ibaPDA-Request-HiPAC

Request data interface to Danieli HiPAC systems

Manual

Issue 1.3

Measurement Systems for Industry and Energy

www.iba-ag.com

Manufacturer

iba AG

Koenigswarterstrasse 44

90762 Fuerth

Germany

Contacts

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

Version	Date	Revision	Author	Version SW
1.3	09-2025	Connection with ibaFOB-R, selection of Codesys version	nm	8.8.0

Contents

1 About this documentation

This documentation describes the function and application of the software interface.

ibaPDA-Request-HiPAC.

The product*ibaPDA-Request-HiPAC*is an extension of *ibaPDA* for the optional acces to variables when recording data from Danieli HiPAC controllers. Only the extensions and differences are described in this documentation. For all other functions and operating options, please refer 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 he/she is capable of assessing safety and recognizing possible consequences and risks on the basis of his/her specialist training, knowledge and experience and knowledge of the standard regulations.

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

- Basic knowledge of *ibaPDA*
- Basic knowledge of network technology
- Knowledge of configuration and operation of the relevant control system

1.2 Notations

In this manual, the following notations are used:

Action	Notation
Menu command	Menu *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.

2 System requirements

The following system requirements are necessary for using the *ibaPDA-Request- HiPAC* data interface:

- *ibaPDA* v8.6 or higher
- Additional license for *ibaPDA-Request-HiPAC*
- If UDP is used as data path:
  - Additional license *ibaPDA-Interface-Generic-UDP*
- If Reflective Memory is used as data path:
  - Additional license *ibaPDA-Interface-Reflective-Memory*
  - FO card of type *ibaFOB-R* or PCIe-5565PIORC (e.g. from ABACO) in the *ibaPDA* computer
- All non VME x86 based HiPAC controllers are supported
- Danieli HiPAC controller HiPAC V2 or HiPAC V3
- Ethernet cponnection to the controller
- Libraries with iba request blocks
  - ibaHiPACRequest.lib for connection via Generic UDP and Reflective Memory (Codesys V2)
  - ibaHiPACRequest.lib for connection via Generic UDP and Reflective Memory (Codesys V3)

System specification

- Maximum of 8 request blocks (IBA_REQ_A) per controller
- Maximum of 1024 requested signals (analog or digital) per request block
- Maximum telegram size:
  - UDP: 4096 bytes
  - RM: specified by HiPAC RM library (typ. 4096 bytes)

Licenses

Order no.	Product name	Description
31.001302	ibaPDA-Request-HiPAC	Extension license for an *ibaPDA* system to be able to use the request functionality with Danieli HiPAC controllers
31.001075	ibaPDA-Interface-Generic-UDP	Extension license for an *ibaPDA* system with an Generic UDP interface Number of connections: 64

Order no.	Product name	Description
31.001220	ibaPDA-Interface-Reflective Memory	Extension license for an *ibaPDA* system with an Reflective Memory interface Number of connections: 64

Hardware

Order no.	Product name	Description
11.112620	ibaFOB-R	Reflective Memory card from iba AG

Other documentation

Further information on the *ibaFOB-R* card can be found in the associated documentation.

3 About ibaPDA-Request-HiPAC

The interface *ibaPDA-Request-HiPAC* is suitable for the measurement data acquisition with a free symbol selection from Danieli HiPAC controllers via Ethernet (UDP/IP) or Reflective Memory. The measurement data is actively sent from the controller to *ibaPDA*. This requires the integration of request blocks into the HiPAC controller. These request blocks are used to cyclically send the current values of the variables selected by the user within *ibaPDA* to *ibaPDA* for recording.

In *ibaPDA*, the variables to be measured are selected using a browser. This enables access to all variables available in the controller. The values of the variables can be sent to *ibaPDA* via the following data paths:

- UDP connection via *ibaPDA-Interface-Generic-UDP*
- Reflective Memory, e.g. via *ibaFOB-R* or PCIe-5565PIORC (Abaco Systems);
  License *ibaPDA-Interface-Reflective-Memory* required

ibaPDA-Request-HiPAC supports HiPAC systems, which are based on a Core i7 CPU with the VxWorks operating system and the Codesys V2.3 or V3 runtime.

The ibaHiPACRequest library must be added to the project in the HiPAC controller. This library requires other libraries on its side that should be available if the HiPAC runtime is up to date. If libraries are missing, please contact Danieli.

The ibaHiPACRequest library contains the “agent” for the request function, which is divided into the following function blocks:

- Management block IBA_REQ_A
- Signal data block IBA_REQ_B

The management block can also be inserted in a (slow) task with low priority. It communicates with *ibaPDA* via the control path (Ethernet TCP/IP) and checks the list of variables.

The signal data block is assigned to a faster task with a higher priority. It collects the data and sends it to *ibaPDA* with each access on the data path.

You can find the libraries as archive files on the "iba Software & Manuals" data medium at
`\04_Libraries_and_Examples\10_Libraries\05_HIPAC\`

Note

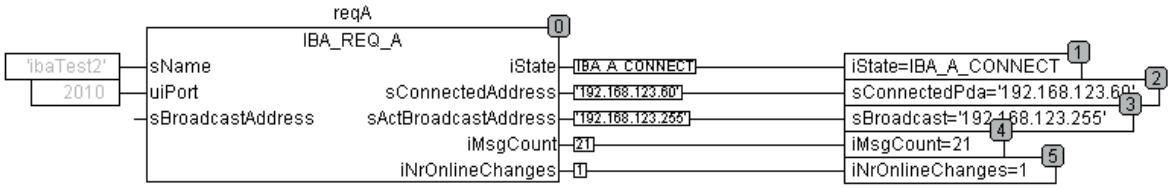
The interface *ibaPDA-Interface-Codesys-Xplorer* can also be used to establish a connection with an HiPAC controller. However, no request function blocks are used and the measured values are only transmitted via Ethernet TCP/IP and not with cycle accuracy.

3.1 Request blocks

The request blocks are used to initialize and control communication between the HiPAC controller and *ibaPDA*.

A request block set always consists of a management block and a signal data block. The same signal data blocks are used for the connection via UDP and Reflective Memory. The blocks are part of the ibaHiPACRequest library.

Management block IBA_REQ_A

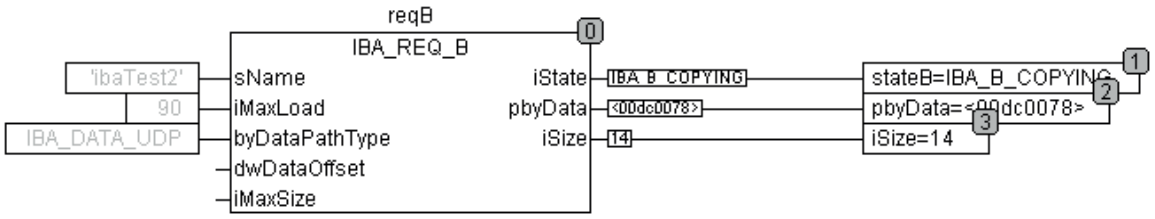


Name	Type	In/Out	Description
sName	STRING(20)	IN	Name of the function block The same name must be used for the corresponding IBA_REQ_B function block. The name must be unique across all HiPAC CPUs that are connected to the same *ibaPDA*.
uiPort	UINT	IN	Port number of the TCP socket for listening
sBroadcastAddress	STRING(20)	IN	Optional IP address to which the function block sends the broadcast telegrams If this input parameter remains empty, the block automatically attempts to determine the address by calling up its own IP address and assuming a subnet mask of 255.255.255.0.
iState	IBA_STATE_A	OUT	Status of the function block
sConnectedAddress	STRING(20)	OUT	IP address of the connected *ibaPDA* system
sActBroadcastAddress	STRING(20)	OUT	IP address used for broadcast telegrams
iMsgCount	INT	OUT	Telegram counter for telegrams sent to *ibaPDA*
iNrOnlineChanges	INT	OUT	Number of online changes detected by the function block

The block IBA_REQ_A can assume the following states (IBA_STATE_A):

Status	Description
IBA_REQ_A_INIT	Initial state before the block has registered with its name
IBA_REQ_A_OPEN	Block attempts to open a socket to listen to port *uiport*.
IBA_REQ_A_WAIT_FOR_CONNECT	Socket for listening is opened and block waits for incoming connection from *ibaPDA*.
IBA_REQ_A_CONNECT	Connection with *ibaPDA* is established and telegrams are exchanged.

Signal data block IBA_REQ_B



Name	Type	In/Out	Description
sName	STRING(20)	IN	Name of the function block The same name must be used for the corresponding IBA_REQ_A function block.
iMaxLoad		INTIN	Maximum CPU load in % If the CPU load exceeds this value, the copying process is stopped.
byDataPathType	IBA_DATA_PATH_TYPE	IN	Type of the data path used. There are 2 possible options: ■ IBA_DATA_RM: Reflective Memory. The offset and buffer size that are reserved on the reflective memory card for request to this CPU are automatically determined by functions of the RFMPDA library. ■ IBA_DATA_UDP: UDP connection. The destination address and port number are automatically read out by the ibaPDA instance, which is connected to the A module to start the acquisition.
dwDataOffset	DWORD	IN	Optional additional offset within the data path buffer. This is only needed if several B blocks on the same CPU write to the same data path.

Name	Type	In/Out	Description
iMaxSize	INT	IN	Maximum size that the block may write on the data path Set to 0 if the entire data path may be used. If several B blocks on the same CPU write to the same data path, then the maximum permissible size is entered here that the block may occupy on the data path.
iState	IBA_STATE_B	OUT	Status of the function block
pbyData	POINTER TO BYTE	OUT	Pointer to the data buffer
iSize	INT	OUT	Current size of the data in the buffer. Valid if *iState* = IBA_B_COPYING.

The signal data block can assume the following states (IBA_STATE_B):

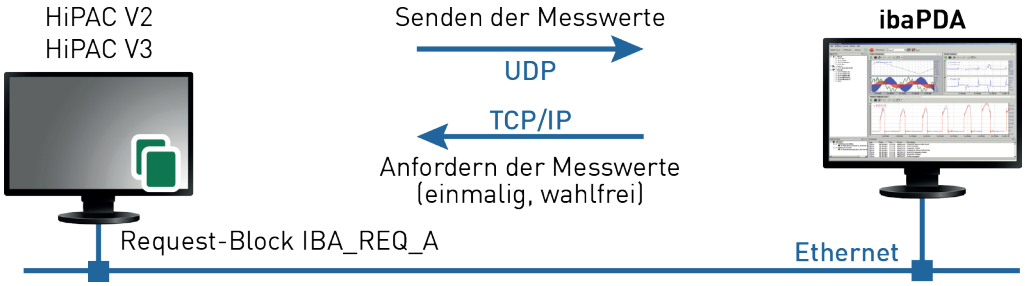
Status	Description
IBA_B_INIT	Initial state. Search for the block of the same name IBA_REQ_A.
IBA_B_NO_DATA_PATH	Connected to the A block, but no data path available.
IBA_B_READY	Connected to the A block and data path found. The variable list is empty.
IBA_B_VALIDATE	A new variable list is validated.
IBA_B_COPYING	Copy data for the variable list.
IBA_B_OVERLOAD	An overload of the controller was detected during validation or copying. Copying has been stopped.
IBA_B_ONLINECHANGE	An online change has occurred; wait for the A block to respond.

4 Request HiPAC via UDP

4.1 System integration via data path UDP

The measurement data is transmitted to *ibaPDA* via UDP. Prerequisite in *ibaPDA* is the license for the communication interface *ibaPDA-Interface-Generic-UDP*.

You need an Ethernet connection via standard network cards.



An additional prerequisite is the the ibaHiPACRequest library in the HiPAC controller.

4.2 Configuration and engineering of the the HiPAC controller

Add the ibaHiPACRequest library from the `04_Libraries_and_Examples\10_Libraries\05_HiPAC` directory of the "iba Software & Manuals" data medium to your project.

Create an instance of a management block IBA_REQ_A and a signal data module IBA_REQ_B.

The management and signal data blocks can be in the same program or in separate programs.

4.3 Configuration in ibaPDA

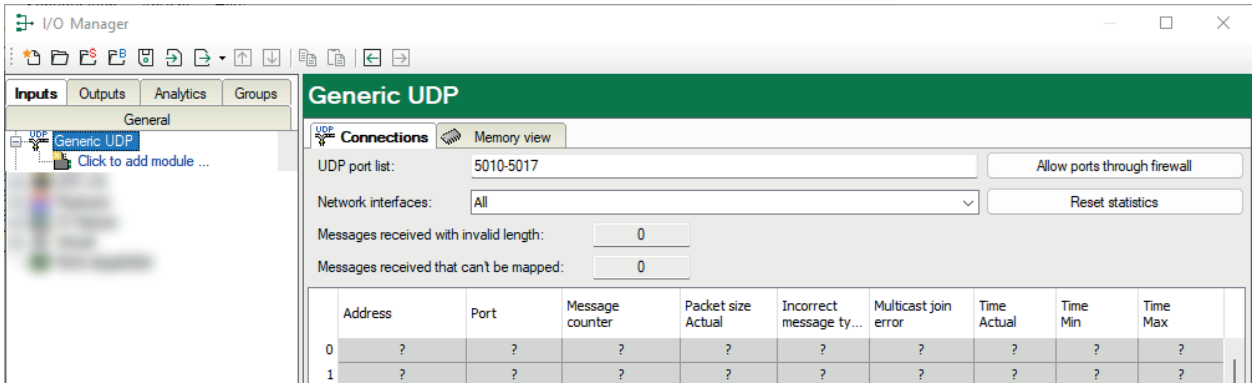
The configuration is done in the I/O Manager of *ibaPDA*. First set up the connection from *ibaPDA* to the HiPAC controller via *ibaPDA-Interface-Generic-UDP*.

Once the connection has been set up, add a HiPAC request module accordingly, see chapter ↗ *Adding a module*, page 16.

The configuration of the signals and selection in the symbol browser is described in chapter ↗ *Selecting symbols*, page 20.

4.3.1 Setting up the connection

A prerequisite for the use of UDP as a data path is the interface *ibaPDA-Interface-Generic-UDP*. If all system requirements are met, the *Generic UDP* interface will be displayed in the interface tree. HiPAC-Request is a module of this interface.



The interface provides the following functions and configuration options:

UDP Port-Liste
Ports on which *ibaPDA* waits for incoming UDP telegrams.

You can enter the ports as port range, as list of individual ports or as a combination of both. If you enter a range, please separate it by a hyphen. Do not separate successive port numbers by commas. The default setting is the range 5010-5017. The port number must be identical in the controller (see *Configuring the controller*, in the *ibaPDA-Interface-Generic-UDP* documentation).

Network interfaces
Using this drop-down list, you can select which network adapters on your computer are used for this interface. The sockets will be opened for communication only on the selected network adapters. In case a network adapter has multiple IP addresses configured, a socket will be opened for all of these IP addresses. At least one network adapter should be selected to get the interface configuration validated. If you select *None*, an error message will be displayed when validating the I/O configuration. By default, the option *All* is selected.

<Allow ports through firewall>
When installing *ibaPDA*, the standard port numbers of the protocols used are automatically entered in the firewall. If the port number is changed or if the interface was subsequently enabled, this port has to be enabled in the firewall here by clicking on this button.

Other documentation

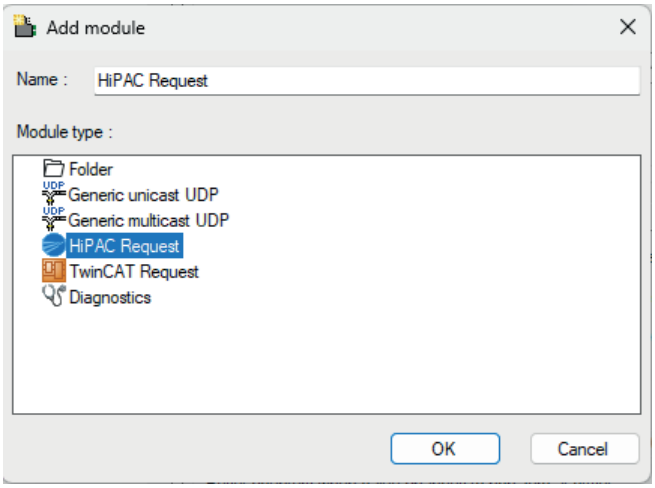
Further information on the general settings of *ibaPDA-interface-Generic-UDP* interface, e.g. on the error counters and the connection table, can be found in the associated documentation.

4.3.2 HiPAC request module

4.3.2.1 Adding a module

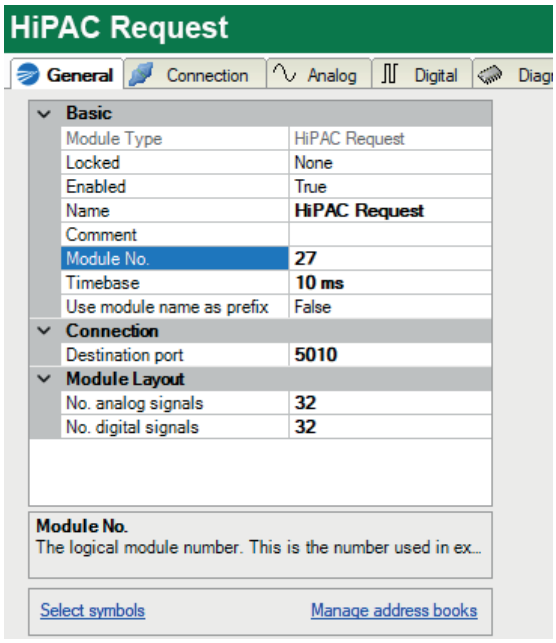
To add a HiPAC Requst module to the *Generic UDP* interface, proceed as follows:

1. Add a module by clicking on the blue *Click to add module* command below the interface.
2. Select the module type *HiPAC Request*, select a name if required and click <OK>.



4.3.2.2 General module settings

You can configure the following module settings in the *General* tab.



Basic settings

Module Type (information only)
Indicates the type of the current module.

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

Enabled
Enable the module to record signals.

Name
You can enter a name for the module here.

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

Module No.
This internal reference number of the module determines the order of the modules in the signal tree of *ibaPDA* client and *ibaAnalyzer*.

Timebase
All signals of the module are sampled on this timebase.

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

Module Layout

No. analog signals
Definition of the number of analog signals for this module (max. 1024)

No. digital signals
Definition of the number of digital signals for this module (max. 1024)

Connection

Destination port
Port used by the controller to send data via this module or this connection to ibaPDA. Possible values: 5010 - 5017, default value = 5010

“Select symbols” and “Manage address books” link

For information on the signal configuration, see ↗ *Selecting symbols*, page 20.

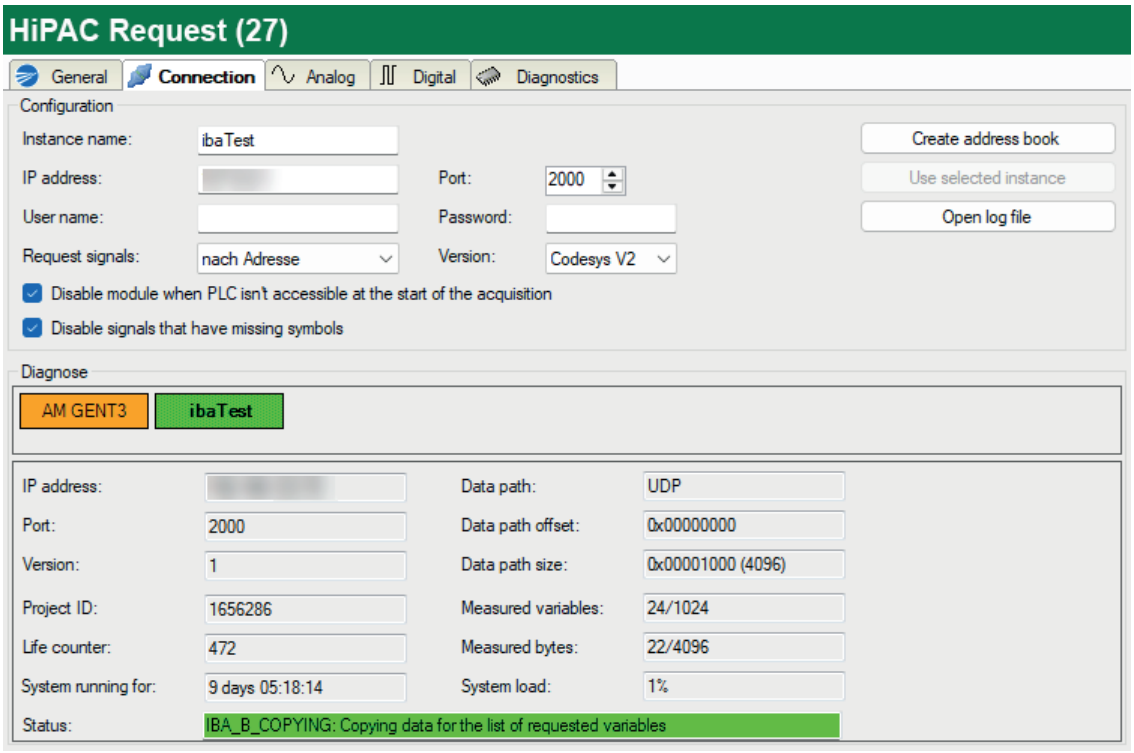
If you click on the *Manage address books* link, the *Address books* node opens in the general settings of *ibaPDA*.

Other documentation

Further information on the configuration of address books can be found in the *ibaPDA* documentation.

4.3.2.3 Configuration of the connection

You will find the following settings and information in the *Connection* tab:



Configuration

Instance name
Enter the name of the corresponding instance of the management function block (IBA_REQ_A) in the HiPAC controller (corresponds to the input parameter sName of the block).

IP address
Enter the IP address of the HiPAC controller here.

Port
Set the port number of the corresponding instance of the management function block (IBA_REQ_A) in the HiPAC controller here (corresponds to the input parameter uiPort of the block).

Tip

If *ibaPDA* is already connected to active HiPAC controllers, you can have the parameters *Instance name*, *IP address* and *port number* automatically adopted by double-clicking on a colored block in the diagnostics area. Alternatively, you can click on the <Use selected instance> button if you have marked a block.

User name/Password
The user name and password needed to access the PLC according to the PLC configuration.

Request signals
Select whether the signals should be requested by address or by name.

The request by address is usually faster, because the PLC does not have to resolve the name. This is done by *ibaPDA*. If, however, the address book in *ibaPDA* is no longer current, then the addresses may be incorrect. *ibaPDA* checks whether the address book is still current when starting the acquisition.

Requesting by address is only possible with HiPAC V2, not with HiPAC V3, since the address book with V3 does not contain any addresses.

Issue
Specification of the Codesys version.

<Create address book>
Once you have configured the instance (name, IP address and port number), you can click this button to create the address book for the signals (symbols) to be measured. All symbols that are part of the symbol configuration in the HiPAC controller are then available to choose from in the Codesys symbol browser of *ibaPDA*.

<Use selected instance>
Clicking on this button will adopt the instance name, IP address and port number of a marked colored block in the diagnostics area as the configuration parameters for the module.

<Open log file>
The log file entries created during the establishment of the connection are shown in the standard editor.

Disable module when PLC is not accessible at the start of the acquisition
If this option is activated, the recording is started even if no connection to the PLC can be established. The module is disabled. During the measurement, *ibaPDA* attempts to reconnect to the PLC. If successful, the acquisition is restarted. If this option is not activated, the recording is not started if no connection to the PLC is possible.

Disable signals that have missing symbols
If the symbol configuration has changed, the module may contain a symbol that is no longer available. If *ibaPDA* then tries to read the data for this variable, the PLC will return an error. If the option “Disable signals that have missing symbols” is enabled, *ibaPDA* ignores this signal and starts the acquisition without this signal. If this option is not enabled, the acquisition is not started.

Diagnostics

In the diagnostics area, all “A” blocks in the form of colored blocks are displayed from which *ibaPDA* receives broadcast telegrams or that are created in the I/O configuration.

The color of a block provides information about its status:

Appearance	Configured	Broadcast reception	TCP connection OK	Data path OK
Orange		X		
Red	X			
Flahing red	X	X		
Yellow	X	X	X	

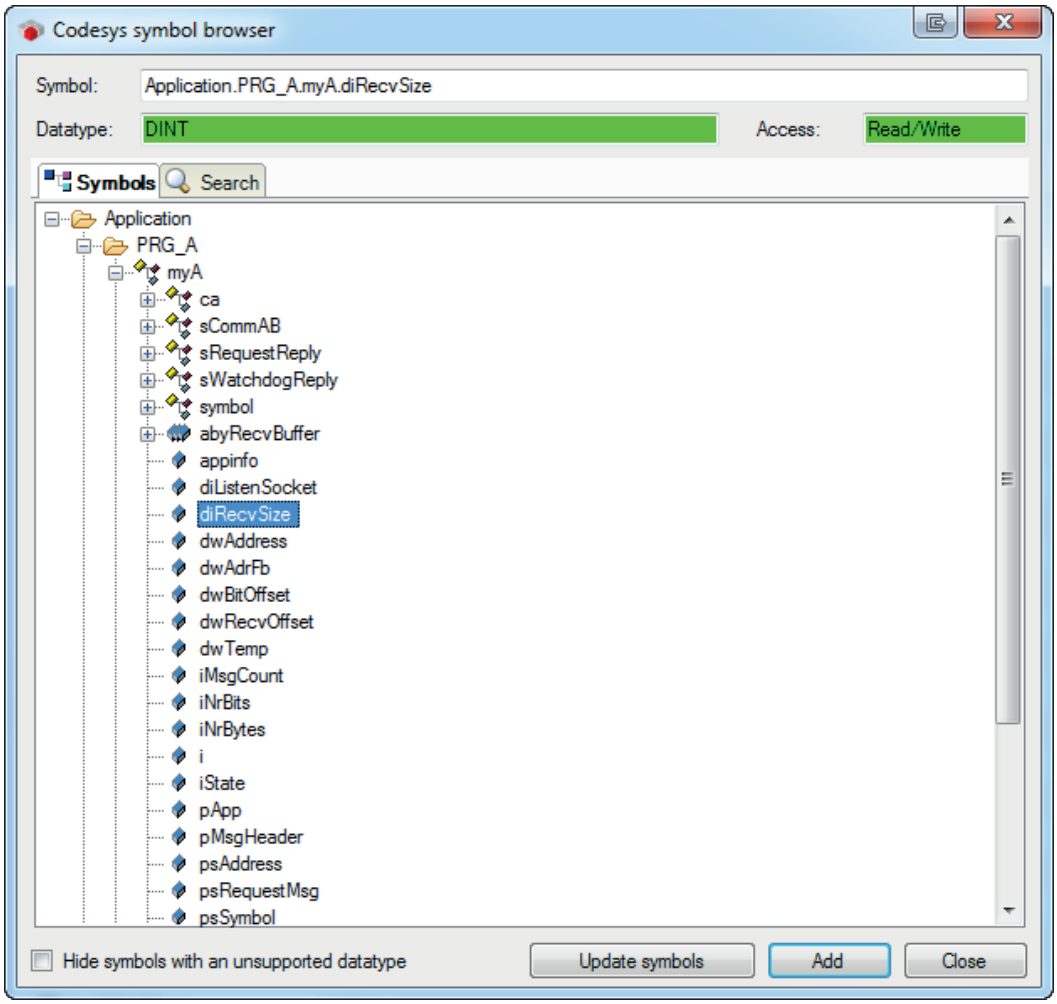
Appearance	Configured	Broadcast reception	TCP connection OK	Data path OK
Yellow with exclamation mark	X		X	
Green	X	X	X	X
Green with exclamation mark	X		X	X

If you click on a block, more information about this instance will be displayed below it. You can also double-click on a block to adopt the parameters *Instance name*, *IP address* and *port number* in the configuration area. Alternatively, you can click on the <Use selected instance> button if you have marked a block.

4.3.3 Selecting symbols

Once the connection to the controller has been successfully established and the address book has been generated, the symbols are loaded and can be selected in the symbol browser.

Open the symbol browser by clicking on the *Select symbols* link in the *General* tab of the HiPAC request module. Since the HiPAC controllers are based on Codesys, the Codesys symbol browser opens as it is also used with the Codesys-Xplorer interface.



In the *Symbols* tab, you can select individual or multiple symbols in the tree. Clicking on <Add> inserts the symbols in the corresponding signal tree (analog or digital).

If you have selected a single symbol, the next symbol will be selected after you have clicked on <Add>. This allows you to add consecutive symbols by clicking <Add> several times.

Double-click on the symbol to transfer it to the signal table.

The symbols are reloaded from the PLC with <Update symbols>.

You can search for symbols by name in the *Search* tab. The handling and selection in the search results tree is identical to the selection in the symbol tree.

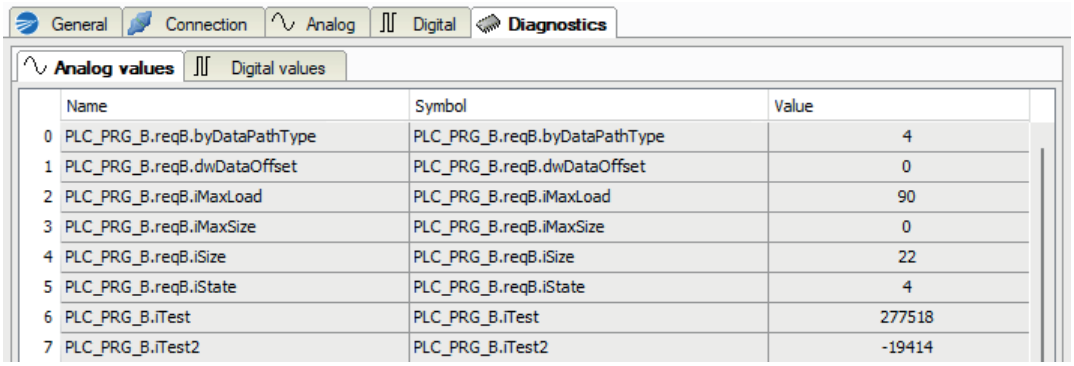
You can also open the symbol browser in the signal tables, *Analog* and *Digital* tabs, by clicking on the browser button (<...>) in the *Symbol* column.

Note

The module *TDC TCP/UDP Generic* supports the acquisition and processing of strings as text signals. Therefore, you can select the data type STRING[32] in the *Analog* tab. In order to convert a text signal or to split it up into several text signals use the *text splitter* module under the *Virtual* interface.

4.3.4 Diagnostics

Once you have selected the symbols for the *Analog* and *Digital* signal table and accepted the I/O configuration, you can see the actual values of the requested symbols in the *Diagnostics* tab of the module.

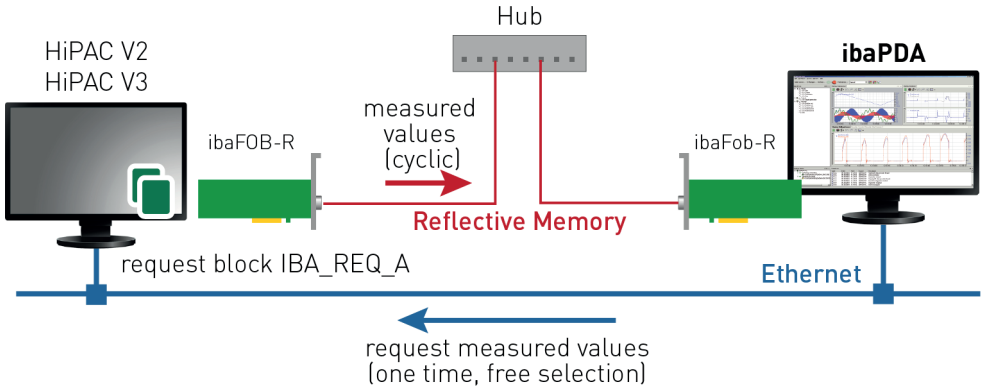


	Name	Symbol	Value
0	PLC_PRG_B.reqB.byDataPathType	PLC_PRG_B.reqB.byDataPathType	4
1	PLC_PRG_B.reqB.dwDataOffset	PLC_PRG_B.reqB.dwDataOffset	0
2	PLC_PRG_B.reqB.iMaxLoad	PLC_PRG_B.reqB.iMaxLoad	90
3	PLC_PRG_B.reqB.iMaxSize	PLC_PRG_B.reqB.iMaxSize	0
4	PLC_PRG_B.reqB.iSize	PLC_PRG_B.reqB.iSize	22
5	PLC_PRG_B.reqB.iState	PLC_PRG_B.reqB.iState	4
6	PLC_PRG_B.iTest	PLC_PRG_B.iTest	277518
7	PLC_PRG_B.iTest2	PLC_PRG_B.iTest2	-19414

5 Request HiPAC via reflective memory

5.1 System integration with the Reflective Memory data path

The measurement data is transmitted via Reflective Memory either directly to *ibaPDA* or via a Reflective Memory hub.



You need the following connections:

- Ethernet connection between *ibaPDA* and the HiPAC PLC
- FO link via Reflective Memory card in the *ibaPDA* computer and in the HiPAC computer (e.g. *ibaFOB-R* or PCIe-5565PIORC from ABACO)

An additional prerequisite is the the ibaHiPACRequest library in the HiPAC controller.

5.2 Configuration and engineering of the the HiPAC controller

On the HiPAC side, the following configuration and engineering steps must be carried out in HiPAC:

Hardware engineering
Integrating the reflective memory card into the device configuration.

Software engineering
Add the ibaHiPACRequest library from the `04_Libraries_and_Examples\10_Libraries\05_HiPAC` directory of the "iba Software & Manuals" data medium to your project.

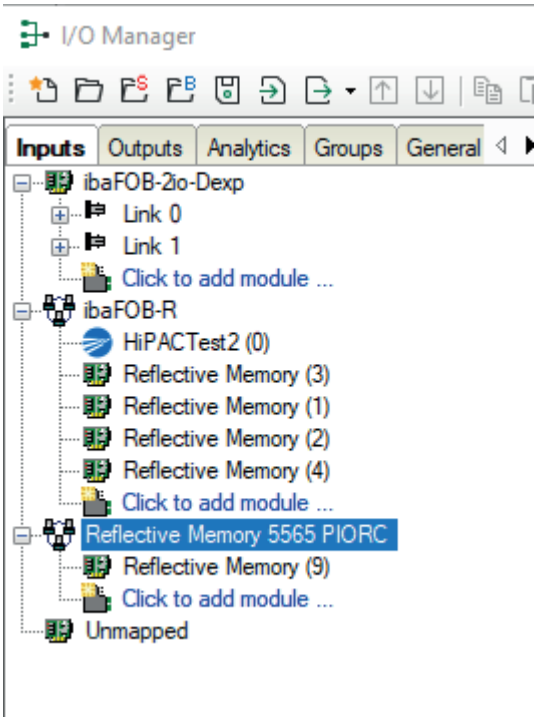
Create an instance of a management block IBA_REQ_A and a signal data module IBA_REQ_B.

The management and signal data blocks can be in the same program or in separate programs.

5.3 Configuration in ibaPDA

The configuration is done in the I/O Manager of *ibaPDA*. First set up the connection from *ibaPDA* to the HiPAC controller via the *ibaFOB-R* interface or Reflective Memory interface.

The interface is only visible if the Reflective Memory license is enabled **and** a Reflective Memory interface card is installed in the PC.



Once the connection has been set up, add a HiPAC request module accordingly, see chapter ↗ *Adding a module*, page 26.

The configuration of the signals and selection in the symbol browser is described in chapter ↗ *Selecting symbols*, page 20.

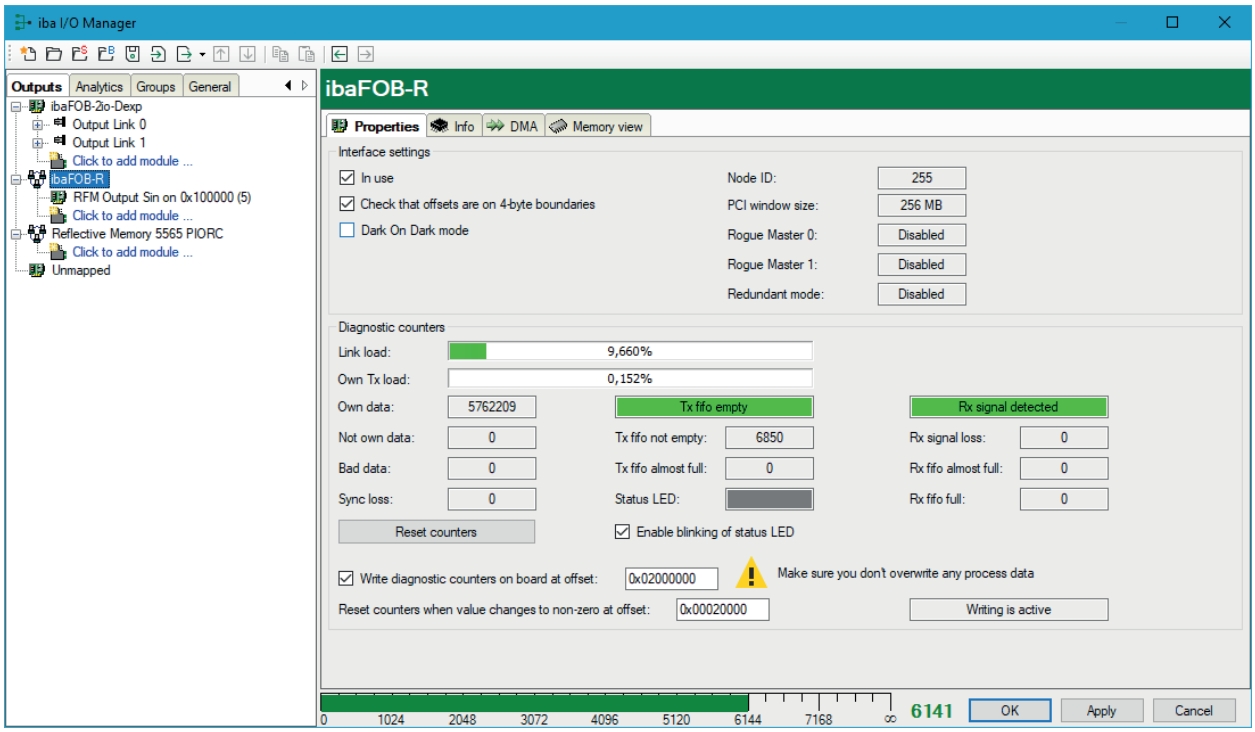
5.3.1 Setting up the connection

The interface settings for the *ibaFOB-R* card are described below.

Note

Detailed information on the *ibaPDA-Interface-Reflective Memory* interface and on configuring the interface with older Reflective Memory cards can be found in the associated documentation.

5.3.1.1 Properties ibaFOB-R



Interface settings

"In use" check box
If the *ibaFOB-R* board should be used by *ibaPDA*, you must check this box.

This is, for example, necessary if *ibaPDA* and *ibaLogic* are active in a hybrid configuration on the PC, with each application having to use its own boards. One board must only be used by one application.

Check that offsets are on 4-byte limits
Usually, the checking of the 4-byte limits is selected by default in order to guarantee a data addressing without gaps. Data of 4-byte size (DINT, DWORD, FLOAT) must always be addressed on a 4-byte offset, relative to the start address. If not, an error message will be generated and the configuration is not valid.

When addressing data, otherwise than on 4-byte limits, be sure to disable this option in order to suppress error messages.

Dark on Dark mode
This checkbox enables the dark-on-dark feature on the *ibaFOB-R* board.

If enabled, the board's transmitter will be turned OFF if the board's receiver does not detect a signal or if the receiver detects invalid data patterns. The dark-on-dark feature is especially useful in hub configurations.

Changes in this setting have immediate effect unless acquisition is running. If the acquisition is running at this time, the changing applies only after pressing <OK> or <Apply>. The acquisition is then stopped and restarted.

Node ID
This is the node ID as set on the *ibaFOB-R* board in the *ibaPDA* computer. It is for display only and cannot be altered here.

PCI window size
This value shows the selected window size for the reflective memory accesses. The default as displayed here, is the full installed memory size. The reduced memory window size choices are 64 MB, 16 MB or 2 MB. The value is for display only and cannot be altered here.

Rogue Master 0
This value indicates whether the Rogue Master 0 functionality is enabled on the board. Rogue Master functionality removes rogue packages from the network. The value is for display only and cannot be altered here.

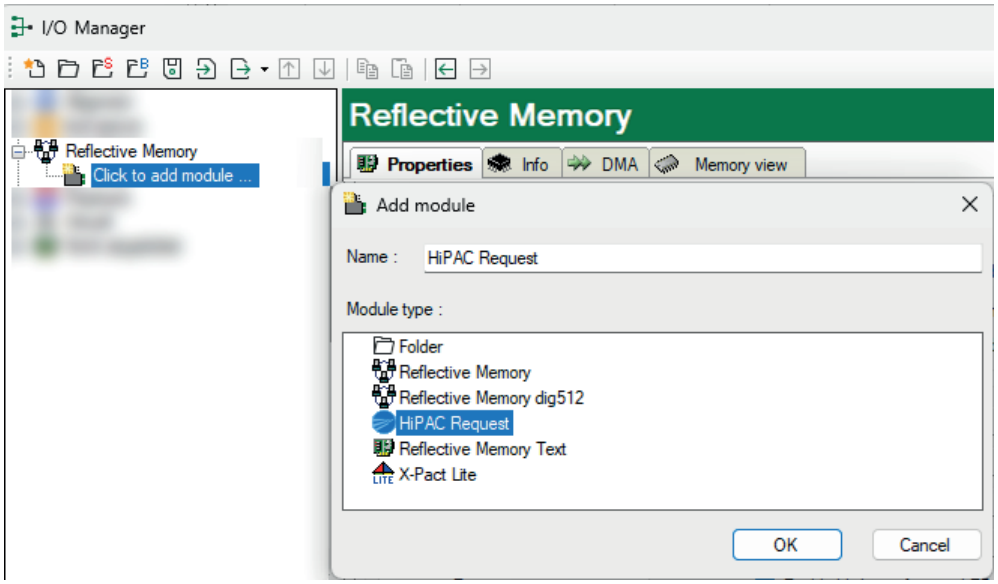
Rogue Master 1
This value indicates whether the Rogue Master 1 functionality is enabled on the board. Rogue Master 0 and 1 if both enabled, cross check each other. The value is for display only and cannot be altered here.

Redundant mode
This value indicates whether the redundant transfer mode is enabled. In the redundant transfer mode, each generated packet transfers twice, the receiving circuitry evaluates both packages. The value is for display only and cannot be altered here.

5.3.2 HiPAC request module

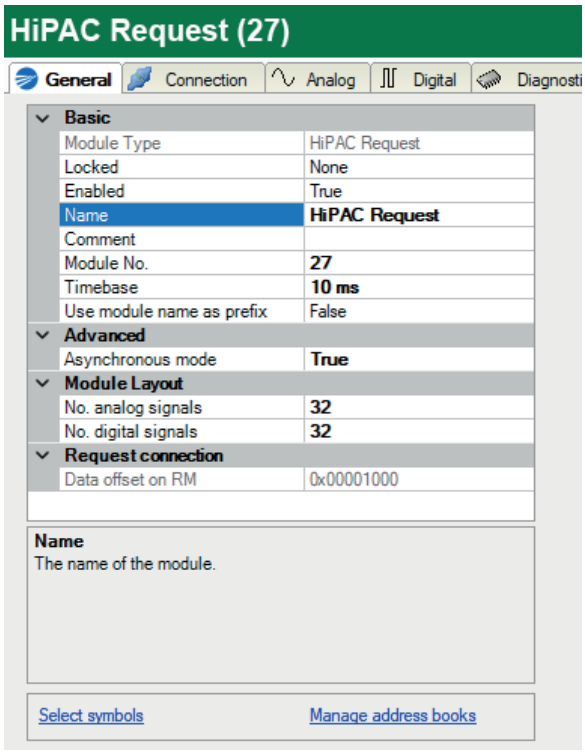
5.3.2.1 Adding a module

Add a HiPAC request module in the I/O Manager by clicking under the *Reflective Memory* or *iba-FOB-R* interface. Select the desired module type and click <OK>.



5.3.2.2 General module settings

In the *General* tab, you can configure the following settings for the HiPAC Request module with reflective memory.



Basic, Module Layout

The basic settings and the module layout under *Basic* are identical to UDP, see chapter ↗ *General module settings*, page 16.

Advanced

Asynchronous mode
If you enable asynchronous mode (True), then the data will be copied from the card memory outside the interrupt service routine (ISR). This mode can be used to measure large amounts of data with a slower acquisition rate than the interrupt.

If you disable the asynchronous mode, then *ibaPDA* attempts to copy the data within the ISR. Data loss occurs if the ISR does not have enough time for the amount of data.

Request connection

Data offset on RM
This is the initial offset in the reflective memory where the data for this module is written.

The value is only for informational purposes and cannot be changed.

5.3.2.3 Configuration of the connection

The connection settings are to be made as with UDP. See chapter ↗ *Configuration of the connection*, page 18.

5.3.3 Selecting symbols

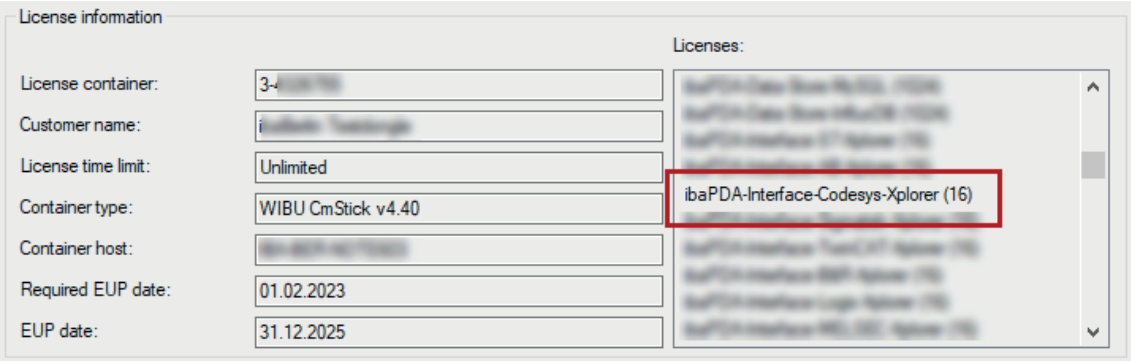
The symbols to be measured are selected as with UDP via the Codesys symbol browser. See chapter ↗ *Selecting symbols*, page 20 .

6 Diagnostics

6.1 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-HiPAC* 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.



6.2 Log files

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

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

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

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

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

Examples:

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

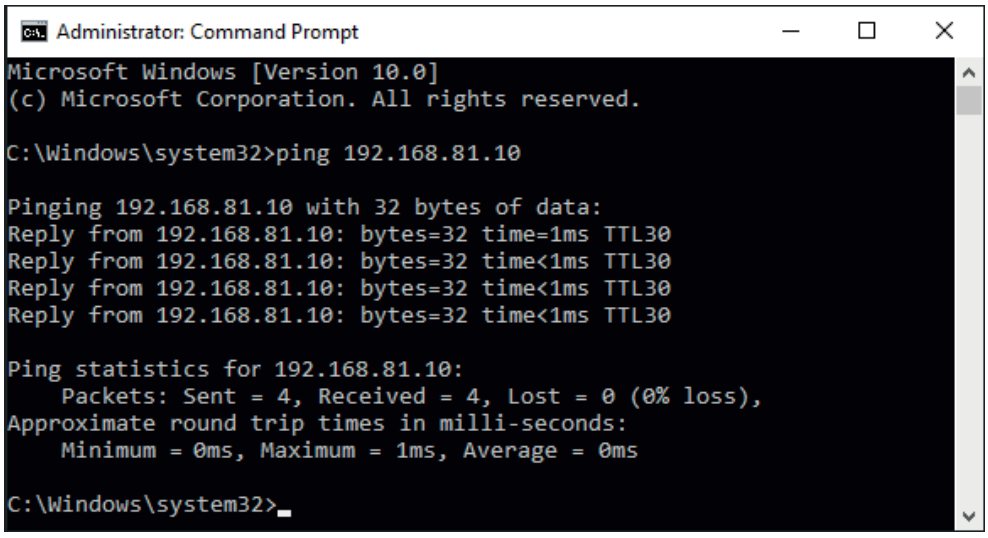
6.3 Connection diagnostics with PING

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

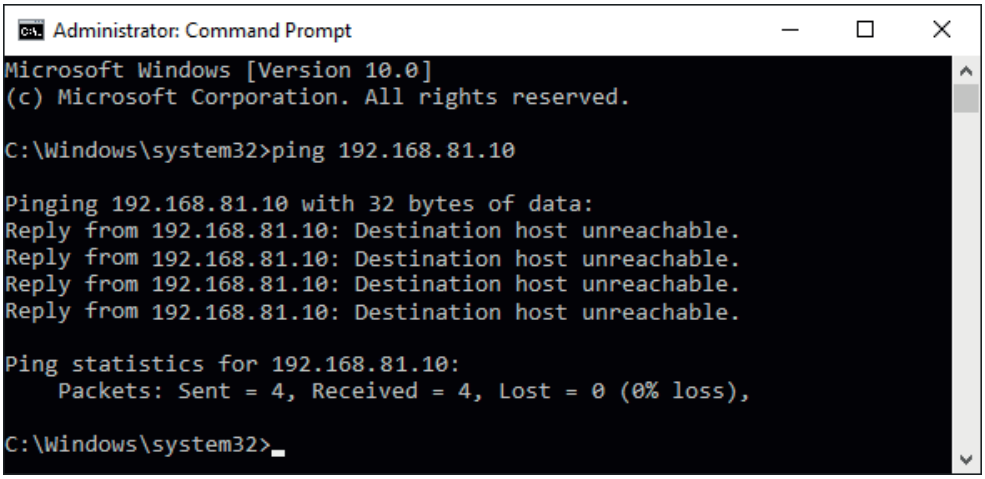
1. Open a Windows command prompt.

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.



7 Support and contact

Support

Phone: +49 911 97282-14

Email: support@iba-ag.com

Note

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

Contact

Headquarters

iba AG
Koenigswarterstrasse 44
90762 Fuerth
Germany

Phone: +49 911 97282-0

Email: iba@iba-ag.com

Mailing address

iba AG
Postbox 1828
D-90708 Fuerth, Germany

Delivery address

iba AG
Gebhardtstrasse 10
90762 Fuerth, Germany

Regional and Worldwide

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

www.iba-ag.com