



# ibaPDA-Request-HiPAC

Request Data Interface to Danieli HiPAC Systems

Manual  
Issue 1.1

Measurement Systems for Industry and Energy

[www.iba-ag.com](http://www.iba-ag.com)

---

## Manufacturer

iba AG  
Koenigswarterstr. 44  
90762 Fuerth  
Germany

## Contacts

Main office +49 911 97282-0  
Fax +49 911 97282-33  
Support +49 911 97282-14  
Engineering +49 911 97282-13  
E-mail [iba@iba-ag.com](mailto:iba@iba-ag.com)  
Web [www.iba-ag.com](http://www.iba-ag.com)

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

© iba AG 2020, All rights reserved.

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

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

Version	Date	Revision - Chapter / Page	Author	Version SW
1.1	02-2020	Supported controller	rm/ip	6.33

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

## Content

<b>1</b>	<b>About this Manual .....</b>	<b>5</b>
1.1	Target group and previous knowledge .....	5
1.2	Notations .....	6
1.3	Used symbols.....	7
<b>2</b>	<b>System requirements .....</b>	<b>8</b>
<b>3</b>	<b>ibaPDA-Request-HiPAC.....</b>	<b>10</b>
3.1	General Information .....	10
3.2	Request blocks.....	11
<b>4</b>	<b>Request HiPAC via UDP.....</b>	<b>14</b>
4.1	System integration with data path UDP.....	14
4.2	Configuration and engineering of the HiPAC controller.....	14
4.3	Configuration in ibaPDA.....	15
4.3.1	Setting up the connection.....	15
4.3.2	HiPAC request module.....	16
4.3.2.1	Add module .....	16
4.3.2.2	General Settings.....	17
4.3.2.3	Configuration of the connection.....	18
4.3.3	Selecting symbols .....	21
4.3.4	Diagnosis.....	22
<b>5</b>	<b>Request HiPAC via reflective memory.....</b>	<b>23</b>
5.1	System integration with the Reflective Memory data path .....	23
5.2	Configuration and engineering of the HiPAC controller.....	23
5.3	Configuration in ibaPDA.....	24
5.3.1	Setting up the connection.....	25
5.3.1.1	Properties .....	25
5.3.1.2	DMA.....	28
5.3.2	HiPAC request module.....	29
5.3.2.1	Add module .....	29
5.3.2.2	General Settings.....	29
5.3.2.3	Configuration of the connection.....	30
5.3.3	Selecting symbols .....	30

---

- 6    **Diagnosis** ..... **31****
- 6.1    Checking the license ..... 31
- 6.2    Log files..... 32
- 6.3    Connection diagnostics with PING..... 33
- 7    **Support and contact**..... **34****

# 1 About this Manual

This document 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 access to variables when recording data from Danieli HiPAC controllers. This manual only shows the extensions and differences. Refer to the manual from *ibaPDA* for all other functions and operating options.

## 1.1 Target group and previous knowledge

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

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

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

## 1.3 Used symbols

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

---

### Danger!



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

- 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

- *ibaPDA* V6.33 or higher
- Additional license for *ibaPDA-Request-HiPAC*
- If UDP is used as a data path:
  - Additional license *ibaPDA-Interface-Generic-UDP*
- If Reflective Memory is used as a data path:
  - Additional license *ibaPDA-Interface-Reflective-Memory*
  - Fibre Optic card of the type PCIe-5565PIORC in the *ibaPDA* computer
- All non VME x86 based HiPAC controllers are supported
- Danieli HiPAC controller HiPAC V2 or HiPAC V3
- Ethernet connection to the controller
- Libraries with iba Request blocks
  - *ibaHiPACRequest.lib* for connection via generic UDP and reflective memory (Codesys V2)
  - *ibaHiPACRequestV3.lib* for connection via generic UDP and reflective memory (Codesys V3)

### System specification

- Maximum 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 for a generic UDP interface Number of connections: 64
31.001220	ibaPDA-Interface-Reflective Memory	Extension license for an ibaPDA system for a Reflective Memory interface Number of connections: 64

Table 1: Available licenses



**Hardware**

Order no.	Product name	Description
19.114003	RTNET-PCIE-5565PIORC,	PC plug-in board for Reflective Memory

Table 2: Hardware

The Reflective memory card as well as other modules for the Reflective memory communication are manufactured and sold by Abaco Systems.

## 3 ibaPDA-Request-HiPAC

### 3.1 General Information

The interface *ibaPDA-Request-HiPAC* is suitable for the measuring data acquisition with a free symbol selection from Danieli HiPAC controllers via Ethernet (UDP/IP) or Reflective Memory. The measuring data is actively sent here from the controller to *ibaPDA*. For this purpose, it is necessary to integrate request blocks into the HiPAC controller. These request blocks serve 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 with a browser. This makes it possible to access all of the 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 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 Vx-Works 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 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 call up of the data path.

You can find the library as an archive file on the DVD “iba Software & Manuals” 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 here and the measured values are only transmitted via Ethernet TCP/IP and not exactly to the cycle.

---

### 3.2 Request blocks

The request blocks are used to initialize and control the communication between the HiPAC controller and the *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 components of the *ibaHiPACRequest* library.

#### Management block IBA\_REQ\_A

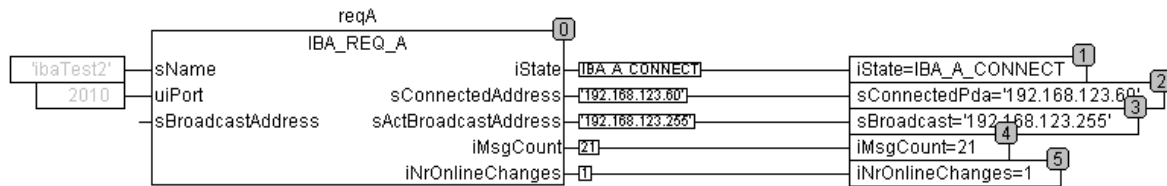


Fig. 1: Management block IBA\_REQ\_A

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 unambiguous across all HiPAC CPUs that are connected with 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, then the block automatically attempts to determine the address by calling up its own IP address and assuming a subnet mask 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 messages sent to ibaPDA
iNrOnlineChanges	INT	OUT	Number of online changes detected by the function block

The block IBA\_REQ\_A may assume the following states (IBA\_STATE\_A):

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

### Signal data block IBA\_REQ\_B

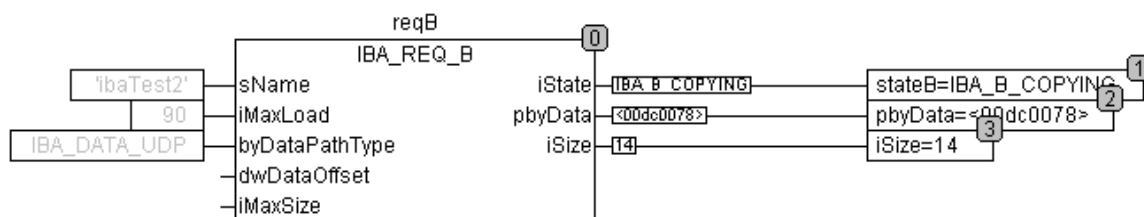


Fig. 2: Signal data block IBA\_REQ\_B

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 capacity utilization rises above this value, then the copy process is stopped.
byDataPath-Type	IBA_DATA_PATH_TYPE	IN	Type of data path used. There are 2 possible options: <ul style="list-style-type: none"> <li>■ 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.</li> <li>■ 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.</li> </ul>
dwDataOffset	DWORD	IN	Optional additional offset within the data path buffer. This is only needed if several B-blocks on the same CPU write 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 on 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 on the data buffer
iSize	INT	OUT	Current size of the data in the buffer. Valid if <i>iState</i> = IBA_B_COPYING.

The signal data block may assume the following states (IBA\_STATE\_B):

Status	Description
IBA_B_INIT	Initial state. Search for the block with 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 variables list is empty.
IBA_B_VALIDATE	A new variables list is validated.
IBA_B_COPYING	Copy data for the variables 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 A block to respond to this.

## 4 Request HiPAC via UDP

### 4.1 System integration with data path UDP

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

You need an Ethernet connection via standard network cards.

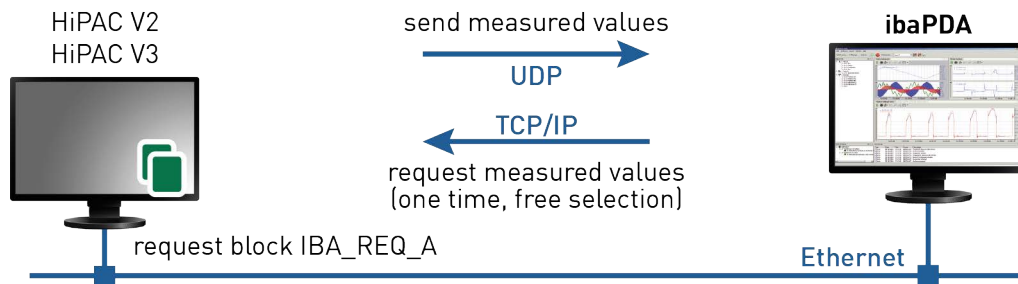


Fig. 3: Connections for control path via TCP/IP and data path via UDP

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

### 4.2 Configuration and engineering of the HiPAC controller

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

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

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

## 4.3 Configuration in ibaPDA

The configuration takes place in the I/O manager of *ibaPDA*. First establish the connection of *ibaPDA* to the HiPAC controller via *ibaPDA-Interface-Generic-UDP*.

Once the connection is established, add a HiPAC request module accordingly. See chapter [➤ Add module](#), page 16.

The configuration of the signals and the selection in the symbol browser is described in the chapter [➤ Selecting symbols](#), page 21.

### 4.3.1 Setting up the connection

The prerequisite for using 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.

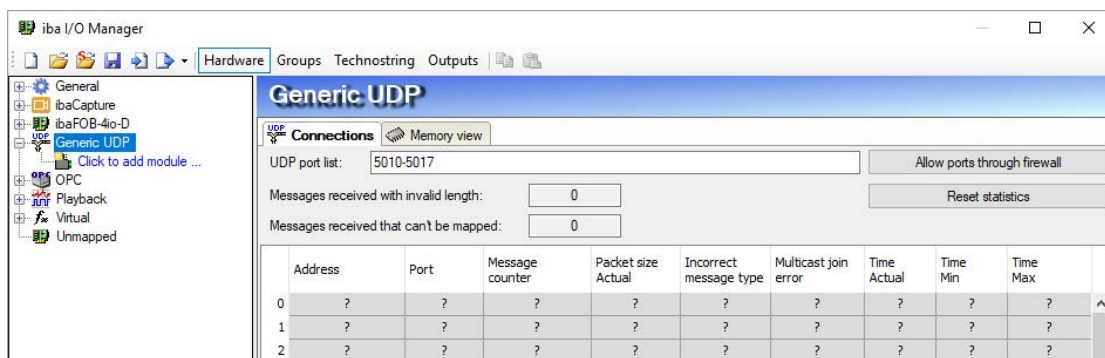


Fig. 4: "Generic UDP" interface

The interface itself has the following functions and configuration options.

#### UDP port list

Ports where *ibaPDA* waits for incoming UDP telegrams. You can enter the port numbers as a range or as an enumeration or both combined. Enter a range with a hyphen and separate non-consecutive numbers with commas. The range 5010-5017 is standard. The port number must be identical in the controller (see *Configuring the controller* and in the manual *ibaPDA-Interface-Generic-UDP*).

#### Allow ports through firewall

When installing *ibaPDA*, the default port numbers of the protocols used are automatically entered in the firewall. When the port number is changed or when the interface was activated subsequently, it is necessary to allow the ports in the firewall here by clicking on this button.

#### Counter for invalid telegrams

#### Connection table

#### Other documentation



You can find additional information about the interface *ibaPDA-Interface-Generic-UDP* in the associated manual.

## 4.3.2 HiPAC request module

### 4.3.2.1 Add module

Add a HiPAC request module in the I/O manager by clicking below the interface generic UDP. Select the desired module type and click <OK>.

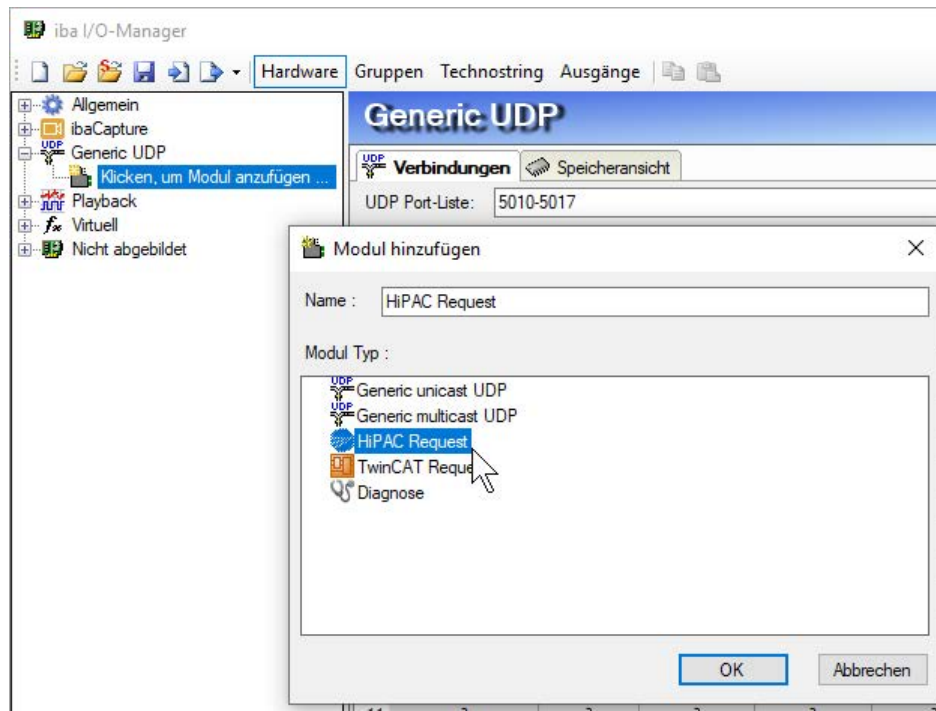


Fig. 5: Adding a module in the I/O manager



### 4.3.2.2 General Settings

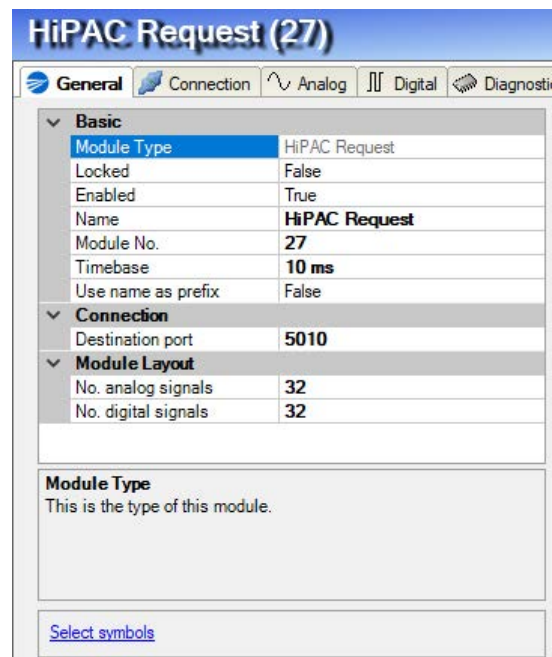


Fig. 6: Module HiPAC request, General tab for UDP connection

#### Basic settings

##### Module Type (information only)

Indicates the type of the current module.

##### Locked

A module can be locked to avoid unintentional or unauthorized changing of the module settings.

##### Enabled

Disabled modules are excluded from signal acquisition.

##### Name

The plain text name should be entered here as the module designation.

##### Module No.

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

##### Time base

All signals of the module will be sampled on this time base.

##### Use name as prefix

Puts the module name in front of the signal names.

#### 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

## Module layout

### Number of analog signals

Determination of the number of analog signals for this module (max. 1024)

### Number of digital signals

Determination of the number of digital signals for this module (max. 1024)

## 4.3.2.3 Configuration of the connection

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

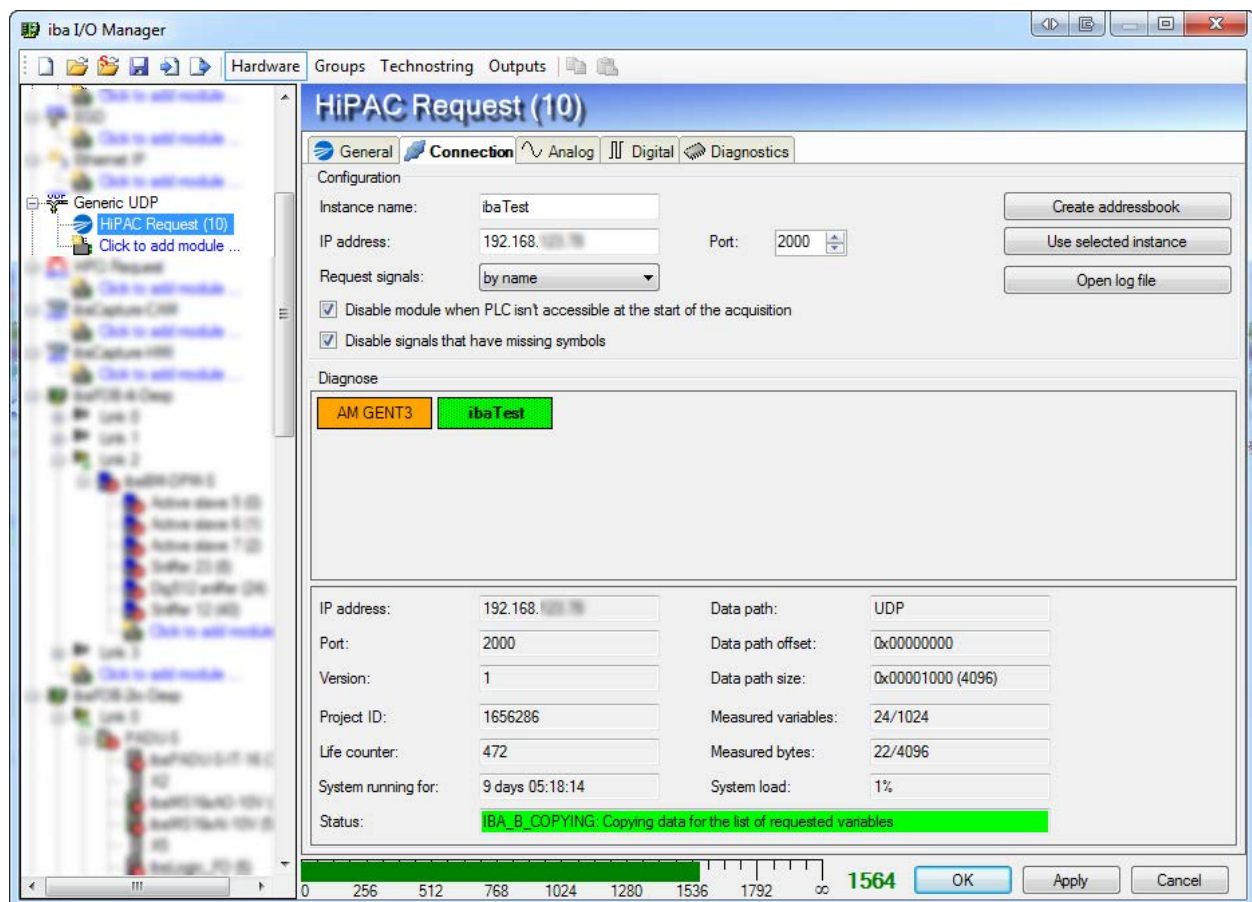


Fig. 7: Configuration of the connection of the HiPAC request module

## 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.

**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.

**<Create address book> button**

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> button**

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> button**

The log book entries generated during the connection setup are displayed in the default editor.

**Disable module when PLC is not accessible at the start of the acquisition.**

If this option is enabled, the acquisition is started, even if no connection to the PLC can be established. The module is disabled. During the acquisition, *ibaPDA* tries again to connect with the PLC. If successful, the acquisition is restarted. If this option is not enabled, the acquisition is not started, even 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 tab, 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.

**Diagnosis**

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			
Flashing red	X	X		
Yellow	X	X	X	
Yellow with exclamation point	X		X	
Green	X	X	X	X
Green with exclamation point	X		X	X

Table 3: Status indication of the management blocks (IBA\_REQ\_A) in ibaPDA

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 PLC 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 link “Select symbols” 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.

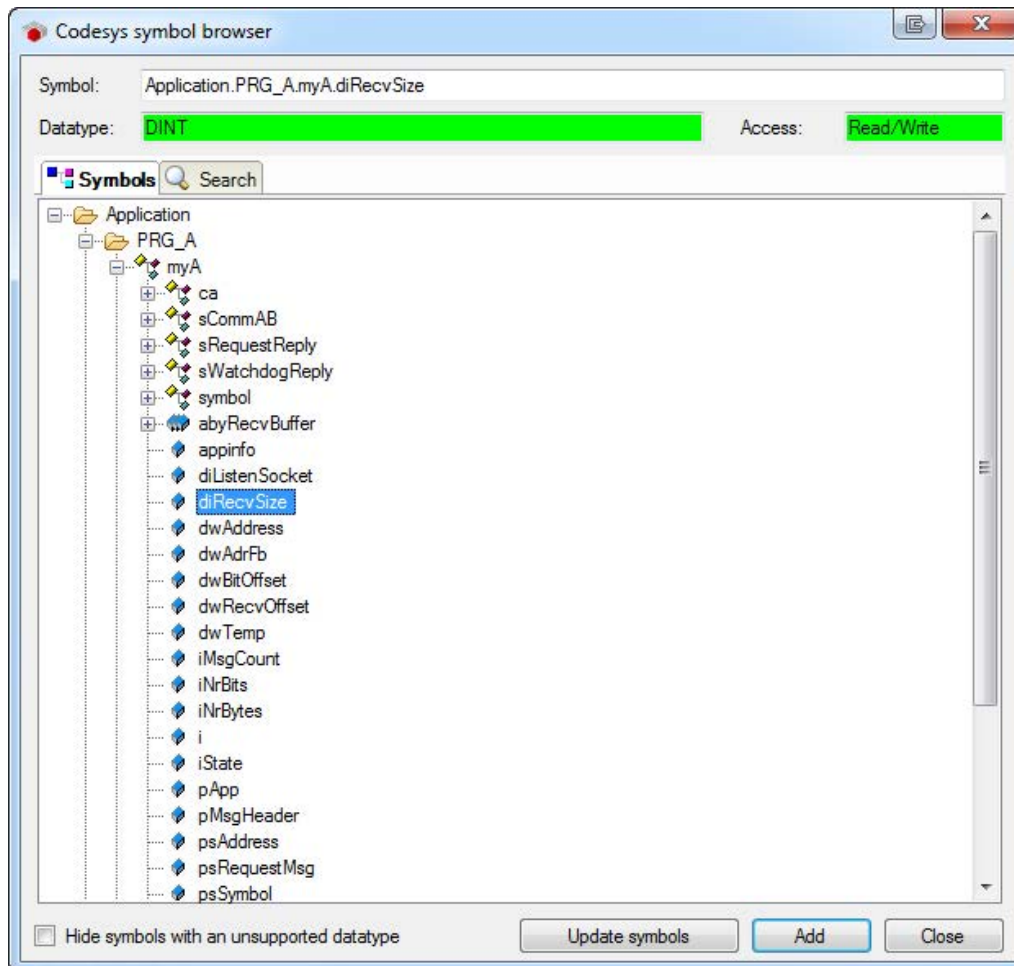


Fig. 8: Codesys symbol browser for selecting signals to be measured

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

If you have selected an individual symbol, the next symbol is selected after you have clicked on <Add>. You can add consecutive symbols by clicking on <Add> several times.

By double clicking on the symbol, this is also adopted in the symbol table.

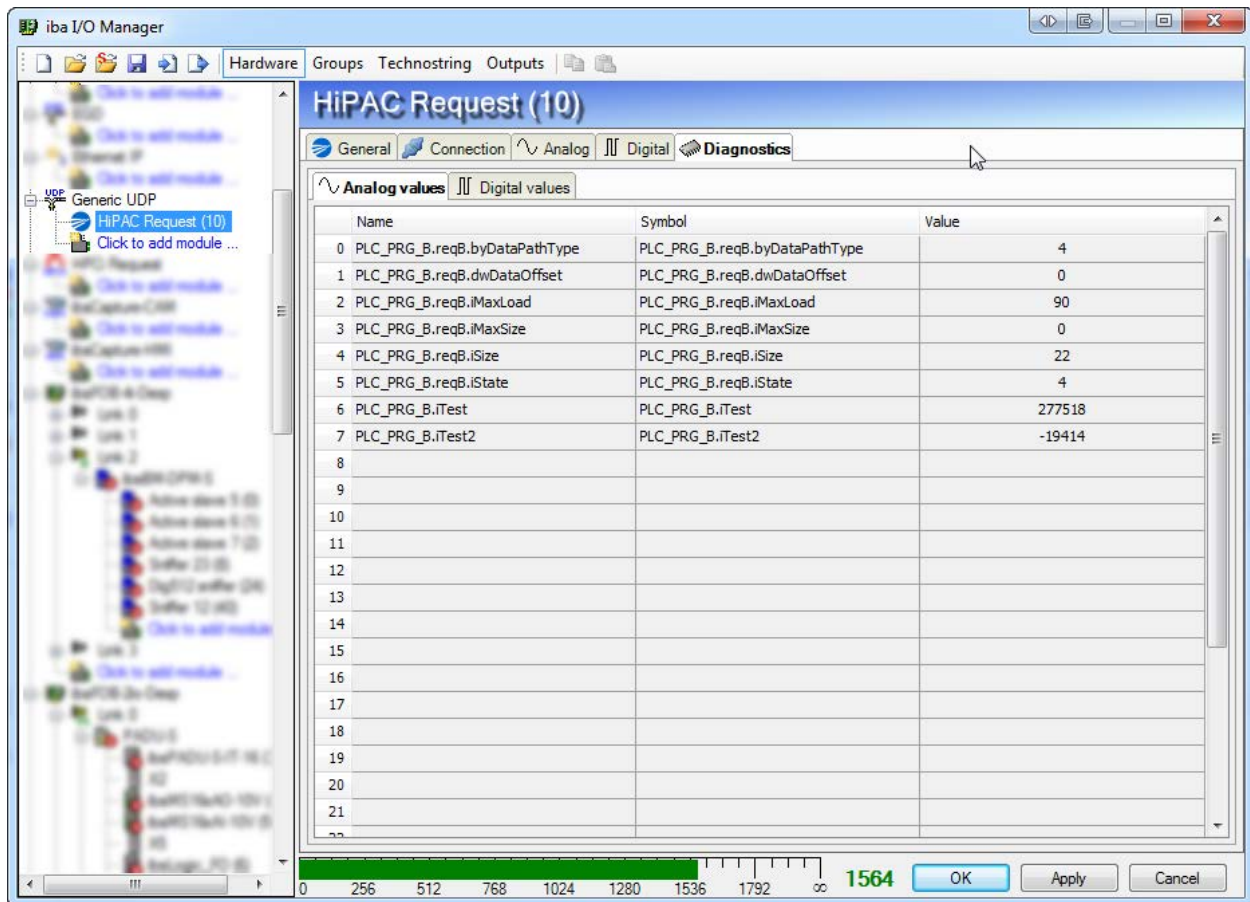
The symbols are re-loaded from the PLC with <Update symbols>.

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

You can also open the symbol browser in the signal tables, *Analog* and *Digital* register, if you click on the browser button in the *Symbol* column (<...>).

### 4.3.4 Diagnosis

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



The screenshot shows the 'Diagnosis' tab for the 'HiPAC Request (10)' module. The 'Analog values' section is active, displaying a table with the following data:

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
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20		
21		
22		

At the bottom of the window, a progress bar shows a value of 1564, with buttons for 'OK', 'Apply', and 'Cancel'.

Fig. 9: HiPAC request module, actual values of the symbols in the module's Diagnostics tab

## 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.

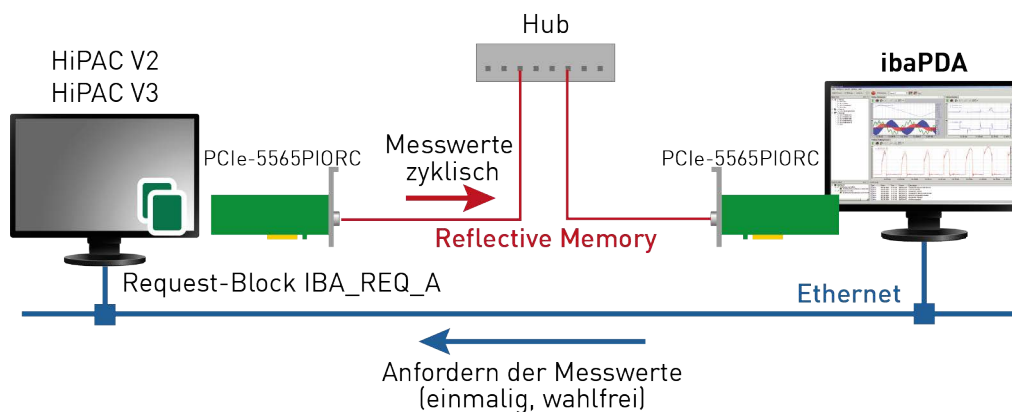


Fig. 10: Connections for control path via TCP/IP and data path via Reflective Memory

You need the following connections:

- Ethernet connection between *ibaPDA* and the HiPAC PLC
- Fiber optic link via reflective memory card in the *ibaPDA* computer and in the HiPAC computer (e.g. PCIe-5565PIORC, sales: abaco Systems)

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

### 5.2 Configuration and engineering of the HiPAC controller

The following configuration and engineering steps are generally to be made in HiPAC on the HiPAC side:

#### Hardware engineering

Installing the reflective memory card in the device configuration.

#### Software engineering

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

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

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

### 5.3 Configuration in ibaPDA

The configuration takes place in the I/O manager of *ibaPDA*. First establish the connection of *ibaPDA* to the HiPAC controller via Reflective Memory interface.

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

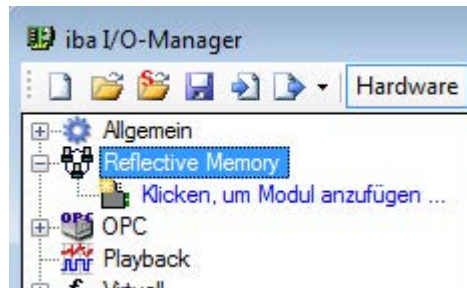


Fig. 11: Reflective Memory interface in the I/O manager

Once the connection is established, add a HiPAC request module accordingly. See chapter [↗ Add module](#), page 29.

The configuration of the signals and the selection in the symbol browser is described in the chapter [↗ Selecting symbols](#), page 21.



### 5.3.1 Setting up the connection

#### Other documentation



You can find extensive information about the interface *ibaPDA-Interface- Reflective Memory* in the associated manual.

#### 5.3.1.1 Properties

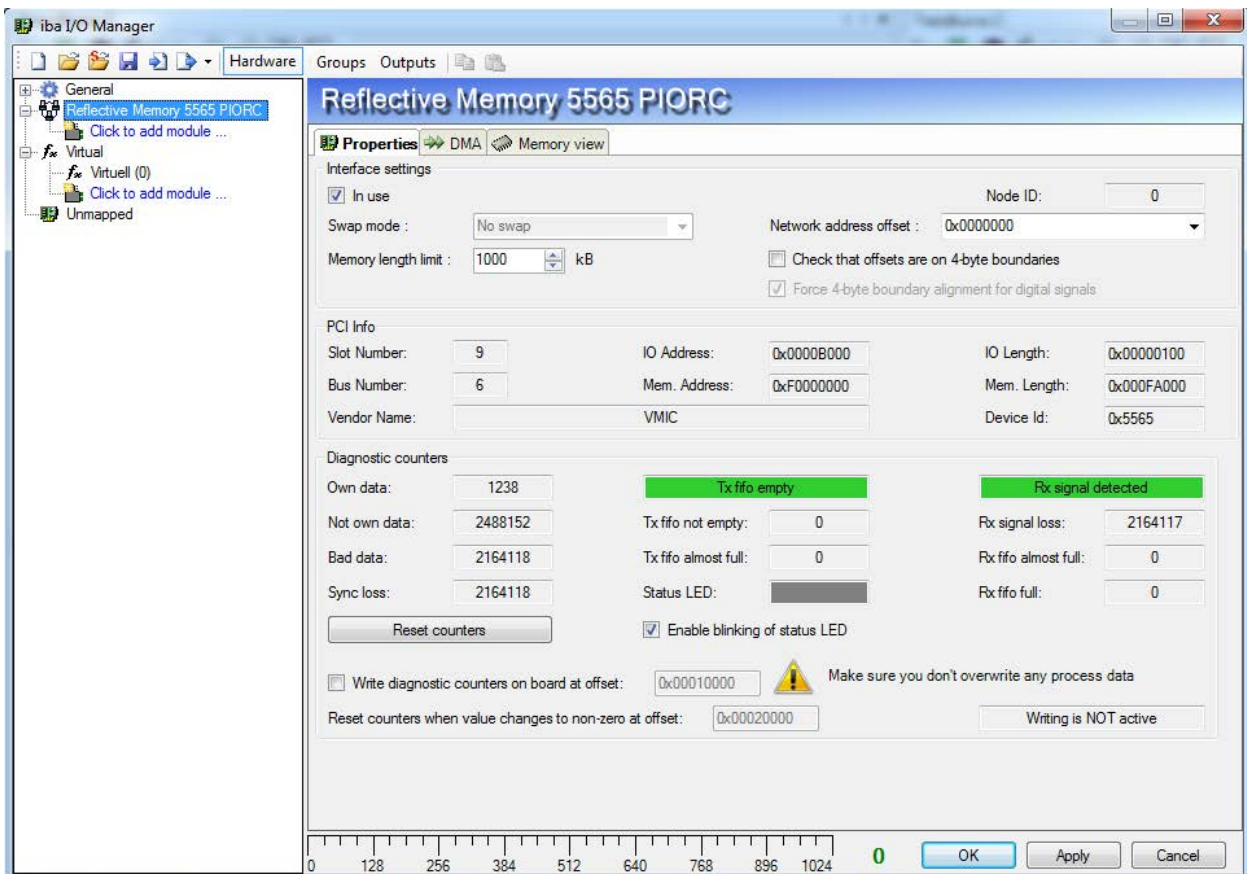


Fig. 12: Interface properties. The part Diagnostic counters is not available for all boards

#### Interface settings

##### "In use" check box

If the RM interface card 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.

##### Swap mode

Select the appropriate swap mode from the drop-down list in this field. The drop-down list provides several options of high- and low byte swapping (Endian Control). Which swap mode is suitable for your configuration depends on the connected source system. 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>. The acquisition is then stopped and restarted.

This setting is disabled with recent boards such as PCI 5565PIORC. You can select the swap mode in the settings of the data module, see *Reflective Memory - General tab*, page .

### **Memory length limit**

This parameter describes the size of the mapped memory space. You should adjust the memory size according to your needs, either by means of the up/down arrows or by entering a value. Reduce the size if you do not need that much memory length. This will save memory space in the *ibaPDA* computer.

### **Node ID**

This is the node ID as set on the RM interface board in the *ibaPDA* computer. It is for display only and cannot be altered here.

### **Network address offset**

This optional setting is only available if a card of VMIC 5576 type is used. The exact setting of a network address offset is required if a 256 kB or a 512 kB card is used in a 1 MB ring.

### **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.

### **Force 4-byte limit alignment for digital signals**

If this option is enabled, it makes sure that the data is always read along 4-byte limits. This is done to prevent sending of wrong data by some Reflective Memory boards if not reading exactly along 4-byte boundaries.

This option is enabled by default when a 5565PIORC board is used.

### **PCI Info**

Besides vendor name and device ID you'll find slot and bus number, PCI-memory addresses and IO-addresses. If the fields are empty or contain implausible values, then the board is plugged into the wrong PCI slot.

### **Diagnostic counters (available for PCI/PCIE 5565PIORC only)**

In the section *Diagnostic counters* you will find a couple of counters and status information which could be helpful when verifying the interface activity between *ibaPDA* and the Reflective Memory board.

---

## Other documentation



A detailed description of the diagnostic counters and status information can be found in the user manual of the *Reflective memory* module.

Example PCI-5565PIORC: Hardware Reference, Publication No: Publication no. 500-9367855565-000 Rev. C

There, you will find the related information in the chapters 3.3.5 "Local Control and Status Register 1" (LCSR) and 3.3.6 "Local Interrupt Status Register" (LISR).

---

- Own data  
Number of times LCSR bit 0 was 1
- Not own data  
Number of times LCSR bit 0 was 0
- Bad data  
Number of times LISR bit 8 was 1
- Sync loss  
Number of times LISR bit 11 was 1
- Tx Fifo not empty  
Number of times LCSR bit 7 was 0  
The field above this counter shows the status of the Tx Fifo as text.  
Therefore, the LCSR bit 8 is evaluated:  
Status 0 = Tx Fifo empty + green background  
Status 1 = Tx Fifo not empty + red background
- Tx Fifo almost full  
Number of times LCSR bit 6 was 1
- Status LED  
Status LCSR bit 31, refers to the red status LED on the board
- Rx signal loss  
Number of times LCSR bit 2 was 0  
The field above this counter shows the status of the Rx signal as text.  
Therefore, the LCSR bit 2 is evaluated:  
Status 0 = Rx no signal + red background  
Status 1 = Rx signal detected + green background
- Rx Fifo almost full  
Number of times LISR bit 9 was 1
- Rx Fifo full  
Number of times LISR bit 10 was 1
- Button <Reset counters>  
Click on this button in order to reset all counters to 0 (zero).

- Enable blinking status LED

If you enable this option, *ibaPDA* will toggle the LCSR bit 31 in 0.5 Hz clock. This function can be used for monitoring of the communication between *ibaPDA* and the Reflective Memory board.

- Write diagnostic counters on board at offset...

If you enable this option, then counter values and status information will cyclically be written into a memory range, which you can address by an offset in the adjacent entry field.

Make sure that this range is not used for other data.

This function is disabled by default because it is only needed for extended diagnostics.

- Reset counters when value changes to non-zero at offset

If you enable this option, then a memory, address which you can enter in the adjacent field, will be monitored.

The display field further on the right indicates whether *ibaPDA* writes the diagnostic counters in the Reflective Memory or not (Writing is active/Writing is NOT active).

### 5.3.1.2 DMA

If the DMA mode is enabled (see Reflective Memory - General tab), you can find in the DMA tab information for diagnostic purposes about the data exchange between *ibaPDA* software and the Reflective Memory interface board.

### 5.3.2 HiPAC request module

#### 5.3.2.1 Add module

Add a HiPAC request module in the I/O manager by clicking below the Reflective Memory interface. Select the desired module type and click <OK>.

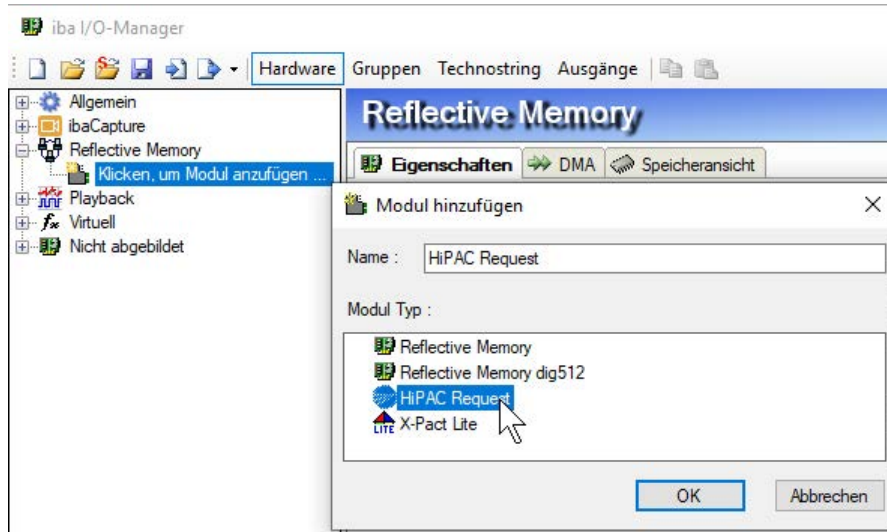


Fig. 13: Adding a module in the I/O manager

#### 5.3.2.2 General Settings

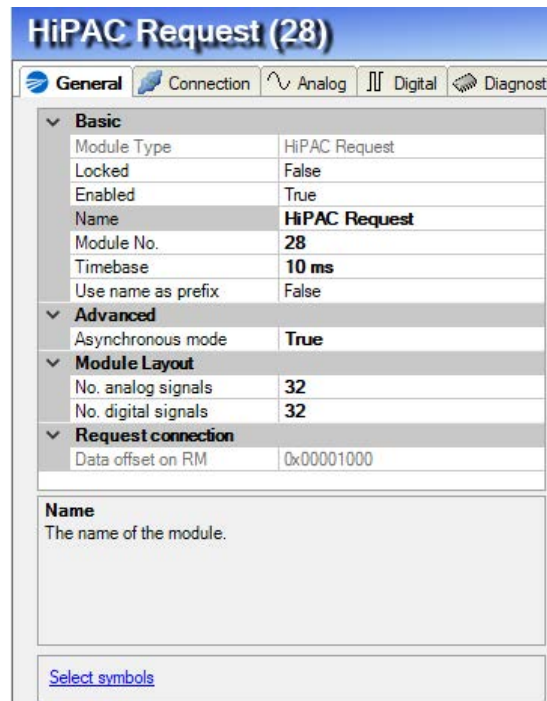


Fig. 14: HiPAC request module, general settings with reflective memory

**Basic settings, module layout**

The basic settings and the module layout under the *General Settings* are identical to UDP. See chapter [↗ General Settings](#), page 17.

**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 21 .

## 6 Diagnosis

### 6.1 Checking the license

If the “HiPAC request” modules are not shown in the signal tree, you can check in the I/O manager under "General - Settings - License info" whether your license *ibaPDA-Request-HiPAC* is properly detected.

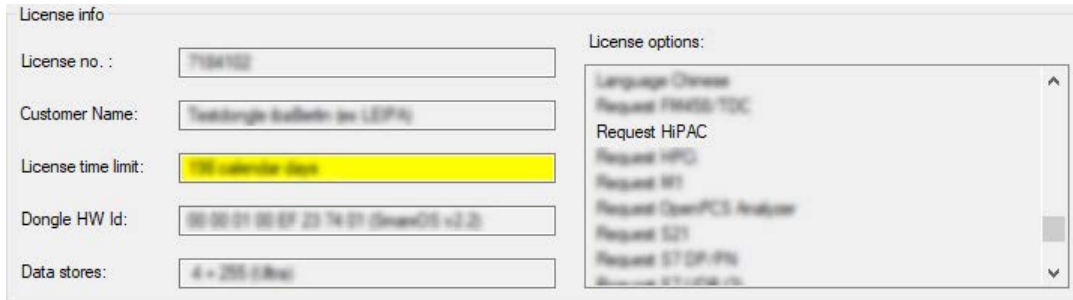


Fig. 15: Display of the license for request HiPAC in the I/O manager

In addition to the license for request HiPAC, other licenses must also be present, depending on which data path is to be used.

For generic UDP:

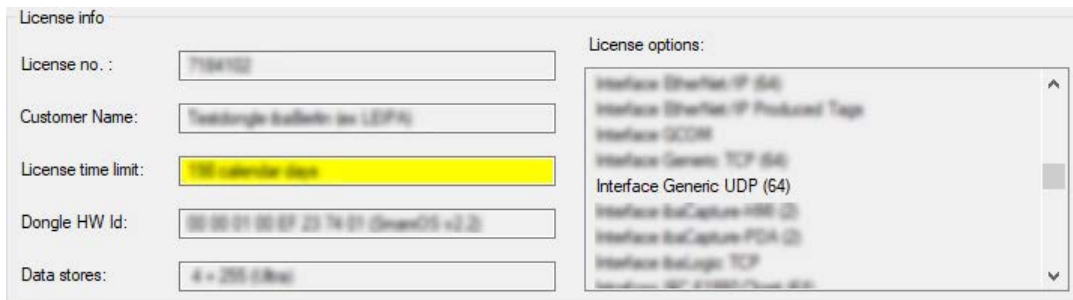


Fig. 16: Display of the license for generic UDP in the I/O manager

For reflective memory:

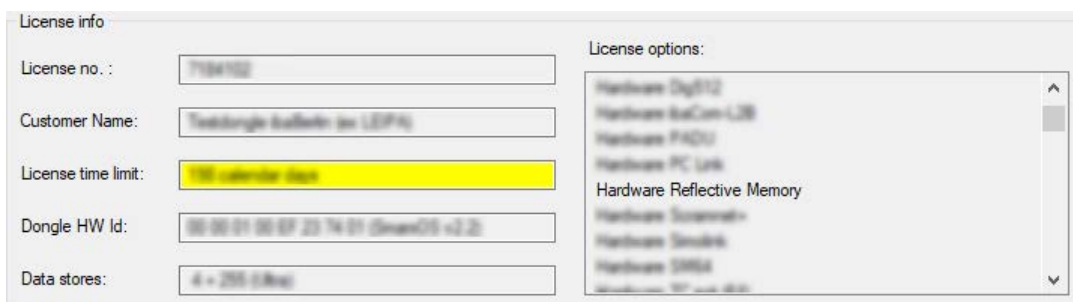


Fig. 17: Display of the license for Reflective Memory in the I/O manager

## 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.

The log file can be opened 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 will find the log files in the program path of the *ibaPDA* server (...\\Programs\\iba\\ibaPDA\\Server\\Log\\). The file names of the log files include the name or abbreviation of the interface type.

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

Examples:

- `ethernetipLog.txt` (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.

Open a Windows command prompt.



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

With an existing connection you receive several replies.

```
Administrator: C:\Windows\system32\cmd.exe
C:\Users>ping 192.168.21.120
Pinging 192.168.21.120 with 32 bytes of data:
Reply from 192.168.21.120: bytes=32 time<1ms TTL=128
Reply from 192.168.21.120: bytes=32 time<1ms TTL=128
Reply from 192.168.21.120: bytes=32 time=1ms TTL=128
Reply from 192.168.21.120: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.21.120:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms
C:\Users>
```

Fig. 18: PING successful

With no existing connection you receive error messages.

```
Administrator: C:\Windows\system32\cmd.exe
C:\Users>ping 192.168.21.121
Pinging 192.168.21.121 with 32 bytes of data:
Reply from 192.168.21.104: Destination host unreachable.
Reply from 192.168.21.104: Destination host unreachable.
Reply from 192.168.21.104: Destination host unreachable.
Reply from 192.168.21.104: Destination host unreachable.

Ping statistics for 192.168.21.121:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
C:\Users>_
```

Fig. 19: PING unsuccessful

## 7 Support and contact

### Support

Phone: +49 911 97282-14  
Fax: +49 911 97282-33  
Email: support@iba-ag.com

---

### Note



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

---

### Contact

#### Head office

iba AG  
Koenigswarterstrasse 44  
90762 Fuerth  
Germany

Phone: +49 911 97282-0  
Fax: +49 911 97282-33  
Email: iba@iba-ag.com  
Contact: Harald Opel

#### 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](http://www.iba-ag.com)**.