7* 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

1.2 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>
Filenames, paths	`Filename`, `Path` Example: `Test.docx`

1.3 Used symbols

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

Danger!

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

Observe the specified measures.

Warning!

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

Observe the specified measures.

Caution!

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

Observe the specified measures.

Note

A note specifies special requirements or actions to be observed.

Tip

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

Other documentation

Reference to additional documentation or further reading.

1.4 Documentation structure

The documentation for the *ibaPDA-Request-S7* interface (PDF edition) is divided into two separate parts. Each part has its own chapter and page numbering beginning at 1 and is independently updated.

Part	Title	Content
Part 1	Applications of the Request data interface for SIMATIC S7 TIA Portal/ S7-1500	Features, settings, and Request blocks for the Request interface in conjunction with SIMATIC TIA Portal and S7-1500 S7 controller
Part 2	Applications of the Request data interface for SIMATIC S7-300/S7-400	Features, settings, and Request blocks for the Request interface in conjunction with SIMATIC STEP 7 and S7-300 and S7-400 S7 controllers

2 Request-S7 variants

This chapter describes the Request-S7 variants for applications with S7-300/400 controllers. For applications with S7-1500 controllers, refer to the manual Part 1.

S7 family	Firmware	Addressing type	iba block family	iba data paths
S7-300	Unlimited	Operands	ibaREQ	DP, PN, UDP
S7-400	Unlimited	Operands	ibaREQ	DP, PN, UDP

2.1 Request-S7 via UDP

The Request-S7 variant via UDP is described below.

2.1.1 General information

You can configure Request S7 for UDP in the following system configurations.

SIMATIC S7-CPU	SIMATIC STEP 7 V5.x (SIMATIC Manager)	SIMATIC STEP 7 V1x Professional (TIA Portal)
S7-300 integrated PN interface and CP343-1	X	X
S7-400 integrated PN interface and CP443-1	X	X

The prerequisite for use of Request S7 using ibaNet-E is a license for *ibaPDA-Interface-S7-TCP/UDP* (item no. 31.001040).

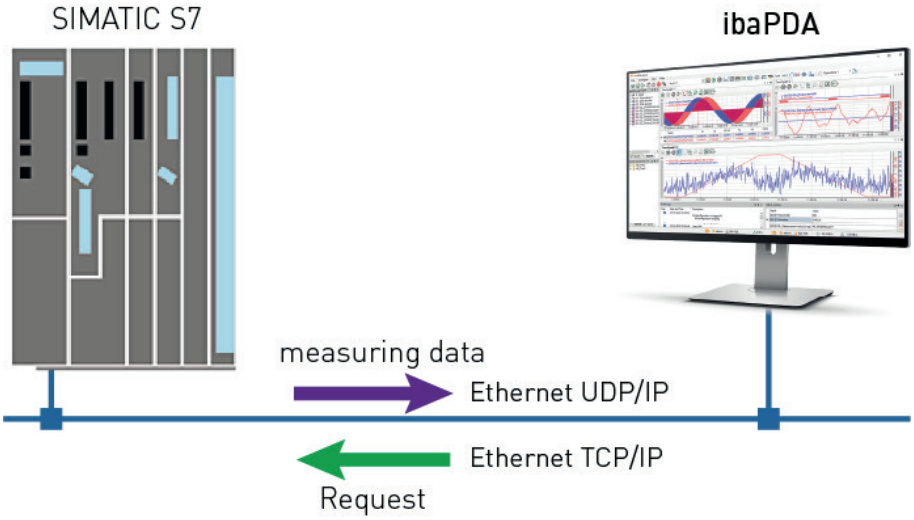
With the UDP Request, the measurement values (Request Handshake) are requested via a separate UDP connection.

Depending on the hardware and software, different access points can be selected for the Request:

- TCP/IP: The connection to the SIMATIC S7 is established using an integrated PN interface on the S7 CPU or corresponding CP modules in the PLC and the standard network interface on the computer. No additional Siemens software is required for the connection.
- PC/CP: This is the name of the various SIMATIC specific access points. In contrast to the TCP/IP connection, the common feature of all connection types in the PC/CP group is that the SIMATIC communication software has to be installed on the computer with the corresponding functions enabled.
  - MPI, PROFIBUS: The connection to the SIMATIC S7 is established using the MPI or PROFIBUS interface on the computer; e.g. using the CP5611 PCI card or the MPI adapter for USB or serial PC interfaces.
  - TCP/IP, ISO: Here, either the standard network interface on the computer or a suitable interface card is used for the connection to the S7.

A transfer rate of max. 1446 Byte is possible for each configured module. The number of available modules depends on the number of licenses.

The following figure shows the system topology for an S7 Request using UDP.



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

Other documentation

You can find detailed information about *ibaPDA-Interface-S7-TCP/UDP* in the interface manual.

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

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

- Configuration Software (STEP 7 V5):
  Integration of the Request blocks in the S7 program
- Connection configuration:
  If a communication processor CP x43-1 is being used, a programmed connection has to be configured in NetPro. This is not necessary when using a PN interface integrated on the CPU.

2.1.2.1 Configuration in STEP 7

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

Copy the required blocks from the iba S7 library to the blocks folder of your STEP 7 project, see ↗ *iba S7 library, page 128*.

Note

The request blocks do not support multi-instance calls.

Note

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

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

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

For further information on customizing, see ↗ *Adaptation to the renumbered system functions, page 137*.

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

The following blocks are required:

- ibaREQ_M (FB140), see ↗ *ibaREQ_M (FB140), page 87*
- ibaREQ_UDPact (FB145), see ↗ *ibaREQ_UDPact (FB145), page 93*
- ibaREQ_UDPint (FB146), see ↗ *ibaREQ_UDPint (FB146), page 95*
- ibaREQ_DB (DB15)
- ibaUDT_UDPact (UDT145)

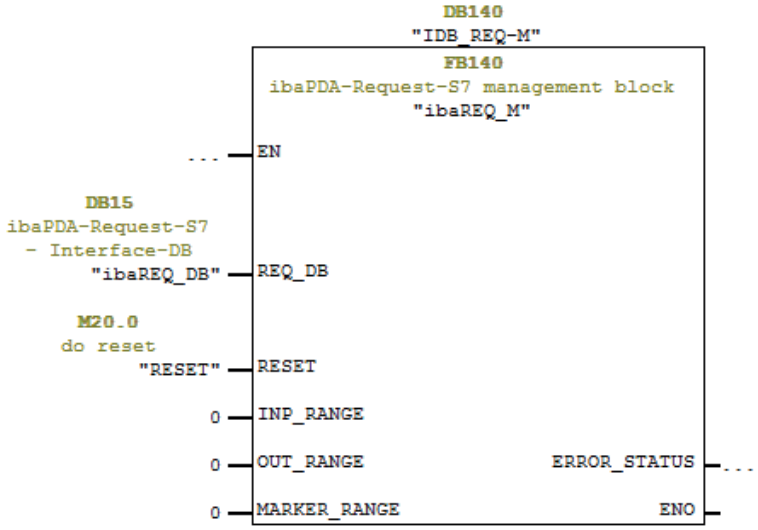
Note

Only use Request blocks from the latest iba S7 library!

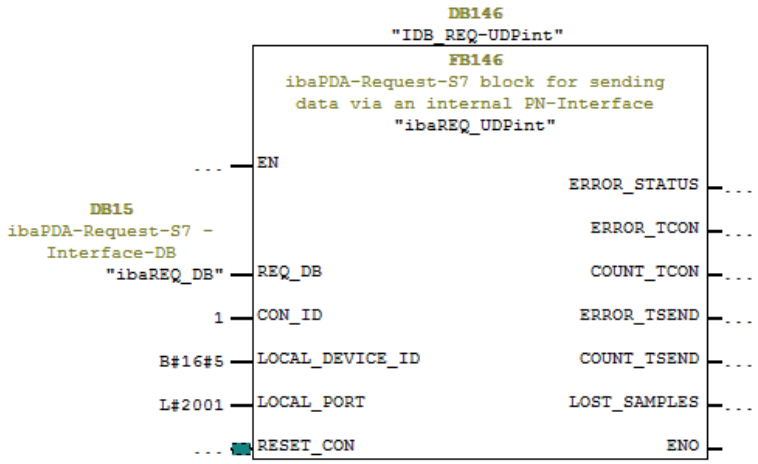
Request blocks in application examples can be outdated and, thus, cause errors.

For each Request module

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



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



For each additional Request module

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

Final

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

2.1.2.1.2 CPU S7-300 with CP343-1

The following function blocks are required:

- ibaREQ_M (FB140), see ↗ *ibaREQ_M (FB140), page 87*
- ibaREQ_UDPact (FB145), see ↗ *ibaREQ_UDPact (FB145), page 93*
- ibaREQ_UDPext3 (FB147), see ↗ *ibaREQ_UDPext3 (FB147), page 100*
- ibaREQ_DB
- ibaUDT_UDPact (UDT145)

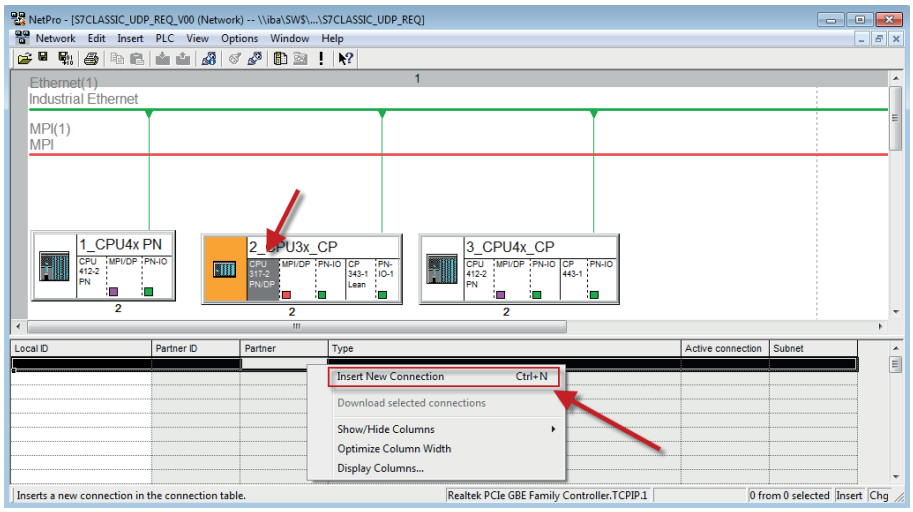
Note

Only use Request blocks from the latest iba S7 library!

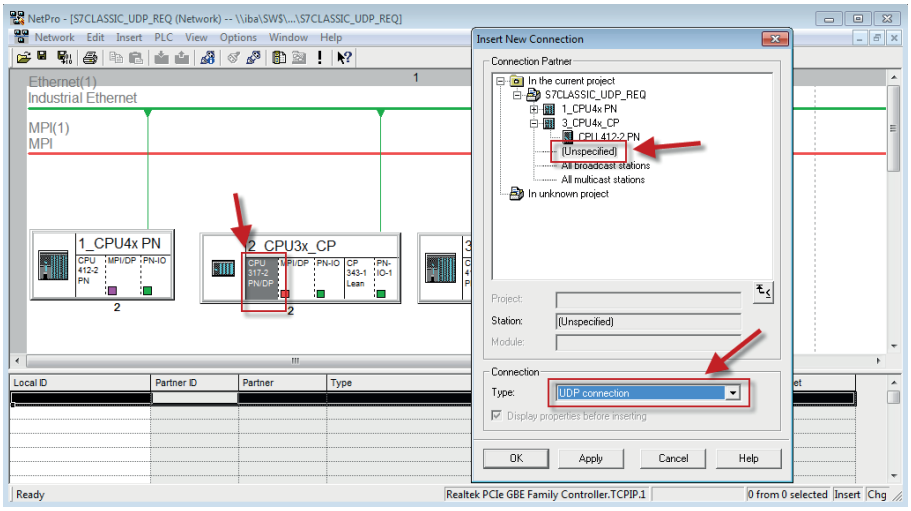
Request blocks in application examples can be outdated and, thus, cause errors.

For each Request module

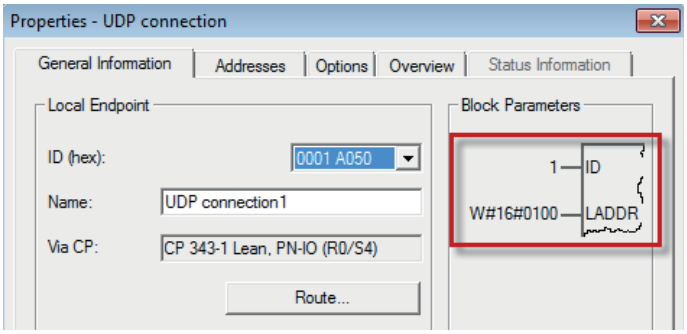
1. Configure a new connection in NetPro.



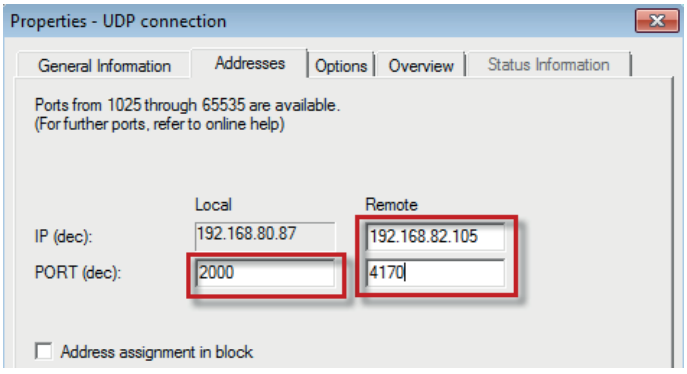
2. Select the connection partner *unspecified* and connection type *UDP connection*.



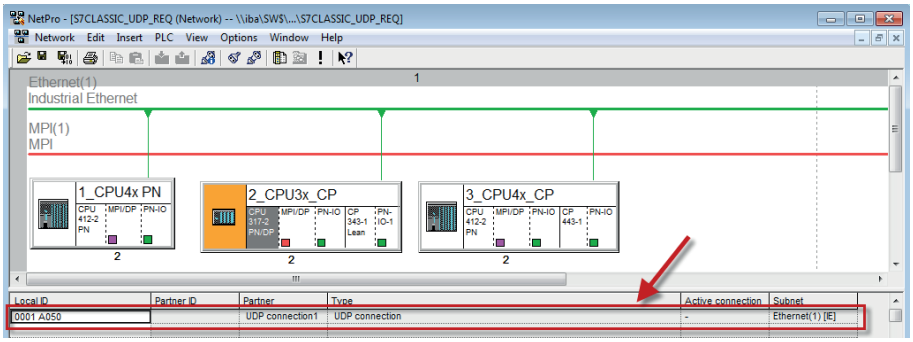
3. The automatically assigned function block parameter for the connection ID (ID) and the hardware starting address (LADDR) are needed later in step 6.



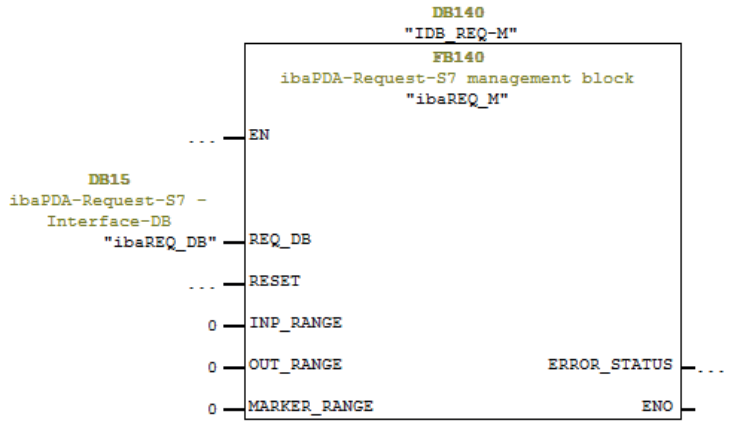
4. Enter the IP address of the *ibaPDA* computer as partner IP address and the configured port number (standard: 4170) and select a unique local port number.



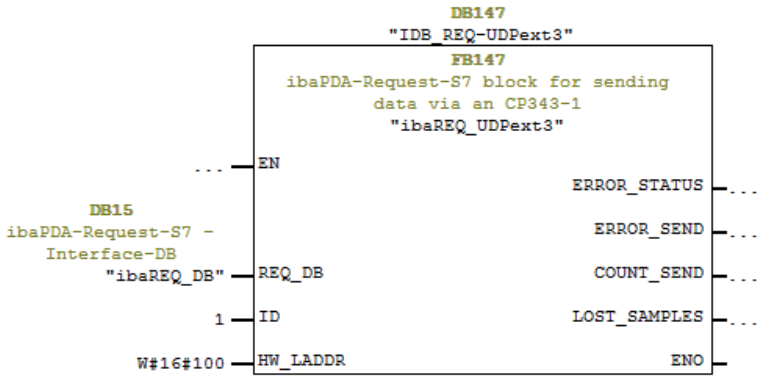
→ The connection table of the CPU now displays the newly configured connection.



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



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



For each additional Request module

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

Final

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

2.1.2.1.3 CPU S7-400 with CP443-1

The following blocks are required:

- ibaREQ_M (FB140), see ↗ *ibaREQ_M (FB140), page 87*
- ibaREQ_UDPact (FB145), see ↗ *ibaREQ_UDPact (FB145), page 93*
- ibaREQ_UDPext4 (FB148), see ↗ *ibaREQ_UDPext4 (FB148), page 101*
- ibaREQ_DB
- ibaUDT_UDPact (UDT145)

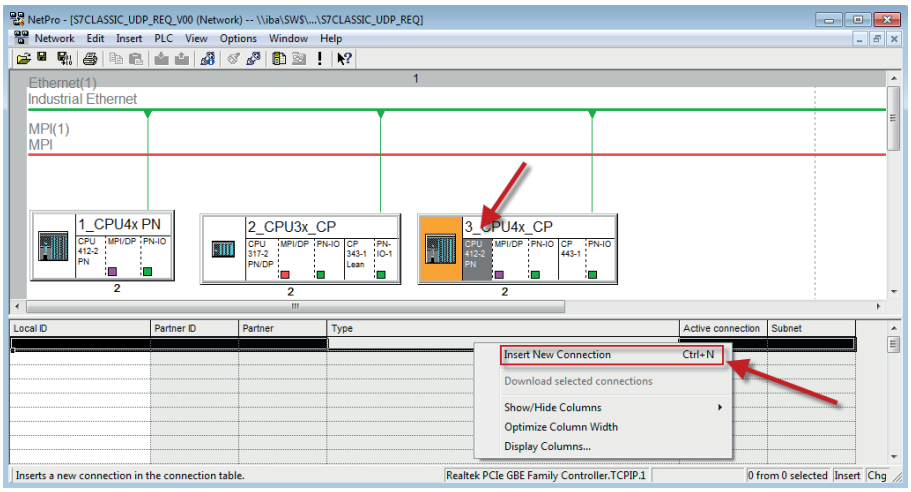
Note

Only use Request blocks from the latest iba S7 library!

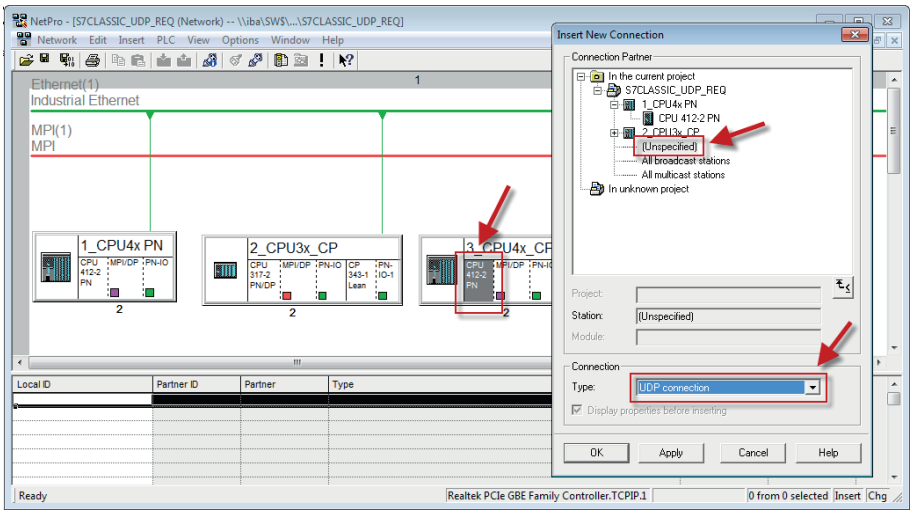
Request blocks in application examples can be outdated and, thus, cause errors.

For each Request module

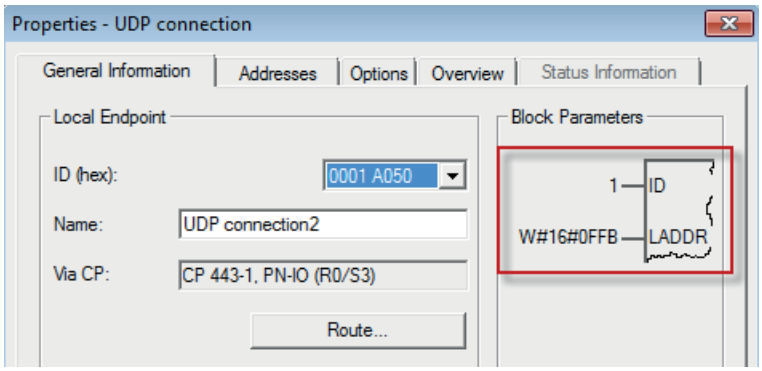
1. Configure a new connection in NetPro.



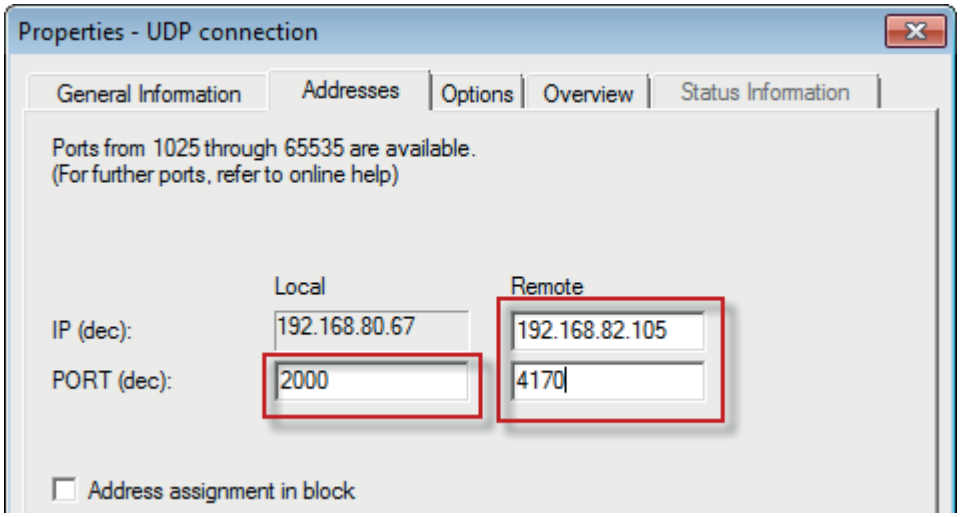
2. Select the connection partner *unspecified* and connection type *UDP connection*.



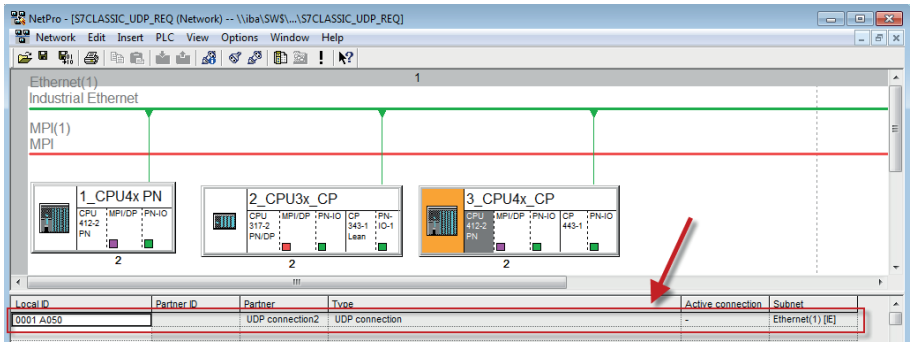
3. The automatically assigned function block parameters are needed later in step 6.



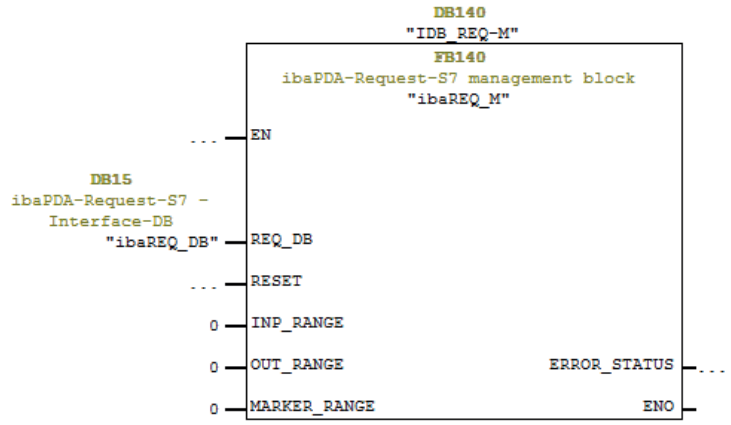
4. Enter the IP address of the *ibaPDA* computer as partner IP address and the configured port number (standard: 4170) and select a unique local port number.



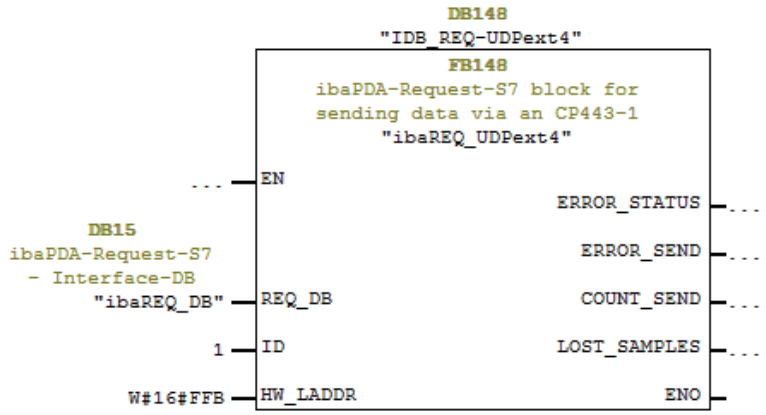
→ The connection table of the CPU now displays the newly configured connection.



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



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



For each additional Request module

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

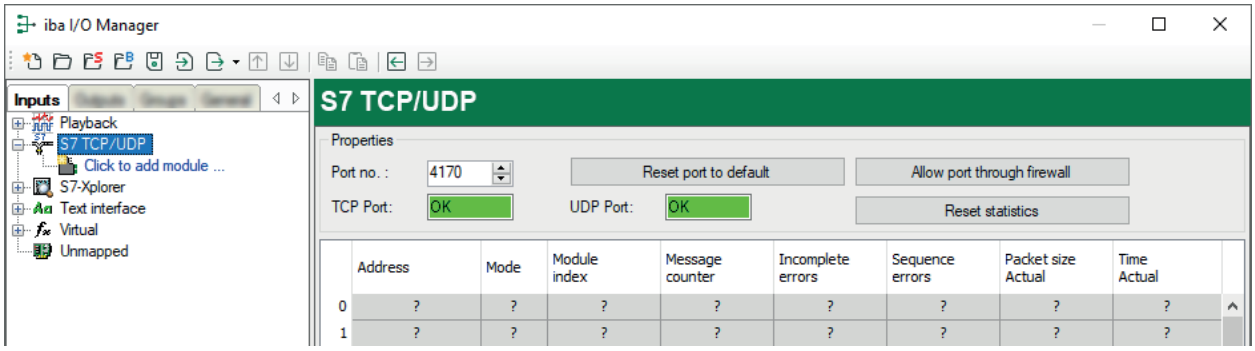
Final

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

2.1.3 Configuration and engineering ibaPDA

2.1.3.1 General interface settings

If all system requirements are fulfilled, *ibaPDA* displays the *S7 TCP/UDP* interface in the interface tree of the I/O Manager.



Other documentation

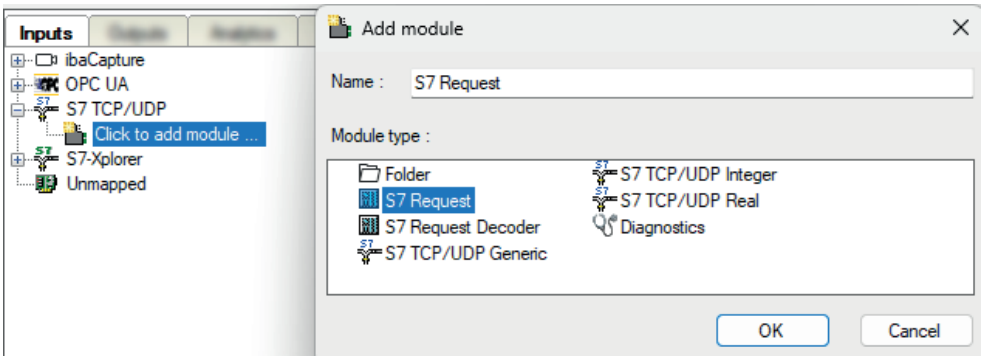
For more information about the interface *ibaPDA-Interface-S7-TCP/UDP*, see the corresponding manual.

Configuring S7 TCP/UDP for Request

1. Start the *ibaPDA* client and open the I/O Manager.
2. Click on the blue command *Click to add module* under the interface.

   Select a *Request S7* module in the dialog box and assign a name via the input field if required. Confirm the selection with <OK>.

   - S7 Request (for acquiring analog signals and digital signals)
   - S7 Request Decoder (for acquiring up to 11728 digital signals)



3. Configure the required module settings and signals as described in the following chapters.

   The *General* and *Connection* tabs are identical for all Request modules. The Request modules only differ in the *Analog* and *Digital* tabs.

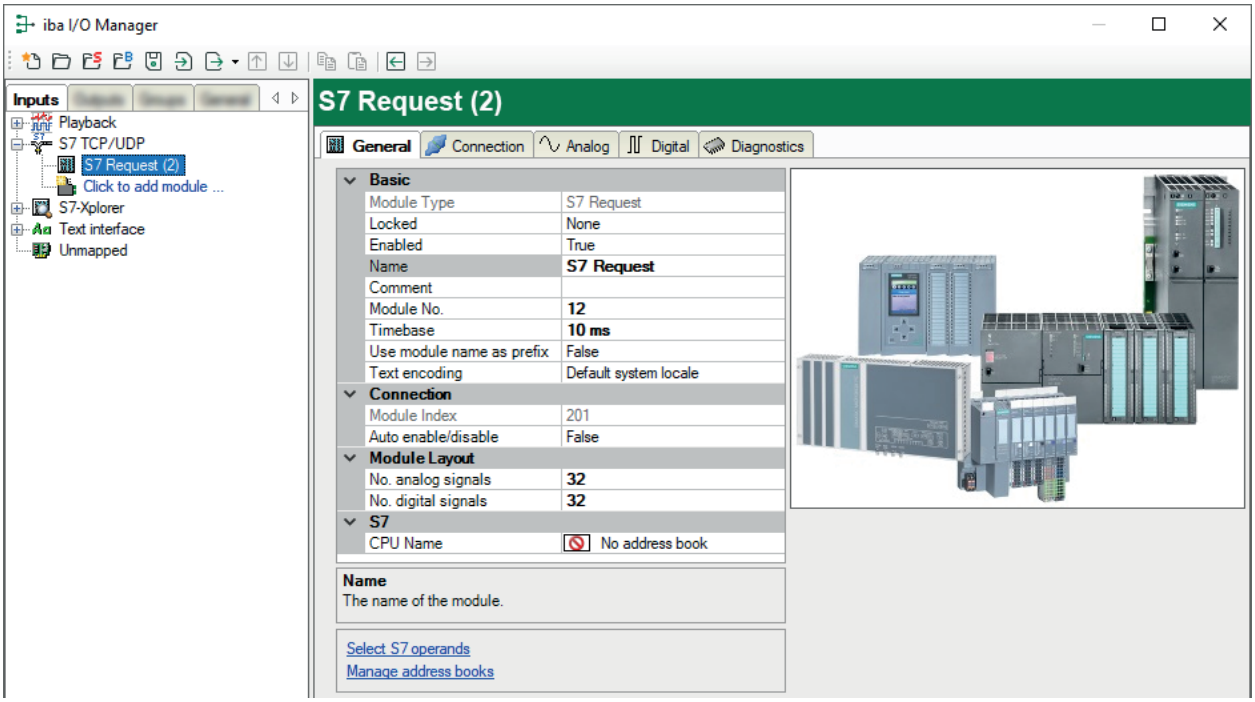
4. After you have finished the configuration, click <Apply> or <OK> to transfer the new configuration to the device and start data acquisition with *ibaPDA*.

2.1.3.2 General module settings

You can find the description of all settings that are identical for all Request-S7 modules in the general chapter *General module settings* in part 1 of the manual.

You can find information on connection settings from the *Connection* tab in the general chapter *General module settings* in part 1 of the manual.

The *S7 Request* module for *S7 TCP/UDP* has the following specific setting options in the *General* tab:



Connection

Module index (information only)
Internal reference number of the module.

Auto enable/disable
If TRUE, the acquisition is started, even if no connection can be established to the S7-CPU. The module is deactivated. During the acquisition, *ibaPDA* tries to reconnect to the S7-CPU. When it succeeds, the acquisition is restarted with this module enabled.
If FALSE, the acquisition is not started, if a connection to the configured S7-CPU cannot be established.

Module Layout

No of analog signals/digital signals
Define the number of configurable analog and digital signals in the signal tables. The default value is 32 for each. The maximum value is 1024. The signal tables are adjusted accordingly.

2.1.3.3 Module S7 Request

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

Configure a separate Request block call for each module.

For more information about the module settings, see ↗ *General module settings, page 22*.

The request modules of *S7 TCP/UDP* have the following specific option in the *Connection* tab.

Connection configuration

ibaPDA IP address
If multiple network cards are installed in the computer, specifiy here the *ibaPDA* IP address to which the controller should send data.

When using the *Auto* option, the system automatically uses the card that can reach the controller based on its IP address.

2.1.3.4 Module S7 Request Decoder

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

General tab

For more information on the module settings, see ↗ *General module settings, page 34*, and the general chapter *General module settings* in part 1 of the manual.

Module specific settings

Module layout – No. of decoders
Define the number of configurable decoders in the table of digital signals. The default value is 32. The maximum value is 733. The signal tables are adjusted accordingly.

Connection configuration

Configure the connection of the *S7 Request Decoder* module in the same way as the connection for an S7 Request module, see the general chapter *Connection settings* in part 1 of the manual.

The request modules of *S7 TCP/UDP* have the following specific option in the *Connection* tab.

ibaPDA IP address
If multiple network cards are installed in the computer, specifiy here the *ibaPDA* IP address to which the controller should send data.

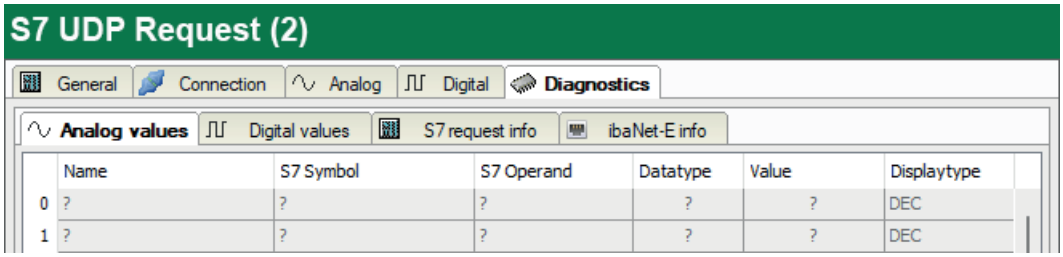
When using the *Auto* option, the system automatically uses the card that can reach the controller based on its IP address.

Digital tab

The signals are configured in the same way as for the *ibaBM-DP* device, see ↗ *Digital tab, page 51*.

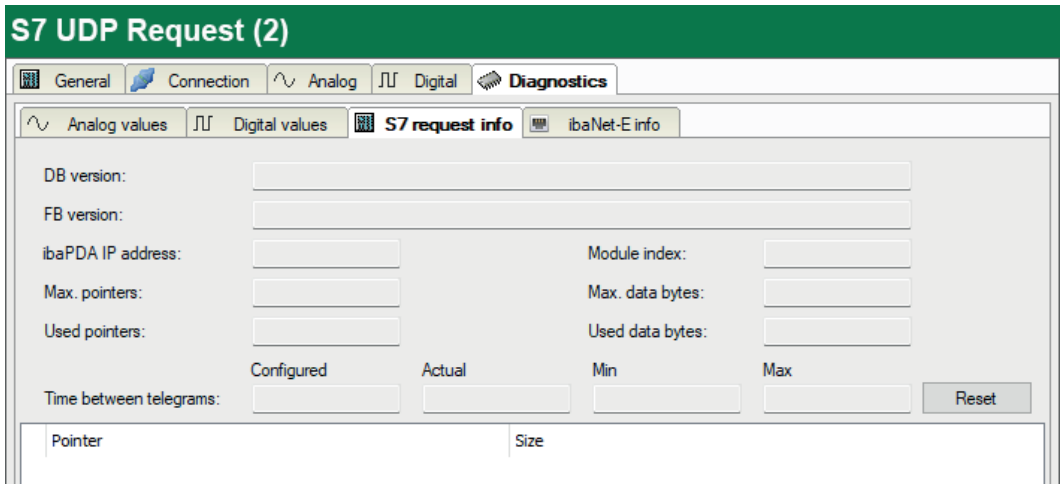
2.1.4 Module diagnostics

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



Subtab S7 request info

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



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 22*)

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

Used pointers
Currently used number of pointers.

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

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

Time between telegrams

- *Configured*: Corresponds to the setting *Timebase* on the *General* tab
- *Actual*: Time between the two last received telegrams
- *Min*: shortest time
- *Max*: longest time

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

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

<Reset>
Resetting the minimum and maximum values

Pointer table
Currently required data pointer with address and length

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

2.2 Request-S7 for ibaBM-PN

This section describes the Request-S7 variant for the PROFINET bus module *ibaBM-PN*.

2.2.1 General information

With the PN Request, the measurement values (Request Handshake) are requested via a separate TCP/IP connection.

SIMATIC S7-CPU	SIMATIC STEP 7 V5.x (SIMATIC Manager)	SIMATIC STEP 7 V1x Professional (TIA Portal)
S7-300 integrated PN interface	X	X
S7-400 integrated PN interface and CP443-1	X	X

Depending on the hardware and software, you can select different access points for the Request:

- TCP/IP: The connection to the SIMATIC S7 is established using an integrated PN interface of the S7 CPU or the respective CP modules in the PLC and the standard network interface of the computer. No additional Siemens software is required for the connection.
- PC/CP: This is the designation for different SIMATIC specific access points. In contrast to the TCP/IP connection, for the PC/CP group the SIMATIC communication software (and the corresponding licenses) must be installed on the computer with the corresponding functions enabled.
  - MPI, PROFIBUS: The connection to the SIMATIC S7 is established using the MPI or PROFIBUS interface of the computer; e.g. with the CP5611 PCI card or the MPI adapter for USB interface or serial PC interface.
  - TCP/IP, ISO: Here, either the standard network interface of the computer or a suitable interface card is used for the connection to the S7.

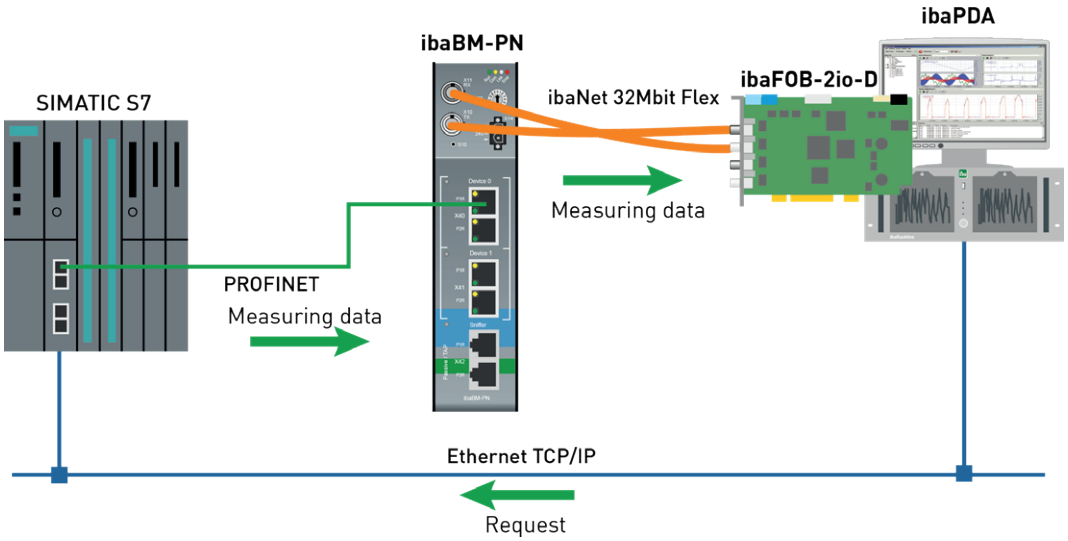
System integration with ibaBM-PN

The measured data are transmitted to the *ibaBM-PN* device over PROFINET.

You need the following connections:

- Online connection between *ibaPDA* and S7-CPU (TCP/IP, MPI or DP)
- Fiber optic connection between *ibaPDA*/*ibaFOB-io-D* and *ibaBM-PN*
- PROFINET connection between *ibaBM-PN* and S7-CPU

The bus monitor offers two separate PROFINET devices. A transfer rate of max. 1440 Byte is possible for each device.



Other documentation

Detailed information about *ibaBM-PN* can be found in the device manual.

For information about application examples, see chapter ↗ *Application examples, page 131*.

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

This section describes the configuration and engineering on the SIMATIC S7 side with SIMATIC Manager (STEP 7 Version ≤ V5) for *ibaBM-PN*.

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

- Hardware configuration:
  Integration of the PROFINET device in the hardware configuration
- Software configuration (STEP 7 V5):
  Integration of the Request blocks in the S7 program

2.2.2.1 Hardware configuration

For *ibaBM-PN*, configure a separate PROFINET device for each PROFINET device in use.

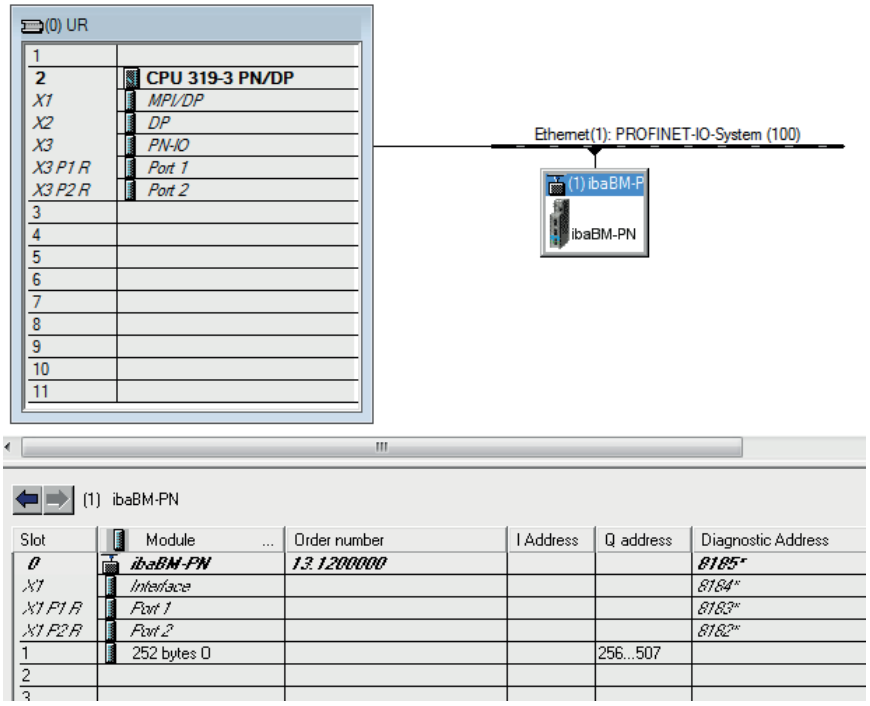
Use the GSDML file `GSDML-Vx.yy-ibaBM-PN-yyyymmdd.xml`.

Note

You find the latest version of the GSDML file on the data storage medium "iba Software & Manuals" in the following directory:

`\02_iba_Hardware\ibaBM-PN\01_GSD_Files\`

S7 CPUs provide consistent slots with a max. of 252 Bytes. You need one slot for each Request block. You can also use slots with a smaller size.



2.2.2.2 Configuration in STEP 7

This section describes how to configure the Request blocks in STEP 7 V5 for *ibaBM-PN*.

Request-S7 is intended for the use with CPUs S7-300/400 with integrated PN interface or CPUs S7-400 with external interface CP 443-1 (PROFINET controller).

Note

The use of the external PN interface CP343-1 of a CPU S7-300 is not supported!

Note

Only use Request blocks from the latest iba S7 library!

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

- Copy the following blocks from the iba S7 library to the blocks folder of your STEP 7 project, see ↗ *iba S7 library, page 128*. One set of Request blocks has to be called for each Request module (connection) in *ibaPDA*.
  - ibaREQ_M (FB140), see ↗ *ibaREQ_M (FB140), page 87*
  - ibaREQ_PN (FB141), see ↗ *ibaREQ_PN (FB141), page 90*
  - ibaREQ_PNdev (FB150), see ↗ *ibaREQ_PNdev (FB150), page 102*
  - ibaREQ_DB (DB15)

Note

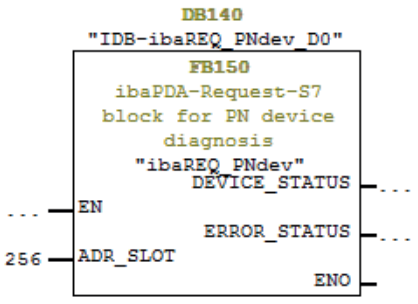
If the block numbers are already used in your project, assign other free numbers to the blocks from the iba S7 library when copying.

Note

The request blocks do not support multi-instance calls.

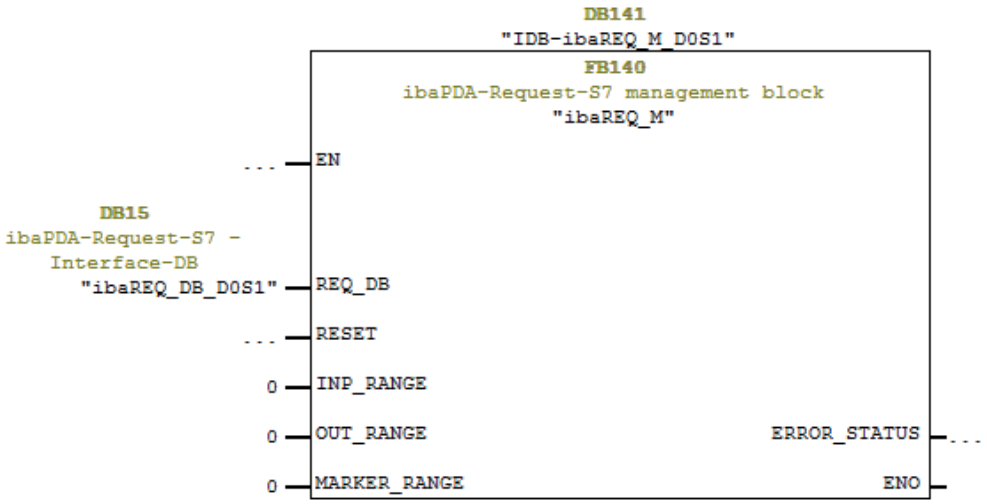
For each used PROFINET device of an ibaBM-PN

- Call the ibaREQ_PNdev (FB150) preferably within the OB1 or a cyclic interrupt (OB3x).

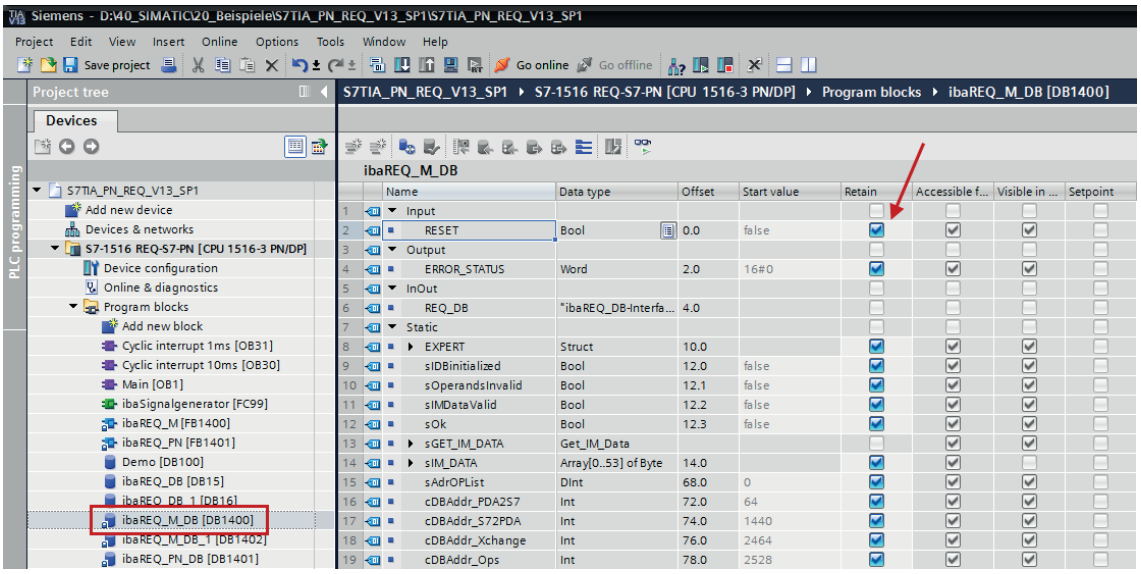


For each Request module

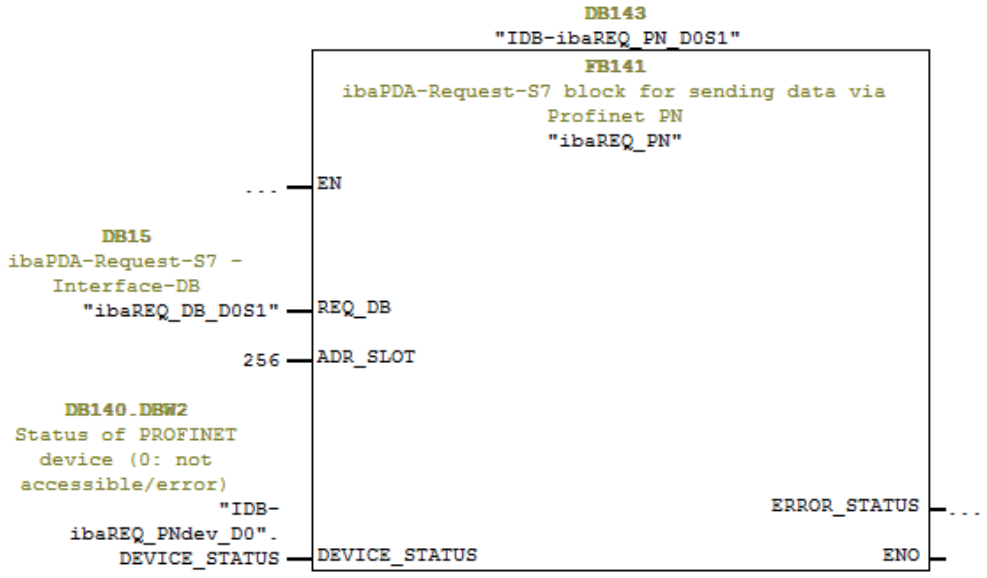
1. Call the ibaREQ_M (FB140) preferably within the OB1.



2. Enable the option *Retain* for the entire instance data block you have just created.



3. Call the ibaREQ_PN (FB141) preferably within the OB1 or a cyclic interrupt OB (OB3x).



For each additional Request module

- In the blocks folder, an ibaREQ_DB (DB15) data block has to be available for each Request module. Copy the data block and assign a new unique DB number.
- Within the OB1, the ibaREQ_M (FB140) has to be called once more with the new DB number (input REQ_DB) for each Request module.
- Within the OB1 or a cyclic interrupt OB (OB3x), the ibaREQ_PN (FB141) has to be called once more with the new DB number (input REQ_DB) for each Request module.
- Make sure that all instance data blocks are unique and that unique values are assigned for the ADR_SLOT.
- It is sufficient to call the ibaREQ_PNdev just once per used PROFINET device and not for each Request module.

Final

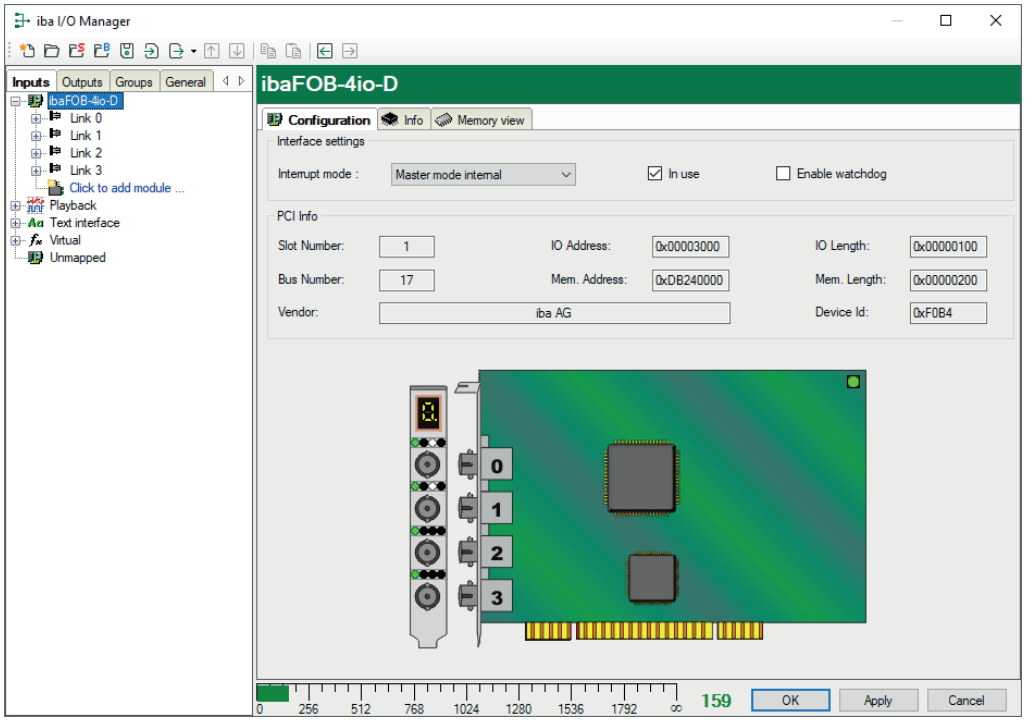
- Load all blocks into the S7 CPU and restart the S7 CPU.

2.2.3 Configuration and engineering ibaPDA

This section describes the configuration of the Request modules in *ibaPDA* for *ibaBM-PN*.

2.2.3.1 General interface settings

If an *ibaFOB-D* card is installed in the *ibaPDA* computer, *ibaPDA* displays the interface of the *ibaFOB-D* card in the interface tree of the I/O Manager.



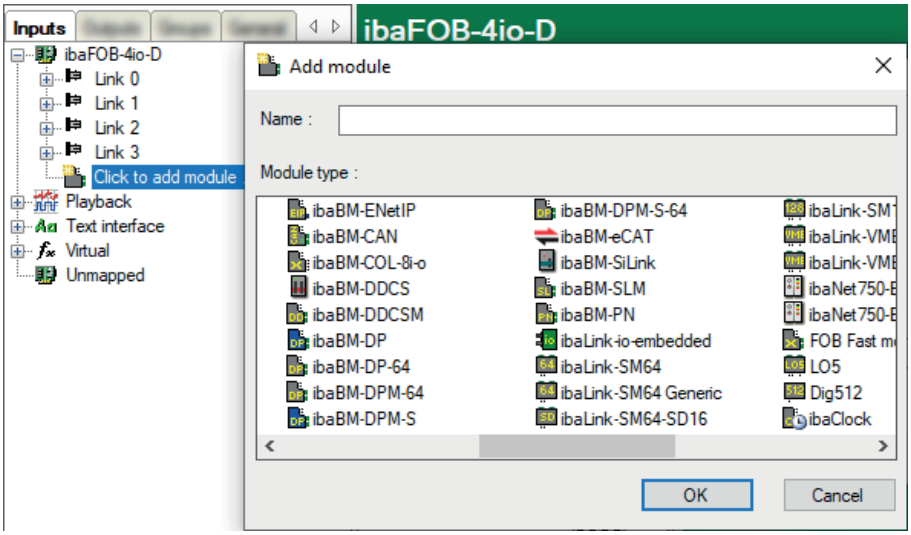
Other documentation

Detailed information about the *ibaFOB-D* card can be found in the respective manual.

Configuring ibaBM-PN

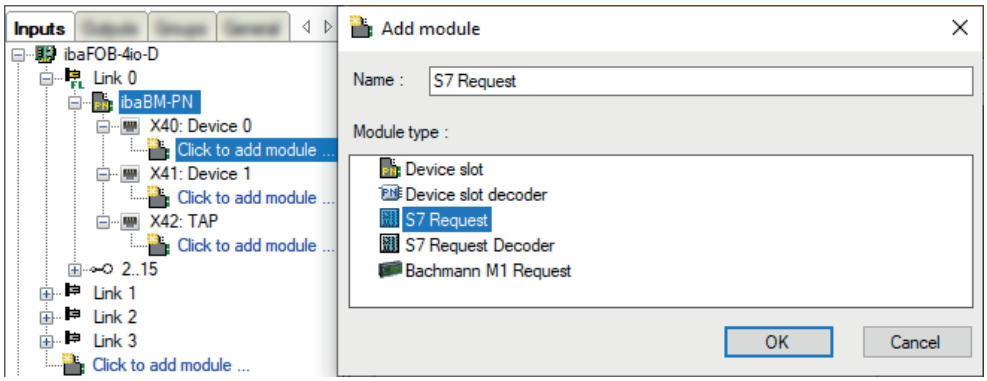
1. Start the *ibaPDA* client and open the I/O Manager.
2. In the interface tree (left), select the link of the *ibaFOB* card to which *ibaBM-PN* is connected.

   Click on the blue command *Click to add module*. Select the *ibaBM-PN* module in the dialog box and assign a name via the input field if required. Confirm the selection with <OK>.



3. Add a Request module (or several modules, in case you need more connections to one or to different S7-CPUs) to the *ibaBM-PN* module on the respective PROFINET connection Device 0 or Device 1. Available modules are:

   - S7 Request (for acquiring analog signals and digital signals)
   - S7 Request Decoder (for acquiring up to 1024 digital signals)



4. Configure the required module settings and signals as described in the following chapters. The *General* and *Connection* tabs are identical for all Request modules. The Request modules only differ in the *Analog* and *Digital* tabs.
5. After you have finished the configuration, click <Apply> or <OK> to transfer the new configuration to the device and start data acquisition with *ibaPDA*.

2.2.3.2 General module settings

You can find the description of all settings that are identical for all Request-S7 modules in the general chapter *General module settings* in part 1 of the manual.

You can find information on connection settings from the *Connection* tab in the general chapter *General module settings* in part 1 of the manual.

The *ibaBM-PN* modules have the following specific options in the *General* tab:

PROFINET

Device
Select the number of the PROFINET device to which the module is assigned.

Slot
Select the number of the slot to which the module is assigned.

Available slots
Shows the slots on the PROFINET device that can be used for the data of this module.

S2 Redundancy
FALSE: Redundancy mode disabled. For further information on the redundancy mode, see ↗ *Request-S7 for ibaBM-PN in redundancy mode, page 37*.

2.2.3.3 Module S7 Request

With the *S7 Request* module, you can acquire data according to the size of the PROFINET slots used. A maximum of 254 bytes is possible.

Configure a separate PROFINET slot and Request block call for each module.

For more information on the module settings, see chapter ↗ *General module settings, page 34* and the general chapter *General module settings* in part 1 of this manual.

2.2.3.4 Module S7 Request Decoder

The *S7 Request Decoder* module allows the acquisition of up to 16384 digital signals sent as a maximum of 128 words (16 Bit). This module type is suited for applications where large amounts of digital signals have to be acquired and for which the max. 1024 directly addressable digital values of the *ibaBM-PN* are not sufficient.

Configure a separate PROFINET slot and Request block call for each module.

General tab

For more information on the module settings, see chapter ↗ *General module settings, page 34* and the general chapter *General module settings* in part 1 of the manual.

Module specific settings

Module layout – No. of decoders
Define the number of configurable decoders in the digital signal table. The default value is 32. The maximum value is 126. The signal tables are adjusted accordingly.

Connection configuration

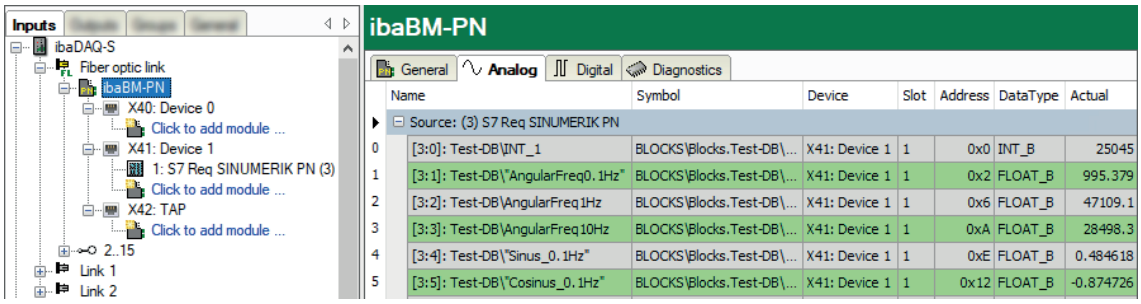
Configure the connection of the *S7 Request Decoder* module in the same way as the connection for an S7-Request module, see the general chapter *Connection settings* in part 1 of the manual.

Digital tab

Configure the signals in the same way as for an *ibaBM-DP* device, see chapter *Module S7 Request Decoder* for *ibaBM-DP* in part 1 of the manual.

2.2.4 Diagnostics

When you select the bus module node in the signal tree and choose the *Analog* or *Digital* tab, you get a list of all operands that have been acquired in the bus module with data type and actual value.

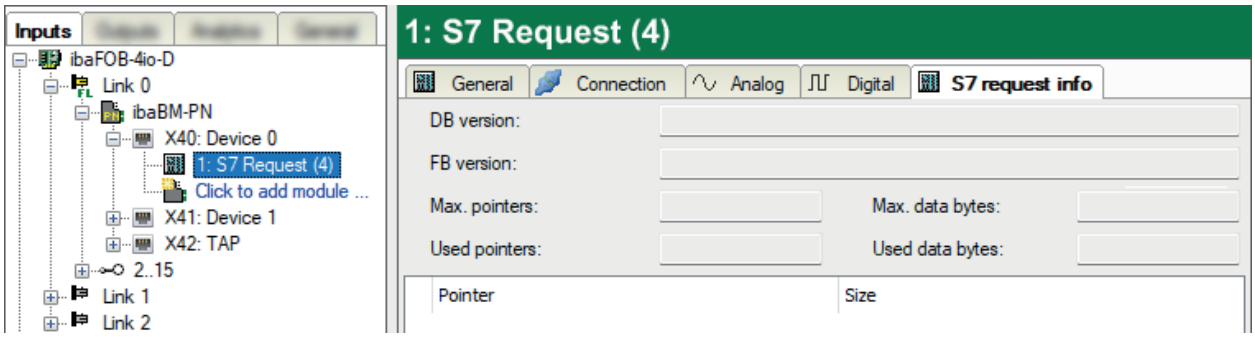


Other documentation

You find a detailed description of the device-specific diagnostic options of *ibaBM-PN* in the device manual.

Module diagnostics on the S7 request info tab

In addition to interface diagnostics, the request module provides the *S7 request info* tab.



DB version
Version of the data block used in the CPU

FB version
Version of the function block used in the CPU

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.

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).

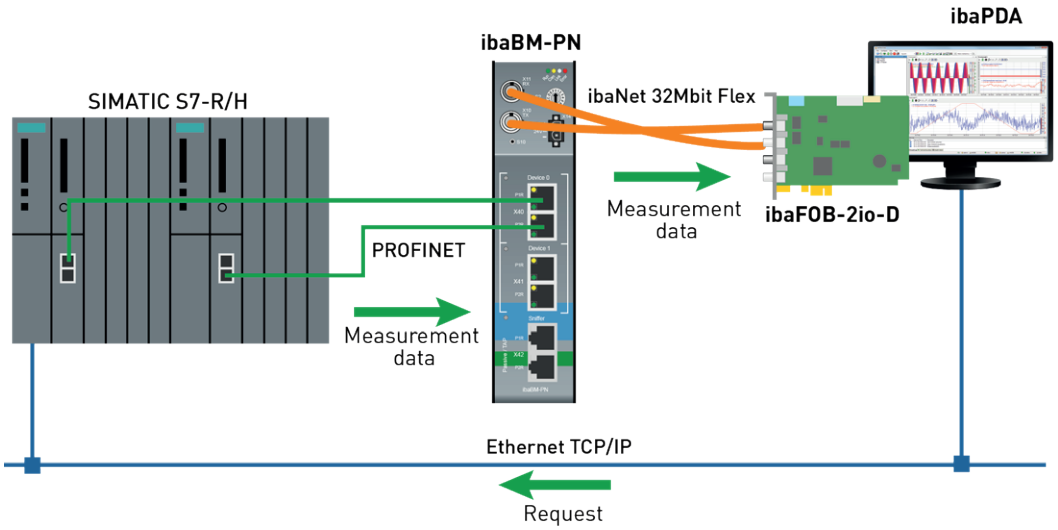
2.3 Request-S7 for ibaBM-PN in redundancy mode

This section describes the Request-S7 variant for the PROFINET bus module *ibaBM-PN* in S2 redundancy mode.

2.3.1 General information

With the S2 redundancy mode of *ibaBM-PN*, the device can be operated on redundant PROFINET systems (S2 system redundancy) in combination with SIMATIC S7-R/H controllers, whose measurement data is to be acquired. You need an additional license for the S2 redundancy mode. Contact the iba Support.

The following figure shows an exemplary integration of an *ibaBM-PN* in S2 redundancy mode:



Other documentation

Detailed information about the S2 redundancy mode of the *ibaBM-PN* can be found the device manual.

The functionality of the Request-S7 for *ibaBM-PN* in redundancy mode mostly corresponds to the functionality in standard mode (see chapter ↗ *Request-S7 for ibaBM-PN, page 26*). The differences and extensions are described in the following.

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

This section describes the configuration and engineering on the SIMATIC S7 side with SIMATIC Manager (STEP 7 Version ≤ V5) for *ibaBM-PN* in redundancy mode.

2.3.2.1 Hardware configuration

For *ibaBM-PN* in redundancy mode, configure a separate PROFINET device for each PROFINET device in use.

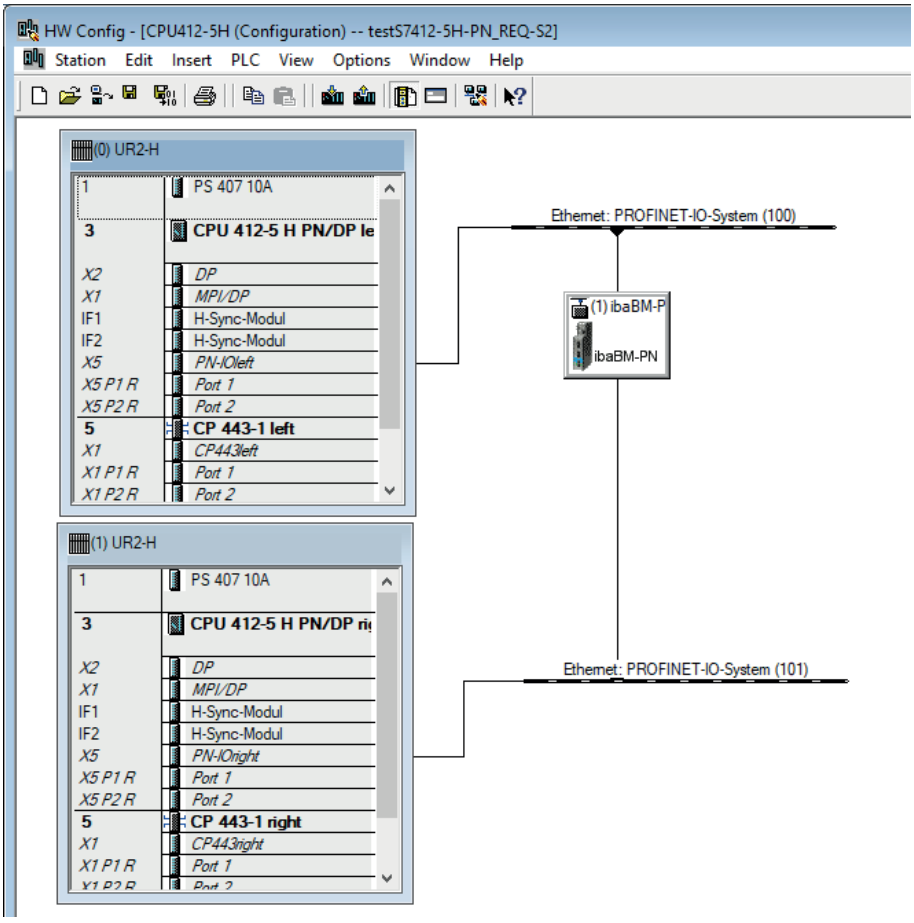
Use the same GSDML file as for normal operation (from version V2.35-20200101).

Note

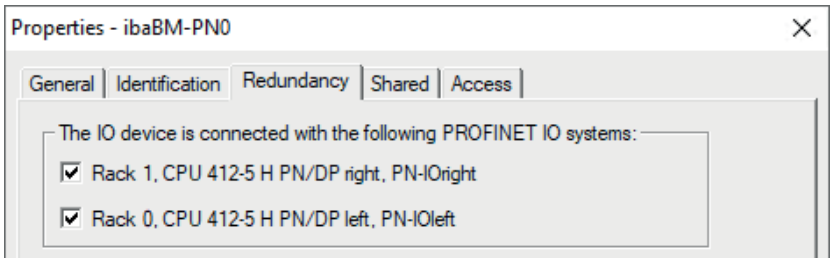
You find the latest version of the GSDML file on the data storage medium "iba Software & Manuals" in the following directory:

`\02_iba_Hardware\ibaBM-PN\01_GSD_Files\`

When inserting an *ibaBM-PN* in a redundant hardware configuration, this is automatically connected with both PROFINET IO systems.



You can configure the connection in the *Redundancy* tab of Properties.



Apply this hardware configuration to operate *ibaBM-PN* in the S2 redundancy mode.

2.3.2.2 Configuration in STEP 7

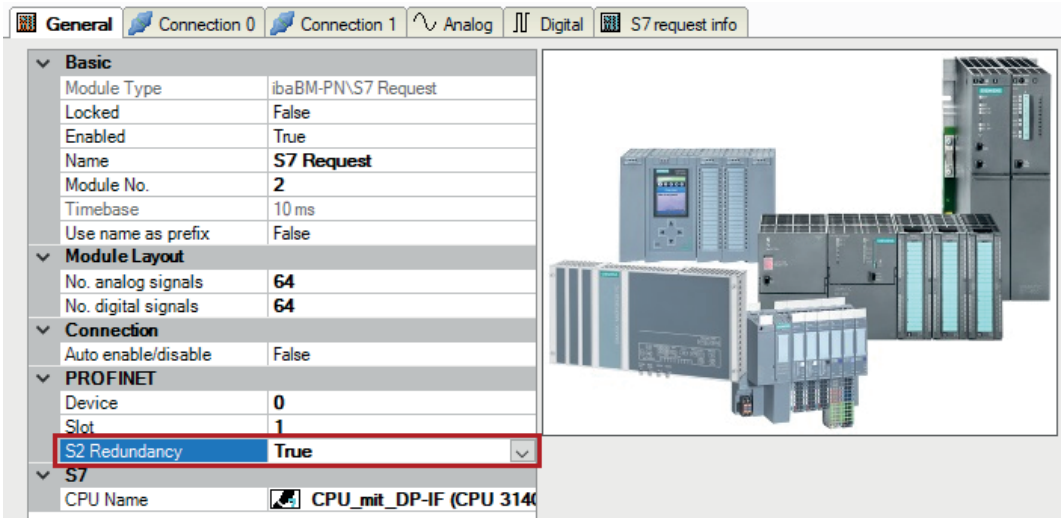
For *ibaBM-PN* in redundancy mode, the configuration of the request blocks corresponds to the configuration in normal operation, see ↗ *Configuration in STEP 7, page 29*.

2.3.3 Configuration and engineering ibaPDA

First connect *ibaBM-PN* to a free link of an *ibaFOB-D* card. In the I/O Manager, add a device module *ibaBM-PN* to the corresponding link.

Then add a module *S7 Request*.

In the *General* tab, set the parameter *S2 Redundancy* to "True". The orange colored icon of the Request module indicates the redundancy mode.

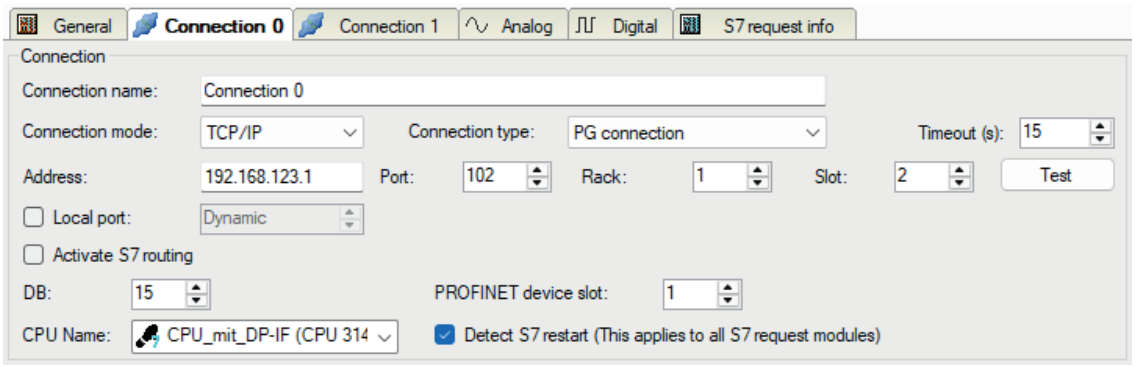


The configuration of the modules otherwise corresponds to the configuration in standard mode, see chapter *General module settings* in part 1.

Connection settings

For transmitting the operand data to both CPUs of the SIMATIC S7-R/H, set up two different connections. *ibaPDA* switches between these connections depending on availability and requirements.

Configure both connections on the tabs *Connection 0* and *Connection 1*. You can change the name of the tabs in the *Connection name* fields.



The other settings are identical to the settings for operation without redundancy mode, see chapter *Connection settings* in part 1.

Note these special features:

- The rack numbers for an H system are 0 or 1 for the both redundant CPUs.
- Set the DB number, the PROFINET device slots and the CPU name for assigning an address book only once in the *Connection 0* tab.

2.4 Request-S7 for ibaBM-DP

This section describes the Request-S7 variant for the PROFIBUS bus module *ibaBM-DP*.

2.4.1 General information

Request-S7 for *ibaBM-DP* can be configured in the following system configurations:

SIMATIC S7 CPU	SIMATIC STEP 7 V5.x (SIMATIC Manager)	SIMATIC STEP 7 V1x Professional (TIA Portal)
S7-300 integrated DP interface	X	X
S7-400 integrated DP interface and CP443-5	X	X
S7-400H integrated DP interface and CP443-5	X	-

For DP Request, the measurement values (Request Handshake) are not requested via the PROFIBUS, but via a separate connection.

Depending on the hardware and software, different access points can be selected for the Request:

- TCP/IP: The connection to the SIMATIC S7 is established using an integrated PN interface of the S7 CPU or the respective CP modules in the PLC and the standard network interface of the computer. No additional Siemens software is required for the connection.
- PC/CP: This is the designation for different SIMATIC specific access points. In contrast to the TCP/IP connection, for the PC/CP group the SIMATIC communication software (and corresponding licenses) must be installed on the computer with the corresponding functions enabled.
  - MPI, PROFIBUS: The connection to the SIMATIC S7 is established using the MPI or PROFIBUS interface of the computer; e.g. with the CP5611 PCI card or the MPI adapter for USB interface or serial PC interface.
  - TCP/IP, ISO: Here, either the standard network interface of the computer or a suitable interface card is used for the connection to the S7.

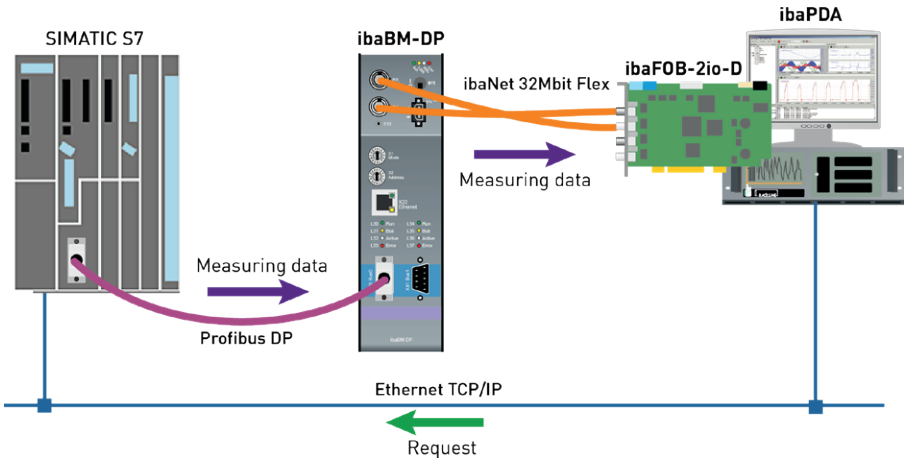
System integration with ibaBM-DP

The measured data are transmitted to the *ibaBM-DP* device over PROFIBUS DP.

You need the following connections:

- Online connection between *ibaPDA* and S7-CPU (TCP/IP, MPI or DP)
- Fiber optic connection between *ibaPDA*/*ibaFOB-io-D* and *ibaBM-DP*
- PROFIBUS connection between *ibaBM-DP* and S7 PROFIBUS master
- A connection from *ibaBM-DP* to the network (TCP/IP over Ethernet) is only needed if the device is operated in compatibility mode, i.e. not with 32Mbit Flex (short: Flex). In this case the system integration is like with ibaBM-DPM-S, see ↗ *Request-S7 for ibaBM-DPM-S, page 62*.

In the standard version, you can configure a maximum of 8 connections per device, i.e. 8 PROFIBUS slaves. Up to 244 Byte data can be transmitted per slave.



Note

The following description refers to the Request blocks V4.0 or higher. If you require information regarding older versions, contact the iba Support.

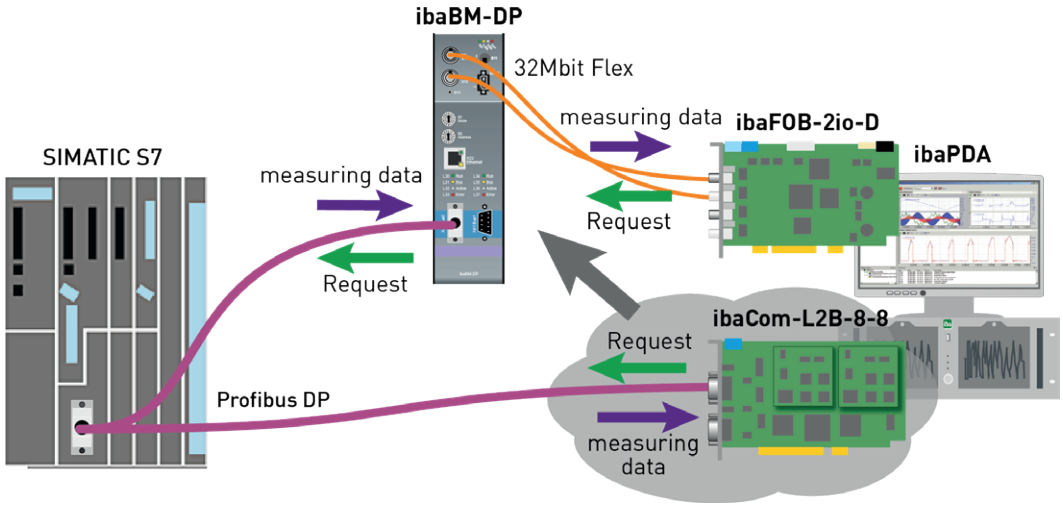
Other documentation

Detailed information about *ibaBM-DP* can be found in the device manual.

For information about application examples, see chapter ↗ *Application examples, page 131*.

2.4.1.1 ibaCom-L2B compatibility mode

You can use *ibaBM-DP* for the Request-S7 functionality of the *ibaCom-L2B*-PROFIBUS card as a successor with full functional compatibility.



For this purpose, you can use the following modules:

- S7 Request (ibaCom-L2B compatible)
- S7 Request Dig512 (ibaCom-L2B compatible)

Compared to the original configuration on the basis of the *ibaCom-L2B* card, modifications in the S7 configuration are not necessary (Hardware or S7 program).

For detailed information about this subject, see ↗ *Request-S7 for ibaCom-L2B, page 71* and ↗ *Replacing Request-S7 on ibaCom-L2B by ibaBM-DP, page 147*.

Note

The *ibaCom-L2B* compatibility mode can only be used in the Flex-mode of *ibaBM-DP*.

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

This section describes the configuration and engineering on the SIMATIC S7 side with SIMATIC Manager (STEP 7 Version ≤ V5) for *ibaBM-DP* when using the modules *S7 Request* and *S7 Request Decoder*.

The configuration and engineering when using the compatibility modules *S7 Request (ibaCom-L2B compatible)* and *S7 Request Dig512 (ibaCom-L2B compatible)* corresponds to using an *ibaCom-L2B* card, see ↗ *Configuration and engineering SIMATIC S7-300, S7-400 and WinAC, page 71*.

2.4.2.1 Hardware configuration

For *ibaBM-DP*, specify an iba-PROFIBUS slave for each module.

Use the GSD file `ibaDPMSi.gsd` (version V2.2 and higher).

Note

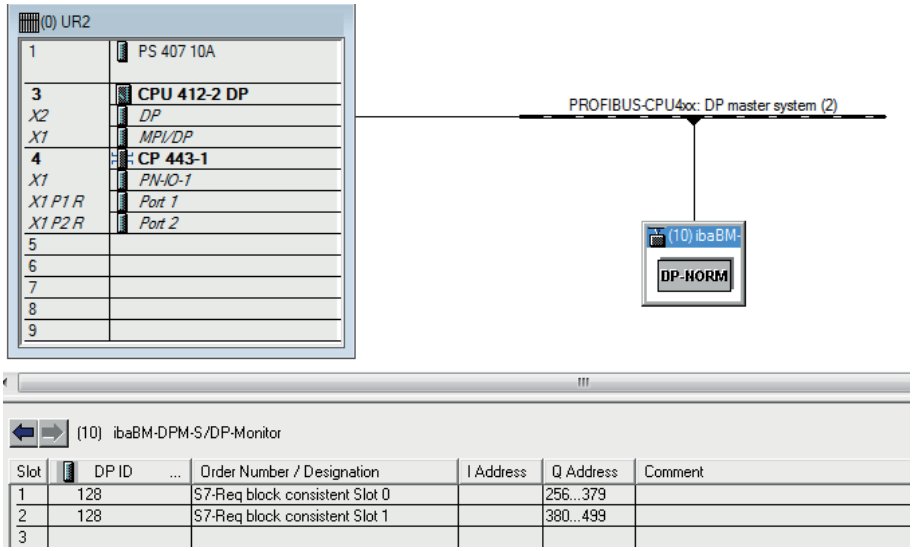
You find the GSD file `ibaDPMSi.gsd` on the data storage medium "iba Software & Manuals" in the following directory:

`\02_iba_Hardware\ibaBM-DP\02_GSD_Files\01_General\`

Note

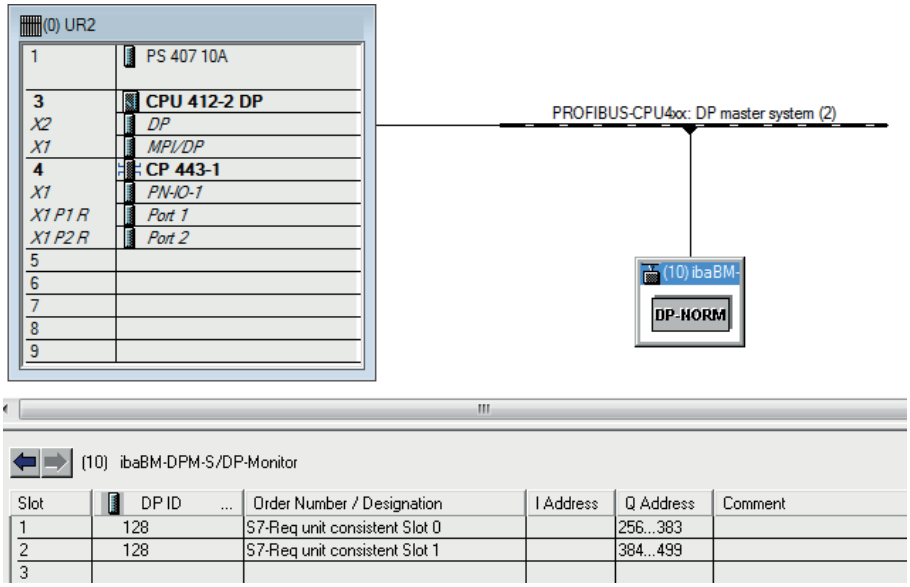
The slots have to be created without gaps and with consecutive address range.

S7-CPUs of newer type[1] provide consistent slots with a max. of 128 Bytes.
Use the elements "S7-Req **block consistent** Slot 0 / Slot 1".



[1] More recent S7 CPU models are generally: S7-400 with firmware version 3.0 or higher and S7-300 with firmware version 2.0 or higher

For **S7-CPUs and for CP443-5 of older types**, you cannot use no long consistent slots.
Use the elements "S7-Req **unit consistent** Slot 0 / Slot 1"



2.4.2.2 Configuration in STEP 7 (STL, LAD, FBD)

Request-S7 for *ibaBM-DP* is intended for the use with CPUs S7-300/400 with integrated DP interface or CPUs S7-400 with external interface CP 443-5 (PROFIBUS master).

Note

The use of the external DP interface CP342-5 of a CPU S7-300 is not supported!

For Request-S7 with one PROFIBUS slave

1. Copy the following blocks from the iba S7 library to the blocks folder of your STEP 7 project, see ↗ *iba S7 library, page 128*. The ibaDP_Req has to be called once for each slave in the cyclic program.
   - FC122 (ibaDP_Req), see ↗ *Request-FC ibaDP_Req (FC122), page 103*
   - DB10 (ibaDP_DB_PDA)
   - DB25 (ibaDP_DB_work)

Note

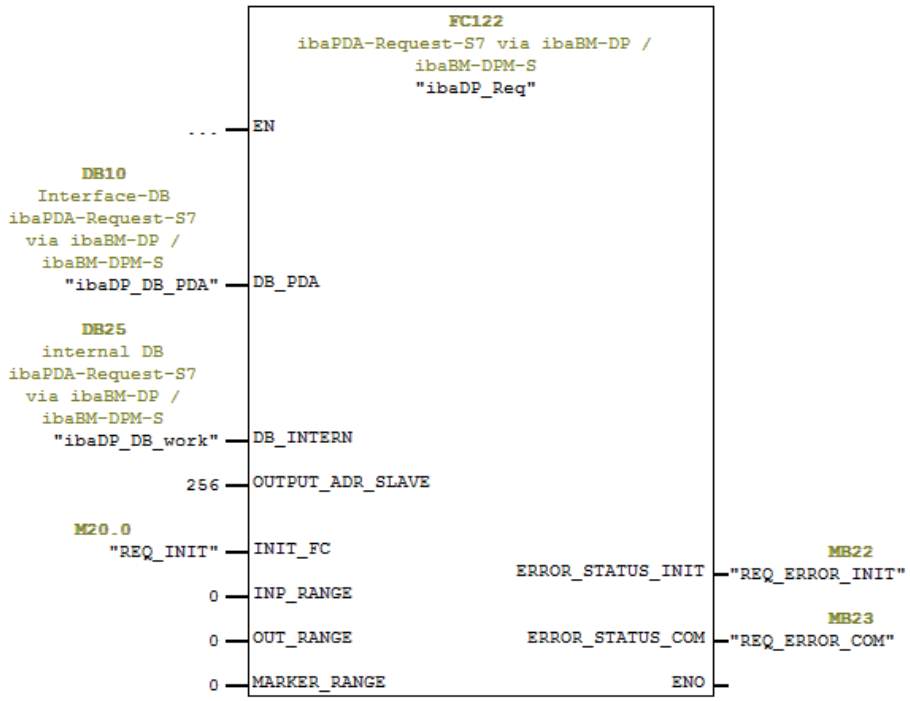
Only use Request blocks from the latest iba S7 library!

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

Note

If the block numbers FC122, DB10 and DB25 are already used in your project, assign other free numbers to the blocks from the iba S7 library when copying.

2. Create the error OBs (OB82, OB85, OB86, OB87, OB122) in order to prevent CPU stops in case of an error.
3. Call and parameterize the ibaDP_Req (FC122) in the cyclic program.



For Request-S7 with multiple PROFIBUS slaves

- In the blocks folder, a data block ibaDP_DB_PDA (DB10) has to be available for each slave. Copy the DB10 in a DB with a new DB number.
- In the blocks folder, a data block ibaDP_DB_work (DB25) has to be available for each slave. Copy the DB25 in a DB with a new DB number.
- In the cyclic OB, the ibaDP_Req (FC122) has to be called once more for each slave with the new DB numbers and the peripheral addresses of the new PROFIBUS slave.
- The input INIT_FC and the outputs ERROR_STATUS_INIT and ERROR_STATUS_COM should have unique markers (or DB elements) for each slave.

Final

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

2.4.2.3 Configuration in STEP 7 (CFC)

For Request-S7 with one PROFIBUS slave

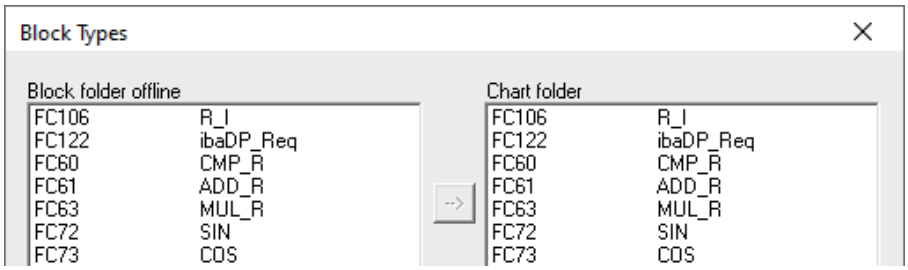
1. Copy the following blocks from the iba S7 library to the blocks folder of your STEP 7 project, see ↗ *iba S7 library, page 128*. The ibaDP_Req has to be called once for each slave in the cyclic program.
   - FC122 (ibaDP_Req), see ↗ *Request-FC ibaDP_Req (FC122), page 103*
   - DB10 (ibaDP_DB_PDA)
   - DB25 (ibaDP_DB_work)

Note

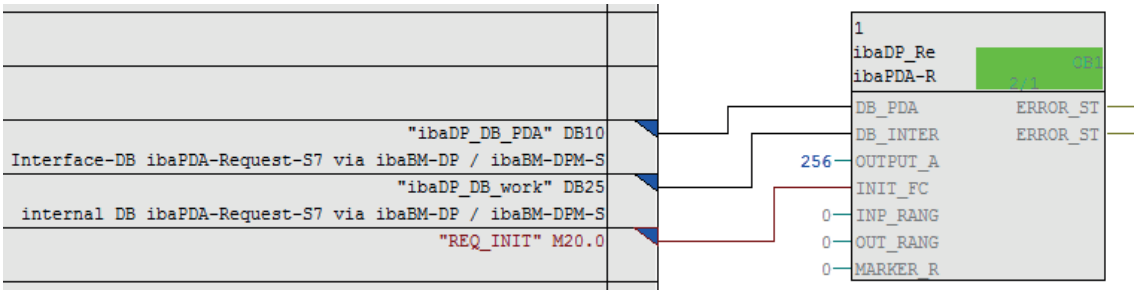
If the block numbers FC122, DB10 and DB25 are already used in your project, assign other free numbers to the blocks from the iba S7 library when copying.

The chosen block numbers have to match the reserved number ranges for DBs and FCs under the CFC settings for *Translating/Loading*.

2. Start the CFC editor and import the ibaDP_Req (FC122) (under *Options – Block Types*). The FC is stored in the group "ibaPDA" in the function block library.



3. Drag the ibaDP_Req (FC122) to your CFC chart. Pay attention to the run sequence. The function block must be called in a cyclic OB (cyclic interrupt OB or OB1).



Make sure to create the error OBs (OB82, OB85, OB86, OB87 and OB122) in order to prevent the CPU from running into STOP in case of an access failure.
To do so, you have to define an empty runtime group within the run sequence of these tasks if they are empty. Alternatively, you may place any dummy block into the error OB. This block must not be deleted, too.

Note

Do **not** choose the option *Delete empty runtime groups* for compilation otherwise the error OBs are removed! Some older CFC versions have this option in the compilation dialog.

For Request-S7 with multiple PROFIBUS slaves

- In the blocks folder, a data block ibaDP_DB_PDA (DB10) has to be available for each slave. Copy the DB10 in a DB with a new DB number.
- In the blocks folder, a data block ibaDP_DB_work (DB25) has to be available for each slave. Copy the DB25 in a DB with a new DB number.
- In the CFC chart, you have to create another instance of the ibaDP_Req (FC122) for each slave. Configure the connectors with the new DB numbers and the peripheral addresses of the new PROFIBUS slave.

After each program modification

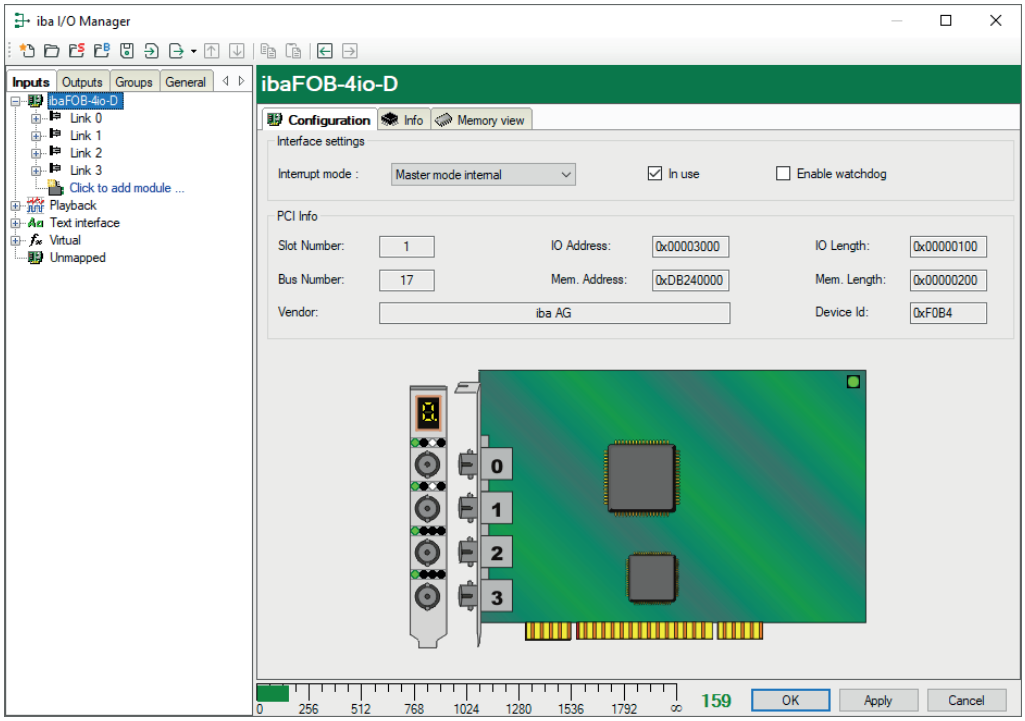
1. Compile the complete program:
   a) Activate the option *Entire program*.
   b) Activate the option *Generate SCL source* (for SIMATIC CFC version 6.1 or higher).
   c) Deactivate the option *Delete empty runtime groups* if available.
2. Load the program and restart the S7-CPU.

2.4.3 Configuration and engineering ibaPDA

This section describes the configuration of the Request modules in *ibaPDA* for *ibaBM-DP*.

2.4.3.1 General interface settings

If an *ibaFOB-D* card is installed in the *ibaPDA* computer, *ibaPDA* displays the interface of the *ibaFOB-D* card in the interface tree of the I/O Manager.



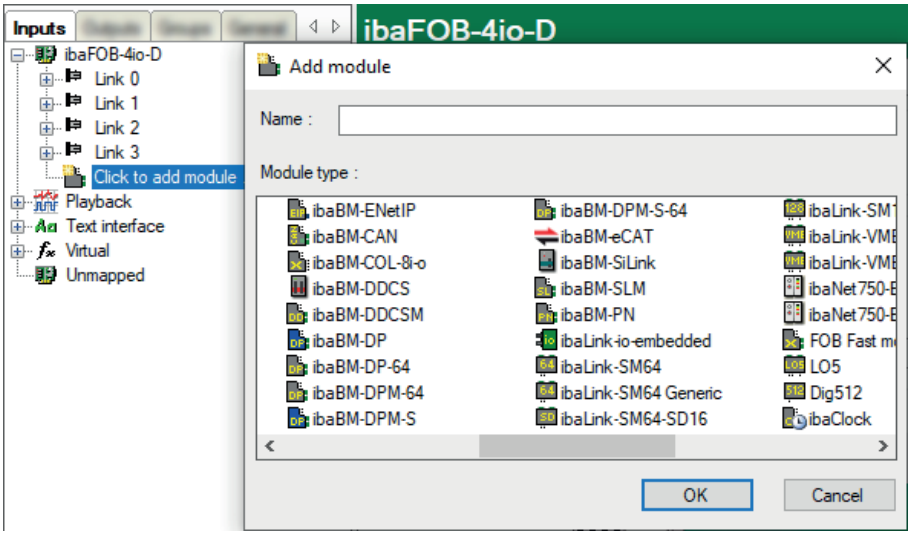
Other documentation

Detailed information about the *ibaFOB-D* card can be found in the respective manual.

Configuring ibaBM-DP

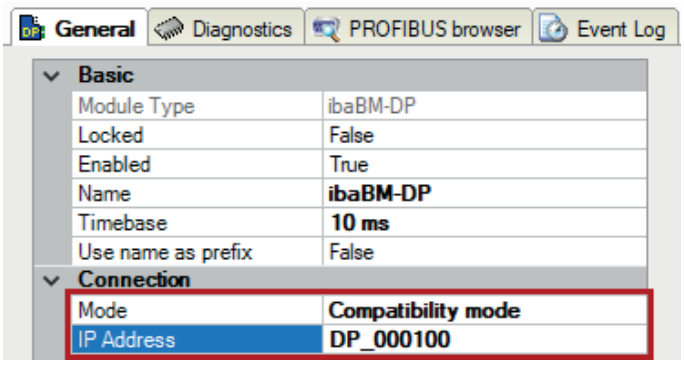
1. Start the *ibaPDA* client and open the I/O Manager.
2. In the interface tree (left), select the link of the *ibaFOB* card to which *ibaBM-DP* is connected.

   Click on the blue command *Click to add module*. Select the *ibaBM-DP* module in the dialog box and assign a name via the input field if required. Confirm the selection with <OK>.



3. If you operate the device in Flex mode, *ibaPDA* sets the IP address of the device on the *General* tab under *Connection* automatically.

   If you operate the device in compatibility mode, enter the IP address of the device on the *General* tab under *Connection*: either as name e.g. "DP_000100" or as IP address e.g. "192.168.81.123".

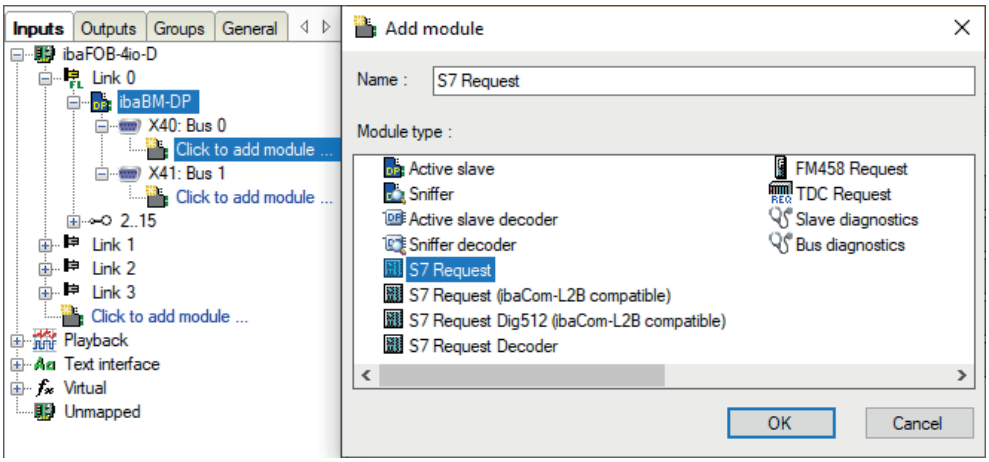


Note

An online connection to the device via Ethernet or USB is mandatory in compatibility mode. The manual of the *ibaBM-DP* device explains how to establish and check this connection.

The device manual also describes differences between Flex mode and compatibility mode in detail.

4. Add a request module (or several modules, in case you need more connections to one or to different S7-CPUs) to the *ibaBM-DP* module on the respective PROFIBUS connection Bus 0 or Bus 1. Available modules are:
   - S7 request (for acquiring analog signals and digital signals)
   - S7 request decoder (for the acquisition of up to 1024 digital signals)



5. Configure the required module settings and signals as described in the following chapters.

   The *General* and *Connection* tabs are identical for all Request modules. The Request modules only differ in the *Analog* and *Digital* tabs.

6. After you have finished the configuration, click <Apply> or <OK> to transfer the new configuration to the device and start data acquisition with *ibaPDA*.

2.4.3.2 General module settings

You can find the description of all settings that are identical for all Request-S7 modules in the general chapter *General module settings* in part 1 of the manual.

You can find information on connection settings from the *Connection* tab in the general chapter *General module settings* in part 1 of the manual.

The *ibaBM-DP* modules have the following specific options in the *General* tab:

PROFIBUS

Bus number
0 = connector X40 left, 1 = connector X41 right

Slave number
PROFIBUS slave address assigned to the module.

Connection

Auto enable/disable
If TRUE, the acquisition is started, even if no connection can be established to the S7-CPU. The module is deactivated. During the acquisition, *ibaPDA* tries to reconnect to the S7-CPU. If successful, the acquisition is restarted with this module enabled.
If FALSE, the acquisition is not started, if a connection to the configured S7-CPU cannot be established.

2.4.3.3 Module S7 Request

With the *S7 request* module, you can acquire up to 64 analog signals and 64 digital signals.

Configure a separate PROFIBUS slave and Request block call for each module.

For a description of the module settings, see the general chapter *General module settings* and *General module settings* for *ibaBM-DP* in part 1 of the manual.

2.4.3.4 Module S7 Request Decoder

The *S7 Request Decoder* module allows the acquisition of up to 1024 digital signals sent as a maximum of 64 words (16 Bit). This module type is suited for applications where large amounts of digital signals have to be acquired and for which the max. 1024 directly addressable digital values of the *ibaBM-DP* are not sufficient.

Configure a separate PROFIBUS slave and Request block call for each module.

General tab

For more information on the module settings, see ↗ *General module settings, page 50*, and the general chapter *General module settings* in part 1 of the manual.

Connection configuration

Configure the connection of the *S7 Request Decoder* module in the same way as the connection for an S7-Request module, see the general chapter *Connection settings* in part 1 of the manual.

Digital tab

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

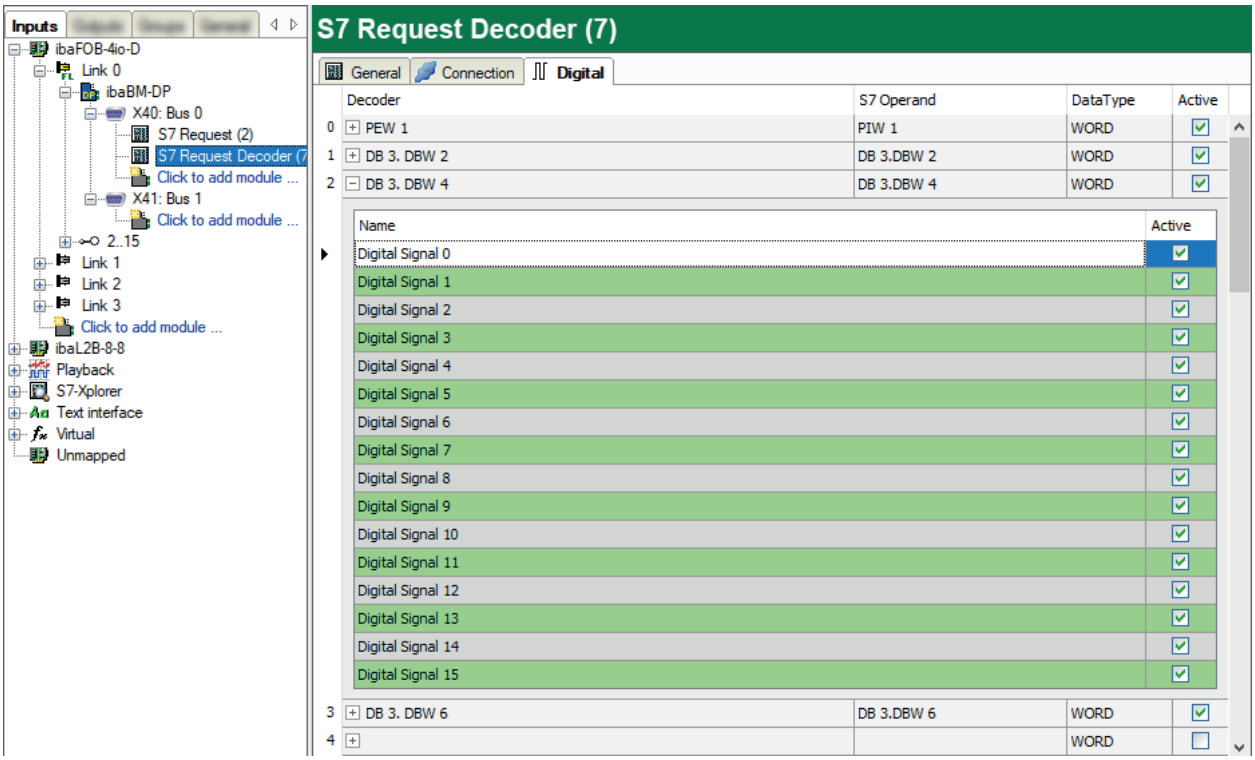
- First define the words (source signals), which are broken down for the digital signals (bits).

  You can directly enter the words as basic signals for decoding via the absolute S7 operands. Only word operands (e.g. PIW, MW, DBW) are allowed.

  You can also use S7 symbols by generating address books. For detailed information, see chapter *Selection via the symbolic operand addresses* in part 1 of the manual. 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.

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

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



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

Source signal

Decoder
Enter a name for the source signal.

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

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

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

Individual digital signals (bits)

Name
Enter a name for the individual digital signals.

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

Note

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

2.4.3.5 Module S7 Request (ibaCom-L2B compatible)

The module *S7 Request (ibaCom-L2B compatible)* is fully compatible to the module *L2B S7 Request* when using an *ibaCom-L2B* card, see↗ *Configuration and engineering ibaPDA, page 80*.

Configure the SIMATIC S7 side according to chapter ↗ *Configuration and engineering SIMATIC S7-300, S7-400 and WinAC, page 71*.

Note

This module type usually is used to replace an existing Request-S7 solution based on an *ibaCom-L2B* card by an *ibaBM-DP* device. For more information about this application case, see chapter ↗ *Replacing Request-S7 on ibaCom-L2B by ibaBM-DP, page 147*.

2.4.3.6 Module S7 Request Dig512 (ibaCom-L2B compatible)

The module *S7 Request Dig512 (ibaCom-L2B compatible)* is fully compatible to the module *L2B S7 Request Dig512* when using an *ibaCom-L2B* card, see ↗ *Configuration and engineering ibaPDA, page 80*.

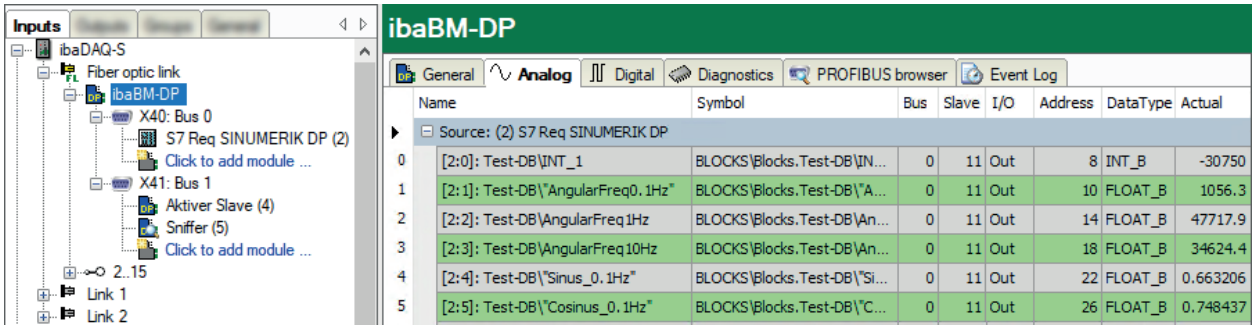
Configure the SIMATIC S7 side according to ↗ *Configuration and engineering SIMATIC S7-300, S7-400 and WinAC, page 71*.

Note

This module type usually is used to replace an existing Request-S7 solution based on an *ibaCom-L2B* card by an *ibaBM-DP* device. For more information about this application case, see chapter ↗ *Replacing Request-S7 on ibaCom-L2B by ibaBM-DP, page 147*.

2.4.4 Diagnostics

When you select the bus module node in the signal tree and select the *Analog* or *Digital* tab, you get a list of all operands that are acquired in the bus module with data type and actual value.

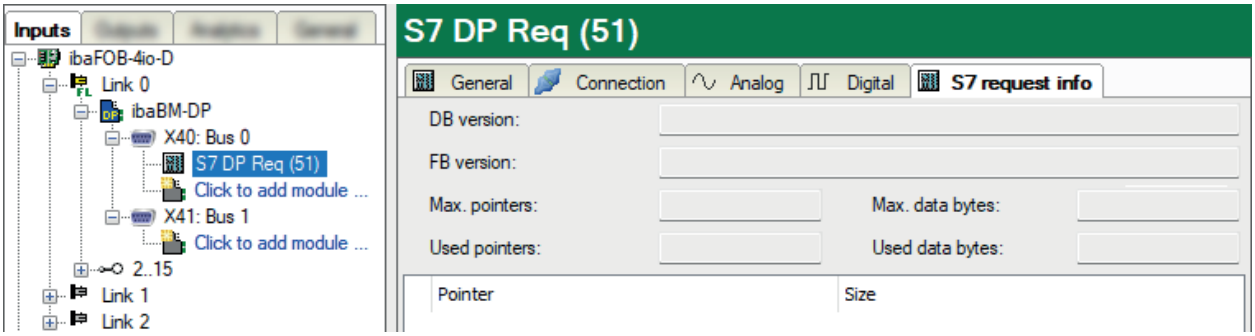


Other documentation

Detailed information about the device-specific diagnostic options of *ibaBM-DP* can be found in the device manual.

Module diagnostics on the S7 request info tab

In addition to interface diagnostics, the request module provides the *S7 request info* tab.



DB version
Version of the data block used in the CPU

FB version
Version of the function block used in the CPU

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.

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).

2.5 Request-S7 for ibaBM-DP in redundancy mode

This section describes the Request-S7 version for the PROFIBUS bus module *ibaBM-DP* in redundancy mode is described.

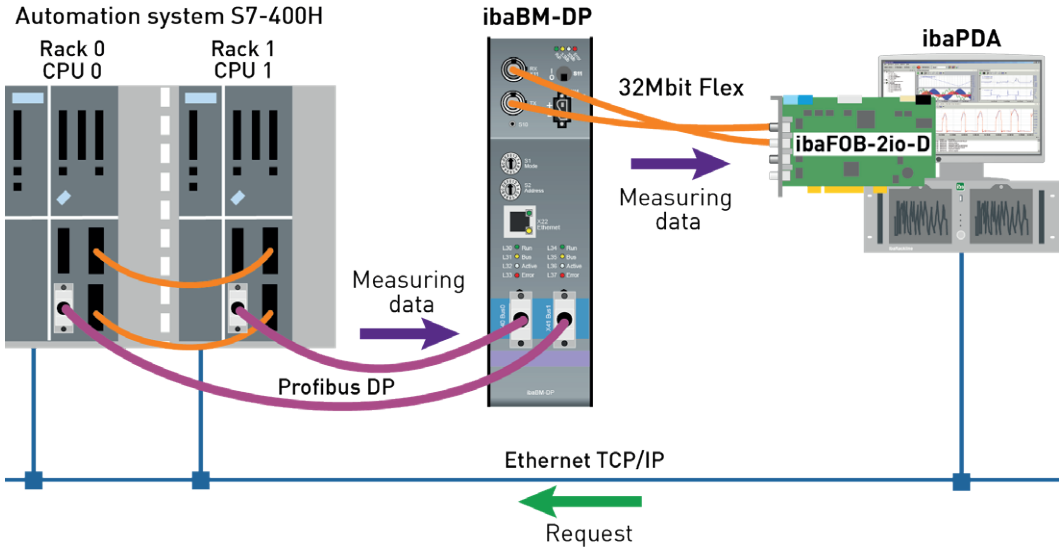
2.5.1 General information

With the redundancy mode of *ibaBM-DP*, the device can be operated on redundant PROFIBUS systems in combination with SIMATIC-400H controllers, whose measurement data are to be acquired.

You need an additional license for the redundancy mode. Contact the iba Support. The license is released using the administrator functions in the Web dialog of the *ibaBM-DP*.

Request-S7 redundant is suitable for operating *ibaBM-DP* with the Request functionality as single-channel periphery on a highly available SIMATIC S7-400H controller.

The following figure shows an exemplary integration of an *ibaBM-DP* in redundancy mode.



Note

The following description refers to the Request blocks V4.0 or higher.

Other documentation

Detailed information about the redundancy mode of the *ibaBM-DP* can be found the device manual.

The functionality of the Request-S7 for *ibaBM-DP* in redundancy mode mostly corresponds to the functionality in standard mode, see ↗ *Request-S7 for ibaBM-DP, page 41*. The differences and extensions are described in the following.

For information about application examples, see chapter ↗ *Application examples, page 131*.

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

This section describes the configuration and engineering on the SIMATIC S7 side with SIMATIC Manager (STEP 7 Version ≤ V5) for *ibaBM-DP* in redundancy mode.

2.5.2.1 Hardware configuration

For *ibaBM-DP* in redundancy mode, Specify a separate ibaBM-PROFIBUS slave for each PROFIBUS master.

Use the GSD file `ibaDPMSi.gsd` (version V2.2 and higher).

Note

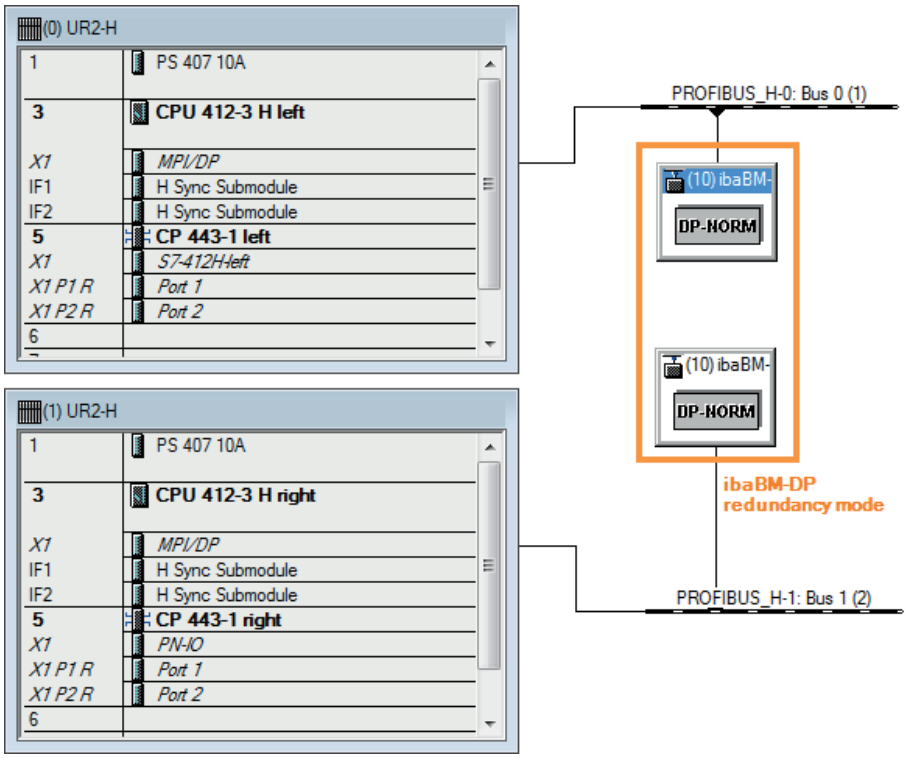
You find the GSD file `ibaDPMSi.gsd` on the data storage medium "iba Software & Manuals" in the following directory:

`\02_iba_Hardware\ibaBM-DP\02_GSD_Files\01_General\`

Note

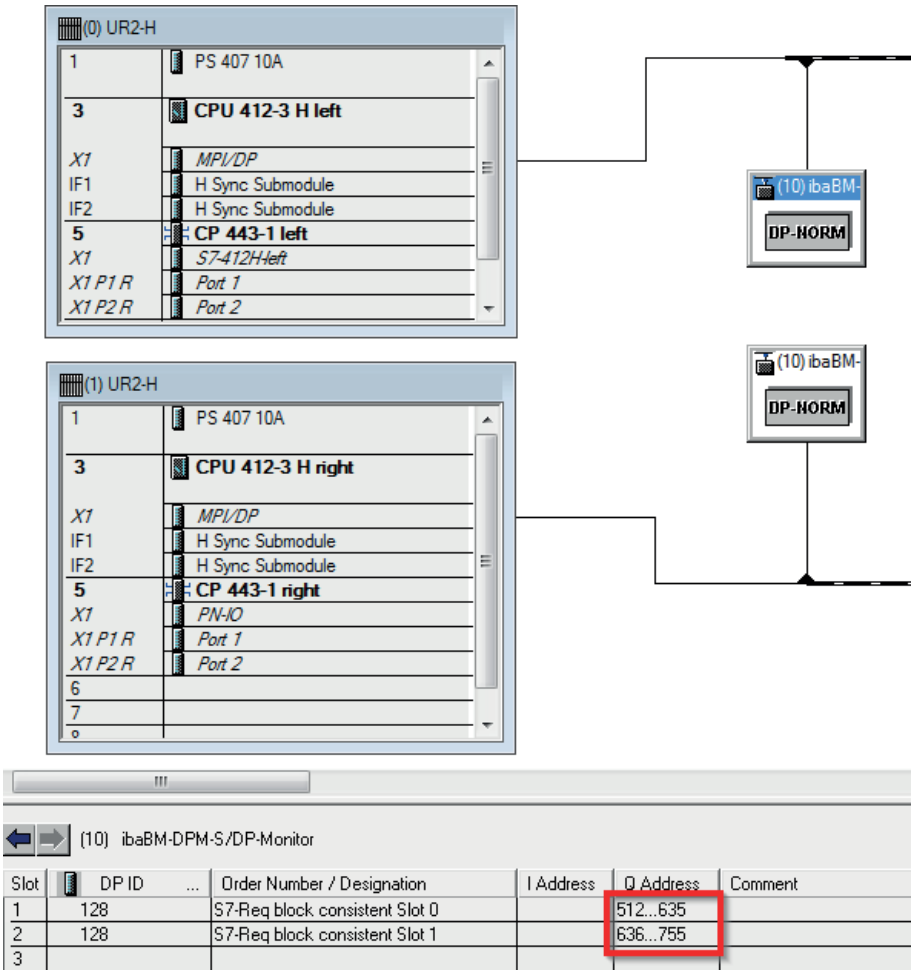
The modules have to be created for each slave without gaps and with sequential start addresses.

The following figure shows an exemplary hardware configuration for a redundant system.

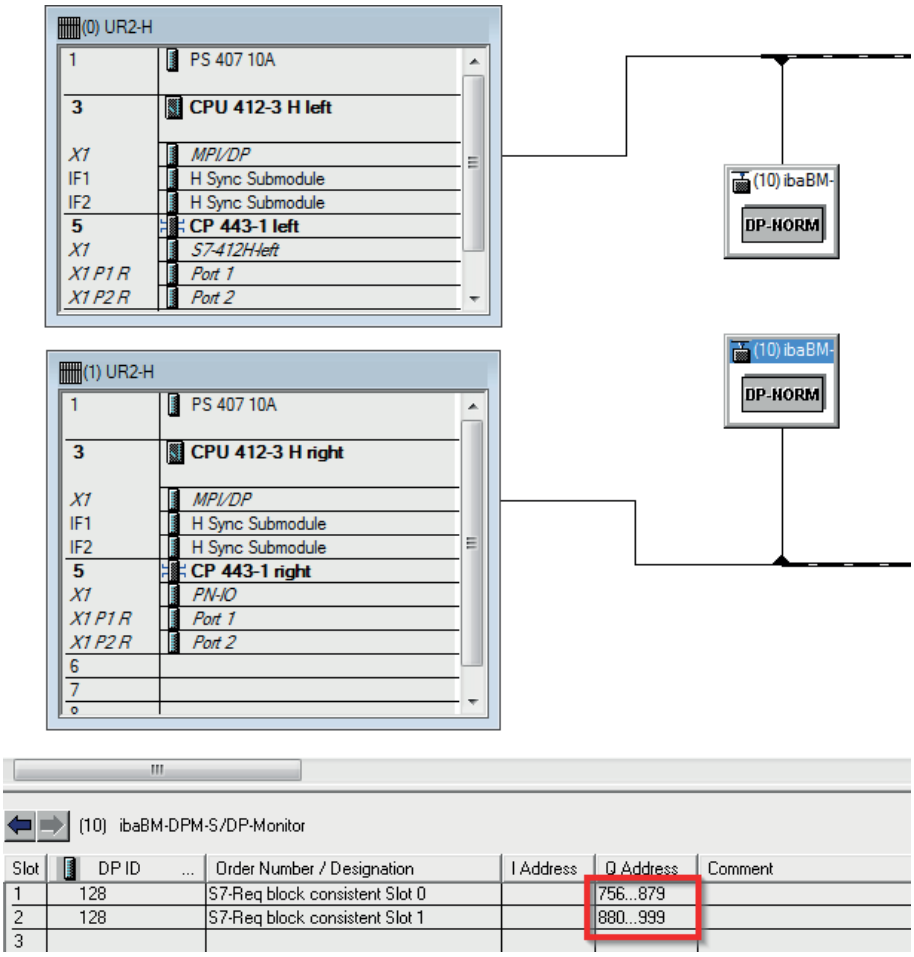


Both configured slaves have to be configured with identical bus addresses. The I/O address ranges are different. Use the modules "S7-Req block consistent Slot 0 / Slot 1".

The following figure shows the slave I/O addresses for Bus 0.



The following figure shows the slave I/O addresses for Bus 1.



2.5.2.2 Configuration in STEP 7 (STL, LAD, FBD)

For *ibaBM-DP*, Request-S7 redundant is intended for the use with S7-400H CPUs with integrated DP interface as well as with external interface CP 443-5 (PROFIBUS master).

For Request-S7 with one PROFIBUS slave

1. Copy the following blocks from the iba S7 library to the blocks folder of your STEP 7 project, see ↗ *iba S7 library, page 128*. The ibaDP_Req_H function has to be called once per slave pair in the cyclic program.
   - FC123 (ibaDP_Req_H) , see ↗ *Request-FC ibaDP_Req_H (FC123), page 106*
   - DB10 (ibaDP_DB_PDA)
   - DB25 (ibaDP_DB_work)

Note

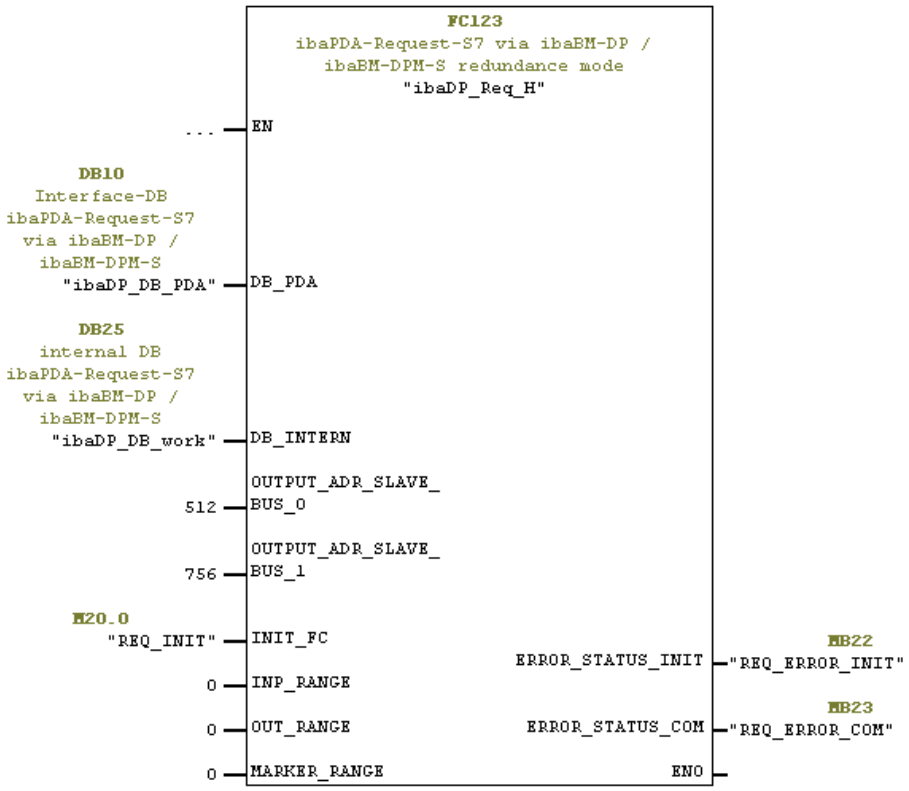
Only use Request blocks from the latest iba S7 library!

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

Note

If the block numbers FC123, DB10 and DB25 are already used in your project, assign other free numbers to the blocks from the iba S7 library when copying.

2. Create the error OBs (OB82, OB85, OB86, OB87, OB122) in order to prevent CPU stops in case of an error.
3. Call and parametrize the ibaDP_Req_H (FC123) in the cyclic program.



For Request-S7 with multiple PROFIBUS slaves

- In the blocks folder, a data block ibaDP_DB_PDA (DB10) has to be available for each slave pair. Copy the DB10 in a DB with a new DB number.
- In the blocks folder, a data block ibaDP_DB_work (DB25) has to be available for each slave pair. Copy the DB25 in a DB with a new DB number.
- In the cyclic program, another call of the ibaDP_Req_H (FC123) with the new DB numbers and the respective peripheral addresses of the new PROFIBUS slave pair has to be done for each slave pair.
- The assignment of markers to the input INIT_FC and the outputs ERROR_STATUS_INIT or ERROR_STATUS_COM (or DB elements) should be unambiguous for each slave pair.

Final

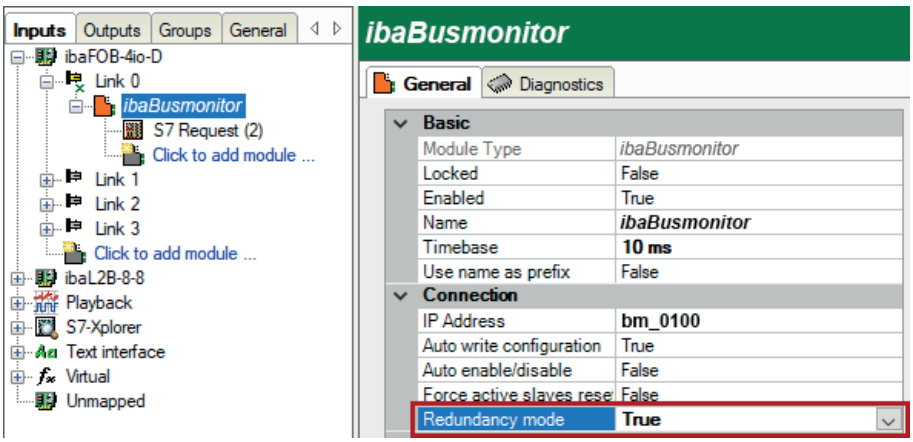
- Load all blocks into the S7 CPU and restart the S7 CPU.

2.5.3 Configuration and engineering ibaPDA

First connect *ibaBM-DP* to a free link of an *ibaFOB-D* card. Insert a device module *ibaBM-DP* on the respective link in the I/O Manager.

In the *General* tab, set the parameter *Redundancy mode* to "True". The orange colored icon of the device module indicates the redundancy mode.

You can operate the device in the 32Mbit Flex mode as well as in compatibility mode with 32Mbit.



As on the redundant PROFIBUS both bus systems 0 and 1 are operated in parallel, there are only settings for *Redundant PROFIBUS* on the *General* tab.

In redundancy mode, the following modules are available with Request-S7:

- S7 Request
- S7 Request Decoder

Note

The following modules cannot be used in redundancy mode:

- Module S7 Request (ibaCom-L2B compatible)
- Module S7 Request Dig512 (ibaCom-L2B compatible)

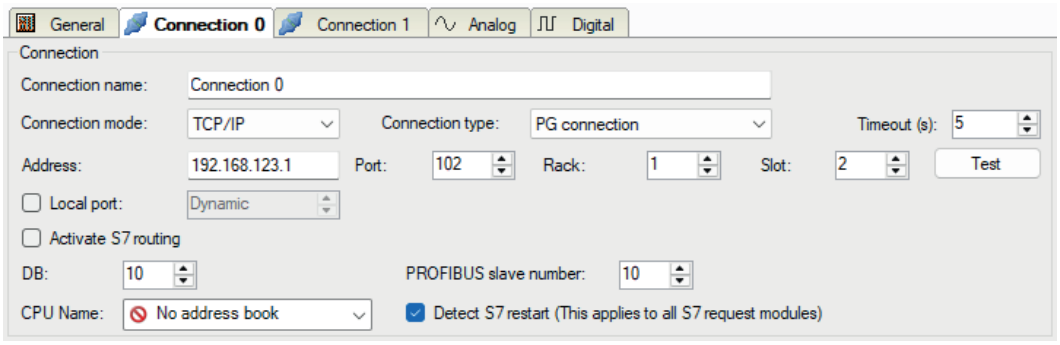
These modules are deactivated automatically and cannot be activated manually, either.

The configuration of the modules corresponds to the configuration in standard mode, see ↗ *General module settings, page 50*.

Connection settings

For transmitting the operand data to both CPUs of the SIMATIC S7-400H, set up two different connections. *ibaPDA* switches between these connections depending on availability and requirements.

Configure both connections on the tabs *Connection 0* and *Connection 1*. You can change the name of the tabs in the *Connection name* fields.



The other settings are identical to the settings for operation without redundancy mode, see general chapter *Connection settings* in part 1 of the manual.

Note these special features:

- The rack numbers for an H system are 0 or 1 for the both redundant CPUs.
- Set the DB number, the PROFIBUS slave number and the CPU name for assigning an address book only once in the *Connection 0* tab.

2.6 Request-S7 for ibaBM-DPM-S

In the following, the Request-S7 version for the PROFIBUS bus module *ibaBM-DPM-S* is described.

The solution Request-S7 for *ibaBM-DPM-S* is replaced by Request-S7 for *ibaBM-DP* with full functional compatibility.

2.6.1 General information

For DP Request, the measurement values (Request Handshake) are not requested via the PROFIBUS, but via a separate connection.

Depending on the hardware and software, different access points can be selected for the Request:

- TCP/IP: The connection to the SIMATIC S7 is established using an integrated PN interface of the S7-CPU or the respective CP modules in the PLC and the standard network interface of the computer. No additional Siemens software is required for the connection.
- PC/CP: This is the designation for different SIMATIC specific access points. In contrast to the TCP/IP connection, for the PC/CP group the SIMATIC communication software (and the corresponding licenses) must be installed on the computer with the corresponding functions enabled.
  - MPI, PROFIBUS: The connection to the SIMATIC S7 is established using the MPI or PROFIBUS interface of the computer; e.g. with the CP5611 PCI card or the MPI adapter for USB interface or serial PC interface.
  - TCP/IP, ISO: Here, either the standard network interface of the computer or a suitable interface card is used for the connection to the S7.

System integration with ibaBM-DPM-S

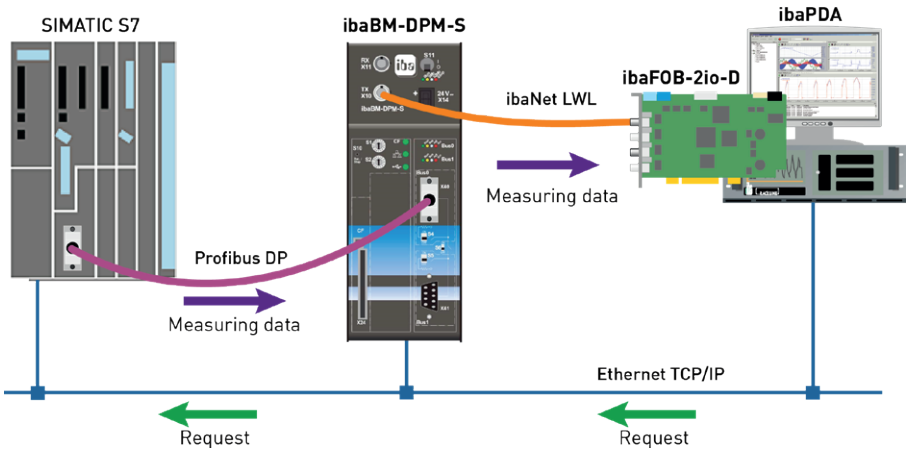
The measured data is transmitted to the *ibaBM-DPM-S* device over PROFIBUS DP.

You need the following connections:

- Online connection between *ibaPDA* and S7-CPU (TCP/IP, MPI or DP)
- Online connection between *ibaPDA* and *ibaBM-DPM-S* (TCP/IP over Ethernet or USB)
- Fiber optic connection between *ibaPDA*/*ibaFOB-io-D* and *ibaBM-DPM-S*
- PROFIBUS connection between *ibaBM-DPM-S* and S7 PROFIBUS master

In the standard version, you can configure a maximum of 8 connections per device, i.e. 8 PROFIBUS slaves. Up to 244 Byte data can be transmitted per slave.

The following illustration shows the configuration using a TCP/IP online connection between *ibaPDA* and S7-CPU.



Note

The following description refers to the Request blocks V4.0 or higher. In case you require information regarding older versions, contact the iba Support.

Other documentation

Detailed information about *ibaBM-DPM-S* can be found in the device manual.

For information about application examples, see chapter ↗ *Application examples, page 131*.

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

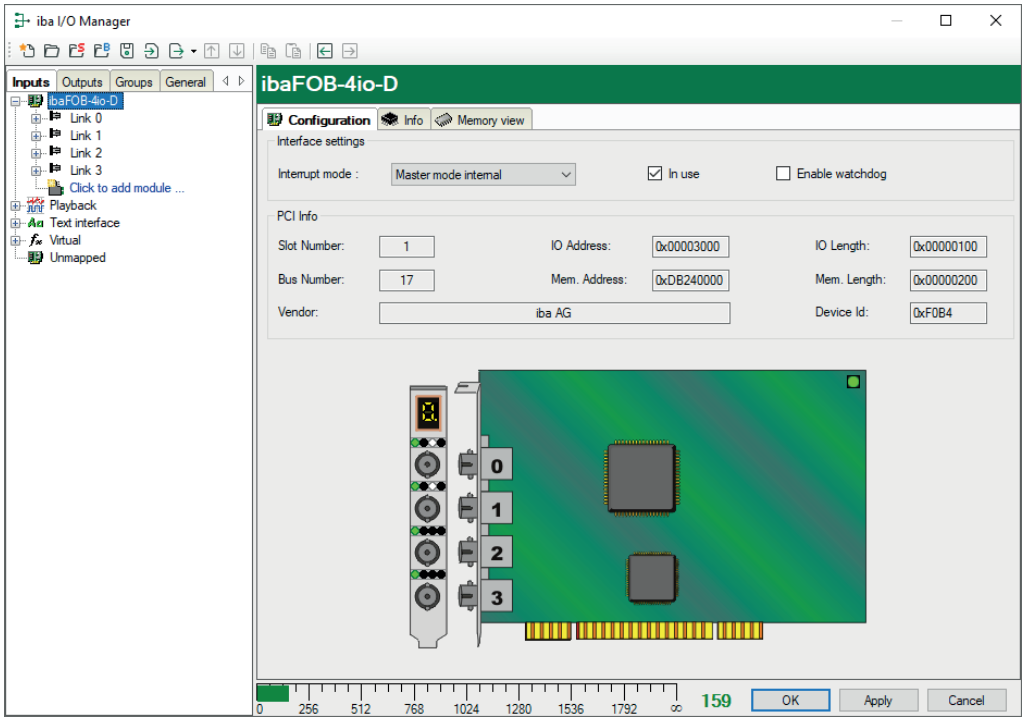
The configuration on the SIMATIC S7 side corresponds to the configuration of *ibaBM-DP* (see chapter ↗ *Configuration and engineering SIMATIC S7-300, S7-400 and WinAC, page 43*).

2.6.3 Configuration and engineering ibaPDA

This section describes the configuration of the Request modules in *ibaPDA* for *ibaBM-DPM-S*.

2.6.3.1 General interface settings

If an *ibaFOB-D* card is installed in the *ibaPDA* computer, *ibaPDA* displays the interface of the *ibaFOB-D* card in the interface tree of the I/O Manager.



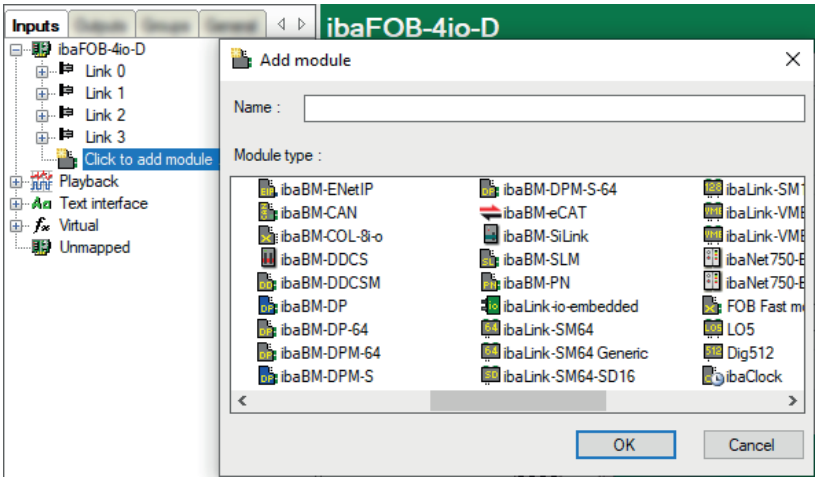
Other documentation

Detailed information about the *ibaFOB-D* card can be found in the respective manual.

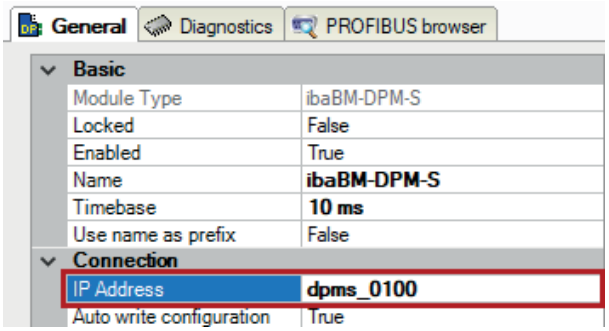
Configuring ibaBM-DPM-S

1. Start the *ibaPDA* client and open the I/O Manager.
2. In the interface tree (left), select the link of the *ibaFOB* card to which *ibaBM-DPM-S* is connected.

   Click on the blue command *Click to add module*. Select the *ibaBM-DPM-S* module in the dialog box and assign a name via the input field if required. Confirm the selection with <OK>.



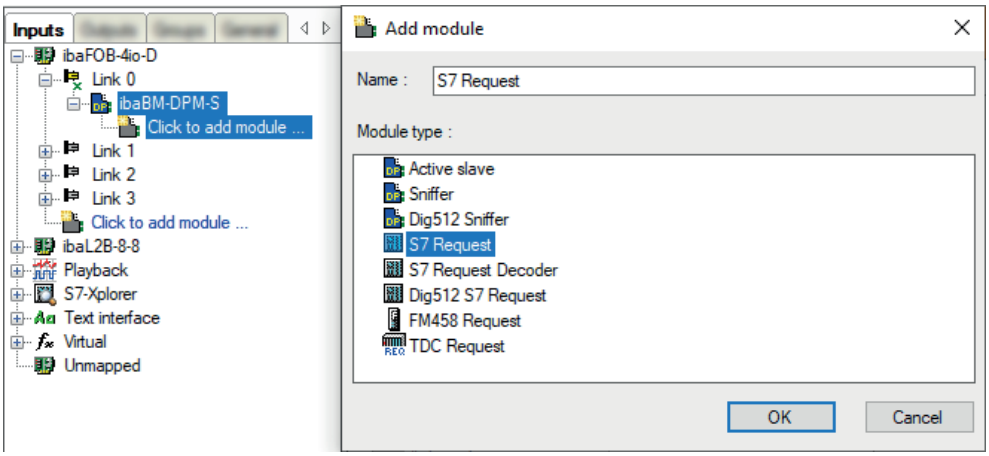
3. Enter the IP address of the device on the *General* tab under *Connection*: either as name e.g. "dpms_000100" or as IP address e.g. "192.168.81.123".



Note

An online connection to the device via Ethernet or USB is mandatory. The manual of the *ibaBM-DPMS* device explains how to establish and check this connection.

4. Add a request module (or several modules, in case you need more connections to one or to different S7-CPUs) to the *ibaBM-DPM-S* module. Available modules are:
   - S7 request (for acquiring analog signals and digital signals)
   - S7 request decoder (for the acquisition of up to 1024 digital signals)
   - S7 request Dig512 (for acquiring up to 512 digital signals)



5. Configure the required module settings and signals as described in the following chapters. The *General* and *Connection* tabs are identical for all Request modules. The Request modules only differ in the *Analog* and *Digital* tabs.
6. After you have finished the configuration, click <Apply> or <OK> to transfer the new configuration to the device and start data acquisition with *ibaPDA*.

2.6.3.2 General module settings

You find the description of all settings that are identical for all Request-S7 modules in part 1 of the manual.

The *ibaBM-DPM-S* modules all have the following common specific settings options.

PROFIBUS

Bus number
0 = connector X40 top, 1 = connector X41 bottom

Slave No.
PROFIBUS slave address assigned to the module.

Connection

Auto enable/disable
If TRUE, the acquisition is started, even if no connection can be established to the S7-CPU. The module is disabled. During the acquisition, *ibaPDA* tries to reconnect to the S7-CPU. When it succeeds, the acquisition is restarted with this module enabled.
In case of FALSE, the acquisition is not started, if a connection to the configured S7-CPU cannot be established.

2.6.3.3 Connection settings

The connection settings are identical to those in the general chapter *Connection settings* in part 1.

PROFIBUS slave number
PROFIBUS slave address assigned to the module. If you change the entry in the *Connection* tab, the entry in the *General* tab changes accordingly and vice versa.

2.6.3.4 Module S7 Request

With this module type, up to 64 analog signals and 64 digital signals can be acquired.

Configure a separate PROFIBUS slave and Request block call for each module.

For a description of the module settings, see chapter ↗ *General module settings, page 66* and in the general chapter *General module settings* in part 1 of this manual.

2.6.3.5 Module S7 Request Decoder

The *S7 Request Decoder* module allows the acquisition of up to 1024 digital signals sent as a maximum of 64 words (16 Bit). This module type is suited for applications where large amounts of digital signals have to be acquired and for which the max. 512 directly addressable digital values of the *ibaBM-DPM-S* are not sufficient.

For a detailed description of the *S7 Request Decoder* module, see the module description of the *ibaBM-DP* device in part 1 of the manual.

2.6.3.6 Module Dig512 S7 Request

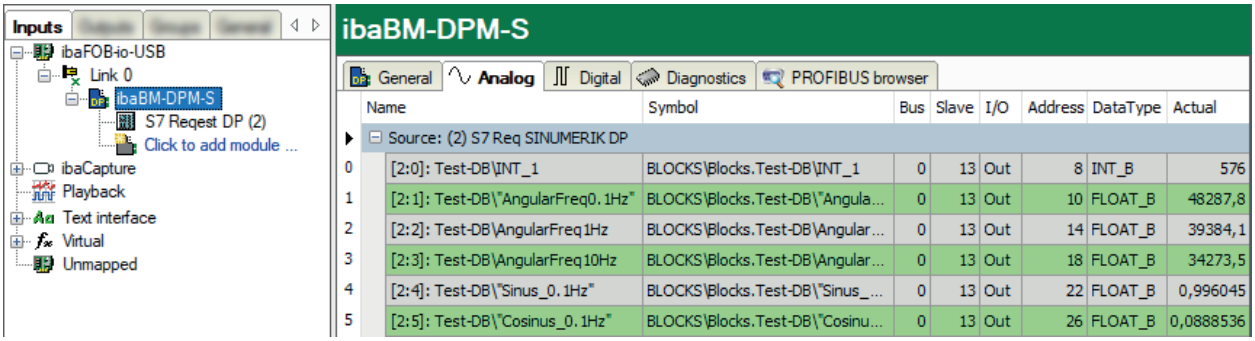
For the *Dig512 S7 request* module the same applies as for the module *S7 Request Decoder*. But you can only acquire 32 words (512 digital values).

The *Dig512 S7 Request* module is the predecessor of the *S7 Request Decoder* module and is still supported in *ibaPDA* for reasons of downward compatibility.

For new configurations, use the *S7 Request Decoder* module, see ↗ *Module S7 Request Decoder, page 67*.

2.6.4 Diagnostics

When you select the bus module node in the signal tree and choose the *Analog* or *Digital* tab, you get a list of all operands that are acquired in the bus module with data type and actual value.



Other documentation

You find a detailed description of the device-specific diagnostic options of *ibaBM-DPM-S* in the device manual.

2.7 Request-S7 for ibaBM-DPM-S in redundancy mode

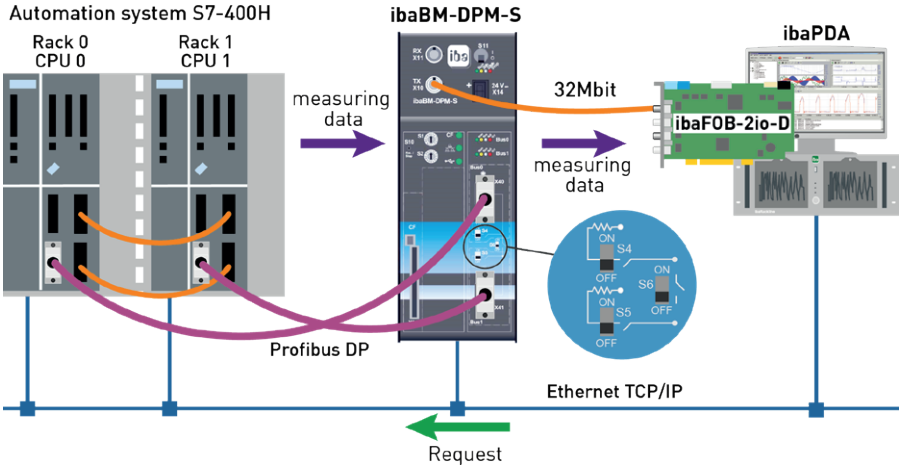
In the following, the Request-S7 version for the PROFIBUS bus module *ibaBM-DPM-S* in redundancy mode is described. The solution "Request-S7 for *ibaBM-DPM-S* in redundancy mode" is replaced by "Request-S7 for *ibaBM-DP* in redundancy mode" with full functional compatibility.

2.7.1 General information

With the redundancy mode of *ibaBM-DPM-S*, the device can be operated on redundant PROFIBUS systems in combination with SIMATIC S7-400H controllers, whose measurement data are to be acquired. You need an additional license for the redundancy mode. Contact the iba Support. The license is released using the administrator functions in the Web dialog of the *ibaBM-DPM-S*.

Request-S7 redundant is suitable for operating *ibaBM-DPM-S* with the Request functionality as single-channel periphery on a highly available SIMATIC S7-400H controller.

The following figure shows an exemplary integration of an *ibaBM-DPM-S* in redundancy mode.



Note

The following description refers to the Request blocks V4.0 or higher.

Other documentation

Detailed information about the redundancy mode of *ibaBM-DPM-S* can be found in the device manual.

The functionality of the Request-S7 for *ibaBM-DPM-S* in redundancy mode mostly corresponds to the functionality in standard mode, see ↗ *Request-S7 for ibaBM-DPM-S, page 62*. The differences and extensions are described in the following.

For information about application examples, see chapter ↗ *Application examples, page 131*.

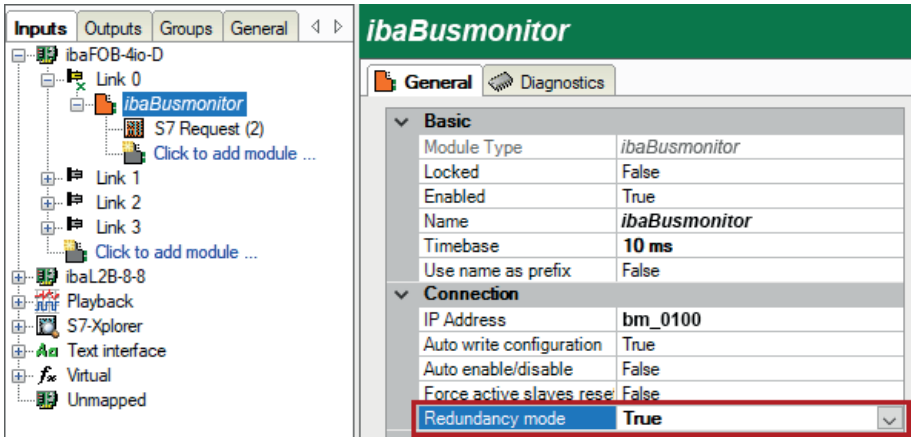
2.7.2 Configuration and engineering SIMATIC S7-300, S7-400 and WinAC

The configuration on the SIMATIC S7 side corresponds to the configuration of *ibaBM-DP* (see chapter ↗ *Configuration and engineering SIMATIC S7-300, S7-400 and WinAC, page 56*).

2.7.3 Configuration and engineering ibaPDA

First connect *ibaBM-DPM-S* to a free link of an *ibaFOB-D* card. In the I/O Manager, add a device module *ibaBM-DPM-S* to the corresponding link.

In the *General* tab, set the parameter *Redundancy mode* to "True". The orange colored icon of the Request module indicates the redundancy mode.



As on the redundant PROFIBUS, both bus systems 0 and 1 are operated in parallel, there are only settings for *Redundant PROFIBUS* on the *General* tab.

In redundancy mode, the following modules are available with Request-S7:

- S7 Request
- S7 Request Decoder
- Dig512 S7 request

The configuration of the modules corresponds to the configuration in standard mode, see ↗ *General module settings, page 66*.

Connection settings

For transmitting the operand data to both CPUs of the SIMATIC S7-400H, set up two different connections. *ibaPDA* switches between these connections depending on availability and requirements.

The connection settings are identical to the Request-S7 for *ibaBM-DP* in redundancy mode (see chapter ↗ *Configuration and engineering SIMATIC S7-300, S7-400 and WinAC, page 56*).

2.8 Request-S7 for ibaCom-L2B

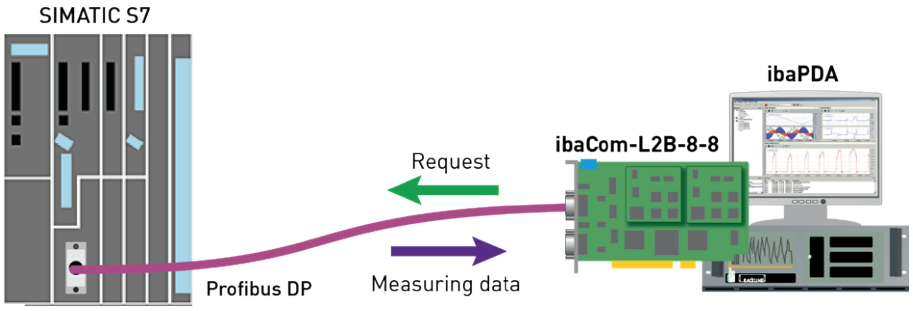
In the following, the Request-S7 version for the *ibaCom-L2B* PROFIBUS card is described.

The solution Request-S7 for *ibaCom-L2B* is replaced by Request-S7 for *ibaBM-DP* with full functional compatibility.

For more information, see chapter ↗ *Replacing Request-S7 on ibaCom-L2B by ibaBM-DP, page 147*.

2.8.1 General information

With the L2B Request, the measurement request (Request Handshake) is sent via PROFIBUS. A section of the IO range of the respective PROFIBUS slave is used for this purpose. No TCP/IP connection between the S7 controller and the *ibaPDA* computer is required for configuring.



Note

The following description refers to the Request blocks V3.4 or higher. In case you require information regarding older versions, contact the iba Support.

Older versions are required if *ibaPDA-Request-S7* is to be used in combination with older S7-CPUs, e.g. CPU 315 before release version 2AF03 or CPU 314 before release version 2AF04.

For information about application examples, see chapter ↗ *Application examples, page 131*.

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

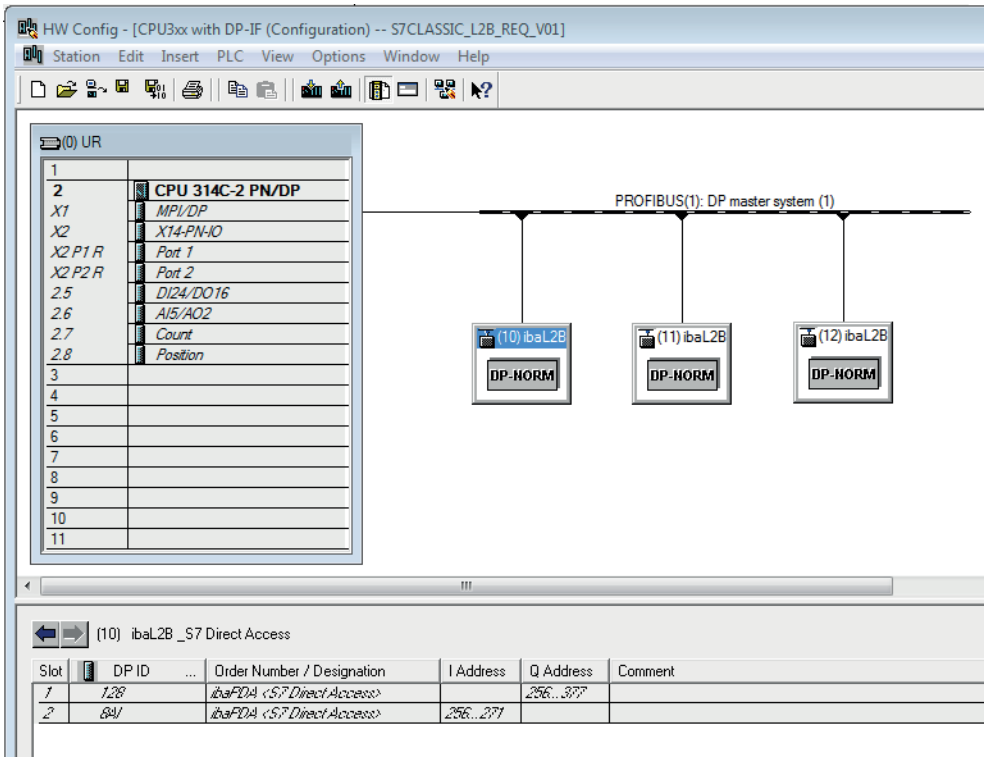
This section describes the configuration and engineering on the SIMATIC S7 side with SIMATIC Manager (STEP 7 Version ≤ V5) for *ibaCom-L2B*.

2.8.2.1 Hardware configuration

Perform the following steps for *ibaCom-L2B*:

1. Link the DP interface of the CPU or CP as DP master to a PROFIBUS network.
2. Install the GSD file `IBA_0F05.GSD` via the menu *Options – Installing GSD-files*.
3. After the installation finished, the iba-GSD-file is in the HW catalog under:
   Profibus-DP – Additional Field Devices – General – ibaL2B <S7 Direct Access>
4. Attach the desired number of DP slaves "ibaL2B <S7 Direct Access>" to the PROFIBUS and set the DP addresses. The peripheral addresses are assigned automatically to the slaves as addresses in ascending order.

5. Load the HW config/system data to the S7-CPU.



2.8.2.2 Configuration in STEP 7 (STL, LAD, FBD)

This section describes how to configure the Request blocks in STEP 7 V5 for *ibaCom-L2B*.

2.8.2.2.1 CPU-internal DP interface or CP 443-5 (for S7-400)

If you use a CPU S7-300/400 with integrated DP interface or use the external interface CP 443-5 for a S7-400, proceed as follows for *ibaCom-L2B*.

If you use the external DP interface CP342-5 of a CPU S7-300, continue in chapter ↗ *External DP interface CP342-5, page 74*.

For Request-S7 with one PROFIBUS slave

1. Copy the following blocks from the iba S7 library to the blocks folder of your STEP 7 project, see ↗ *iba S7 library, page 128*. The functions have to be called once for each activated PROFIBUS slave in the program.
   - FC111 (ibaL2B_Init) and FC112 (ibaL2B_Req), see ↗ *Initialization-FC ibaL2B_Init (FC111), page 110* and ↗ *Communication-FC ibaL2B_Req (FC112), page 112*
   - DB22 (ibaL2B_DB_work) and UDT22 (ibaL2B_DB_Struct)

Note

Only use Request blocks from the latest iba S7 library!

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

Note

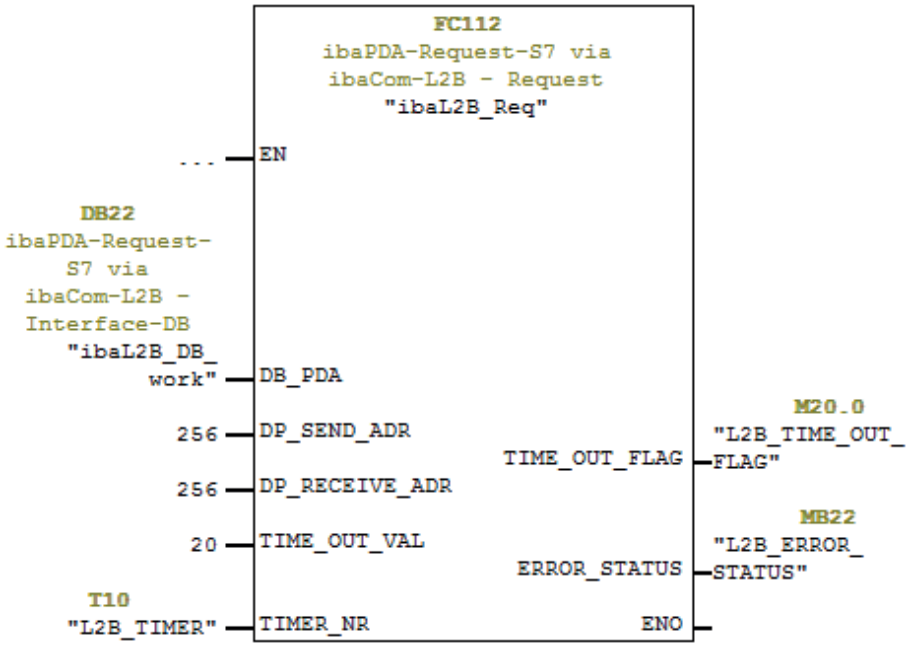
In case the block numbers FC111, FC112, DB22 and UDT22 are already used in your project, assign other free numbers to the blocks from the iba S7 library when copying.

2. Create the error OBs (OB82, OB85, OB86, OB87, OB122) in order to prevent CPU stops in case of an error.

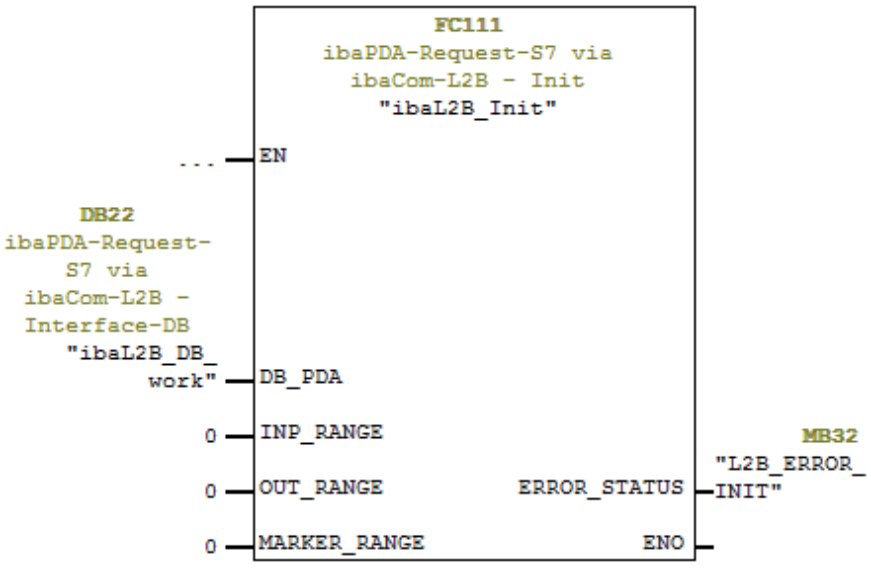
Note

DB22 and UDT22 contain a data structure which is not required for processing. A DB with a length of 1500 bytes is sufficient. The data structure is useful for diagnostic purposes only.

3. Call and parameterize the ibaL2B_Req (FC112) in the cyclic program.



4. Call and parameterize the ibaL2B_Init (FC111) in the start-up OBs (OB100, OB101, OB102).



For Request-S7 with multiple PROFIBUS slaves

- In the blocks folder, a data block (DB) that is referenced by both FCs has to be available for each Request slave. Copy the ibaL2B_DB_work (DB22) to a DB with a new DB number.
- Call the ibaL2B_Init (FC111) once more with a new DB number in the start-up OBs.
- In the cyclic OB, you have to call the ibaL2B_Req (FC112) once more with the new DB number and the peripheral addresses of the new PROFIBUS slave.

Final

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

2.8.2.2.2 External DP interface CP342-5

When using an external DP interface CP342-5, perform the following steps for *ibaCom-L2B*.

For Request-S7 with one PROFIBUS slave

1. Copy the following blocks from the iba S7 library to the blocks folder of your STEP 7 project, see ↗ *iba S7 library, page 128*. The functions have to be called once for each activated PROFIBUS slave in the program.
   - FC113 (ibaL2B_Req_CP) and FC111 (ibaL2B_Init), see ↗ *Communication-FC ibaL2B_Req_CP (FC113) for CP342-5, page 114* and ↗ *Initialization-FC ibaL2B_Init (FC111), page 110*
   - DB10 (ibaL2B_CP_SNDRCV), DB22 (ibaL2B_DB_work) and UDT22 (ibaL2B_DB_Struct)

   Note

   Only use Request blocks from the latest iba S7 library!

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

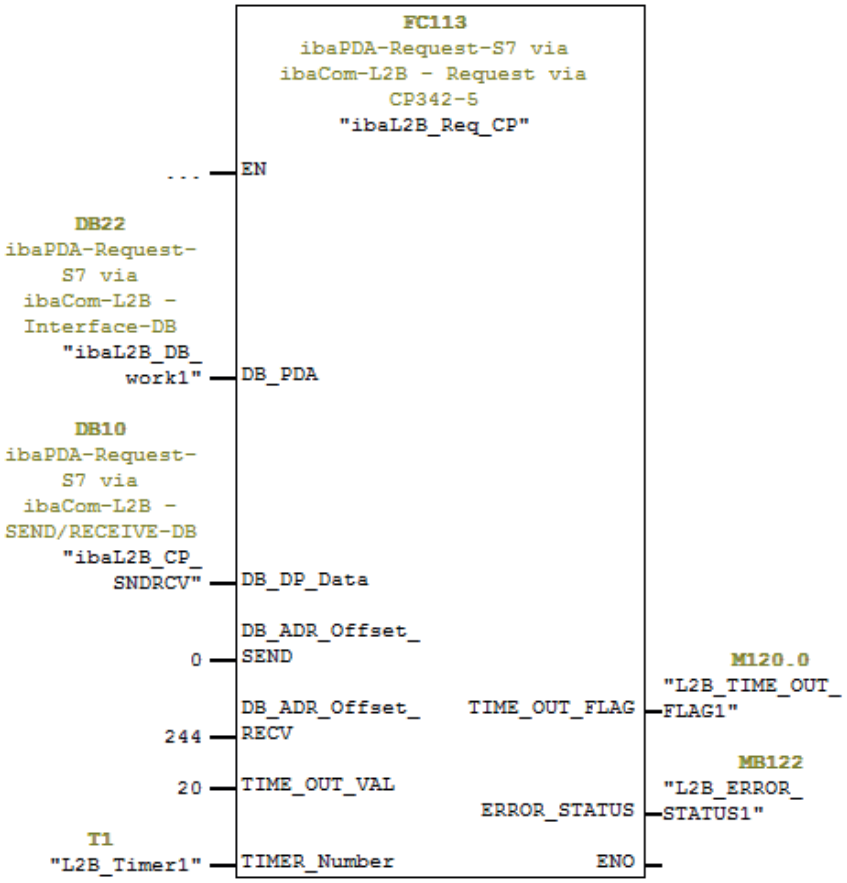
   Note

   In case the block numbers FC111, FC113, DB10, DB22 and UDT22 are already used in your project, assign other free numbers to the blocks from the iba S7 library when copying.

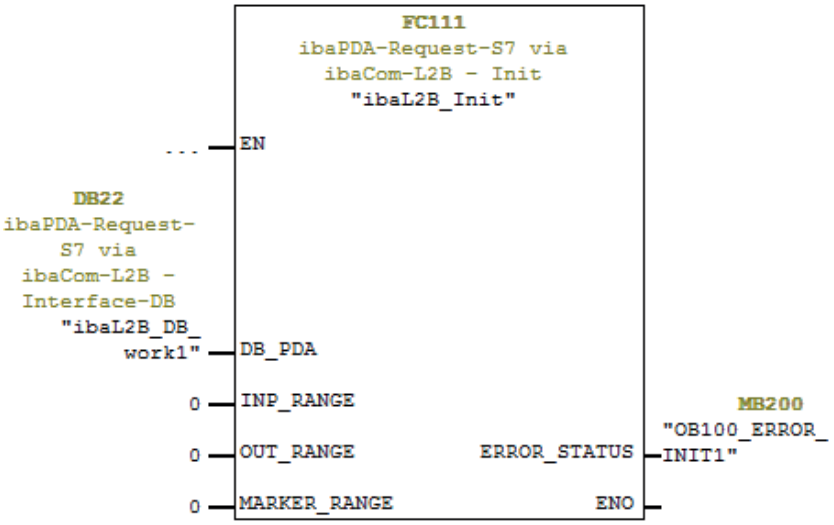
2. Create the error OBs (OB82, OB85, OB86, OB87, OB122) in order to prevent CPU stops in case of an error.

3. Call and parameterize the ibaL2B_Req_CP (FC113) in the cyclic program.

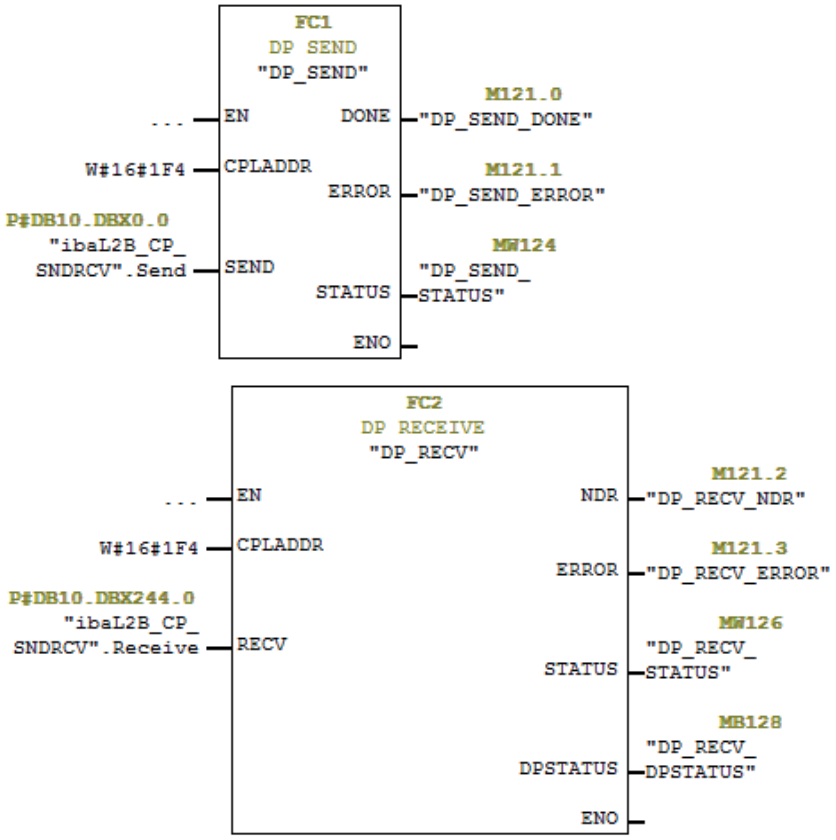
   As offset addresses for send and receive data, own ranges within the DB "ibaPDA_SENDRECV" (DB10) have to be defined for each slave and handed over as parameters "DB_ADR_Offset_SEND" or "DB_ADR_Offset_RECV".



4. Call and parameterize the ibaL2B_Init (FC111) in the start-up OBs (OB100, OB101, OB102).



5. Call and parameterize FC1 (DP_SEND) and FC2 (DP_RECV) in the cyclic program.



For Request-S7 with multiple PROFIBUS slaves

- In the blocks folder, a data block (DB) that is referenced by both FCs has to be available for each Request slave. Copy the ibaL2B_DB_work (DB22) to a DB with a new DB number.
- In the ibaL2B_CP_SNDRCV (DB10), 122 Bytes have to be reserved in the "Send" array and 16 Bytes in the "Receive" array for each Request slave. If necessary, you have to extend the arrays.
- Call the ibaL2B_Init (FC111) once more with a new DB number in the start-up OBs.
- In the cyclic OBs, you have to call the ibaL2B_Req_CP (FC113) once more with a new DB number and the respective address offsets for the send and receive range in the ibaL2B_CP_SNDRCV (DB10) of the new PROFIBUS slave.

Final

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

Note

The structure of the send and receive data in DB10 must correspond to the output addresses and input addresses of the DP slaves.

If there are more than two slaves, you have to extend the arrays in DB10 by 122 Bytes for sending and 16 Bytes for receiving direction, respectively.

In case of four slaves, for example, the following address assignment applies:

	DB10		PROFIBUS P-addresses	ibaL2B_Req_CP. …Offset_SEND	ibaL2B_Req_CP. …Offset_RECV
1: Slave	Send	Offset 0	Q-address 0	0	-
2. Slave	-	Offset 122	Q-address 122	122	-
3. Slave	-	Offset 244	Q-address 244	244	-
4. Slave	-	Offset 366	Q-address 366	366	-
1. Slave	Receive	Offset 0	I-address 0	-	488
2. Slave	-	Offset 16	I-address 16	-	504
3. Slave	-	Offset 32	I-address 32	-	520
4. Slave	-	Offset 48	I-address 48	-	536

Note

All parameters like times and markers have to be different!

2.8.2.3 Configuration in STEP 7 (CFC)

If you use a CPU S7-300/400 with integrated DP interface or use the external interface CP 443-5 for a S7-400, proceed as follows for *ibaCom-L2B*.

If you use the external DP interface CP342-5 of a CPU S7-300, continue in chapter ↗ *External DP interface CP342-5, page 74*.

For Request-S7 with one PROFIBUS slave

1. Copy the following blocks from the iba S7 library to the blocks folder of your STEP 7 project, see ↗ *iba S7 library, page 128*. The functions have to be called once for each activated PROFIBUS slave in the program.
   - FC111 (ibaL2B_Init) and FC112 (ibaL2B_Req), see ↗ *Initialization-FC ibaL2B_Init (FC111), page 110* and ↗ *Communication-FC ibaL2B_Req (FC112), page 112*
   - DB22 (ibaL2B_DB_work) and UDT22 (ibaL2B_DB_Struct)

Note

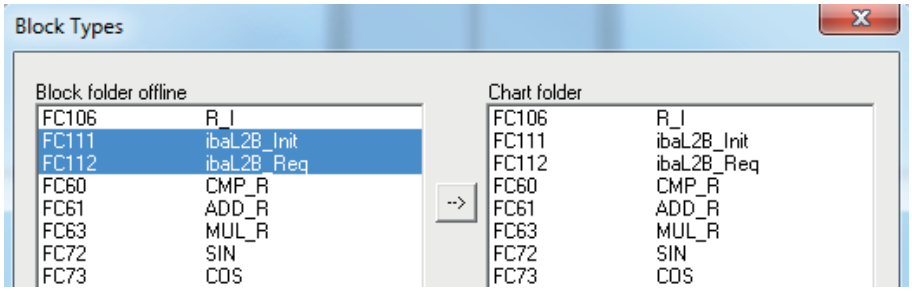
Only use Request blocks from the latest iba S7 library!

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

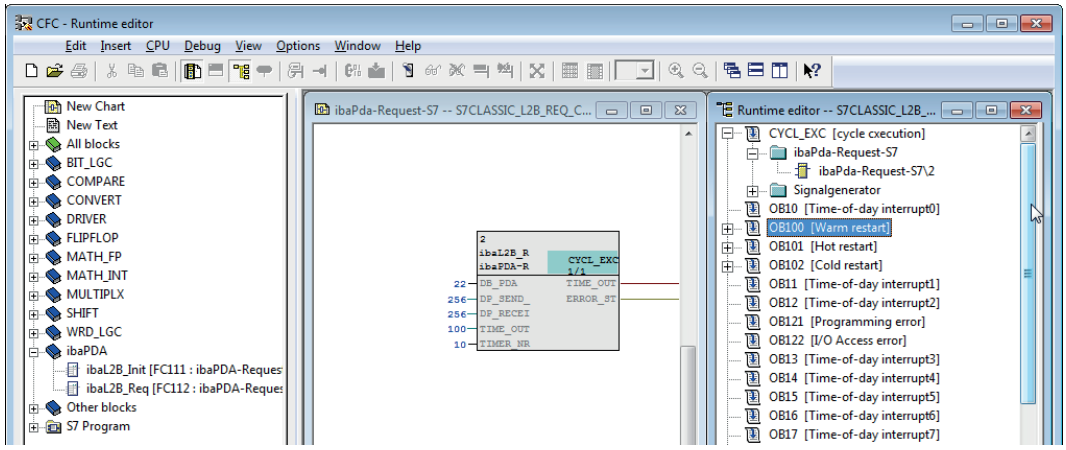
Note

If the block numbers FC111, FC112, DB22 and UDT22 are already used in your project, assign other free numbers to the blocks from the iba S7 library when copying.

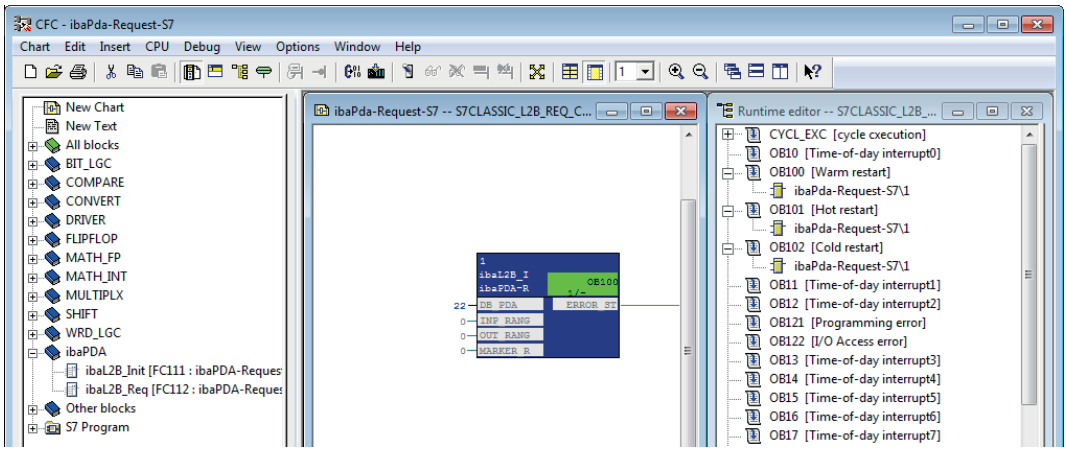
2. Start the CFC editor and import these blocks (under *Options – Block Types*).
   The FCs are stored in the group "ibaPDA" in the function block library.



3. Drag the ibaL2B_Req (FC112) to your CFC chart. Pay attention to the run sequence. The ibaL2B_Req has to be called in a cyclic task (e. g. cyclic interrupt OB35 or free-running OB1).



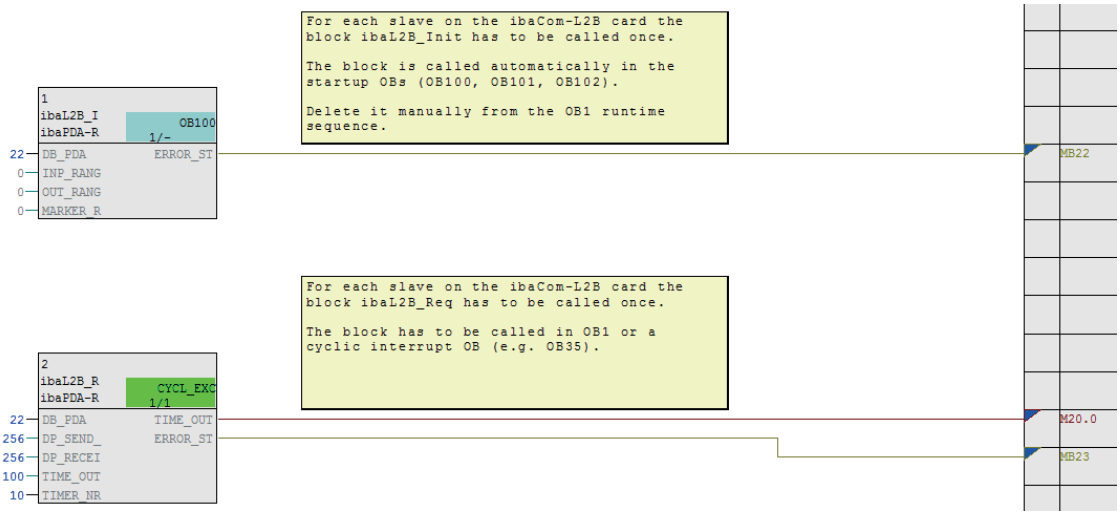
4. Drag the ibaL2B_Init (FC111) to your CFC chart. Pay attention to the run sequence. The ibaL2B_Init is entered automatically in the start-up tasks (OB100, OB101 and OB102). Remove it from the cyclic tasks (e.g. OB35).



5. Connect the blocks:

   a) Enter the DB number of the Interface-DB (DB22) on both FCs on the input DB_NUMBER.

   b) Enter the peripheral addresses of the PROFIBUS slave under DP_SEND_ADR and DP_RECEIVE_ADR of the ibaL2B_Req block from the hardware configuration.



Make sure to create the error OBs (OB82, OB85, OB86, OB87 and OB122) in order to prevent the CPU from running into STOP in case of an access failure.
To do so, you have to define an empty runtime group within the run sequence of these tasks if they are empty. Alternatively, you may place any dummy block into the error OB. This block must not be deleted, too.

Note

Do **not** choose the option *Delete empty runtime groups* for compilation otherwise the error OBs are removed! Some older CFC versions have this option in the compilation dialog.

For Request-S7 with multiple PROFIBUS slaves

- For each Request slave, there must be one data block (DB) in the blocks folder which is referenced by both FCs. Copy the ibaL2B_DB_work (DB22) to a DB with a new DB number.
- The ibaL2B_Init (FC111) has to be called once more on a CFC chart with a new DB number.
- The ibaL2B_Req (FC112) has to be called once more on a CFC chart with the new DB number and the peripheral addresses of the new slave.

After each program modification

1. Compile the complete program:

   a) Activate the option *Entire program*.

   b) Activate the option *Generate SCL source* (for SIMATIC CFC version 6.1 or higher).

   c) Deactivate the option *Delete empty runtime groups* if available.

2. Load the program and restart the S7-CPU.

Note

Depending on the S7-CPU type, you may notice the following warnings after compiling and loading:

- "W: OB101/ OB102 is not supported by this CPU"
- "W: OB101/OB102 could not be downloaded, because it is not supported by the connected online CPU".

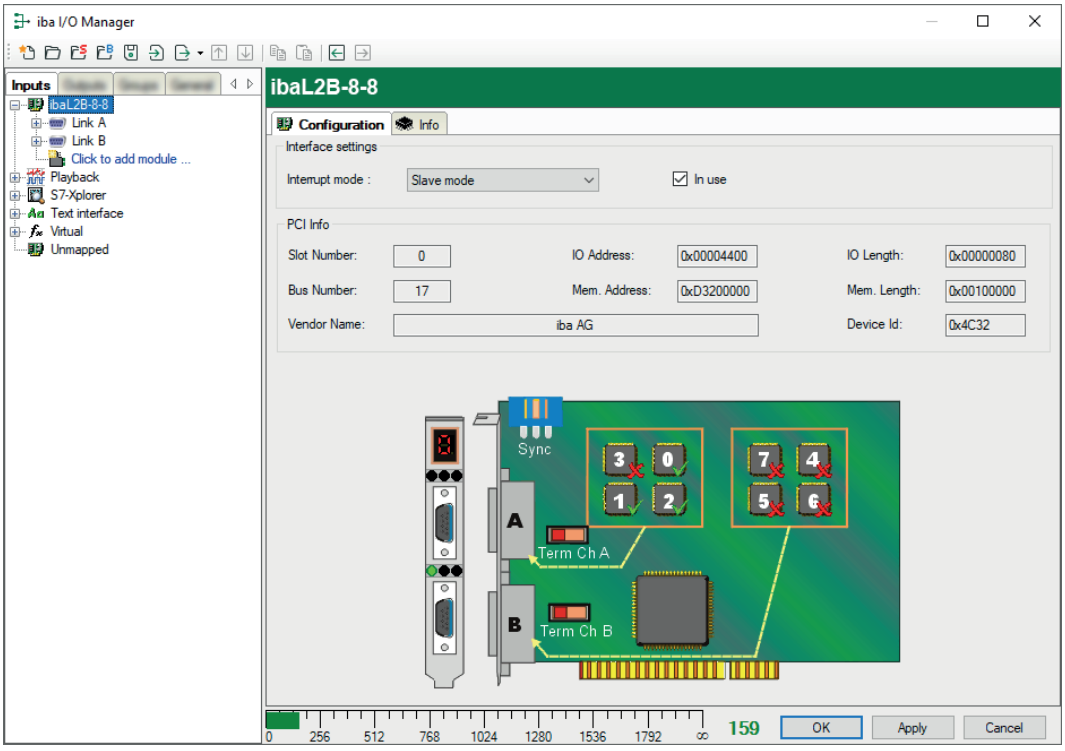
These messages are displayed because the ibaL2B_Init is entered automatically in the start-up tasks OB101 and OB102. However, some S7-CPU types do not support OB101 and OB102. You can ignore the warnings.

2.8.3 Configuration and engineering ibaPDA

This section describes the configuration of the Request modules in *ibaPDA* for *ibaCom-L2B*.

2.8.3.1 General interface settings

If an *ibaCom-L2B* card is installed in the *ibaPDA* computer, *ibaPDA* displays the interface of the *ibaCom-L2B* card in the interface tree of the I/O Manager.

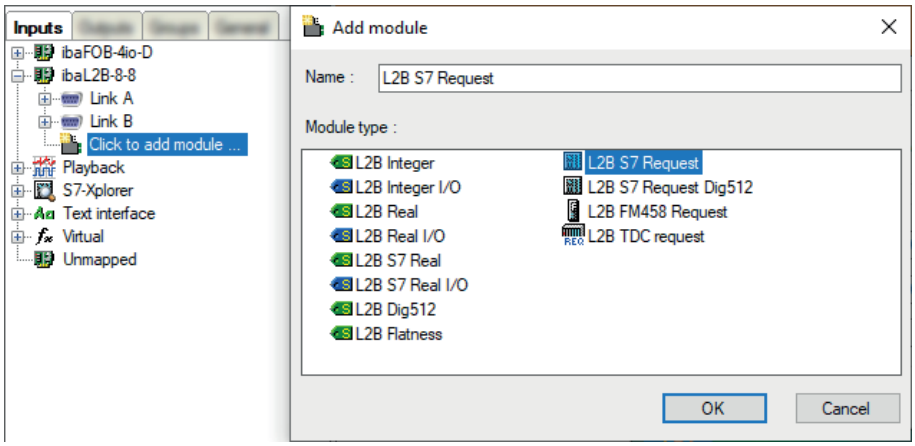


Other documentation

Detailed information about the *ibaCom-L2B* card can be found in the respective manual.

2.8.3.2 General module settings

The module types *L2B S7 Request* and *L2B S7 Request Dig512* are available.



You can find the description of all settings that are identical for all Request-S7 modules in part 1 of the manual.

The L2B modules have the following common specific setting options:

PROFIBUS

Slave number
PROFIBUS slave address assigned to the module.

Timeout
Waiting time in seconds before the firmware regards the connection as interrupted.

Connection settings

In contrast to the Request-S7 solutions with the *ibaBM-DP* and *ibaBM-DPM-S* devices, no additional connection settings are required. The Request Handshake is done via a part of the I/O peripheral area.

2.8.3.3 Module L2B S7 Request

You can acquire up to 32 analog signals and 32 digital signals with the *L2B S7 Request* module.

Configure a separate PROFIBUS slave and Request block call for each module.

For a description of the module settings, see chapter ↗ *General module settings, page 81* and part 1 of this manual.

2.8.3.4 Module L2B S7 Request Dig512

With the *L2B S7 Request Dig512* module, you can acquire up to 512 digital signals that are sent as max. of 32 words (16 Bit). This is why this module type is especially suited for applications where many digital signals have to be acquired and the digital values of the modules *L2B S7 Request* which can be addressed directly are not sufficient.

Configure a separate PROFIBUS slave and Request block call for each module.

General tab

For further information on module settings, see ↗ *General module settings, page 81*.

Digital tab

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

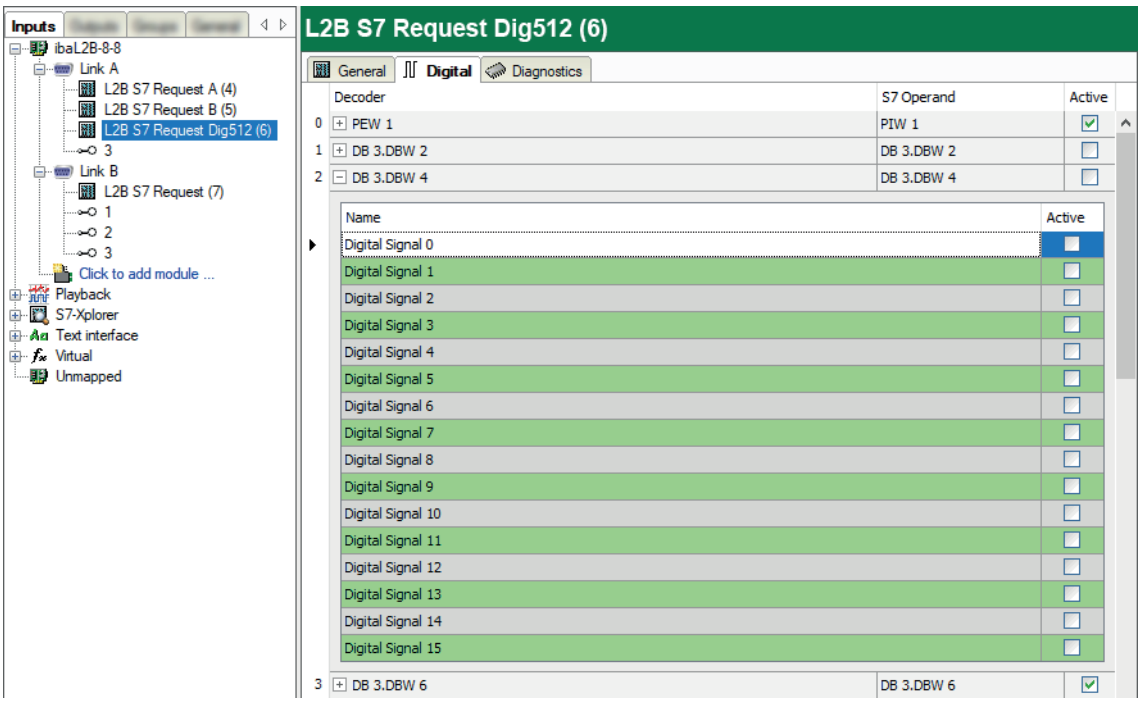
- First define the words (source signals), which are broken down for the digital signals (bits).

  You can directly enter the words as basic signals for decoding via the absolute S7 operands. Only word operands (e.g. PIW, MW, DBW) are allowed.

  It is also possible to use S7 symbols by generating address books. For detailed information, see part 1 of this manual, section *Selection via the symbolic operand addresses*. The signals selected in S7 CFC and symbol browser are applied and the columns *Name*, *S7 symbol* and *S7 Operand* are filled in automatically.

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

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



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

Source signal

Decoder
Enter a name for the source signal.

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

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

Individual digital signals (bits)

Name
Enter a name for the individual digital signals.

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

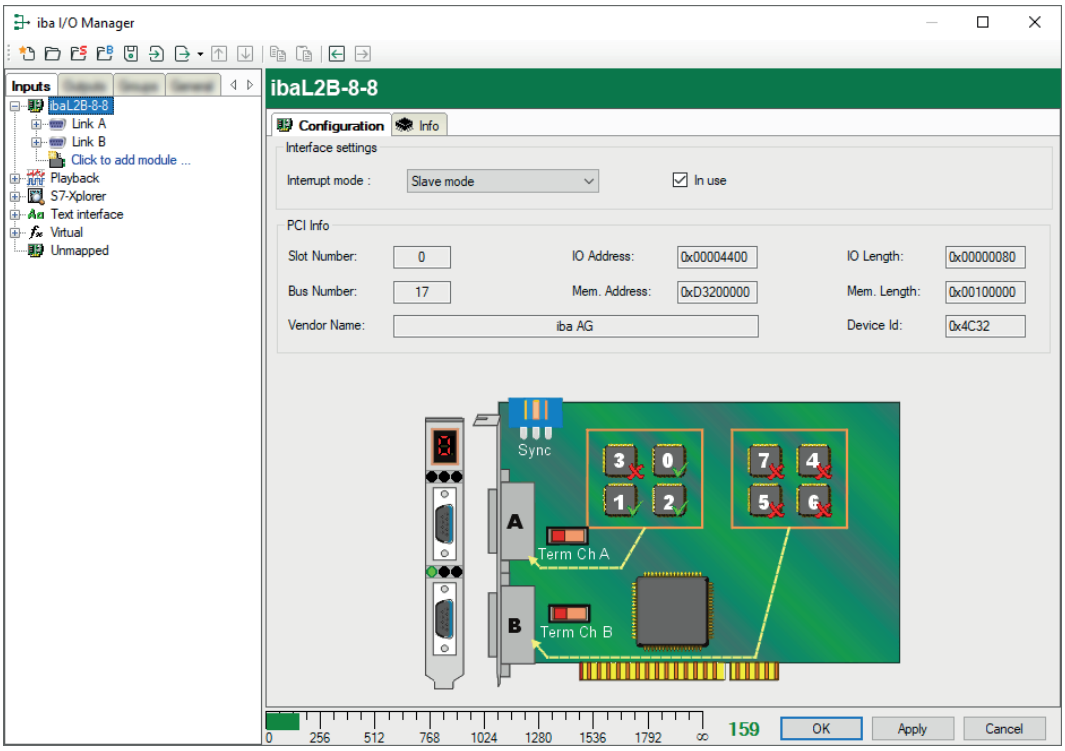
Note

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

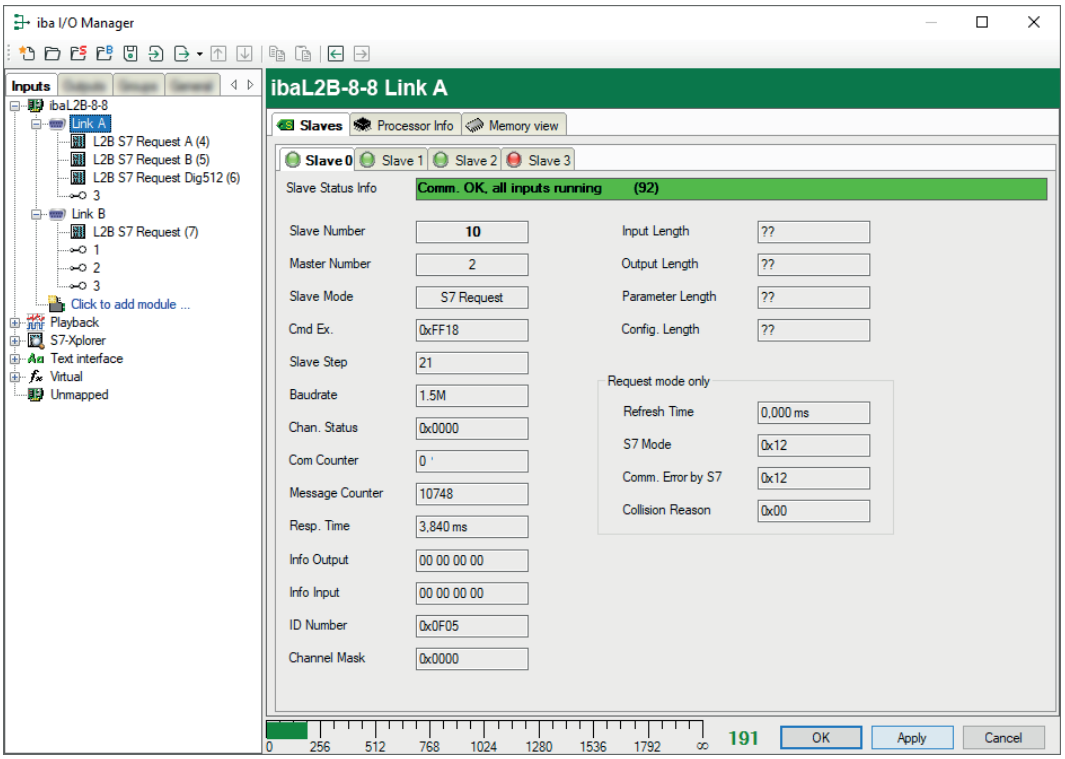
2.8.4 Diagnostics

2.8.4.1 ibaCom-L2B card

When you select an L2B card node in the signal tree, *ibaPDA* displays a status overview of the card. The green checkmark on the slave number indicates an active connection. In the figure slaves 0 to 2 are active.

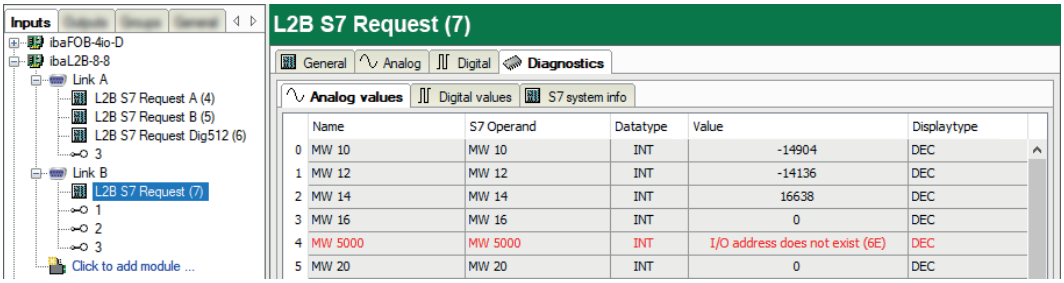


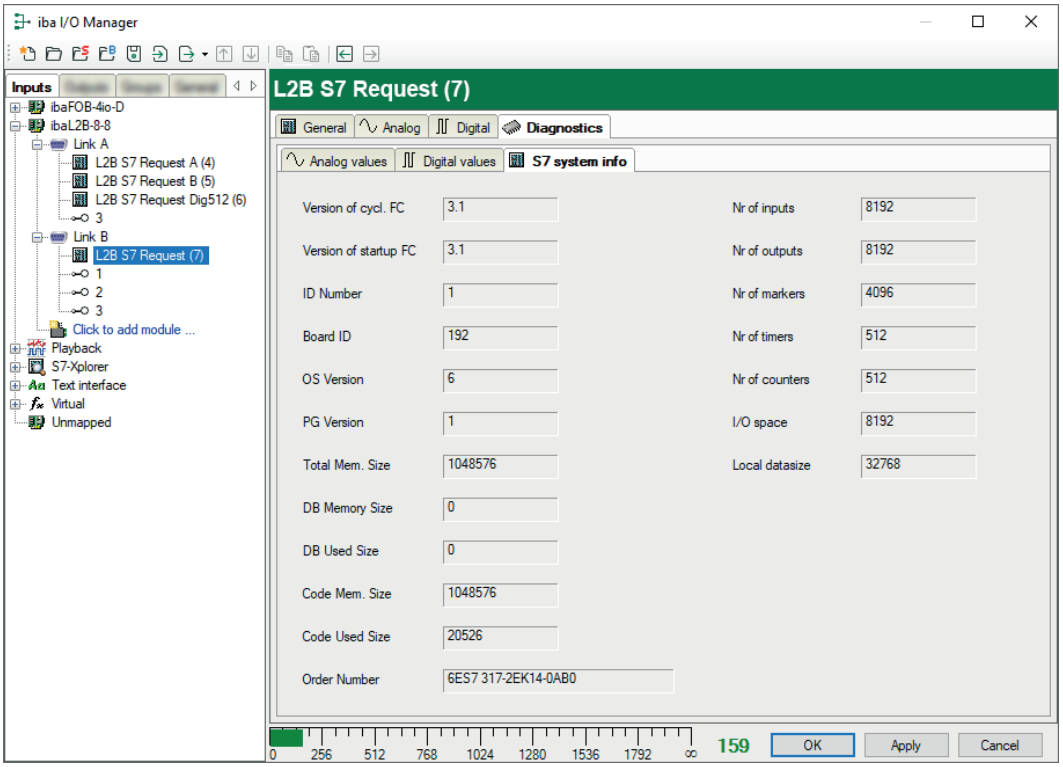
When you select the PROFIBUS link, you can see detailed information regarding the status of each slave. This information is mainly relevant for support purposes.



2.8.4.2 Module diagnostics

When you select an L2B-Request module node in the signal tree, in the *Diagnostics* tab you can see details of the individual measured values and system information of the S7-CPU in the sub-tab *S7 system info*.





3 Description of Request blocks

3.1 ibaREQ iba block family

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

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

One set of Request blocks has to be called for each Request module (connection) in *ibaPDA*. The blocks used are part of the iba S7 library, see ↗ *iba S7 library, page 128*.

Note

Only use Request blocks from the latest iba S7 library!

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

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

Request block	S7-300 with integrated DP interface	S7-300 with integrated PN interface	S7-300 CPU + CP343-1	S7-400(H) with integrated DP interface and CP443-5	S7-400 with integrated PN interface and CP443-1 (PROFINET)	Recommended call level
ibaREQ_M (FB140)	X	X	X	X	X	OB1
ibaREQ_PN (FB141)	X	X	-	X	X	OB1 or OB3x
ibaREQ_DP (FB142)	X	X	-	X	X	OB1 or OB3x
ibaREQ_UDP-act (FB145)	X	X	X	X	X	OB3x
ibaREQ_UDPint (FB146)	-	X	-	-	-	OB3x
ibaREQ_UD-Pext3 (FB147)	-	-	X	-	-	OB3x
ibaREQ_UDP-ext4 (FB148)	-	-	-	-	X	OB3x
ibaREQ_PNdev (FB150)	-	X	-	-	X	OB1 or OB3x
ibaREQ_DB (DB15)	X	X	X	X	X	-
ibaUDT_UDP-act (UDT145)	X	X	X	X	X	-
ibaREQ_DB-Interface	X	X	X	X	X	-

- ibaREQ_M (Management)
  The block realizes the communication with *ibaPDA*. Ideally, the block is called in OB1.
- ibaREQ_PN/ibaREQ_DP (provision and sending of current signal values)
  The block provides the current signal values using the transmission cycle. Ideally, the block is called in a cyclic interrupt OB.
- ibaREQ_UDPact (provision of current signal values)
  The block provides the current signal values using the transmission cycle. The block is called internally in the blocks ibaREQ_UDPint, ibaREQ_UDPext3, or ibaREQ_UDPext4. The block must therefore always be present in the project, but does not have to be called separately.
- ibaREQ-PNdev (diagnostics for PROFINET device)
  The block performs cyclic diagnostics for availability of the PROFINET device used.
- ibaREQ_DB (interface DB)
  This DB acts as an interface to *ibaPDA* and between the different Request blocks.

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

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

Always use the blocks ibaREQ_UDPint, ibaREQ_UDPext3 and ibaREQ_UDPext4 alternatively.

3.1.1 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 (default) TRUE: Block reset
INP_RANGE	IN	INT	Number of input bytes (evaluation only during initialization), 0: automatic detection (recommended)
OUT_RANGE	IN	INT	Number of output bytes (evaluation only during initialization), 0: automatic detection (recommended)
MARKER_RANGE	IN	INT	Number of marker bytes (evaluation only during initialization), 0: automatic detection (recommended)
ERROR_STATUS	OUT	DWORD	Internal error code

The following SFCs are used internally:

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

Detailed description

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

Length:
5280 Bytes: up to 128 Pointers (min.)
9120 Bytes: up to 512 Pointers
14240 Bytes: up to 1024 Pointers (max.)

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

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

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

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

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

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

Error codes

Decimal value ERROR_STATUS	Description
1	datablock ibaREQ_DB is write protected
2	datablock ibaREQ_DB invalid (DB = 0 or > limit of CPU)
3	datablock ibaREQ_DB does not exist
4	datablock ibaREQ_DB undefined error
5	datablock ibaREQ_DB too short
6	datablock ibaREQ_DB too short for ibaREQ_UDP
9	internal error (RD_SINFO)
10	no access to datablock ibaREQ_DB (read)
11	no access to datablock ibaREQ_DB (write)
20	initialization not finished
21	insufficient memory for SZL
22	wrong SZL_ID
23	wrong or invalid index of SZL
24	error while reading I&M data from CPU
25	error while reading PLC data
31	initialization canceled with error
32	initialization not completed
41	too many pointers (ibaREQ_DB too small)
42	too many pointers in one command (>128)
44	invalid command id
45	operand invalid (not defined)
46	operand invalid (datatype)
47	operand invalid (memory area)
200	no connection to PN device / DP slave
300	version of ibaREQ_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 peripheral address is invalid
310	no access to datablock ibaREQ_DB (read)
311	no access to datablock ibaREQ_DB (write)
315	error while masking of synchronous faults
316	error while demasking of synchronous faults

Decimal value ERROR_STATUS	Description
320	operand invalid (datatype)
321	operand invalid (pointer)
401	ADR_SLOT/ADR_SLOT_0 invalid hw-id
402	ADR_SLOT/ADR_SLOT_0 invalid hw-id, no IO-Device or DP-Slave
403	ADR_SLOT/ADR_SLOT_0 invalid hw-id, is no PROFIBUS or PROFINET
406	ADR_SLOT/ADR_SLOT_0 invalid configuration slot (0)
407	ADR_SLOT/ADR_SLOT_0 invalid configuration slot (0)
409	ADR_SLOT_1 invalid configuration slot 1
410	no connection to PN device/DP-Slave or error
411	ADR_SLOT_1 invalid hw-id
412	ADR_SLOT_1 invalid hw-id, no IO-Device or DP-Slave
413	ADR_SLOT_1 invalid hw-id, is no PROFIBUS
416	ADR_SLOT_1 invalid configuration slot 1
0x8yyy	error code of inner TUSEND/AG_SEND/AG_LSEND

3.1.2 ibaREQ_PN (FB141)

Description of the formal parameters

Name	Type	Data Type	Description
REQ_DB	IN	BLOCK_DB	DB of the *ibaPDA* communication interface ibaREQ_DB
ADR_SLOT	IN	INT	Start address of the peripheral output range
DEVICE_STATUS	OUT	INT	Status of the PROFINET device
ERROR_STATUS	OUT	INT	Internal error code

The following SFCs are used internally:

- SFC 15 (DPWR_DAT)
- SFC 20 (BLKMOV)
- SFC 21 (FILL)
- SFC 50 (RD_LGADR)
- SFC 71 (LOG_GEO)

Detailed description

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

ADDR_SLOT
Peripheral start address of the used slot in *ibaBM-PN* in the peripheral output range for sending data.

DEVICE_STATUS
Status of the addressed PROFINET device in the *ibaBM-PN*.
(0: Device not accessible or error; 1: Device accessible)

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

Error codes

Decimal value ERROR_STATUS	Description
200	There is no connection to a PN device.
300	The version of ibaREQ_UDPact does not match with ibaREQ_M (ID).
301	The version of ibaREQ_UDPact does not match with ibaREQ_M (FB).
302	The version of ibaREQ_UDPact does not match with ibaREQ_M (DB).
303	The type of transmit agent does not match with configured request type in *ibaPDA*.
306	The configured "ADR_SLOT" is not assigned to a PROFIBUS PN device.
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 (ANY pointer)
410	Device does not exist.

3.1.3 ibaREQ_DP (FB142)

Description of formal parameters

Name	Type	Data Type	Description
ADR_SLOT_0	IN	VARIANT	Start address of the output range of slot 0
ADR_SLOT_1	IN	VARIANT	Start address of the output range of slot 1
RESET_CON	IN	BOOL	FALSE: No reset (default) TRUE: Communication connection reset
REQ_DB	INOUT	UDT	DB for the *ibaPDA* communication interface ibaREQ_DB
ERROR_STATUS	OUT	WORD	Internal error code

Detailed description

ADR_SLOT_0
Start address of the used slot 0 in *ibaBM-DP* in the output range of the process image.

ADR_SLOT_1
Start address of the used slot 1 in *ibaBM-DP* in the output range of the process image.

RESET_CON
Used for manually resetting the communication connection.

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

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

Error codes

Decimal value ERROR_STATUS	Description
300	The version of ibaREQ_UDPact does not match with ibaREQ_M (ID).
301	The version of ibaREQ_UDPact does not match with ibaREQ_M (FB).
302	The version of ibaREQ_UDPact does not match with ibaREQ_M (DB).
303	The type of transmit agent does not match with configured request type in *ibaPDA*.
305	The hardware configuration of the PROFIBUS DP slave is invalid.
306	The configured "OUTPUT_ADR_SLOT" is not assigned to a PROFIBUS DP slave.
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 (ANY pointer)
410	Slave does not exist.

3.1.4 ibaREQ_UDPact (FB145)

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

Description of the formal parameters

Name	Type	Data type	Description
REQ_DB	IN	BLOCK_DB	DB of the *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
The DB is used for data exchange with *ibaPDA*. For all related Request blocks, configure the identical DB.

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

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

Error codes

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

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

3.1.5 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
The DB is used for data exchange with *ibaPDA*. For all related Request blocks, configure the identical DB.

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

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

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

Table 1: 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/en/view/51339682

LOCAL_PORT
Number of the locally used port.

RESET_CON
Used for manually resetting the communication connection.

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

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

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

0X8yyy error code of TCON block is passed.

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

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

0X8yyy error code of TUSEND block is passed.

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

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

Error codes

Hexadecimal value ERROR_TCON	Description
W#16#8086	The ID parameter is outside the valid range.
W#16#8087	Maximum number of connections reached; no additional connection possible.
W#16#8089	The CONNECT parameter does not point to a data block.
W#16#809A	The CONNECT parameter points to a field that does not match the length of the connection description (UDT65).
W#16#809B	The local_device_id in the connection description does not match the CPU.
W#16#80A0	Group error for error codes W#16#80A1 and W#16#80A2
W#16#80A1	Connection or port is already used by user.
W#16#80A2	Local or remote port is used by the system.
W#16#80A3	Attempt is being made to re-establish an existing connection.
W#16#80A4	IP address of the remote connection endpoint is invalid, or it matches the local IP address.
W#16#80A7	Communications error: You called TDISCON before TCON had completed. TDISCON must first complexly terminate the connection referenced by the ID.
W#16#80B2	The CONNECT parameter points to a data block that was generated with the keyword UNLINKED.
W#16#80B3	Inconsistent parameter assignment: Group error for the error codes W#16#80A0 to W#16#80A2, W#16#80A4, W#16#80B4 to W#16#80B9

Hexadecimal value ERROR_TCON	Description
W#16#80B5	Error in active_est parameter (UDT 65) in the UDP protocol variant
W#16#80B6	Parameter assignment error relating to the connection_type parameter (UDT 65)
W#16#80B7	Error in one of the following parameters of UDT 65: block_length, local_tsap_id_len, rem_subnet_id_len, rem_staddr_len, rem_tsap_id_len, next_staddr_len
W#16#80B8	Parameter ID in the local connection description (UDT 65) and parameter ID are different.
W#16#80C3	All connection resources are in use.
W#16#80C4	Temporary communications error: ■ The connection cannot be established at this time. ■ The interface is receiving new parameters. ■ The configured connection is being removed by a TDISCON instruction. ■ The H system is connecting and updating.

Hexadecimal value ERROR_TSEND	Description
W#16#8085	LEN parameter has the value "0" or is greater than the highest valid value.
W#16#8086	The ID parameter is outside the valid address range.
W#16#8088	LEN parameter is greater than the memory area specified in DATA.
W#16#8089	ADDR parameter does not point to a data block.
W#16#80A1	Communication error: ■ The specified connection between user program and communication layer of the operating system has not yet been established. ■ The specified connection between the user program and the communication layer of the operating system is currently being terminated. Transfer over this connection is not possible. ■ The interface is being re-initialized.
W#16#80A4	IP address of the remote connection endpoint is invalid, or it matches the local IP address.
W#16#80B3	The protocol variant (connection_type parameter in the connection description) is not set to UDP. Please use FB 63 TSEND. ADDR parameter: Invalid settings for port number.
W#16#80C3	■ A block with this ID is already being processed in a different priority class. ■ Internal lack of resources.
W#16#80C4	Temporary communications error: ■ The connection between the user program and the communication layer of the operating system cannot be established at this time. ■ The interface is receiving new parameters.

Other documentation

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

3.1.6 ibaREQ_UDPext3 (FB147)

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_SEND
COUNT_SEND	OUT	WORD	Telegram counter of the send block
LOST_SAMPLES	OUT	WORD	Counter for lost measurement values

The following SFCs are used internally:

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

Detailed description

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

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

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

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

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

0X8yyy error code of AG_SEND block is passed.

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

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

3.1.7 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
The DB is used for data exchange with *ibaPDA*. For all related Request blocks, configure the identical DB.

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

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

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

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

The following error codes can be displayed:

0X8yyy error code of AG_LSEND block will be passed

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

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

3.1.8 ibaREQ_PNdev (FB150)

Description of the formal parameters

Name	Type	Data Type	Description
ADR_SLOT	IN	INT	Start address of the peripheral output range
DEVICE_STATUS	OUT	INT	Status of the PROFINET device
ERROR_STATUS	OUT	INT	Internal error code

The following SFCs are used internally:

- SFC 51 (RDSYSST)
- SFC 71 (LOG_GEO)

Detailed description

ADDR_SLOT
Peripheral start address of the used slot in *ibaBM-PN* in the peripheral output range for sending data.

DEVICE_STATUS
Status of the addressed PROFINET device in the *ibaBM-PN*.
(0: Device not accessible or error; 1: Device accessible)

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

Error codes

Decimal value ERROR_STATUS	Description
306	The configured "ADR_SLOT" is not assigned to a PROFIBUS PN device.
410	Device does not exist.

3.2 Description of Request FCs

These functions initialize and control the communication between the S7 controller (SIMATIC S7-300, S7-400, WinAC only), *ibaPDA*, and the *ibaBM-DP* device or *ibaCom-L2B* board.

The blocks used are part of the iba S7 library, see ↗ *iba S7 library, page 128*.

Note

Only use Request blocks from the latest iba S7 library!

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

3.2.1 Request-FC ibaDP_Req (FC122)

This function initializes and controls the communication between the S7 controller, *ibaPDA* and the *ibaBM-DP* device.

The ibaDP_Req has to be called once for each slave in the cyclic program.

The used blocks are part of the iba S7 library, see ↗ *iba S7 library, page 128*.

Note

Only use Request blocks from the latest iba S7 library!

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

Description of the formal parameters of the ibaDP_Req (FC122)

Name	Type	Data Type	Description
DB_PDA	IN	BLOCK_DB	DB of the *ibaPDA* communication interface ibaDP_DB_PDA (2064 Byte)
DB_INTERN	IN	BLOCK_DB	DB of the internal data interface ibaDP_DB_work (1900 Byte)
OUTPUT_ADR_SLAVE	IN	INT	Start address of the peripheral output range, continuous and 244 Bytes long
INIT_FC	IN	BOOL	TRUE: Initialize
INP_RANGE	IN	INT	Number of output bytes (evaluation only during initialization), 0: automatic detection (recommended)
OUT_RANGE	IN	INT	Number of output bytes (evaluation only during initialization), 0: automatic detection (recommended)
MARKER_RANGE	IN	INT	Number of marker bytes (evaluation only during initialization), 0: automatic detection (recommended)
ERROR_STATUS_INIT	OUT	BYTE	Initialization error
ERROR_STATUS_COM	OUT	BYTE	Communication error

The following SFCs are used internally:

- SFC 13 (DPNRM_DG)
- SFC 15 (DPRD_DAT)
- SFC 20 (BLKMOV)
- SFC 21 (FILL)
- SFC 24 (TEST_DB)
- SFC 49 (LGC_GADR)
- SFC 50 (RD_LGADR)
- SFC 51 (RDSYSST)

Detailed description

DB_PDA
DB in the range of 1 to n (see technical data of the CPU). The DB handles the data exchange with *ibaPDA*. Length at least 2064 Bytes.

DB_INTERN
DB in the range of 1 to n (see technical data of the CPU).
Length at least 1900 Bytes.

The following data are stored in this DB:

- Technical data of CPU which are determined during initialization
- S7 pointers requested by *ibaPDA*
- binary and analog transmission data

OUTPUT_ADR_SLAVE
DP start address of the *ibaBM-DP* in the peripheral output range for sending data. Length 244 Bytes, continuous, without gaps.

INIT_FC
Used for initializing the Request block. The Request block is initialized automatically internally. Additionally, the initialization can be performed manually by an external logic via the input INIT_FC = TRUE.

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

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

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

ERROR_STATUS_INIT
Shows error code of initialization errors.

ERROR_STATUS_COM
Shows error code of communication errors.

Error codes

Decimal value ERROR_STATUS_INIT	Description
1	DB_PDA is read-only.
2	DB_PDA-No = 0 or higher than the max. permissible DB number of this CPU
3	DB_PDA with the specified number is not available.
5	DB_PDA is too short.
11	DB_INTERN is read-only.
12	DB_INTERN-No = 0 or higher than the max. permissible DB number of this CPU
13	DB_INTERN with the specified number is not available.
15	DB_INTERN is too short.
16	Error when reading the identification data of the CPU
19	Initialization not finished
21	Not enough memory space for the data set
22	SZL_ID is wrong or not known in this CPU.
23	The index of the SZL is wrong or not permitted.
30	OUT_ADR_SLAVE is no peripheral output range.
31	OUT_ADR_SLAVE is not assigned to a PROFIBUS.

Decimal value ERROR_STATUS_COM	Description
100	Bit number not 0
101	Bit number not 0–7
103	The operand range is not defined.
104	The data type is not defined.
105	DB0 has been requested as data source.
106	DB number is higher than the max. permitted number of DBs for this CPU.
107	DB with the specified number is not available.
109	DB is too short for the accessed operand.
110	Specified data address does not exist.

Decimal value ERROR_STATUS_COM	Description
111	Initialization aborted with error (error cause can be seen in ERROR_STATUS_INIT)
112	Initialization not carried out (error cause can be seen in ERROR_STATUS_INIT)
150	Order fragmentation is not supported.
151	Order Request: Total number of values does not match values in the order.
152	Order Request: Up to 64 binary values permitted
153	Order Request: Up to 64 analog values permitted
200	DP station is not available.

3.2.2 Request-FC ibaDP_Req_H (FC123)

This function initializes and controls the communication between the S7 controller, *ibaPDA* and the *ibaBM-DP* device.

The ibaDP_Req_H function has to be called once per slave pair in the cyclic program.

The used blocks are part of the iba S7 library, see ↗ *iba S7 library, page 128*.

Note

Only use Request blocks from the latest iba S7 library!

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

Description of the formal parameters of the ibaDP_Req_H

Name	Type	Data Type	Description
DB_PDA	IN	BLOCK_DB	DB of the *ibaPDA* communication interface ibaDP_DB_PDA (2064 Byte)
DB_INTERN	IN	BLOCK_DB	DB of the internal data interface ibaDP_DB_work (1900 Byte)
OUTPUT_ADR_SLAVE	IN	INT	Start address of the peripheral output range, continuous and 244 Bytes long
INIT_FC	IN	BOOL	TRUE: Initialize
INP_RANGE	IN	INT	Number of input bytes (evaluation only during initialization), 0: automatic detection (recommended)
OUT_RANGE	IN	INT	Number of output bytes (evaluation only during initialization), 0: automatic detection (recommended)
MARKER_RANGE	IN	INT	Number of marker bytes (evaluation only during initialization), 0: automatic detection (recommended)
ERROR_STATUS_INIT	OUT	BYTE	Initialization error
ERROR_STATUS_COM	OUT	BYTE	Communication error

The following SFCs are used internally:

- SFC 13 (DPNRM_DG)
- SFC 15 (DPRD_DAT)
- SFC 20 (BLKMOV)
- SFC 21 (FILL)
- SFC 24 (TEST_DB)
- SFC 49 (LGC_GADR)
- SFC 50 (RD_LGADR)
- SFC 51 (RDSYSST)

Detailed description

DB_PDA
DB in the range of 1 to n (see technical data of the CPU). The DB is used for data exchange with *ibaPDA*. Length at least 2064 Bytes.

DB_INTERN
DB in the range of 1 to n (see technical data of the CPU).
Length at least 1900 Bytes.

The following data are stored in this DB:

- technical data of CPU which are determined during initialization
- S7 pointers requested by *ibaPDA*
- binary and analog transmission data

OUTPUT_ADR_SLAVE
DP start address of the *ibaBM-DP* in the peripheral output range for sending data. Length 244 Bytes, continuous, without gaps.

INIT_FC
Used for initializing the Request block. The Request block is initialized automatically internally. Additionally, the initialization can be performed manually by an external logic via the input INIT_FC = TRUE.

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

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

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

ERROR_STATUS_INIT
Shows error code of initialization errors.

ERROR_STATUS_COM
Shows error code of communication errors.

Error codes

Decimal value ERROR_STATUS_INIT	Description
1	DB_PDA is read-only.
2	DB_PDA-No = 0 or higher than the max. permissible DB number of this CPU
3	DB_PDA with the specified number is not available.
5	DB_PDA is too short.
11	DB_INTERN is read-only
12	DB_INTERN-No = 0 or higher than the max. permissible DB number of this CPU
13	DB_INTERN with the specified number is not available.
15	DB_INTERN is too short.
16	Error when reading the identification data of the CPU
19	Initialization not finished
21	Not enough memory space for the data set
22	SZL_ID is wrong or not known in this CPU.
23	The index of the SZL is wrong or not permitted.
30	OUT_ADR_SLAVE is no peripheral output range.
31	OUT_ADR_SLAVE is not assigned to a PROFIBUS.
32	Redundancy mode (RM): The configured "OUTPUT_ADR_SLAVE_BUS_0" is incorrect.
33	RM: The configured "OUTPUT_ADR_SLAVE_BUS_0" is not assigned to a PROFIBUS DP Slave.
34	RM: The configured "OUTPUT_ADR_SLAVE_BUS_1" is incorrect.
35	RM: The configured "OUTPUT_ADR_SLAVE_BUS_1" is not assigned to a PROFIBUS DP Slave.
36	RM: SLAVE BUS0 and SLAVE BUS1 do not have the same DP address.

Decimal value ERROR_STATUS_COM	Description
100	Bit number not 0
101	Bit number not 0–7
103	The operand range is not defined.
104	The data type is not defined.
105	DB0 has been requested as data source.
106	DB number is higher than the max. permitted number of DBs for this CPU.
107	DB with the specified number is not available.
109	DB is too short for the accessed operand.
110	Specified data address does not exist.
111	Initialization aborted with error (error cause can be seen in ERROR_STATUS_INIT)
112	Initialization not carried out (error cause can be seen in ERROR_STATUS_INIT)
150	Order fragmentation is not supported.
151	Order Request: Total number of values does not match values in the order.
152	Order Request: Up to 64 binary values permitted
153	Order Request: Up to 64 analog values permitted
200	DP station is not available.
201	Redundancy mode (RM): Slave Bus 0 not available
202	RM: Slave Bus 1 not available
203	RM: Slaves Bus 0 + 1 are not available.
210	Output modules of the Slaves Bus 0 and 1 are configured differently.

3.2.3 Initialization-FC ibaL2B_Init (FC111)

This function checks parameters of the S7-CPU and defines the default settings for the DBs.

The ibaL2B_Init (FC111) function has to be called in all available start-up OBs (OB100, OB101 and OB102).

Description of the formal parameters of the ibaL2B_Init

Name	Type	Data Type	Description
DB_PDA	IN	BLOCK_DB	DB for the *ibaPDA* communication interface ibaL2B_DB_work
INP_RANGE	IN	INT	Limitation of the number of input bytes to be measured (depends on CPU); Recommendation: specify "0" for automatic detection.
OUT_RANGE	IN	INT	Limitation of the number of output bytes to be measured (depends on CPU); Recommendation: specify "0" for automatic detection.
MARKER_RANGE	IN	INT	Limitation of the number of marker bytes to be measured (depends on CPU); Recommendation: specify "0" for automatic detection.
ERROR_STATUS	OUT	BYTE	Error code for parameterization error

The following SFCs are used internally:

- SFC 6 (RD_SINFO)
- SFC 24 (TEST_DB)
- SFC 51 (RDSYSST)

Detailed description

DB_PDA
DB in the range of 1 to n (see technical data of the CPU). The DB handles the data exchange with *ibaPDA*. Each slave requires its own DB. The DB needs to be the same as the DB referenced by ibaL2B_Init.

INP_RANGE
Limits the number of input bytes to be measured (depending on CPU). It is generally recommended to set "0" for automatic detection. Only if the wrong number is displayed in the "S7 System Info" (*ibaPDA*), you may specify the number of input bytes manually.

OUT_RANGE
Limits the number of output bytes to be measured (depending on CPU). It is generally recommended to set "0" for automatic detection. Only if the wrong number is displayed in the "S7 System Info" (*ibaPDA*), you may specify the number of output bytes manually.

MARKER_RANGE
Limits the number of marker bytes to be measured (depending on CPU). It is generally recommended to set "0" for automatic detection. Only if the wrong number is displayed in the "S7 System Info" (*ibaPDA*), you may specify the number of marker bytes manually.

ERROR_STATUS
Error code for parameterization errors relating to the "DB_NUMBER" parameter or during reading of the CPU parameters.

Error codes

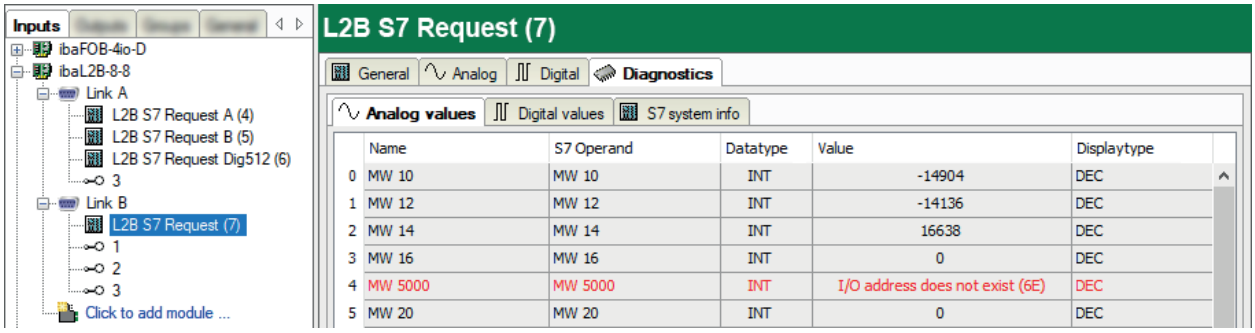
Decimal value ERROR_STATUS	Description	Solution
19	Initialization is not finished.	Contact iba support.
Relating parameter "DB_NUMBER"		
11	DB is read-only.	Change DB attribute.
12	DB no. = 0 or higher than the max. permissible DB number of this CPU	Change to a valid DB.
13	DB with the specified number does not exist.	Copy/rename DB22
14	No sequence-relevant DB	Contact iba support.
15	DB is too short.	Copy/rename DB22
While reading CPU parameters		
21	Not enough memory space is available for the data set.	Contact iba support.
22	SZL_ID is wrong or unknown within this CPU.	Contact iba support.
23	Index is wrong or not allowed.	Contact iba support.

Ranges for number of inputs, outputs and markers

The ibaL2B_Init function can determine the number of inputs, outputs and markers in the CPU can be evaluated reliably. For this purpose, the value "0" has to be assigned to the parameters as in our example.

In exceptional cases, you may adjust or modify these parameters after consultation with the iba support. These parameters help to avoid configuration errors when selecting signals in *ibaPDA*. For example, if you select a marker which does not exist in the S7, *ibaPDA* will generate an error message. In the *ibaPDA* diagnostics of the corresponding module you can recognize such conflicts quite easily.

In the *Value* column of the *Analog values* tab, the message "I/O address does not exist (6e)" appears, if the value does not comply with the range detected in ibaL2B_Init.



In such case, please check the following:

- You selected an operand address in *ibaPDA* which is available in the CPU. If not, use an available operand.
- The automatically determined address ranges in the *ibaPDA* menu *S7 System Info* correspond to the technical data of the S7-CPU. If not, set the number of I, O and M Bytes manually (from the technical data of the S7-CPU) when parameterizing the ibaL2B_Init.

3.2.4 Communication-FC ibaL2B_Req (FC112)

This function provides the communication between the S7 and the *ibaCom-L2B* card within the *ibaPDA* PC. It also supervises the communication links.

The ibaL2B_Req (FC112) function has to be called once per slave in the cyclic program.

Description of the formal parameters of the ibaL2B_Req (FC112)

Name	Type	Data Type	Description
DB_PDA	IN	BLOCK_DB	DB for the *ibaPDA* communication interface ibaL2B_DB_work
DP_SEND_ADR	IN	INT	Start address in the output range for sending data.
DP_RECEIVE_ADR	IN	INT	Start address in the input range for receiving data.
TIME_OUT_VAL	IN	INT	Communication timeout
TIMER_NR	IN	Timer	Timer for the internal stand-by clock generator.
TIME_OUT_FLAG	OUT	BOOL	Communication lost between S7 and *ibaCom- L2B* card
ERROR_STATUS	OUT	BYTE	Error code for parameterization error

The following SFCs are used internally:

- SFC 20 (BLKMOV)
- SFC 24 (TEST_DB)
- SFC 36 (MSK_FLT)
- SFC 37 (DMSK_FLT)
- SFC 50 (RD_LGADR)

Detailed description

DB_PDA
DB in the range of 1 to n (see technical data of the CPU). The DB handles the data exchange with *ibaPDA*. Each slave requires its own DB. The DB needs to be the same as the DB referenced by ibaL2B_Init.

DP_SEND_ADR
DP start address of the *ibaCom-L2B* card in the peripheral output range for sending data.

DP_RECEIVE_ADR
DP start address of the *ibaCom-L2B* card in the peripheral input range for the reception of data.

TIME_OUT_VAL
Communication timeout TIME_OUT_VAL × 100 ms is the maximum waiting time.

TIMER_NR
Timer for the internal stand-by clock generator in the range from T0 to Tn (see technical data of the CPU). Each slave requires its own timer.

TIME_OUT_FLAG
This flag is set when the connection between S7 and *ibaCom-L2B* card is interrupted.

ERROR_STATUS
Error code for parameterization errors.

Error codes

Decimal value ERROR_STATUS	Description	Solution
92	Invalid command from *ibaPDA*	Contact iba support.
100	In the DB address range is not enough space for the requested analog values.	Reduce data amount in *ibaPDA* to a max. of 112 Byte.
101	channel number > 31	Contact iba support.
102	Data type is not supported (analog values only).	Contact iba support.
103	Range identifier is missing or cannot be interpreted.	Contact iba support.
104	This variable type cannot be interpreted.	Contact iba support.
105	DB 0 was requested as data source.	Use other allowed DB.
106	DB number is higher than max. permissible number of DBs of this CPU.	Change to a valid DB.
107	DB with the specified number does not exist.	Request existing DB data.
109	DB is too short for the accessed operand.	Copy/rename DB22

Decimal value ERROR_STATUS	Description	Solution
110	Actual operand address (I, O, M, P) does not exist.	Request available operands.
111	Initialization aborted with error.	Analyze error state of ibaL2B_Init, otherwise contact iba support.
112	Initialization not performed.	Call ibaL2B_Init in the start-up OBs.

3.2.5 Communication-FC ibaL2B_Req_CP (FC113) for CP342-5

Not every CPU in the S7-300 family has an integrated or free (unused) DP interface. In the S7-400 family, as described above, an external communication processor (CP) can also be addressed using the functions ibaL2B_Init (FC111) and ibaL2B_Req (FC112). The external CP is connected to the communication bus and acts like an internal DP interface.

The connection is more complex for CPUs in the S7-300 family, as they do not have a communication bus. Here, the connections to the iba slaves are realized by the standard blocks FC1 (DP_SEND) and FC2 (DP_RCV) from the S7 standard library and not over the peripheral address space. The physical connection is carried out via the external CP342-5. In this case, ibaL2B_Req (FC112) is replaced by ibaL2B_Req_CP (FC113). Moreover, another DB ibaL2B_CP_SNDRCV (in example DB10) is required for all slave connections, where the peripheral data of all slaves are stored temporarily.

The function ibaL2B_Req_CP (FC113) has to be called instead of the ibaL2B_Req (FC112) once per slave in the cyclic program.

Description of the formal parameters of the ibaL2B_Req_CP (FC113)

Name	Type	Data Type	Description
DB_PDA	IN	BLOCK_DB	DB for the *ibaPDA* communication interface ibaL2B_DB_work
DB_DP_Data	IN	BLOCK_DB	DB which buffers the send and receive data for FC_SEND and FC_RECV.
DB_ADR_Offset_ SEND	IN	INT	DB-OFFSET address of the DB range for the SEND data
DB_ADR_Offset_ RECV	IN	INT	DB-OFFSET address of the DB range for the RECEIVE data
TIME_OUT_VAL	IN	INT	Communication timeout
TIMER_NR	IN	Timer	Timer for the internal stand-by clock generator.
TIME_OUT_FLAG	OUT	BOOL	Communication lost between S7 and *ibaCom- L2B* card
ERROR_STATUS	OUT	BYTE	Error code for parameterization error

Detailed description

DB_PDA
DB in the range of 1 to n (see technical data of the CPU). The DB handles the data exchange with *ibaPDA*. Each slave requires its own DB. The DB needs to be the same as the DB referenced by ibaL2B_Init.

DB_DP_Data
DB in the range of 1 to n (see technical data of the CPU). This DB buffers the send and receive data for FC_SEND and FC_RECV. The data of all slaves can be collected in one DB.

DB_ADR_Offset_SEND
DB-OFFSET address of the DB range for the SEND data. 122 Byte are occupied per slave. Each slave must have an own OFFSET address for the SEND data.

DB_ADR_Offset_RECV
DB-OFFSET address of the DB range for the RECEIVE data. 16 Byte are occupied per slave. Each slave must have an own OFFSET address for the RECEIVE data.

TIME_OUT_VAL
Communication timeout TIME_OUT_VAL × 100 ms is the maximum waiting time.

TIMER_NR
Timer for the internal stand-by clock generator in the range from T0 to Tn (see technical data of the CPU). Each slave requires its own timer.

TIME_OUT_FLAG
This flag is set when the connection between S7 and *ibaCom-L2B* card is interrupted.

ERROR_STATUS
Error code for parameterization errors.

Error codes

Decimal value ERROR_STATUS	Description	Solution
92	Invalid command from *ibaPDA*	Contact iba support.
100	In the DB address range is not enough space for the requested analog values.	Reduce data amount in *ibaPDA* to a max. of 112 Byte.
101	channel number > 31	Contact iba support.
102	Data type is not supported (analog values only).	Contact iba support.
103	Range identifier is missing or cannot be interpreted.	Contact iba support.
104	This variable type cannot be interpreted.	Contact iba support.
105	DB 0 was requested as data source.	Use other allowed DB.
106	DB number is higher than max. permissible number of DBs of this CPU.	Change to a valid DB.

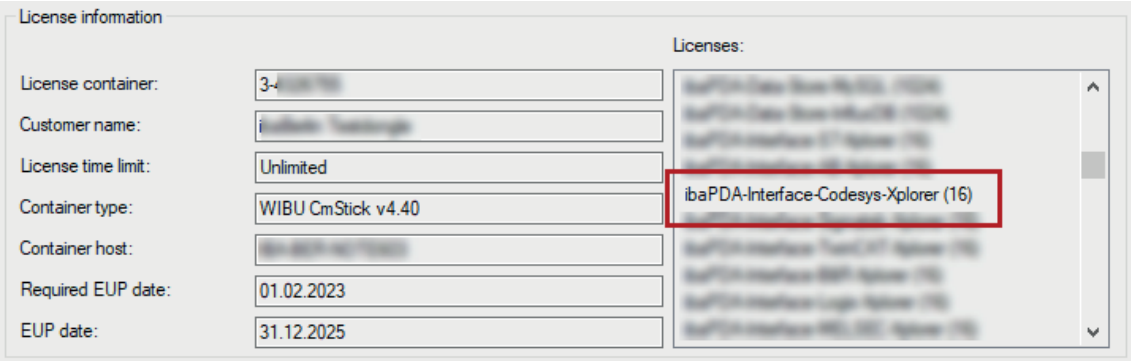
Decimal value ERROR_STATUS	Description	Solution
107	DB with the specified number does not exist.	Request existing DB data.
109	DB is too short for the accessed operand.	Copy/rename DB22
110	Actual operand address (I, O, M, P) does not exist.	Request available operands.
111	Initialization aborted with error.	Analyze error state of ibaL2B_Init, otherwise contact iba support.
112	Initialization not performed.	Call ibaL2B_Init in the start-up OBs.

4 Diagnostics

4.1 License

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

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



4.2 Visibility of the interface

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

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

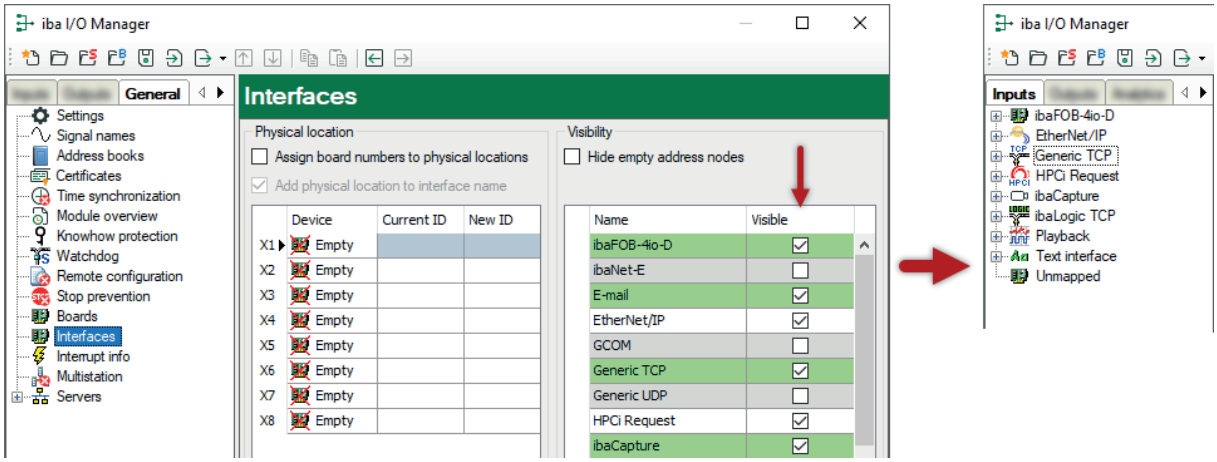
Visibility

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

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

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

Selected interfaces are visible, the others are hidden:



4.3 Log files

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

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

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

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

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

Examples:

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

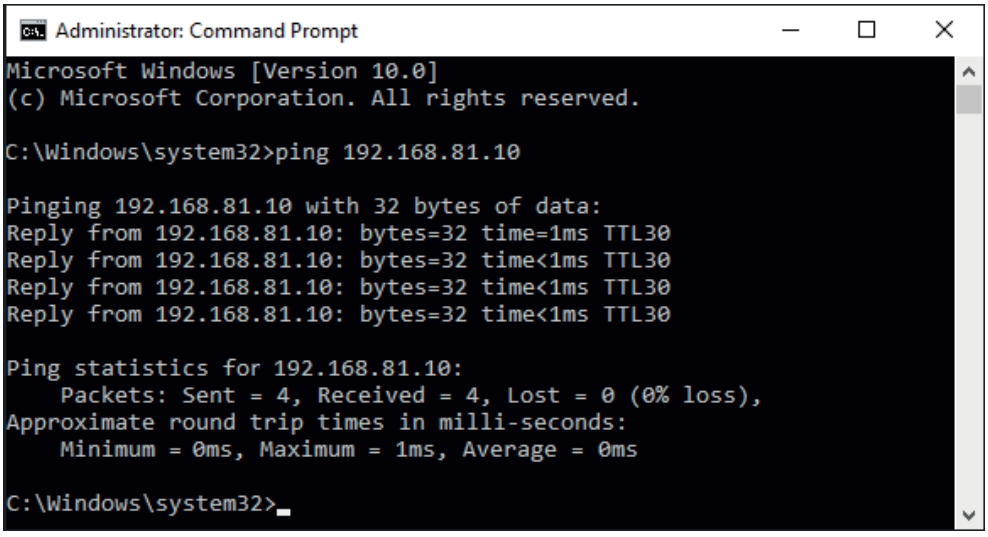
4.4 Connection diagnostics with PING

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

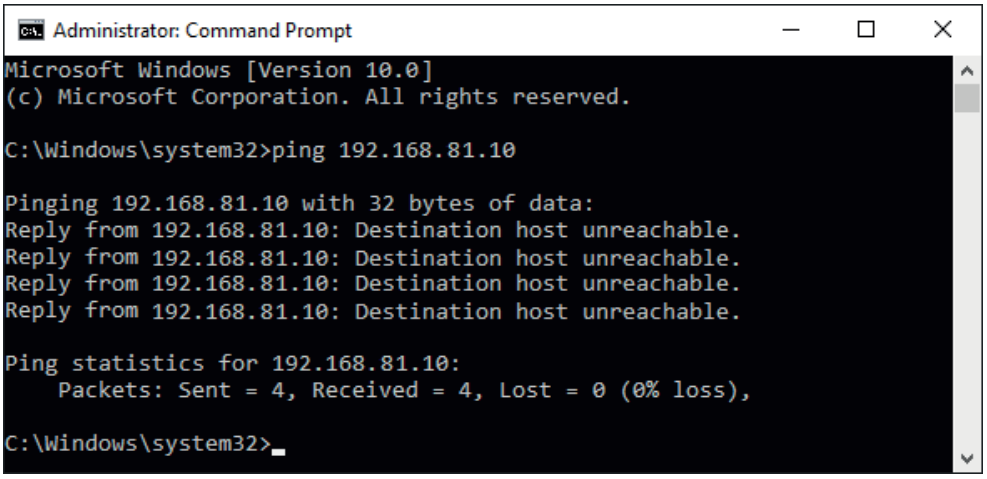
1. Open a Windows command prompt.

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

→ With an existing connection you receive several replies.



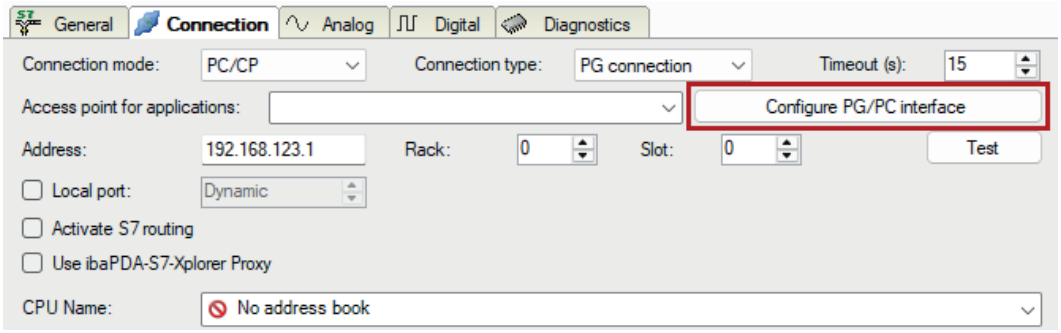
→ With no existing connection you receive error messages.



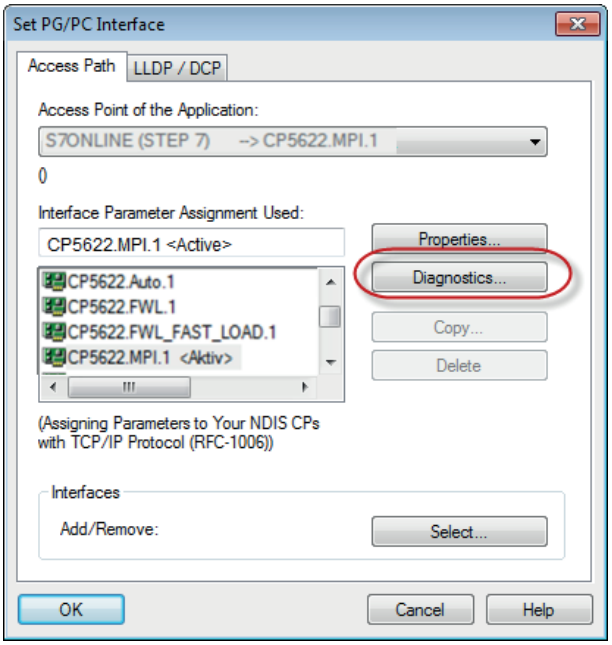
4.5 Connection diagnostics with PG/PC interface

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

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

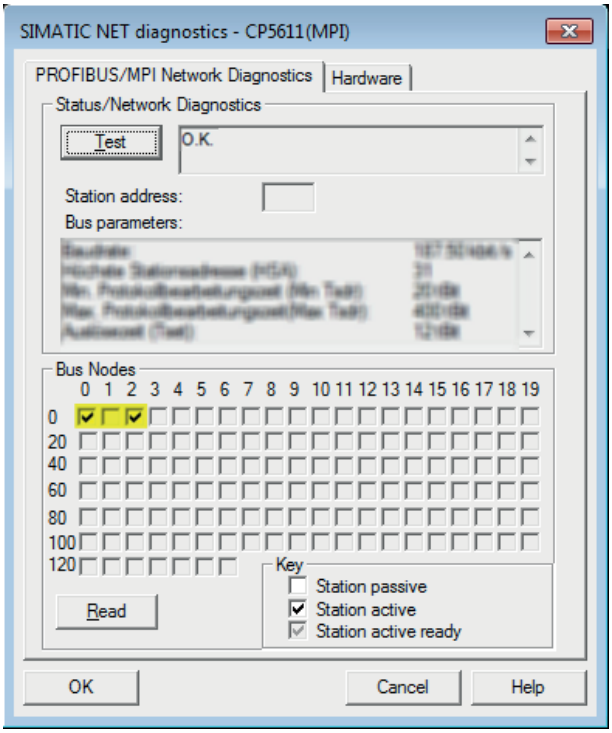


2. Open the diagnostics dialog with the <Diagnostics> button.



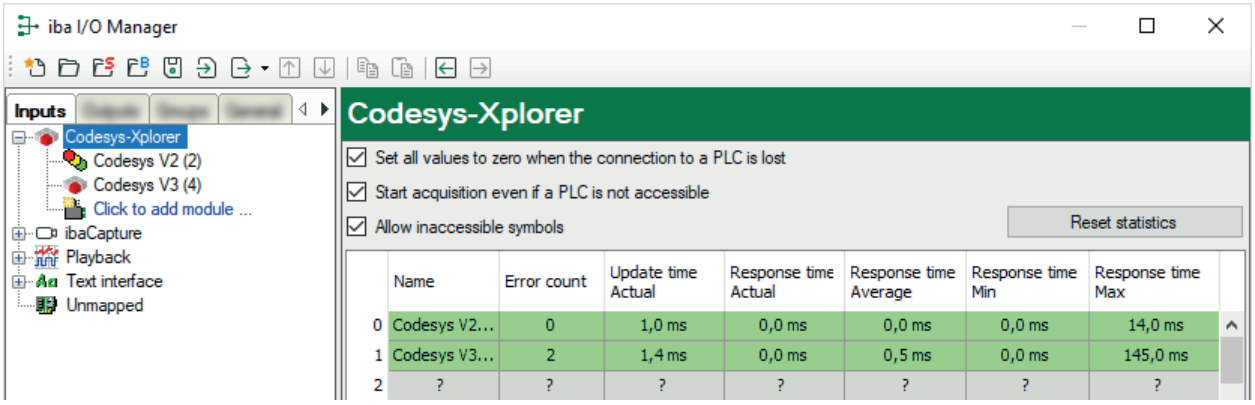
3. Start the network diagnostics with the <Test> button.
   Check the availability of the bus devices with the <Read> button.

→ The following figure shows an example of a diagnostics of a SIMATIC Net CP5622 (PROFIBUS).
There is one active station at each of the addresses 0 and 2.



4.6 Connection table

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



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

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

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

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

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

4.7 Diagnostic modules

Other documentation

You can find a description of the diagnostic modules for the *ibaBM-DP* and *ibaBM-PN* in the respective device documentation.

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

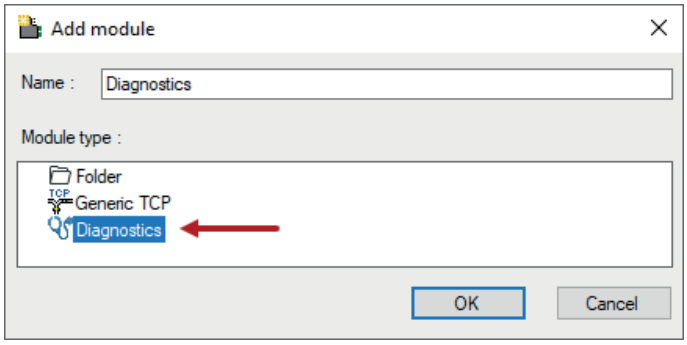
A diagnostic module is always assigned to a data acquisition module of the same interface and supplies its connection information. By using a diagnostic module, you can record and analyze the diagnostic information continuously in the *ibaPDA* system.

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

Example for the use of diagnostic modules:

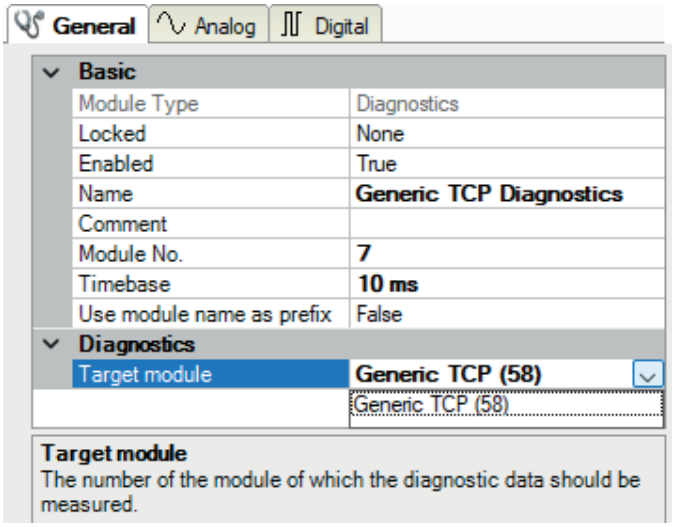
- A notification can be generated, whenever the error counter of a communication connection exceeds a certain value or the connection gets lost.
- In case of a disturbance, the current response times in the telegram traffic may be documented in an incident report.
- The connection status can be visualized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

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



Module settings diagnostic module

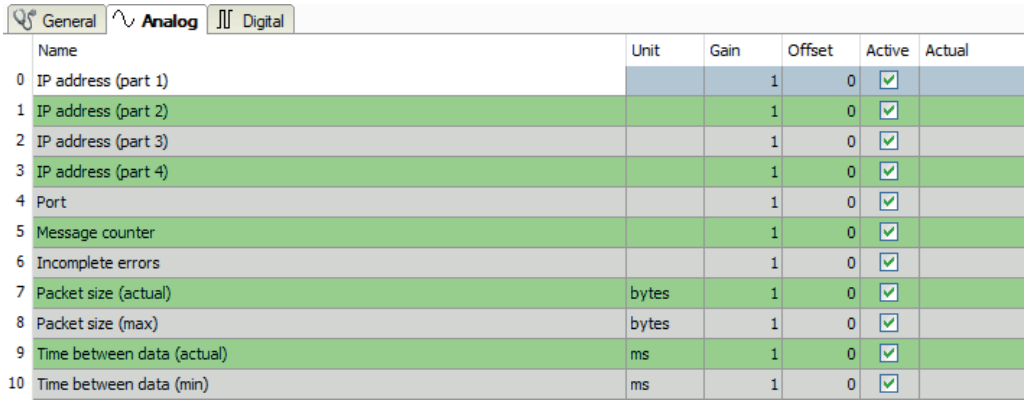
For a diagnostic module, you can make the following settings (example: Generic TCP):



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

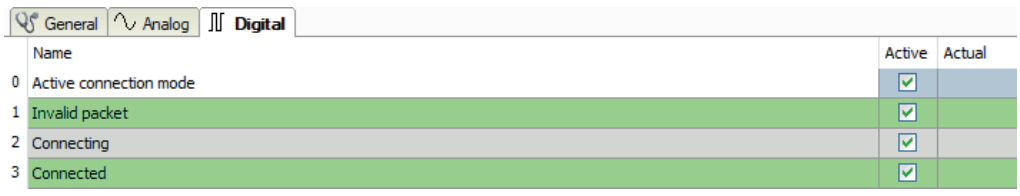
There is only one setting which is specific for the diagnostic module: the target module.

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



	Name	Unit	Gain	Offset	Active	Actual
0	IP address (part 1)		1	0	☑	
1	IP address (part 2)		1	0	☑	
2	IP address (part 3)		1	0	☑	
3	IP address (part 4)		1	0	☑	
4	Port		1	0	☑	
5	Message counter		1	0	☑	
6	Incomplete errors		1	0	☑	
7	Packet size (actual)	bytes	1	0	☑	
8	Packet size (max)	bytes	1	0	☑	
9	Time between data (actual)	ms	1	0	☑	
10	Time between data (min)	ms	1	0	☑	

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



	Name	Active	Actual
0	Active connection mode	☑	
1	Invalid packet	☑	
2	Connecting	☑	
3	Connected	☑	

Diagnostic signals

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

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

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

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

5 Appendix

5.1 iba S7 library

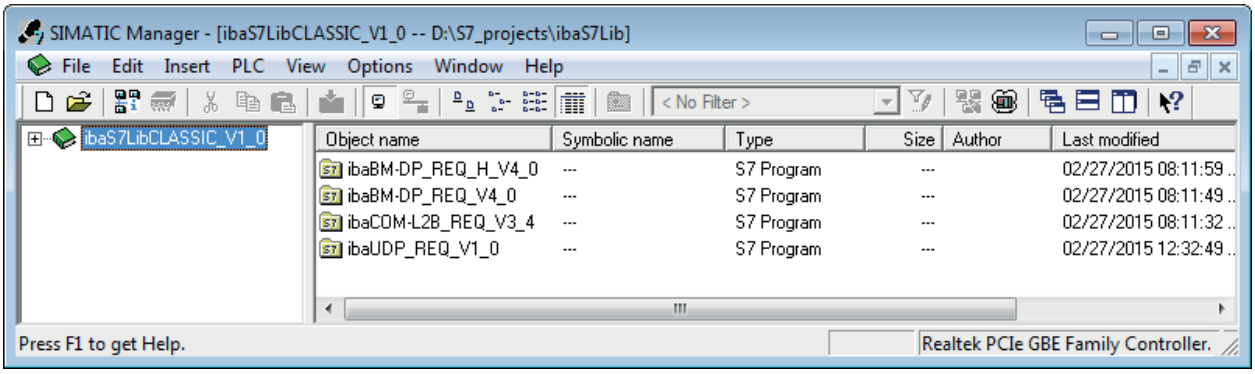
The iba S7 library is available in two versions:

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

5.1.1 iba S7 library for SIMATIC Manager

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

You find the iba S7 library as an archived file on the data storage medium "iba Software & Manuals" in the following directory:
`\04_Libraries_and_Examples\10_Libraries\01_SIMATIC_S7\`



The following components are included:

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

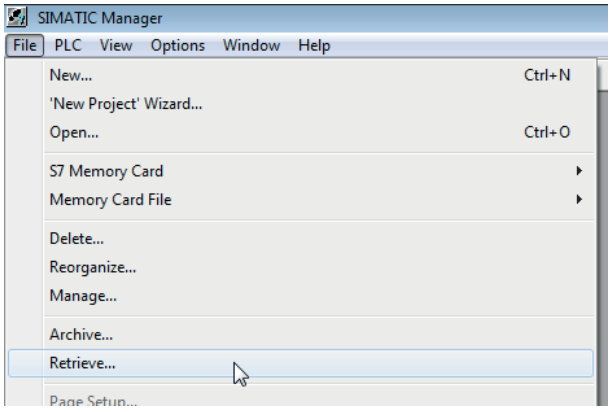
iba connection	Block name	Block no.	Note
ibaCom-L2B	ibaL2B_Init	FC111	formerly FC23 and FC101
	ibaL2B_Req	FC112	formerly FC22 and FC100
	ibaL2B_Req_CP	FC113	formerly FC26 and FC102 only necessary when using a CP342-5 instead of the FC112
	ibaL2B_DB_work	DB22	-
	ibaL2B_DB_Struct	UDT22	-
	ibaL2B_CP_SNDRCV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

Table 2: ibaS7LibCLASSIC block overview

5.1.1.1 Integrating the library into SIMATIC Manager

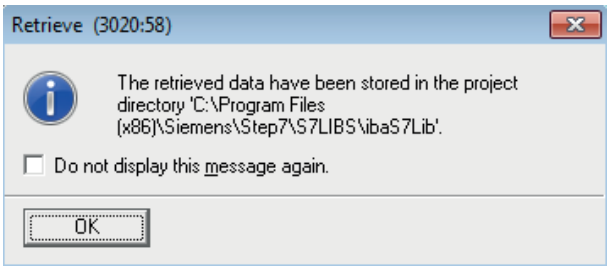
For integrating the library, it has to be retrieved in the SIMATIC Manager. Copy the iba S7 library to a local directory of your computer, on which the SIMATIC Manager is executed.

1. Select the menu *File – Retrieve*.

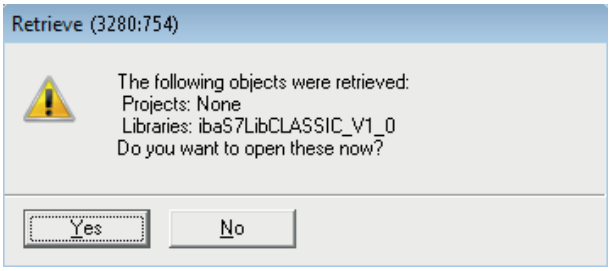


2. Select the archive file of the iba S7 library and select a storage location for the extracted library in a next step.

3. Confirm the message for a successful extraction.



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

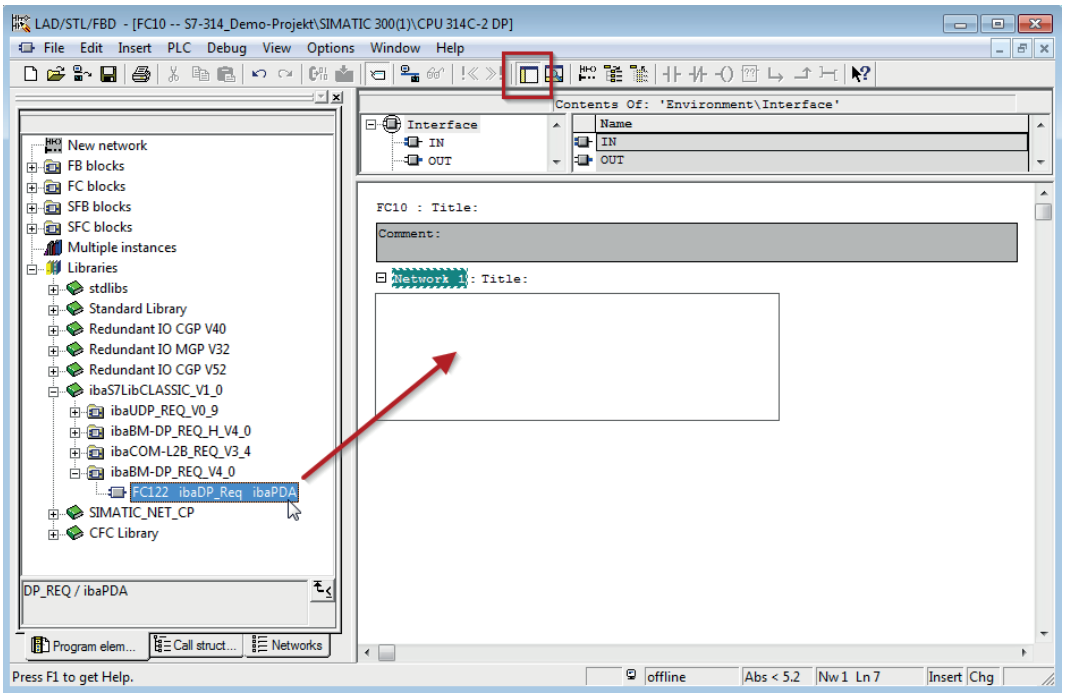


→ Now, the library is integrated and can be closed again.

5.1.1.2 Using the blocks in SIMATIC Manager

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

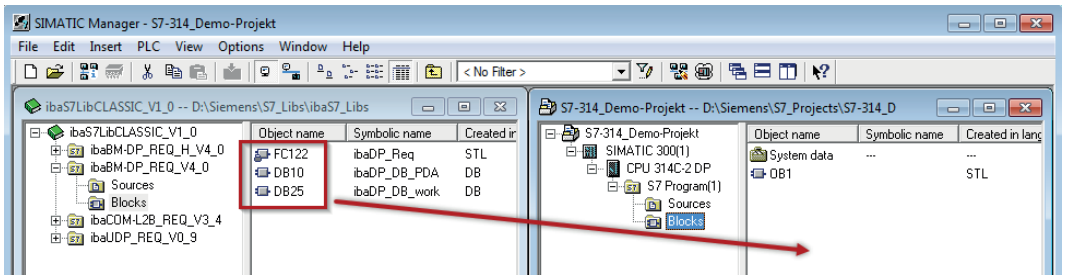
- Display the block library and drag the required blocks to the opened destination block.



- Open the library via *File – Open – Libraries*, and the required destination project in parallel.

  With the <Tile horizontal> button, both projects can be displayed side by side.

  You can either drag & drop the blocks or copy & paste them.



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

5.2 Application examples

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

- For the *ibaBM-DPM-S* device in the directory
  `\04_Libraries_and_Examples\30_ibaBM-DPM-S\01_SIMATIC_S7\Request-S7\`
- For the *ibaBM-DP* device in the directory
  `\04_Libraries_and_Examples\31_ibaBM-DP\01_SIMATIC_S7\Request-S7\`
- For the *ibaBM-PN* device in the directory
  `\04_Libraries_and_Examples\32_ibaBM-PN\01_SIMATIC_S7\Request-S7\`
- For the *ibaCom-L2B* card in the directory
  `\04_Libraries_and_Examples\40_ibaCOM-L2B\01_SIMATIC_S7\Request-S7\`
- For the interface *ibaPDA-Interface-ibaNet-E*
  `\04_Libraries_and_Examples\53_ibaPDA-Interface-ibaNet-E\01_SIMATIC_S7\Request-S7\`
- For the interface *ibaPDA-Interface-S7-TCP/UDP*
  `\04_Libraries_and_examples\50_ibaPDA-Interface-S7-TCP_UDP\Request-S7\`

Examples are available for the following configurations:

iba	S7-CPU	S7 project	ibaPDA project
ibaBM-PN	S7-300 PN-IF	S7CLASSIC_ PN_REQ_Vxx.zip	ibaPDA_S7CLASSIC_ PN_REQ_Vxx.zip
	S7-400 PN-IF		
	S7-1500 PN-IF	S7TIA_ PN_REQ_Vxx.zip	ibaPDA_S7TIA_ PN_REQ_Vxx.zip
		ibaS7TIA_PN_REQsym_ Vxx.zap*	ibaPDA_S7TIA_PN_REQsym
		ibaS7TIA_PN_ REQ3sym_V*.zap*	
ibaBM-DP	S7-300 DP-IF	S7CLASSIC_ DP_REQ_Vxx.zip	ibaPDA_S7CLASSIC_ DP_REQ_CPU3xx_Vxx.zip
	S7-400 DP-IF		ibaPDA_S7CLASSIC_ DP_REQ_CPU4xx_Vxx.zip
	S7-400 + CP443-5		ibaPDA_S7CLASSIC_ DP_REQ_CPU4xx_Vxx.zip
	WinAC		ibaPDA_S7CLASSIC_ DP_REQ_WinAC_Vxx.zip
	S7-400 with CFC	S7CLASSIC_ DP_REQ_CFC_Vxx.zip	ibaPDA_S7CLASSIC_ DP_REQ_CFC_Vxx.zip
	S7-400H	S7CLASSIC_ DP_REQ_H_Vxx.zip	ibaPDA_S7CLASSIC_ DP_REQ_CPU4xxH_Vxx.zip
	S7-300 DP-IF	S7CLASSIC_ L2B_REQ_Vxx.zip	ibaPDA_S7CLASSIC_ DP_L2Bcomp_REQ_Vxx.zip
	S7-1500 DP-IF	S7TIA_ DP_REQ_Vxx.zip	ibaPDA_S7TIA_ DP_REQ_Vxx.zip
ibaBM-DPM-S	S7-300 DP-IF	S7CLASSIC_ DP_REQ_Vxx.zip	ibaPDA_S7CLASSIC_ DPMS_REQ_CPU3xx_Vxx.zip
	S7-400 DP-IF		ibaPDA_S7CLASSIC_ DPMS_REQ_CPU4xx_Vxx.zip
	WinAC		ibaPDA_S7CLASSIC_ DPMS_REQ_WinAC_Vxx.zip
	S7-400 with CFC	S7CLASSIC_ DP_REQ_CFC_Vxx.zip	ibaPDA_S7CLASSIC_ DPMS_REQ_CFC_Vxx.zip
	S7-400H	S7CLASSIC_ DP_REQ_H_Vxx.zip	ibaPDA_S7CLASSIC_ DPMS_REQ_CPU4xxH_Vxx.zip
ibaCom-L2B	S7-300 DP-IF	S7CLASSIC_ L2B_REQ_Vxx.zip	ibaPDA_S7CLASSIC_ L2B_REQ_Vxx.zip
	S7-300 + CP342-5		
	S7-400 with CFC	S7CLASSIC_ L2B_REQ_CFC_Vxx.zip	ibaPDA_S7CLASSIC_ L2B_REQ_CFC_Vxx.zip

iba	S7-CPU	S7 project	ibaPDA project
ibaPDA-Interface-ibaNet-E	S7-1500	ibaS7TIA_NetE_REQ_Vxx.zap*	ibaPDA_S7TIA_NetE_REQ
		ibaS7TIA_NetE_REQsym_Vxx.zap*	ibaPDA_S7TIA_NetE_REQsym
		ibaS7TIA_NetE_REQ3sym_V*.zap*	
ibaPDA-Interface-S7-TCP/UDP	S7-300 PN	S7CLASSIC_UDP_REQ_Vxx.zip	ibaPDA_S7CLASSIC_UDP_REQ_Vxx.zip
	S7-300 + CP343-1 LEAN		
	S7-400 + CP443-1		
	S7-1500	S7TIA_UDP_REQ_Vx_SPx_Vyy.zip	ibaPDA_S7TIA_UDP_REQ_Vyy.zip
		ibaS7TIA_UDP_REQsym_Vx_SPx_Vyy.zip	ibaPDA_S7TIA_UDP_REQsym_Vyy.zip
		ibaS7TIA_UDP_REQ3sym_Vx_SPx_Vyy.zip	

5.3 S7 cycle time measurements

5.3.1 ibaCom-L2B

The following tables provide information on the cycle time extension you can expect for *ibaPDA* in conjunction with L2B-Request depending on the S7-CPU type.

Test series with up to 256 analog signals and 256 digital signals (8 slaves) were used to obtain the time values. The measurement values were determined in a test environment and only serve as reference points. The values may deviate in other system environments.

S7-CPUs with integrated DP interface

S7-CPU type	Max. cycle time extension per slave (32 analog signals + 32 digital signals)
316- 2AG00	13 ms
314- 6CF00	8.63 ms
315- 2AG10	6.88 ms
317- 2EJ10	0.875 ms
317- 6FF00	1.125 ms
318- 2AG00	0.75 ms
416- 2XK00	1.25 ms
413- 2XG02	2.625 ms
414- 2XG03	0.875 ms
416- 2XK04	0.375 ms

S7-CPUs with external DP interface (CP)

S7-CPU type/ CP	Max. cycle time extension per slave (32 analog signals + 32 digital signals)
CPU315 with CP342- 5	6.0 ms
CPU318 with CP342- 5	0.75 ms
CPU416 with CP443- 5	0.625 ms

Note

When using the external PROFIBUS-CP CP342-5 with the S7-300, a cycle-precise measurement is usually not possible because the data transmission over the backplane bus of the S7-300 is too slow. Otherwise, precise cycle measurements are possible if the PROFIBUS cycle is sufficiently fast.

5.3.2 ibaBM-DP

The following tables provide information about the code runtimes that the request blocks require when using *ibaPDA* in combination with Request-S7 for *ibaBM-DP*.

The measurement values were determined in a test environment and only serve as reference points. The values may deviate in other system environments.

SIMATIC S7 CPU	Number of signals	Data amount	ibaDP_Req FC122
CPU412-2 PN 6ES7 412-2EK06-0AB0	1 INT + 0 BOOL (1 Pointer)	2 Byte	564 µs
	59 REAL + 64 BOOL (2 Pointers)	244 Byte	1614 µs
	59 REAL + 64 BOOL (123 Pointers)	244 Byte	1632 µs

SIMATIC S7 CPU	Number of signals	Data amount	ibaREQ_M FB1400	ibaREQ_DP FB1402
CPU1516-3 PN/DP 6ES7 516-3AN00-0AB0	1 INT + 0 BOOL (1 Pointer)	2 Byte	192 µs	286 µs
	59 REAL + 64 BOOL (2 Pointers)	244 Byte	194 µs	296 µs
	59 REAL + 64 BOOL (123 Pointers)	244 Byte	192 µs	834 µs
	122 INT + 0 BOOL (1 Pointer)	244 Byte	195 µs	287 µs
	122 INT + 0 BOOL (122 Pointer)	244 Byte	198 µs	706 µs

5.3.3 ibaBM-PN

The following tables provide information about the code runtimes that the request blocks require when using *ibaPDA* in combination with Request-S7 for *ibaBM-PN*.

The measurement values were determined in a test environment and only serve as reference points. The values may deviate in other system environments.

SIMATIC S7 CPU	Number of signals	Data amount	ibaREQ_M FB140	ibaREQ_PN FB141
CPU412-2 PN 6ES7 412-2EK06-0AB0	1 INT + 0 BOOL (1 Pointer)	2 Byte	128 µs	302 µs
	59 REAL + 64 BOOL (2 Pointers)	244 Byte	126 µs	376 µs
	59 REAL + 64 BOOL (123 Pointers)	244 Byte	132 µs	937 µs
	122 INT + 0 BOOL (1 Pointer)	244 Byte	126 µs	342 µs
	122 INT + 0 BOOL (122 Pointer)	244 Byte	132 µs	954 µs

SIMATIC S7 CPU	Number of signals	Data amount	ibaREQ_M FB1400	ibaREQ_PN FB1401
CPU1516-3 PN/DP 6ES7 516-3AN00-0AB0	1 INT + 0 BOOL (1 Pointer)	2 Byte	198 µs	276 µs
	59 REAL + 64 BOOL (2 Pointers)	244 Byte	194 µs	282 µs
	59 REAL + 64 BOOL (123 Pointers)	244 Byte	192 µs	806 µs
	122 INT + 0 BOOL (1 Pointer)	244 Byte	195 µs	274 µs
	122 INT + 0 BOOL (122 Pointer)	244 Byte	198 µs	672 µs

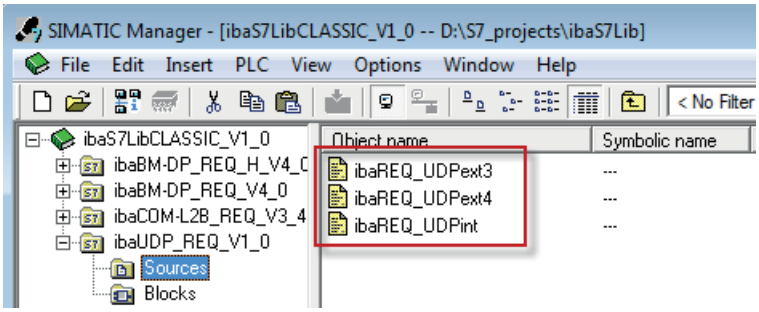
5.4 Adaptation to the renumbered system functions

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.

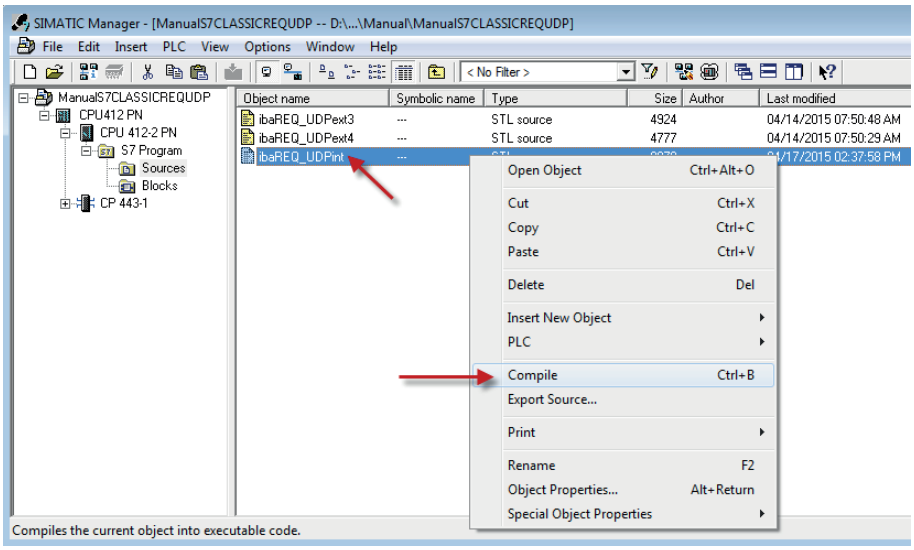
Symbolic name	Standard numbering	Origin
ibaREQ_UDPact	FB145	iba AG, ibaS7Lib
TCON	FB65	Siemens, Standard Library
TDISCON	FB66	Siemens, Standard Library
TUSEND	FB67	Siemens, Standard Library
AG_SEND	FC5	Siemens, SIMATIC_NET_CP
AG_LSEND	FC50	Siemens, SIMATIC_NET_CP
ibaUDT_UDPact	UDT145	iba AG, ibaS7Lib

Table 3: Subordinate blocks

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



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



Note

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

5.5 S7 routing

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

Reference

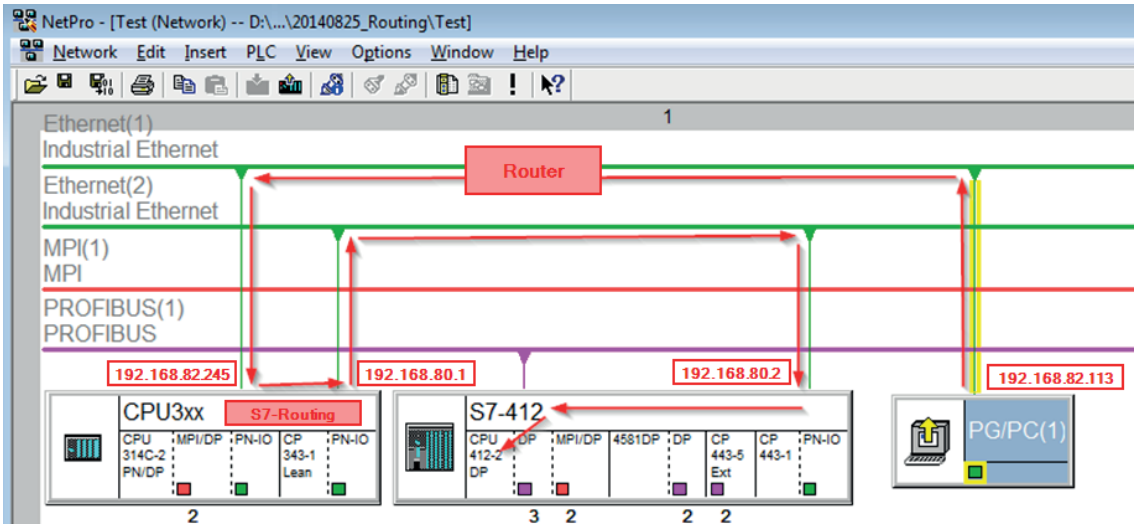
For more information about the S7 Routing, refer to:

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

5.5.1 Routing from Ethernet to Ethernet

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

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



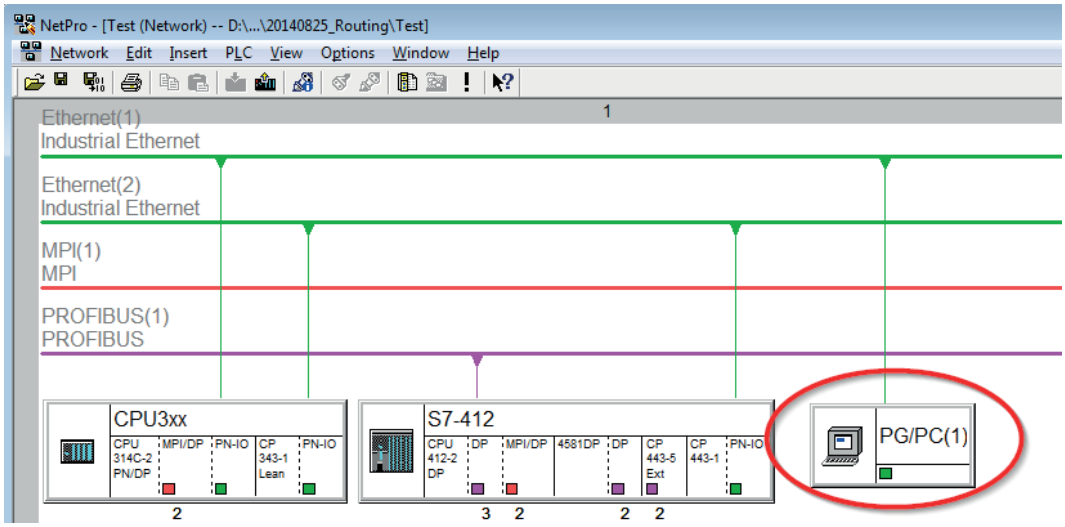
The engineering computer (also with *ibaPDA*) is to access the CPU412 controller. The computer and the controller are not directly connected via a common network or bus. The connection has to run over the CPU314C controller.
"Passing" the communication through this controller is called *S7 Routing*.

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

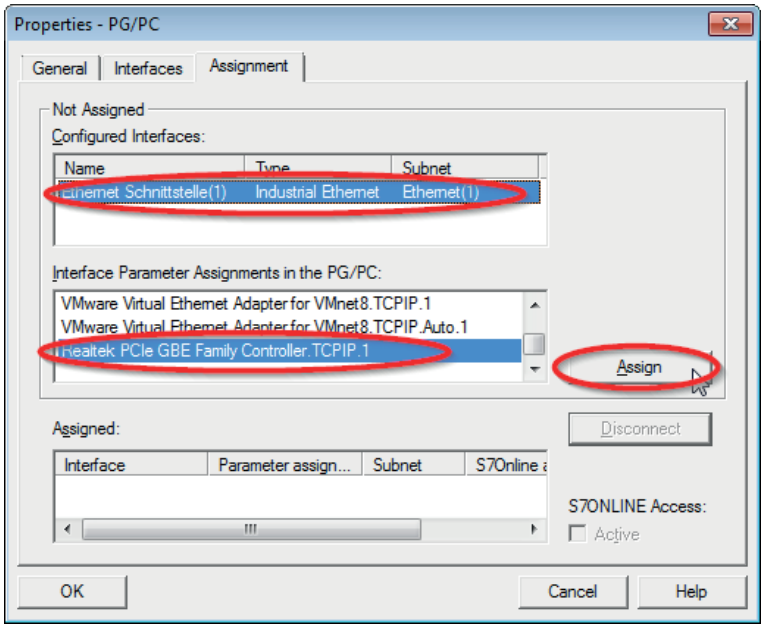
5.5.1.1 Configuration of STEP 7/NetPro

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

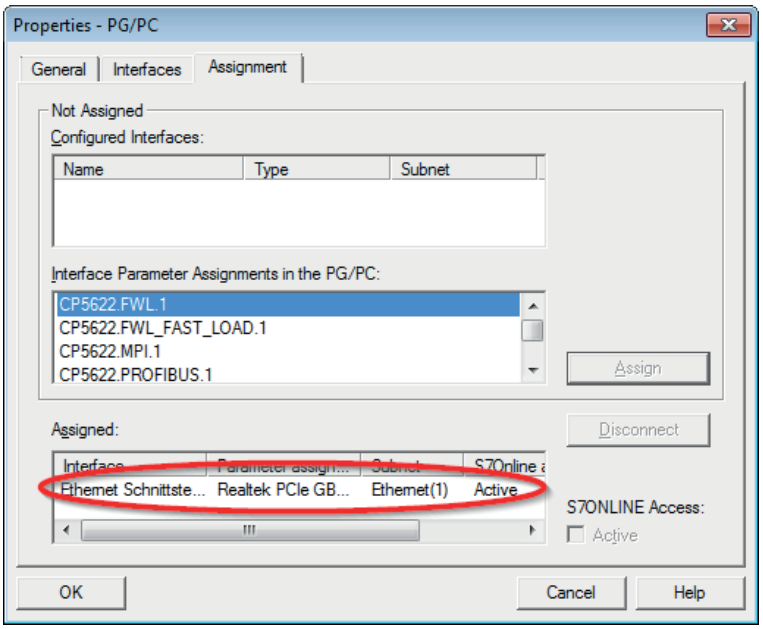
1. Add a PG/PC station and configure it.



2. Assign an interface (network card).

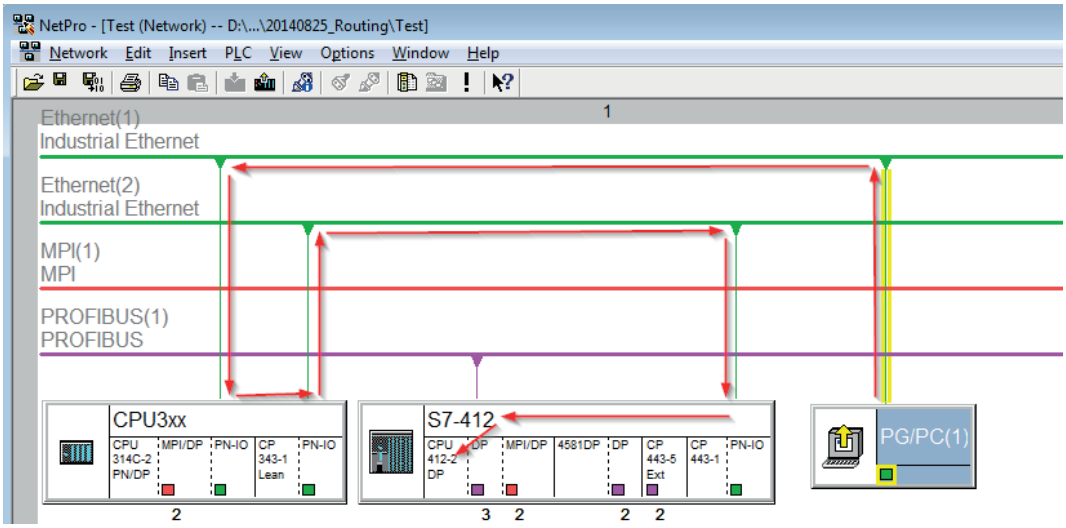


→ Result:



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

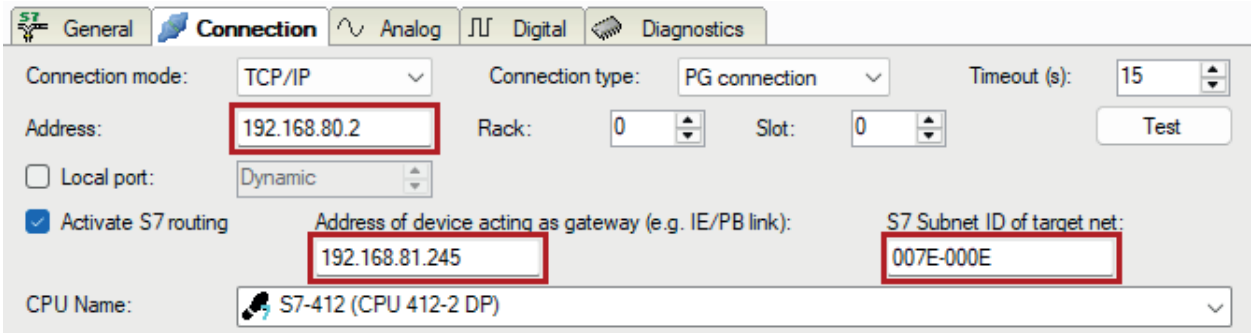
In the following figure, the communication path is shown using arrows (these are not displayed in SIMATIC NetPro).



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

5.5.1.2 Configuration of ibaPDA

Configure the following settings.



Activate S7 routing
Enable this option to use S7 routing.

Address
Enter the address of the target control (here CPU412)

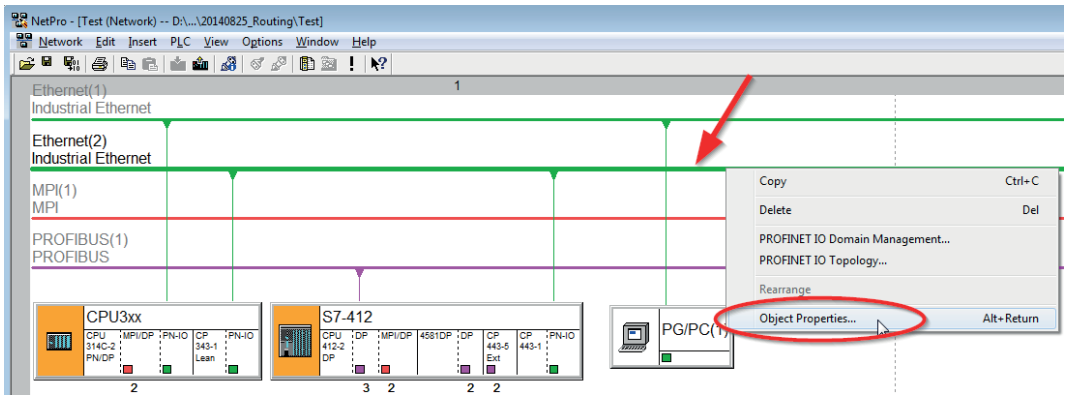
Address of device acting as gateway
Enter the address of the gateway (here CPU314C).

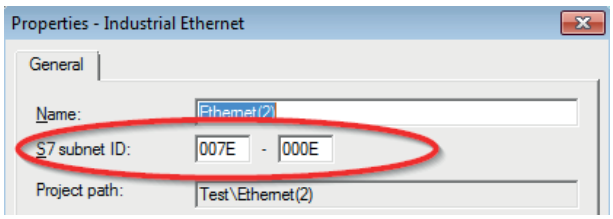
S7 subnet ID of target net
Enter the subnet ID from STEP 7 NetPro or TIA Portal.

Identifying the S7 subnet ID in NetPro

You can identify the S7 subnet ID in NetPro.

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

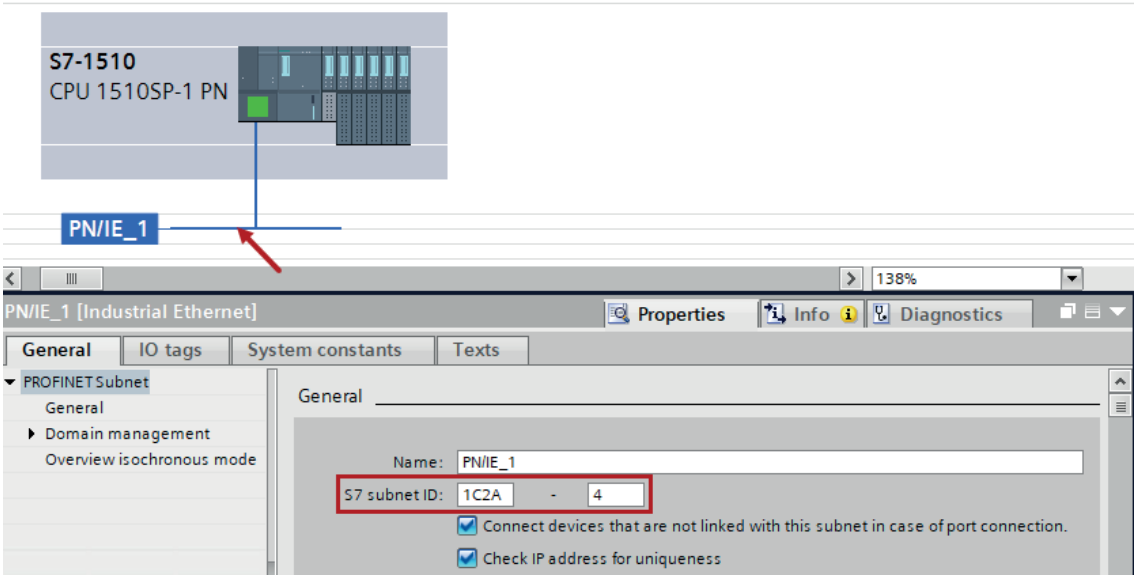




Identifying the S7 subnet ID in TIA Portal

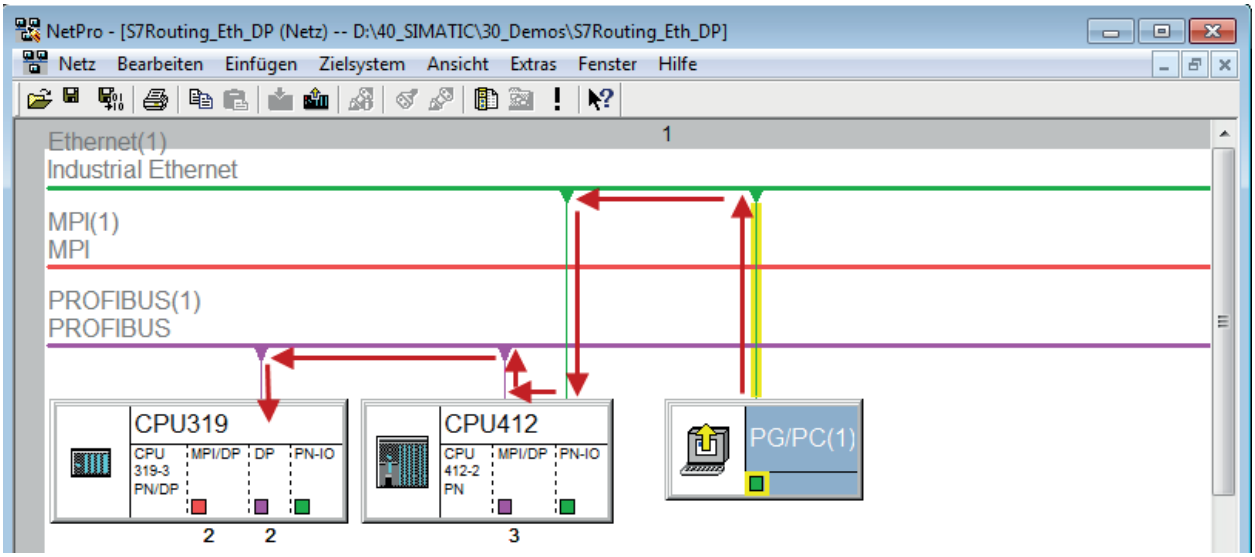
You can identify the S7 subnet ID in TIA Portal.

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



5.5.2 Routing from Ethernet to PROFIBUS

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

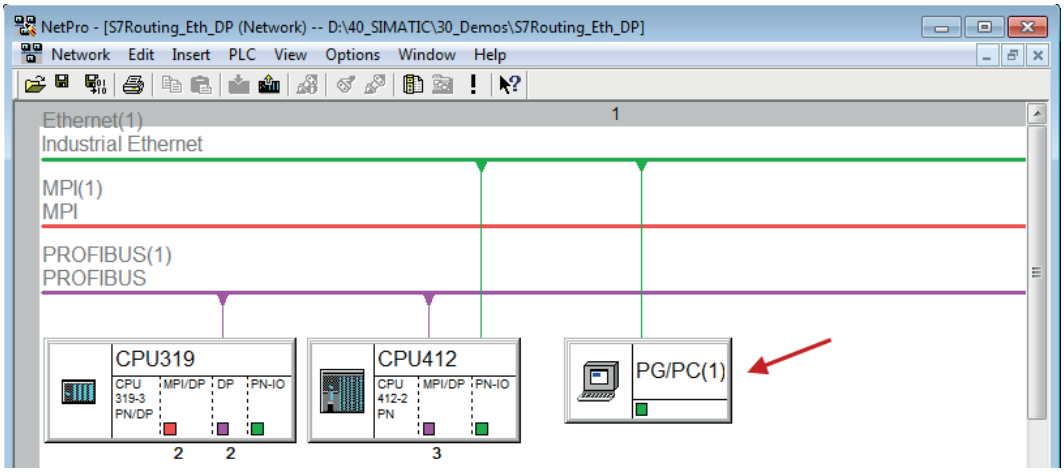


The engineering computer (also with *ibaPDA*) is to access the CPU319 controller. The computer and the controller are not directly connected via a common network or bus. The connection has to run over the CPU412 controller.
"Passing" the communication through this controller is called *S7 Routing*.

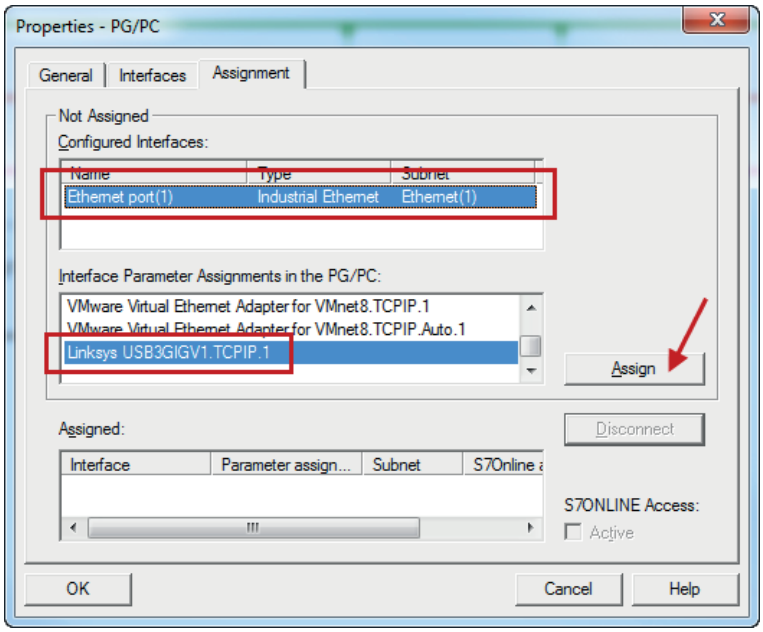
5.5.2.1 Configuration of STEP 7/NetPro

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

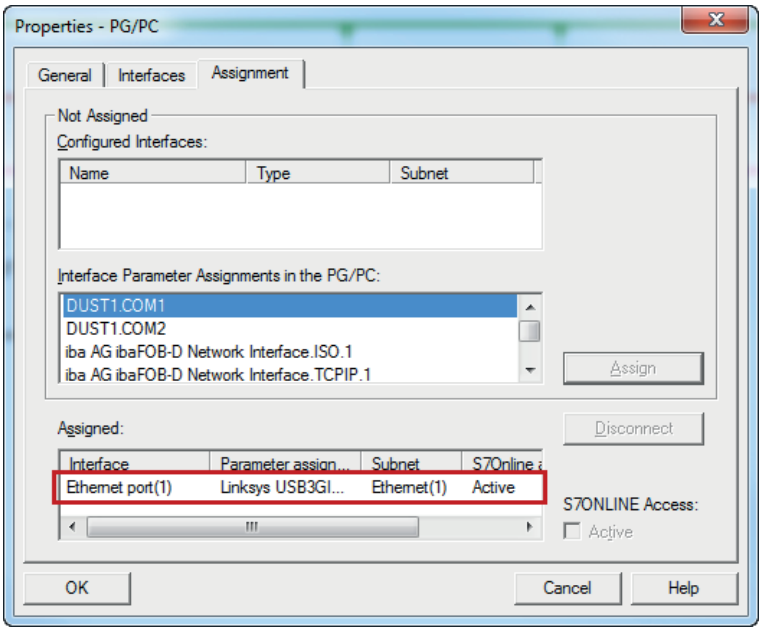
1. Add a PG/PC station and configure it.



2. Assign an interface (network card).

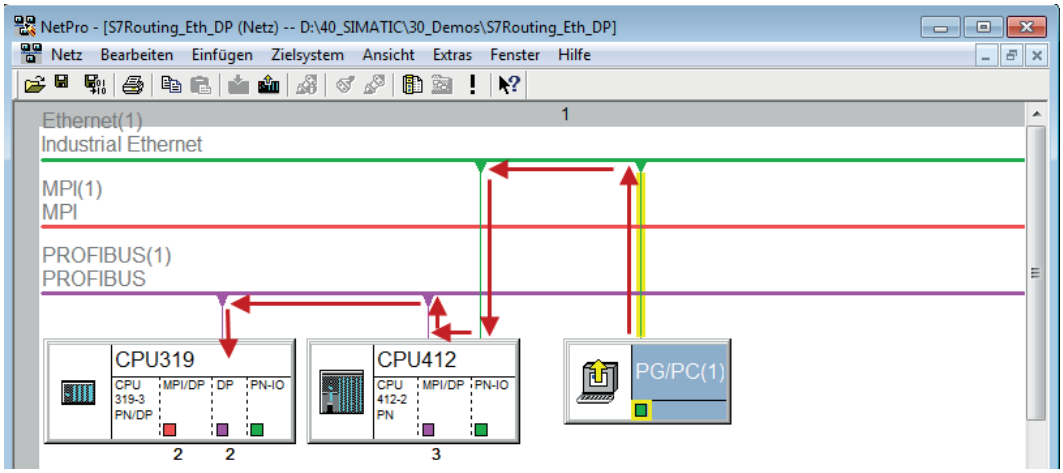


→ Result:



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

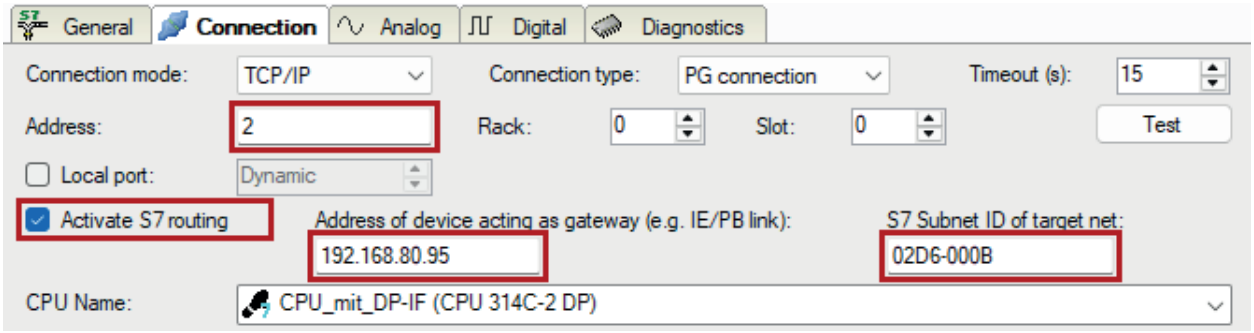
In the following figure, the communication path is shown using arrows (these are not displayed in SIMATIC NetPro).



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

5.5.2.2 Configuration of ibaPDA

Configure the following settings.



Activate S7 routing
Enable this option to use S7 routing.

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

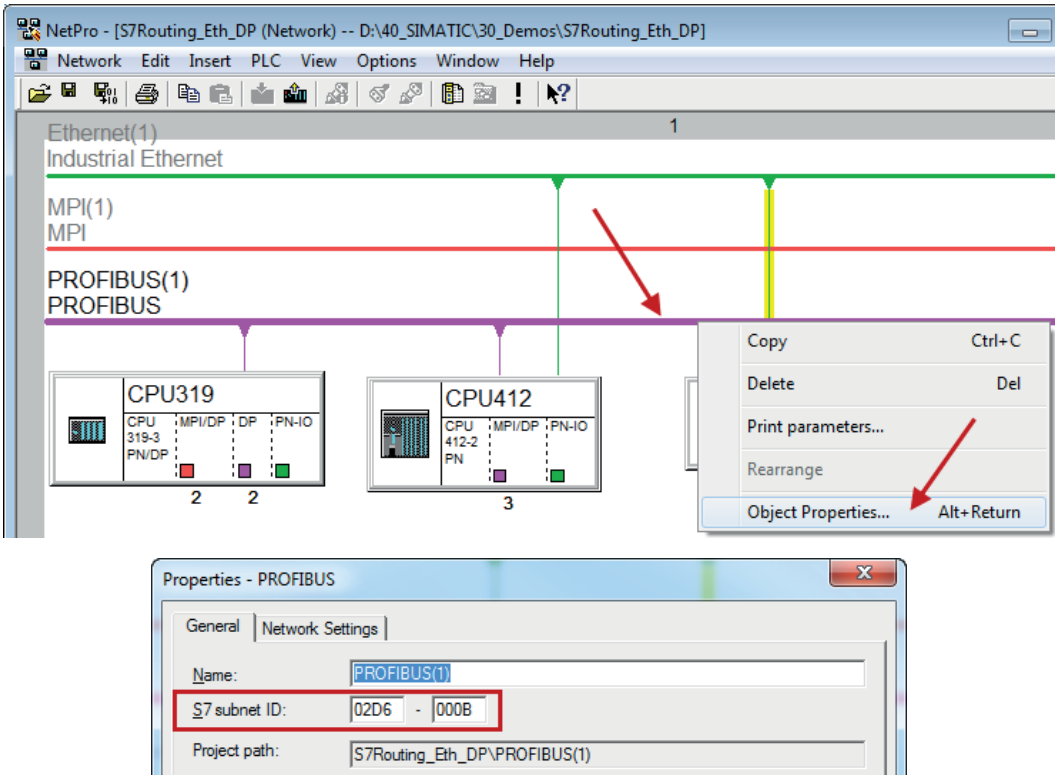
Address of device acting as gateway
Enter the address of the gateway (here CPU412).

S7 subnet ID of target net
Enter the subnet ID from STEP 7 NetPro.

Identifying the S7 subnet ID in NetPro

You can identify the S7 subnet ID in NetPro.

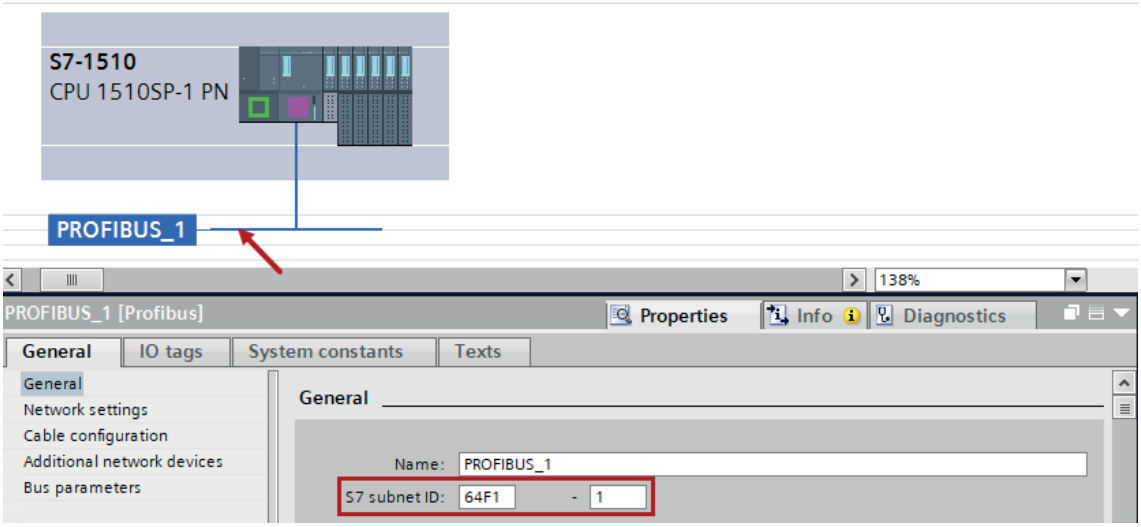
Right-click on the secondary bus system and open the *Object Properties*.



Identifying the S7 subnet ID in TIA Portal

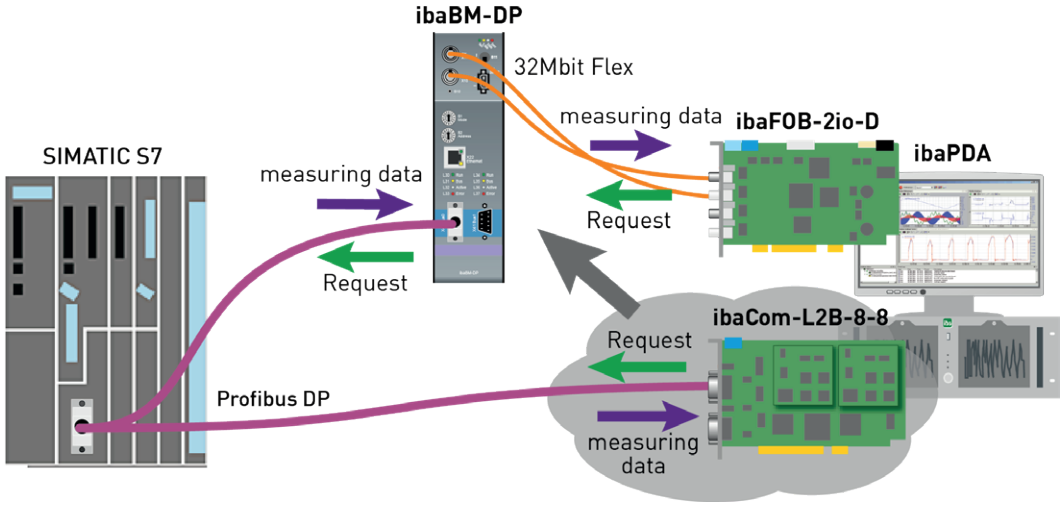
You can identify the S7 subnet ID in TIA Portal.

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



5.6 Replacing Request-S7 on ibaCom-L2B by ibaBM-DP

A common task is the replacement of an existing Request-S7 solution based on the *ibaCom-L2B* card by an *ibaBM-DP*. This might be required in case of spare parts (the *ibaCom-L2B* card is discontinued) or when a new computer without PCI slots is used.



This task can be solved in an easy way by using the *ibaCom-L2B* compatible modules of *ibaBM-DP*.

No modifications in the program and the hardware configuration of the S7-CPU are required.

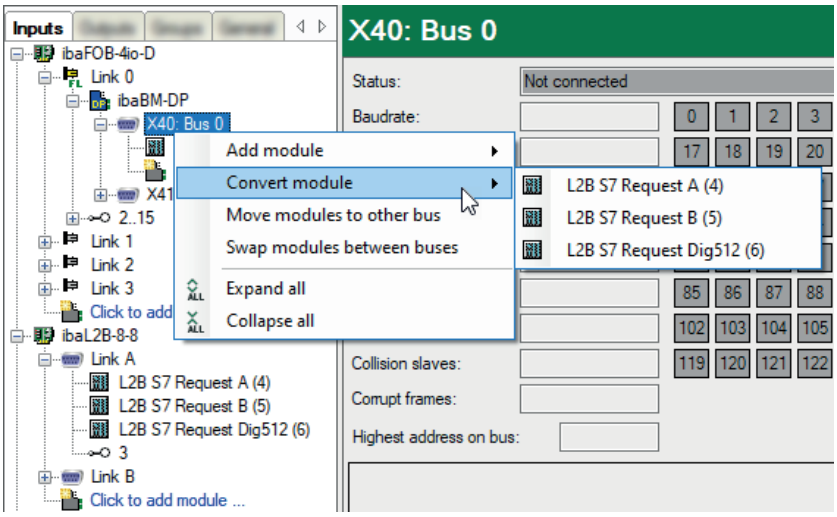
As for the *ibaCom-L2B* card also for the compatible solution with *ibaBM-DP* no network connection between *ibaPDA* computer and the S7-CPU is required. The measurement values (Request handshake) are requested via FO and are sent via the PROFIBUS I/O range.

The following steps have to be performed for the replacement:

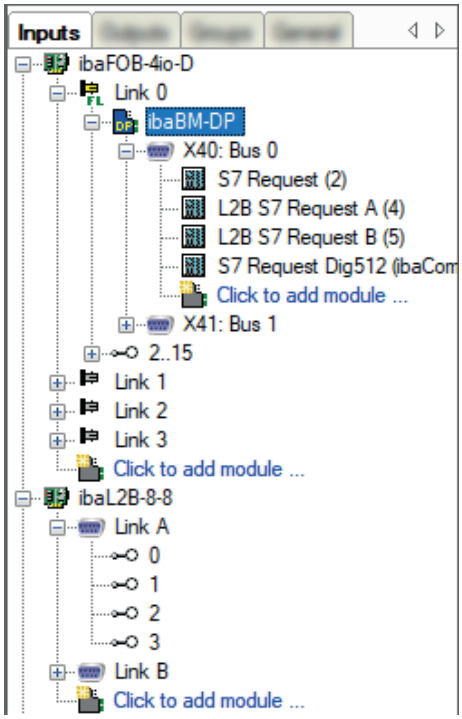
1. If required, install a new *ibaFOB-D* card in the *ibaPDA* computer. A card with input and output is needed (*ibaFOB-io-D*, *ibaFOB-2io-D* or *ibaFOB-4io-D*) as the compatible modules can only be used in the bidirectional 32Mbit Flex mode.
2. Connect the *ibaBM-DP* to the *ibaFOB-D* card via the bidirectional FO cable.
3. Connect the PROFIBUS connector of the *ibaCom-L2B* card to the PROFIBUS plugs of the *ibaBM-DP*.
4. In the *ibaPDA* I/O Manager, a newly installed *ibaFOB-D* card is displayed. Configure an *ibaBM-DP* device on the respective link.
5. Click with the right mouse button on the PROFIBUS links and select *Convert module* in the context menu.

→ All available L2B Request-S7 modules are displayed.

Select here the modules you want to convert to an *ibaCom-L2B* compatible module (usually all modules).



→ The selected modules are deleted on the *ibaCom-L2B* card and converted into compatible modules on the *ibaBM-DP* device.

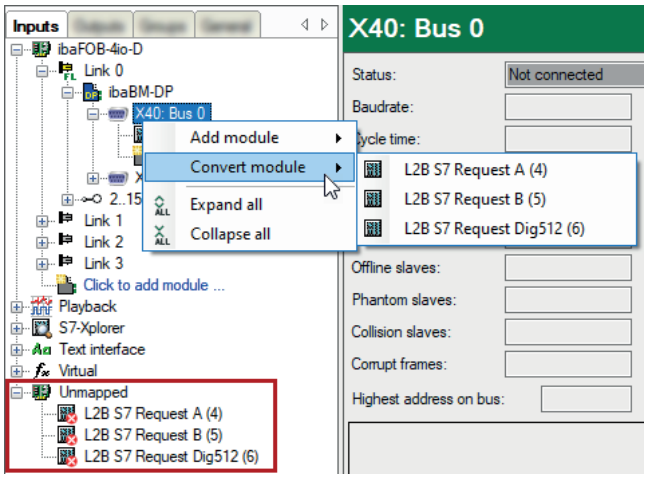


→ The conversion is finished.

6. Apply the new configuration with <OK>.

Note

You can also convert L2B Request-S7 modules if the modules are filed under the unmapped modules. This is e.g. the case if the existing I/O configuration incl. L2B Request S7 modules is loaded on a new *ibaPDA* computer, that does not have an *ibaCom-L2B* anymore, but only the combination of *ibaFOB-D* and *ibaBM-DP*.



5.7 Usage of MPI/DP-TCP-adapters

You can connect Siemens S7 CPUs that do not have an Ethernet interface via MPI/DP-TCP-adapters to their MPI interface.

Different manufacturers offer adapters that convert the S7 communication from TCP/IP to MPI/PROFIBUS DP.

Basically, you can use these adapters together with *ibaPDA*. Therefore, you must configure the connection on the *ibaPDA*-side as a TCP-connection.

The access on the adapter is possible over its IP-address, which can be set by means of the associated software provided by the manufacturer.

In the connection configuration enter "0" for frame and the MPI address or DP address of the S7 CPU as slot.

For more information, refer to the corresponding device documentation.

6 Support and contact

Support

Phone: +49 911 97282-14

Email: support@iba-ag.com

Note

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

Product Security Incident Response Team (PSIRT) at iba

Current advisories: www.iba-ag.com/en/security

Email: psirt@iba-ag.com

Contact

Headquarters
iba AG
Gebhardtstrasse 10–20
90762 Fuerth
Germany

Phone: +49 911 97282-0

Email: iba@iba-ag.com

Mailing address
iba AG
Postbox 1828
D-90708 Fuerth, Germany

Delivery address

iba AG
Gebhardtstrasse 10
90762 Fuerth, Germany

Regional and worldwide

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

www.iba-ag.com