



# ibaPDA

## Data interfaces and modules

Manual  
Issue 8.7

Measurement Systems for Industry and Energy  
[www.iba-ag.com](http://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](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 2024, 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	Author	Version SW
8.7	04-2024	New: HTTP(S) interface, Snapshots; new status module for LMI Gocator	rm	8.7.0

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

## Contents

<b>1</b>	<b>About this documentation .....</b>	<b>9</b>
1.1	Target group and previous knowledge .....	9
1.2	Notations .....	9
1.3	Used symbols.....	10
1.4	Documentation structure .....	11
<b>2</b>	<b>Basics of data interfaces .....</b>	<b>12</b>
2.1	Introduction.....	12
2.2	Handling and availability of interfaces.....	14
<b>3</b>	<b>Basics of modules.....</b>	<b>15</b>
3.1	What is a module?.....	15
3.2	Adding modules.....	15
3.3	Deleting modules.....	16
3.4	Exporting modules.....	17
3.5	Arranging modules by folders.....	18
3.6	Module settings and features .....	20
3.6.1	Common and general module settings.....	20
3.6.2	Customizing signal tables.....	21
3.6.3	Columns in tables with analog and digital signals .....	22
3.6.4	Filling signal tables.....	27
3.6.5	Notes on working in the signal tables.....	28
<b>4</b>	<b>Standard interfaces (always available).....</b>	<b>29</b>
4.1	ibaCapture .....	30
4.1.1	Module types ibaCapture and ibaVision.....	32
4.2	OPC interface.....	34
4.2.1	OPC Client module.....	35
4.2.2	Redundant OPC client module.....	40
4.2.3	OPC Server module.....	41
4.3	Playback.....	42
4.4	Signal mapper.....	46
4.4.1	Signal mapper module.....	46

4.4.2	Add and configure signal mapper profiles .....	47
4.4.3	Signal mapper Example .....	49
4.5	Text interface .....	53
4.6	Virtual .....	54
4.6.1	Diagnosis of the processor load by virtual signals .....	56
4.6.2	16 bit decoder .....	57
4.6.3	32 bit decoder .....	59
4.6.4	Multidecoder module .....	60
4.6.5	16-bit encoder .....	66
4.6.6	32-bit encoder .....	67
4.6.7	Electric modules .....	68
4.6.8	ibaQPanel input .....	72
4.6.9	ibaQPanel text input .....	74
4.6.10	Trigger module .....	75
4.6.11	Virtual .....	80
4.6.12	Virtual retentive .....	83
4.6.13	Shift register .....	84
4.6.14	Computation module .....	85
4.6.15	Computation retentive module .....	93
4.6.16	Lookup table .....	94
4.6.17	Process condition .....	99
4.6.18	Parameter set .....	105
4.6.19	NMEA 0183 decoder .....	109
4.6.20	Vector calculation module .....	112
4.7	Unmapped .....	116
<b>5</b>	<b>ibaNet data interfaces (for iba devices) .....</b>	<b>117</b>
5.1	ibaNet protocols .....	120
5.2	ibaFOB-io-, 2io-, 4io- etc. ....	122
5.2.1	Interface settings .....	123
5.2.2	PCI Info .....	124
5.2.3	Card display .....	124
5.2.4	Info tab (card level) .....	126

5.2.5	Memory view (card level) .....	127
5.2.6	ibaFOB-io on link level .....	127
5.2.7	Info tab (link level) .....	127
5.2.8	Configuration tab (link level) .....	130
5.2.9	Memory view tab (link level) .....	133
5.3	ibaFOB-io-ExpressCard .....	134
5.4	ibaNet-E .....	135
5.5	Modules for ibaPADU-series devices .....	136
5.5.1	Example: ibaPADU-8 .....	137
5.6	Modules for ibaPADU-S-series devices (modular system) .....	138
5.6.1	Example: ibaPADU-S-CM.....	140
5.7	Modules for ibaBM series devices (bus modules) .....	142
5.7.1	Example: ibaBM-DP with active slave .....	145
5.8	Modules for ibaLink-series devices (system interfaces).....	154
5.8.1	Example: ibaLink-VME in 32Mbit Flex mode .....	155
5.9	ibaNet750/ibaNet750-D module type .....	159
5.10	FOB Fast module .....	161
5.10.1	FOB Fast – General tab .....	162
5.10.2	FOB Fast – Analog tab .....	163
5.10.3	FOB Fast – Digital tab.....	165
<b>6</b>	<b>Ethernet-based interfaces .....</b>	<b>166</b>
6.1	General and common settings.....	168
6.2	Connection table .....	169
6.3	AN-X-DCSNet .....	170
6.4	EGD (Ethernet Global Data) .....	171
6.5	EtherNet/IP.....	172
6.6	GCOM .....	174
6.7	Generic TCP .....	175
6.8	Generic UDP .....	176
6.9	ibaLogic TCP .....	178
6.10	IEC 61850 Client.....	179
6.11	IEC 61850-9-2 .....	180

6.12	LANDSCAN .....	181
6.13	LMI-Gocator.....	182
6.14	Micro-Epsilon.....	183
6.15	OPC UA .....	184
6.16	Modbus TCP client.....	185
6.17	Modbus TCP Server .....	186
6.18	Raw Ethernet.....	188
6.19	Raytek.....	189
6.20	S7 TCP/UDP.....	189
6.21	Sisteam TCP .....	191
6.22	TDC TCP/UDP.....	192
6.23	VIP TCP/UDP .....	194
6.24	Modules for Ethernet-based interfaces.....	195
<b>7</b>	<b>PLC-Xplorer interfaces.....</b>	<b>199</b>
7.1	AB-Xplorer .....	200
7.2	ABB-Xplorer .....	201
7.3	Bachmann-Xplorer.....	202
7.4	B&R-Xplorer.....	203
7.5	Codesys-Xplorer.....	205
7.6	Logix-Xplorer.....	206
7.7	MELSEC-Xplorer .....	207
7.8	OMRON-Xplorer .....	208
7.9	S7-Xplorer.....	209
7.10	Sigmatek-Xplorer .....	211
7.11	TwinCAT-Xplorer .....	212
<b>8</b>	<b>Siemens-specific interfaces .....</b>	<b>214</b>
8.1	FOB-SD/-SDexp and FOB-TDC/-TDCexp .....	215
8.2	TDC Request and Simadyn Request.....	217
8.3	CP1616.....	218
8.4	CP1626.....	219
8.5	MMC request.....	221
8.6	ibaFOB-PlusControl.....	222

8.7	SINAMICS-Xplorer .....	223
8.8	SIMOTION-Xplorer .....	224
8.9	SINUMERIK-Xplorer .....	226
<b>9</b>	<b>Other manufacturer-specific interfaces .....</b>	<b>228</b>
9.1	DTBox-Request .....	230
9.2	DGM200E.....	231
9.3	DGM200P .....	233
9.3.1	DGM200P – Board information .....	234
9.3.2	DGM200P module .....	236
9.3.3	DGM200P dig512 module .....	239
9.3.4	HPCi Lite module .....	240
9.3.5	HPCi Lite.....	243
9.4	Hitachi MicroSigma.....	244
9.5	HPCi Request .....	244
9.5.1	HPCi Request – Overview .....	246
9.5.2	HPCi Request – Diagnostics .....	247
9.6	Modbus serial .....	248
9.7	PC Link .....	250
9.7.1	PC Link module .....	252
9.7.2	PC Link Dig512 module type .....	253
9.7.3	PC Link Symbolic module.....	254
9.8	Reflective memory.....	256
9.9	ScramNet+ .....	257
9.9.1	Interface configuration .....	258
9.9.2	ScramNet module type.....	258
9.9.3	ScramNet dig512 module type .....	261
9.10	Toshiba ADMAP JAMI1 .....	262
9.11	X-Pact.....	263
9.11.1	X-Pact lite module (Reflective Memory).....	264
9.12	X-Pact Request.....	266
9.12.1	X-Pact Request module.....	268

- 10 Cloud, database and message broker interfaces ..... 271**
  - 10.1 SQL database interface ..... 271
  - 10.2 MQTT interface ..... 273
  - 10.3 HTTP(S) interface ..... 274
- 11 Other interfaces and ibaPDA add-ons ..... 275**
  - 11.1 Audio ..... 276
  - 11.2 ibalCycle ..... 277
  - 11.3 ibalSpectra ..... 278
  - 11.4 Snapshots ..... 279
- 12 Diagnostic modules ..... 280**
- 13 Support and contact ..... 285**

# 1 About this documentation

This documentation describes the function and application of the software *ibaPDA*.

## 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.

## 1.2 Notations

In this manual, the following notations are used:

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

## 1.3 Used symbols

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

---

### Danger!



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

- Observe the specified measures.

---

### Warning!



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

- Observe the specified measures.

---

### Caution!



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

- Observe the specified measures

---

### Note



A note specifies special requirements or actions to be observed.

---

### Tip



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

---

### Other documentation



Reference to additional documentation or further reading.

## 1.4 Documentation structure

This documentation fully describes the functionality of the *ibaPDA* system. It is designed both as a tutorial as well as a reference document. The sections and chapters essentially follow the procedure for configuring the system.

In addition to this documentation, you can examine the version history in the main menu, *Help – Version history* (file [versions.htm](#)) for the latest information about the installed version of the program. This file not only lists the bugs that have been eliminated, but also refers to extensions of the system in note form.

In addition, special "NewFeatures..." documentation comes with any software update that includes significant new features, which provides a more detailed description of the new features.

The state of the software to which the respective part of this documentation refers is listed in the revision table on page 2.

The *ibaPDA* system documentation (PDF and printed version) is divided into seven separate parts. Each part has its own section and page numbering beginning at 1, and is updated independently.

<b>Part 1</b>	Introduction and installation	General notes, license policy and add-ons Installation and program start User interface, system architecture, client server User management, printing
<b>Part 2</b>	I/O Manager	Basic information about the I/O Manager, general settings Groups and vector signals, text signals, outputs, configuration files
<b>Part 3</b>	Data interfaces and modules	Interfaces for the measurement data acquisition Standard interfaces, ibaFOB, Ethernet-based interfaces and more. For interfaces for which there are separate manuals, these are referred to.
<b>Part 4</b>	Expression builder	All functions for calculating virtual signals
<b>Part 5</b>	Data storage	Types of data recording, recording profiles, signal selection
<b>Part 6</b>	Data visualization	All display modes for live data, their operation and settings
<b>Part 7</b>	Appendix	Various additions, error listings, etc.

## 2 Basics of data interfaces

The data interfaces establish the connection between *ibaPDA* and the plant, process or control. All data and signals to be measured are acquired via them.

### 2.1 Introduction

To connect the *ibaPDA* system to the most common communication standards and most PLC systems there are different types of interfaces. These interfaces support both standard protocols as well as selected system interfaces with special components.

For all interfaces, one or more modules are available for further configuration, which make it possible to customize the measured data acquisition and in which the actual measuring signals are configured.

Interfaces to standard field-bus technology, such as PROFINET, PROFIBUS and CAN-bus, are generally implemented with corresponding devices in combination with *ibaFOB* input boards.

The data interfaces form the trunk of the interface tree in the left field of the I/O Manager.

In the following context, the interfaces are grouped together.

#### ■ Default interfaces

These interfaces are always available in the I/O Manager regardless of licenses. These include, for example, *ibaCapture*, *Playback* and *Virtual*.

See [↗ \*Standard interfaces \(always available\)\*](#), page 29.

#### ■ ibaNet interfaces

These interfaces are available in the I/O Manager if the corresponding board (*ibaFOB*-card) is inserted into the computer. All iba devices with ibaNet fiber-optic interfaces can be connected to these interfaces. These include, for example, *ibaFOB-io-D*, incl. all board versions (-io, -2io, -4i, -4o) and *ibaFOB-io-ExpressCard* (-34, -54).

See [↗ \*ibaNet data interfaces \(for iba devices\)\*](#), page 117.

#### ■ Ethernet-based interfaces

These interfaces are available in the I/O Manager if the appropriate interface license is enabled in the dongle. On the hardware side, the network interface of the computer or an additional network card is used. These include, for example, *EGD*, *ibaLogic TCP* and *MQTT*.

See [↗ \*Ethernet-based interfaces\*](#), page 166.

#### ■ PLC-Xplorer interfaces

These interfaces are available in the I/O Manager if you have purchased the *ibaPDA-PLC-Xplorer* product or licensed the appropriate Xplorer interface in your *ibaPDA* system. With the exception of *S7-Xplorer*, these interfaces only use the network interface of the computer. These include, for example, *ABB-Xplorer*, *S7-Xplorer* and *TwinCat-Xplorer*.

See [↗ \*PLC-Xplorer interfaces\*](#), page 199 and the corresponding manual for the each interface.

**■ Drive-Xplorer interfaces**

These interfaces are available in the I/O Manager if you have purchased the *ibaPDA-Drive-Xplorer* product or licensed the appropriate Xplorer interface in your *ibaPDA* system. These include, for example, SIMOTION-Xplorer and SINAMICS-Xplorer. See [➤ PLC-Xplorer interfaces](#), page 199 and the corresponding manual for the each interface.

**■ Special Siemens interfaces**

These interfaces are available in the I/O Manager if the corresponding board is inserted and/or the corresponding interface license has been acquired. These include, for example, iba-FOB-SD/-SDexp, CP1616/CP1626 and MMC Request.

See [➤ Siemens-specific interfaces](#), page 214 and the corresponding manual for the each interface.

**■ Other interfaces**

These interfaces are available in the I/O Manager if the corresponding board is inserted and/or the corresponding interface license has been acquired. These include, for example, HPCi Request, ibaInSpectra and PC Link.

See [➤ Other manufacturer-specific interfaces](#), page 228, [➤ Other interfaces and ibaPDA add-ons](#), page 275 and the corresponding manual for the each interface.

**■ SQL database interfaces**

These interfaces are available in the I/O Manager if a license for at least one interface out of this group has been acquired. They can be used to retrieve data from or write data to a database by means of SQL statements. These include, for example, MySQL/MariaDB, Oracle and SQL-Server.

See [➤ Cloud, database and message broker interfaces](#), page 271 and the corresponding manual for the each interface.

In addition to acquire measured data, some interfaces also offer the option to output data (values, texts). The configuration of these interfaces and output modules is described in part 2, *Outputs*.

## 2.2 Handling and availability of interfaces

### Availability of interfaces

In the left part of the I/O Manager window, you will find a tree structure containing all data interfaces that...

- a) are installed and have been detected automatically (e.g. ibaFOB-4i-D card) and/or
- b) are enabled according to the license in the dongle (e.g. TCP/IP protocols) and/or
- c) were added manually afterwards.

Some interfaces and branches are always available by default, such as OPC, Playback and Virtual.

### Adding a data interface

Manual addition of data interfaces is not required, as the system automatically detects which interfaces are installed on the computer or activated in the dongle.

The installed and enabled interfaces are displayed in the I/O Manager tree.

---

#### Tip



If you decide, however, to add an interface manually, e.g. in order to prepare an I/O configuration, then proceed as follows:

1. Press and hold <Shift> and make a right mouse click on the node *General* up in the interface tree.
2. In the context menu, click on *Add interface...*
3. Select the desired interface

---

### Removing an interface

Manual removal of data interfaces is not required as the system automatically detects which interfaces are installed on the computer or activated in the dongle.

With startup of *ibaPDA*, the system is checked for available interfaces. If the recent I/O configuration included interfaces, which are no longer available, then the interfaces in question are marked as blocked.

---

#### Tip



If you want, however, to remove an interface manually, then do as follows:

1. Press and hold <Shift> and make a right mouse click on the interface you'd like to remove in the interface tree.
2. In the context menu, click on *Clear*.

## 3 Basics of modules

### 3.1 What is a module?

Generally, a module is the software equivalent of a device or data connection which provides a number of signals for measurement. Modules must be configured beneath the appropriate data interface in the I/O Manager.

There are module types, which correspond directly to hardware devices, e. g. ibaPADU-8 or ibaLink-SM-64, providing 8 or 64 analog and 8 or 64 digital signals, respectively.

For other interfaces or data sources, e. g. Ethernet-based connections, one module corresponds to one connection with up to 1000 analog and 1000 digital signals.

For some interface types, there are module variants that include different data types (integer, real, digital) or communication types (e.g. Active Slave, Sniffer).

Module name and type are stored in the data file and hence are displayed in *ibaAnalyzer*.

The following chapters describe the generally valid properties and settings of the modules. More information about the configuration of individual modules can be found in this part of the *ibaPDA* manual or in the respective interface manual.

### 3.2 Adding modules

All modules are created or added in a similar way. You can choose from the following methods.

#### Right-click on data interface in the tree structure of the I/O Manager

1. Open the context menu with a right-click on a data interface in the tree structure of the *Inputs* or *Outputs* tab.
  2. Select the *Add module* command.
- Depending on the interface type chosen, another submenu opens a selection of available (or permitted) modules. Particularly for the *ibaNet750-BM/-BM-D* module type, there is a multi-level submenu, which helps you selecting the available terminals after you right-clicked on the added terminal module.

#### Via the Click to add module... command in the tree structure of the I/O Manager

1. Click on the blue command *Click to add module...* located under each data interface in the *Inputs* or *Outputs* tab.
2. Select the desired module type in the dialog box and assign a name via the input field if required.
3. Confirm the selection by clicking on <OK>.
4. To add further terminals to the *ibaNet750-BM/-BM-D* module, click on the *Click to add terminal* command below the module.

5. Add the desired terminals, either by double-clicking on the terminal or by clicking the <Add> button.

If you want to add several terminals of the same type, you can first enter the number of terminals to be added and then click <Add>.

→ The corresponding signals are added to the signal lists of the module according to the terminal type.

6. Close the dialog with <Close>.

### Adding connected devices

If you have connected another device to the *ibaPDA* interface, right-click on the corresponding link (connector) in the tree structure beneath the data interface. Select *Autodetect* from the context menu.

→ The system automatically detects the new device and adds the corresponding modules and signals.

### Copying a configured module

If you have already configured a module and would like to add a similar one, just right-click on the existing module and select *Copy* from the context menu.

→ A copy of the module is added to the next available free place or link.

The system automatically detects which module types can be added to an existing configuration and modifies the choice in the context menus accordingly.

## 3.3 Deleting modules

In order to delete a module, click on the module in the tree structure of the I/O Manager and press <DEL>. Or right-click on the module and choose *Delete* from the context menu.

---

### Note



Deleting a module will delete all configured signals of this module, too!

If an existing module cannot be used, e.g. because the corresponding hardware device is temporarily not in operation, it is recommended to shift the module by drag and drop to the "Unmapped" interface node. This will remove the module from the active configuration, but keep the module and signal settings (names, units etc.). Later, when the device is back in operation, you can drag the module back to the original node.

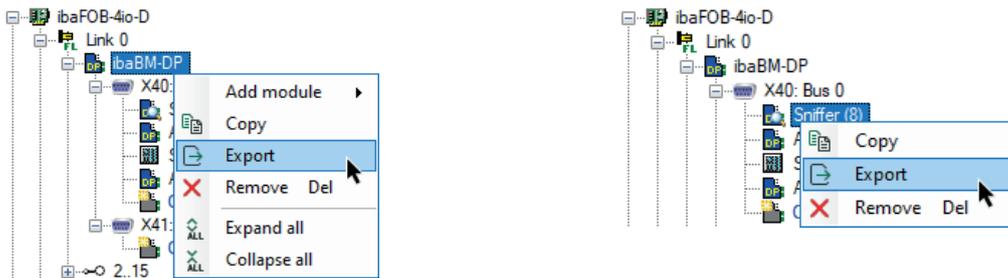
---

### 3.4 Exporting modules

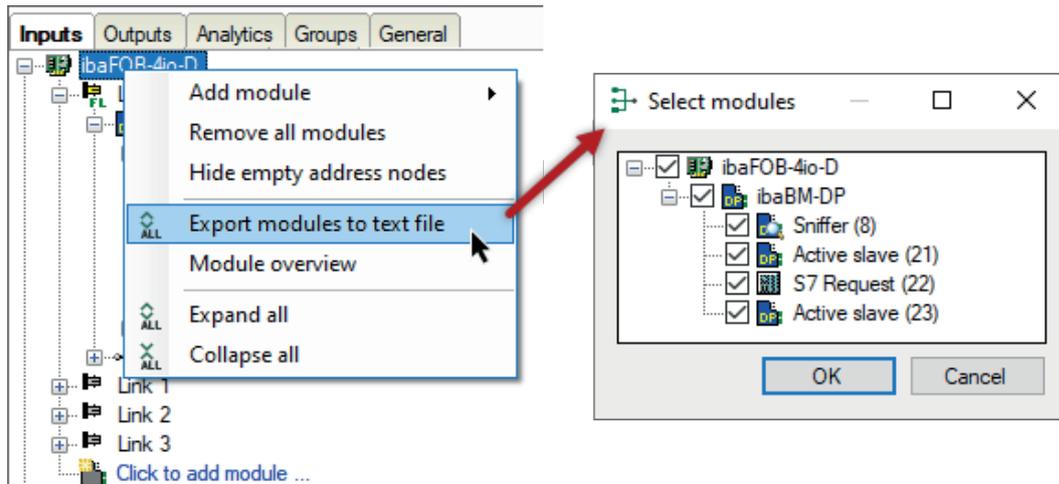
Export of single modules is possible. If there is no need to export an entire I/O configuration, you can export a single module. Just right-click on the module in the I/O Manager tree and select *Export* from the context menu. In case of modules and sub-ordinated sub-modules, the export consists either of the main module and all sub-modules or of the sub-module only, depending on your previously selected node.

By default, the file name of the export file is `Module Name (Module No.).txt`.

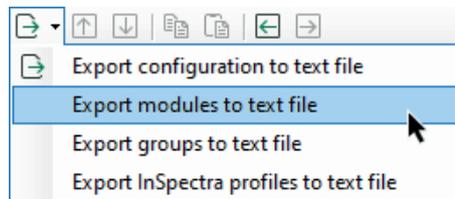
In the example below, on the left side, the ibaBM-DPM-S, Sniffer, S7 Request modules and two Active slave modules are exported. The export on the right side includes just the Sniffer module.



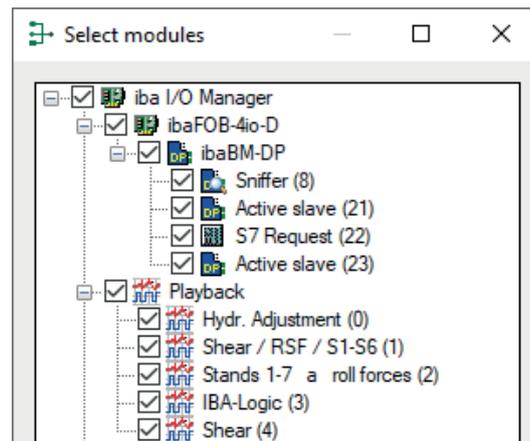
If you call up the export function on the link or interface level, you first see a dialog in which you can deselect modules that you do not wish to export.



A function to export modules is also available via the Export button in the I/O Manager toolbar. To do this, click *Export modules to text file*.



A window opens in which you can deselect modules that you do not wish to export.



### 3.5 Arranging modules by folders

You can create folders under each interface. Particularly for interfaces with lots of modules, folders can help to gain clarity by arranging modules in a folder structure. For instance, you can use folders to collect input signals which have the same technological context while originating from different sources. Or you organize the signals according to their origin or properties, e. g. text signals, computed signals etc.

#### Create folders

You can create folders by clicking on the blue link "Click to add module..." or via the context menu on the interface node. You can create sub-folders too.

A new folder initially gets the name of the interface or of its parent folder. However, you can rename any folder individually.

Existing modules can be moved into the folders by drag and drop.

By using the context menu on a folder you can directly create a module in that folder.

#### Remove folders

If you want to remove a folder which contains modules, a request pops up. Here, you can decide whether to remove the folder including its modules or just the folder structure. In the latter case the folder will be deleted and the included modules will be moved to the parent folder or to the root level of the interface.

#### Copy folders

You can copy folders and their sub-folders you have created for one interface to other interfaces. Thus, you can easily transfer folder structures you've created once to other interfaces.

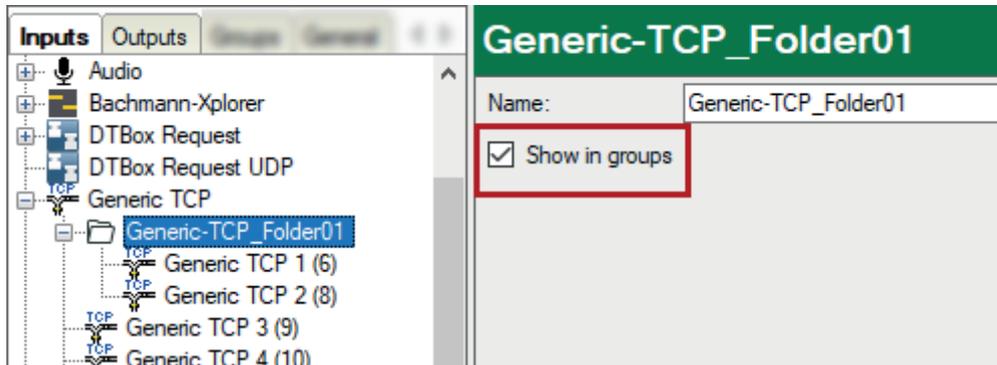
Make a right mouseclick on the folder you want to copy and choose *Copy folder...* in the context menu. A dialog opens where you can select one or more target interfaces. When you click on <OK> the folder or the folder structure will be created under the other interfaces, still empty. You then can assign the desired signals to the folders for each interface by drag and drop.

#### Use folders in the group view

Folders and sub-folders which you created in the *Inputs* or *Outputs* tabs, are automatically available in the *Groups* tab. Hence, your preferred structure is available with the groups immediately. However, folders with the same name and structure level from different interfaces will be

combined to one group folder. For instance, a folder which was created under three interfaces appears only once in the *Groups* tab containing the signals of all three interfaces.

You can disable the listing of the folders in the *Groups* tab for each folder. Just select the folder in the interface tree of the I/O Manager and remove the checkmark at *Show in groups*.



### Folders in the signal tree

If you switch to the "Show groups" mode in the signal tree pane of the *ibaPDA* client then you see the signals in the same order like in the I/O Manager's *Groups* tab.

### Folders in analysis

The folder structure you have created in *ibaPDA* will be stored in the data files or *ibaHD-Server* data storage.

If needed, the folder structure is available in *ibaAnalyzer* for data files as well as for HD queries.

## 3.6 Module settings and features

In principle, the configuration dialog of a module consists of at least 3 tabs:

- General settings
- Analog signals
- Digital signals

Depending on the module type, there may be more or less tabs available.

The majority of all modules share the common properties which should be configured in the *General* tab of a module.

Since most modules have these general properties in common, they are described below.

For some modules, there are additional properties that need to be configured beyond these common items. Those are explained in the chapter of the corresponding module.

The tabs with the signal tables for analog and digital signals of a module are similar and offer the same functions for many modules. Some differences apply regarding data types and addressing. The most common features are also described below.

### 3.6.1 Common and general module settings

#### Basic settings

##### Locked

A module can be locked in order to prevent change of module settings by accident or unauthorized users. The lock function is linked to the user management in *ibaPDA*. Provided the user management is activated, a module can only be locked or unlocked by users with the appropriate access rights.

The following options are available for locking a module:

- None: The module is unlocked. Changes can be made to the module configuration.
- Outputs: The output signals are locked and cannot be changed. The input signals can be changed. If there are module properties that affect the outputs then these are locked as well.
- Inputs and outputs: The complete module is locked. No change of module settings is possible, neither input and output signals nor other properties.

Remark: On module types which do not support outputs, only the options *None* and *Inputs* are available for locking. On module types *ibaW-750*, *ibaNet750-BM* and *ibaNet750-BM-D*, the options *Outputs* and *Inputs and outputs* are not available unless some output terminal has been added.

##### Enabled

By selecting items from the dropdown list in the field on the right side of *Enabled*, or by double-clicking, you determine whether the module is enabled (true) or disabled (false). If a module is disabled, then its signals are excluded from acquisition. This means they are neither available for display nor for recording. Furthermore, the number of signals of a disabled module will not be taken into account in the signal statistics (see signal usage display).

---

**Tip**

For all settings which only allow two states (such as true/false), you can also simply change them by double-clicking in the field.

---

**Name**

Enter a comprehensive name for the module here.

It is recommended to use an project specific nomenclature for a better transparency and comprehension, particularly when many modules are used. The name can be chosen after its technological function or the installation site in the plant.

The number of characters in the name is unlimited. The name of the module is stored in the data file and visible in *ibaAnalyzer*.

**Module No.**

A module number must be entered here. If you add modules to the configuration, the system automatically assigns the numbers in chronological order. However, you may choose a different order later in the data file for analysis. Feel free to change the module numbering. The order of the modules in the signal tree of *ibaAnalyzer* is determined by their numbers.

The module number must be unique. Duplicate module numbers are not accepted

The range for module numbers extends from 0 to 1048575

**Time base**

You can enter a value (in ms) here as the *time base*, which is an integer multiple of the general time base as configured in the *General tab* in the *Settings* node of the I/O Manager. All signals of the module are then captured on this time base. The ratio between the maximum and minimum time base is limited to the value 1000. The time base value is limited to 1000 ms.

**Use name as prefix**

If you set this option to TRUE, then the module name is prepended to the signal names. The combination of module name and signal name is both used for the display in the *ibaPDA* views and also stored in the data file.

### 3.6.2 Customizing signal tables

The signal tables (analog, digital) contain all the signals to be acquired by *ibaPDA*.

The properties of each signal are organized in different table columns.

You can customize the layout of the signal table according to your needs regarding:

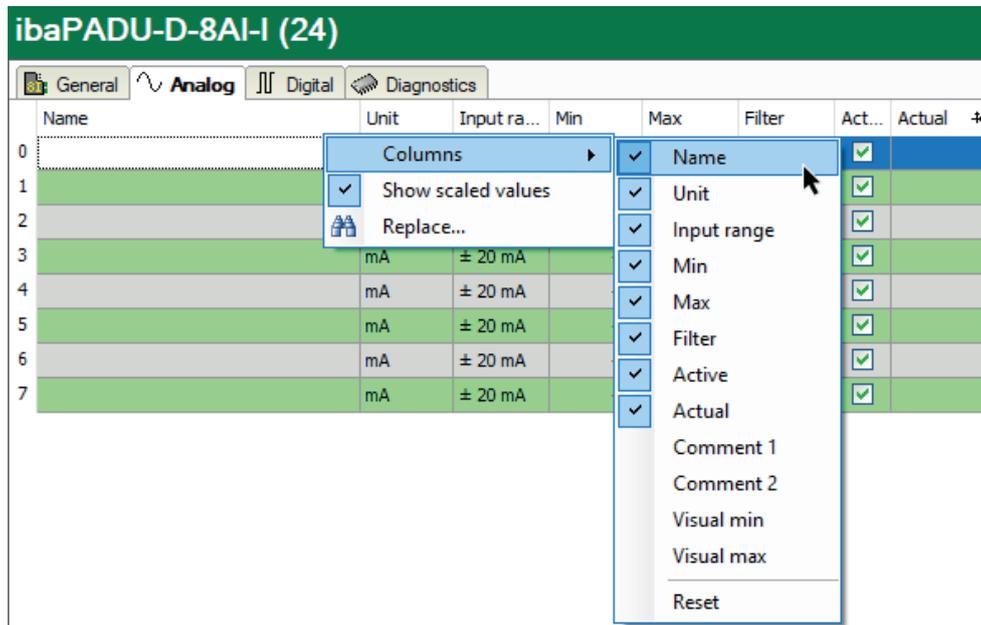
- Column visibility
- Column order
- Column width

The custom settings are saved for each module type, i. e. user-specific settings for the signal tables of a module apply to all other modules of the same type.

Of course, the selection of the available columns depends on the module type.

In order to customize the table layout, follow these steps:

1. Right-click somewhere in the table header. The context menu appears. Follow the Columns submenu.



2. Click on the column name(s) you like to hide or show (toggle). "Reset" restores the default settings. To close, click outside the menu.
3. In order to change the order of the columns, click in the header of the column you like to move, drag it to the desired position and drop it there.
4. In order to change the column width, place the mouse cursor in the header on the column border to be shifted. In the correct position, the mouse cursor changes to a double arrow icon. Click and hold left mouse key and move the mouse to the right or left hand side. Let drop when the width is ok.

### 3.6.3 Columns in tables with analog and digital signals

#### Name

Here, enter a clear text name of the signal.

It is recommended to use an project specific nomenclature for a better transparency and comprehension, particularly when many signals are used. The name for the signal can be chosen after its technological function or the installation site in the plant.

The number of characters is unlimited. The signal names are stored in the data file and indicated in *ibaAnalyzer*.

### Comment 1 and Comment 2

Each signal can be assigned up to 2 comments. Comments can be used to provide additional information about a signal. This could be an extended signal description or information in another language. The comments can be entered in the signal tables in different ways:

- Enter the comment directly in the cell if the comment column is visible.
- In the cell of the signal name, click the  button and enter the comment in the comment field. Then click <OK>.
- When using Request modules for selected PLC systems, the comments are loaded together with the signals if comments are supported by the PLC system.

#### Tip



Comments are of great use because:

- They are stored in the data file and available in *ibaAnalyzer*.
- They can be used as alternative signal names.
- They are subject to the search function (optionally) and can be displayed in the search results.
- They are shown as tooltips on a signal in the signal tree, legend, marker table and digital meter.

### Visual min. and visual max. (analog signals only)

These settings can be used to limit the Y scale of a trend graph to a minimum and a maximum value. These settings only apply to the trend graph if the corresponding Y-axis property "Visual scale configured on signal" is enabled.

This helps to determine the Y scale already in the configuration phase provided the final signal range is known.

Moreover, the Y-axis automatically adjusts the *Visual min.* and *Visual max.* values as soon as you open a trend graph with such a signal. For doing so, you have to set the "Visual scale configured on signal" property of the Y-axis in the trend graph's preferences. The Y scale immediately shows the expected range.

### Min and Max (analog signals on ibaPADU modules and other A/D converting modules only)

Lower and upper limit of measuring range given in physical units.

Preset values are -10/+10 for the standard modules ibaPADU and -20/+20 for ibaPADU-8-I modules. This corresponds to the voltage range of real devices of -10 V to +10 V or from -20 mA to +20 mA.

If you do not want to measure exactly the same range, but other physical quantities like temperature, pressure, speed etc., you should enter the minimum and maximum value corresponding to the limits of the measurement range of the device in physical units.

The value can be entered directly or with the help of the two-point-scaling dialog.

Value range: -32768 to +32767 (Integer)

**Example**

A temperature measurement connected to an ibaPADU-8 module delivers the values -10 V at -10 °C and 10 V at 43 °C. In that case, you should enter -10 (min.) and 43 (max.).

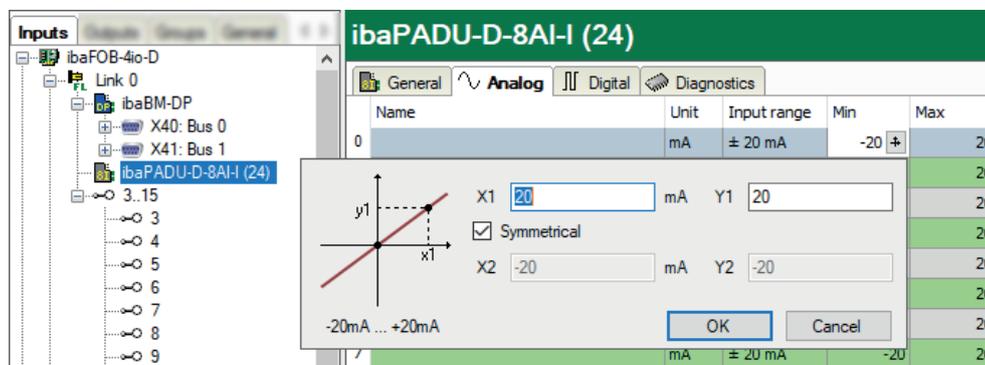
The analog voltage level of -10 V is assigned to the temperature -10 °C.

The analog voltage level of +10 V is assigned to the temperature 43 °C.

**Tip**

By entering these values, you do not set real limits to the measurement, but rather a linear scaling rule.

If you do not know the actual limits of a measurement device, just take two values you know for sure, e. g. from a calibration or test measurement, and use the two-point-scaling dialog.

**Two-point-scaling dialog**

The two-point-scaling dialog can be used to provide a scaling factor for analog signals by entering 2 points (X1/Y1 and X2/Y2) of a straight line. This dialog helps to scale the physical input quantity (e.g. voltage on ibaPADU-8) in order to get a value in physical units of the measured quantity (e. g. pressure, speed, temperature, etc.)

You can open the dialog for two-point scaling with a click on the button the fields "Min", "Max", "Gain" or "Offset". (Cursor must be positioned in the field for the button to appear.)

If you know that one of the values is 0/0, you can check the "Symmetrical" check box and enter only one value. (X1/Y1).

If the connected module has a physical input range, this range is shown in the dialog (e.g. from -10 V to +10 V).

**Unit (analog signals only)**

Assignment of an engineering unit (such as Ampere, Volt, m/s, tons etc.) for the signal. This entry can be up to 11 characters long and is regarded as a comment field only. It is always displayed in conjunction with a numerical display of the values.

### Gain and Offset (analog signals only)

The values for gain and offset describe a linear characteristic curve for scaling. If incoming values are given in physical units, gain and offset can be ignored, i.e. set gain = 1 and offset = 0.

However, control and regulation applications of the connected automation systems often use normalized values so that analog signals vary, e.g., only between 0.0 and 1.0 or -1.0 and +1.0. In order to get a correct scale for display in terms of physical units, a scaling factor needs to be specified. This factor is determined from the gain and offset parameters.

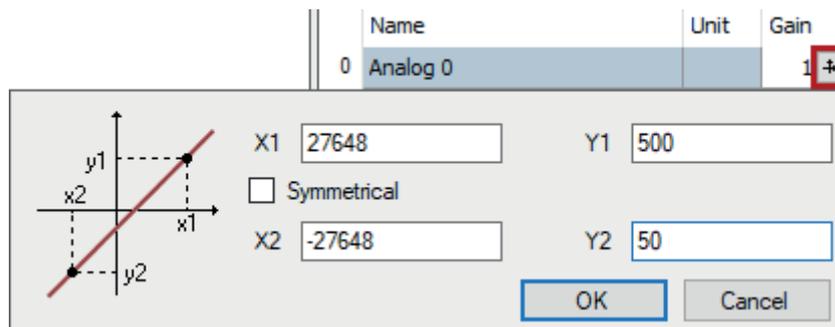
Gain and offset can be entered directly or set by means of the two-point scaling with two pairs of applicable values.

You can open the dialog for two-point scaling with a click on the  button in the fields "Gain" or "Offset." (Cursor must be positioned in the field for the button to appear.)

### Example

With a SIMATIC ET200 AI/AO component, a  $\pm 10$  V signal is transmitted with the value range -27648 to 27648 (corresponding to -10 V to +10 V). The transmitted value has a physical meaning in the control program (e.g., temperature 50 °C up to 500 °C). A conversion can be set via Gain/Offset so that the unitless value captured is recorded converted into the physical unit.

In order to facilitate the gain/offset calculation, by clicking on the coordinate cross a help dialog box appears in the Gain or Offset field in which you simply specify two bases of the linear equation. The gain and offset are then calculated automatically.



### Active

Activation of signals

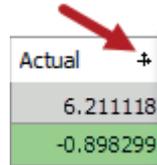
A click on the "Active" column heading enables (check mark) or disables (no check mark) all the signals for the measurement at the same time. Individual signals can be enabled in the signal-specific selection box. No acquisition takes place for disabled channels, so that such channels are available neither for visualization nor for storage.

Furthermore, disabled signals are not taken into account in the signal statistics (signal usage display).

Entries – such as names or physical units – remain unchanged. They are available again right after reactivation of a signal.

### Actual value

The fields in this column show the actual value of the signals. Even if the data acquisition is not running yet, the actual values may be displayed, as they are directly read from the hardware driver. As for the value display, you can choose between scaled and raw values. Click on the column header to toggle the values. If scaled values are on display, the header shows a little icon:



Alternatively, you can use the context menu anywhere in the signal table to enable/disable scaled values display.

In some cases, "NaN" (= Not a Number) instead of a number can be displayed. This can happen with a few modules if no valid value is available (e.g. ibaPACO-4).

For digital signals, only the values 0 and 1 are permitted.

### Address

The *Address* column is mainly available in modules which communicate over network interface, Xplorer interface or other bus systems.

The address determines where the signal in question is located within the net data range of a telegram or Dual-Port RAM (given in bytes). For some modules the address value can be displayed as decimal or hexadecimal value. The signal to be measured is always identified by the combination of address and data type (in the adjacent column). If you enter the data types first, you can let the system determine automatically the addresses by a double-click on the column header. This is useful if signals of different data types are consecutively packed for transmission.

### Data type/Bit no.

The interface type, respectively the module type determines which data types are supported. For analog signals the selection of these data types can be found in a drop-down list in the fields of the *Data type* column. The data type determines the address of the next signal.

For digital signals only the bit number, relative to the byte address, is given in the column aside to identify the digital signal.

### Symbol/Operand

Notably, the modules of Xplorer or Request interfaces have a column *Symbol* and/or *Operand* instead of *Address* und *Data type*.

These interfaces are exactly adapted to the controls or PLC systems which supply the signals to be measured. Mostly the symbolic addressing of the signals to be measured is supported. Therefore, in a first step, an address book is generated in the configuration of the system.

Usually you can open a symbol browser in the *Symbol* column in order to select the desired signals. The symbol name is copied automatically to the *Name* column, but you can overwrite it with any clear name.

### 3.6.4 Filling signal tables

For filling the signal tables with data, i.e. for the configuration of signals to be measured, there are some helpful functions which could ease your work, particularly if you have large amounts of data.

In the following, find the description of two functions which apply to signal tables in general:

#### Copying signal tables over clipboard

In the toolbar of the I/O Manager, you find two buttons for the copy and paste function.

 (Copy)  (Paste)

The buttons are active only if a signal table (*Analog or Digital*) of a module is open.

For example, if you want to configure several modules (e.g. *ibaPADU-8*) with similar signals, configure the first signal table completely in one module.

Click on the *Copy* button and the entire signal table is copied to the clipboard.

Now, open a signal table of the same type (analog/digital) in another module and click *Paste*. All data from the clipboard is entered in the table.

Afterwards, just make minor adjustments in the new signal table, e.g. edit signal names. The automatic fill function may help to optimize your work.

A special feature of the copy/paste function is the compatibility with tables in MS Excel or MS Word. If you have the signal data already available in such a table or if you simply prefer to work with these programs, you can edit your signal tables in MS Excel or MS Word and copy them to *ibaPDA* over the clipboard.

Prerequisite: the number of lines and columns as well as the order of columns must be identical.

#### Import and export of I/O configuration data

You can use the text files to edit entire I/O configurations and import them into *ibaPDA*. The import feature supports decimal as well as hexadecimal (prefix "0x") signal addresses.

You can find more information about this in part 7, *The I/O configuration file*.

The following provides useful information on working in the different columns of a signal table.

### 3.6.5 Notes on working in the signal tables

---

#### Auto-filling of columns



You can fill out the cells of a column automatically by clicking on the column head. In most cases the content of the cell where the cursor stands will be copied into the empty consecutive cells below. Cells already containing values will not be overwritten.

If you enter a signal name, which has a digit at the end, and click on the column head, then the underlying empty fields will automatically be filled with the same name and ascending number until the next occupied line.

If you press <SHIFT> while you click on the column head, the cells containing values will be overwritten.

---

#### Module types with S7 operands



If you click the header of the S7 operand column, *ibaPDA* fills in S7 operands with ascending numbering. This works on all module types having an S7 operand column

---

#### Replacing names and units



The "Replace..." function is in the context menu of the signal table. This is an effective function for searching and replacing strings in all fields and columns of the table.

---

#### Enabling / disabling signals



In order to enable or disable signals, just check the box in the first row of the table and click on the "Active" column header. Whenever you intend to enable or disable multiple signals, starting from the beginning or somewhere in the middle of the table, just change the activation of the first signal of this group and click on the column caption. The change will be applied to all channels below.

Entries, such as names or physical units, remain unchanged. They are available again right after reactivation of a signal.

---

#### Setting up the address (not for all modules)



For a default assignment of the addresses within the column, just click on the column header. The addresses are set automatically with reference to the address in the first line and according to the data type of each signal. The bit no. is incremented automatically.

First, configure name, unit and data type of all signals, then enter the first offset address. If you click on the column header, all addresses are set correctly, provided the signals are configured consecutively in the data source. Addresses can be entered or displayed in decimal or hexadecimal ("0x") values.

---

## 4 Standard interfaces (always available)

The standard interfaces include the interfaces that are always available in the I/O Manager of *ibaPDA* even if no interface cards are installed in your computer or interface licenses activated in the dongle.

Name	Connection to...	Link
ibaNet-E	Connection to ibaNet-E-compatible devices, e.g. ibaW-750	<a href="#">↗ ibaNet-E, page 135</a>
ibaCapture	Connection to ibaCapture server, ibaVision process	<a href="#">↗ ibaCapture, page 30</a>
OPC	OPC (DA) interface for OPC Client and Server	<a href="#">↗ OPC interface, page 34</a>
Playback	Program interface for playback of iba data files	<a href="#">↗ Playback, page 42</a>
Signal mapper	Program interface for assigning input signals to internal signals	<a href="#">↗ Signal mapper, page 46</a>
Text interface	Connection to sources of text signals, such as files, serial interface, TCP, UDP	<a href="#">↗ Text interface, page 53</a>
Virtual	Program interface for computation and processing of virtual signals by means of mathematical, technological and many more functions	<a href="#">↗ Virtual, page 54</a>
Unmapped	Storage for unused/unassigned modules	<a href="#">↗ Unmapped, page 116</a>

## 4.1 ibaCapture

*ibaCapture* is an image acquisition and recording system integrated in the *ibaPDA* data acquisition system. It facilitates video recording synchronized to measurement value acquisition. The synchronized combination of image data and measured data allows relationships between the process and the measurements to be easily recognized and communicated.

### Description

The *ibaCapture* interface is displayed by default in the I/O Manager. However, it can only be used in conjunction with an *ibaCapture* or *ibaVision* server.<sup>1)</sup>

There is only one *ibaCapture* interface per *ibaPDA* system. You can add up to 64 subordinate *ibaCapture* and/or *ibaVision* modules to this interface. Each module is assigned to one *ibaCapture* or *ibaVision* process.

The interface provides an overview of all active *ibaCapture* server connections.

A TCP/IP connection is required between the *ibaPDA* server and the *ibaCapture* server.

The *ibaCapture* interface also provides output modules. With these output modules, it is possible to control the camera recording and, with PTZ cameras, to control the cameras.

For more information see part 2, *Outputs*.

### Interface configuration

#### Port no.

The port number is specified in the entry field in the top left. The port number is used to receive synchronization data from one or more *ibaCapture* servers via a TCP/IP connection. The default port number, 9121, can generally be used.

If the default port number is already used in your network for other purposes, you can change it here. The port number is always transmitted to the *ibaCapture* server to inform the video server about the correct port number for exchanging the synchronization messages.

#### <Reset port to default>

The port number 9121 is set.

#### <Allow port through firewall>

When installing *ibaPDA*, the default port numbers of the protocols used are automatically entered in the firewall. If the port number is changed here or the interface has subsequently been activated, it is necessary to allow this port through firewall.

### Network interfaces

Using this drop-down list, you can select which network adapters on your computer should be 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 of the network adapters, an error message will be produced when validating the I/O configuration. By default the option *All* is selected.

<sup>1)</sup> *ibaVision* is a system for intelligent image processing.

### ibaCapture interface licenses

The available *ibaCapture* licenses are shown in the top right of the screen. You need one license for each connection to an *ibaCapture* server.

### Start acquisition even if an ibaCapture server/ibaVision process/ibaCapture-ScreenCAM host is not accessible

If you want to make sure that the measurement starts regardless of whether an *ibaCapture* server is accessible or not, you should check this option.

### Restart acquisition when the configuration of an ibaCapture server has changed

If you select this option, the acquisition of *ibaPDA* automatically starts after a configuration change on the *ibaCapture* server and the changed settings on the *ibaCapture* server take effect immediately.

### Rename camera signals with XML file

If there is the possibility or requirement that the system configuration of one or more *ibaCapture* server(s) will be changed from time to time, e.g., due to the renaming of cameras, then the XML file is a helpful tool.

If this option is selected, then *ibaPDA* reads the XML file which is specified in the "Rename file" field. If necessary, you have to enter the username and password if the XML file is stored on a network drive.

With the <Show example> button you can open a sample file in the standard text editor of the computer (e.g., Notepad), where you can see the correct syntax.

### TCP Port / UDP Port

"OK" is displayed here if the socket can be opened on this port. "ERROR" is displayed if conflicts occur, e.g., if the port is already busy.

### Messages received with invalid length/ID

These counter values should ideally always be 0 (zero). If the values rise, this indicates errors in the synchronization between the *ibaPDA* server and the *ibaCapture* server.

### Connection table

The table contains diagnostic information about the current connections to the *ibaCapture* servers and *ibaVision* processes. One row is created in the table for each connection.

### Available modules

- ibaCapture (input and output module)
- ibaVision input (input module)
- ibaVision output (output module)

Also see ↗ *Module types ibaCapture and ibaVision*, page 32 for a brief description of the modules.

---

## Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaCapture*.

---

### 4.1.1 Module types *ibaCapture* and *ibaVision*

The available *ibaCapture* and/or *ibaVision* modules in a tree structure will be displayed in the signal tree below the *ibaCapture* interface. The modules can only be used together with licensed *ibaCapture* or *ibaVision* systems. The *ibaCapture* interface supports up to 64 modules. Each module is assigned to an *ibaCapture* server or *ibaVision* process.

#### 4.1.1.1 *ibaCapture* module

An *ibaCapture* module is always connected to an *ibaCapture* server. All cameras must have been configured on the *ibaCapture* server by means of the software *ibaCapture Manager*. The virtual cameras like screen cams for screen capturing must also be configured on the *ibaCapture* server.

An *ibaCapture* server supports up to 64 cameras per computer, whereas only a part can be used for video recording at the same time. For the purpose of visualization in *ibaPDA*, all connected cameras can be used.

#### Principle

In order to establish a connection with an *ibaCapture* server, you should add an *ibaCapture* module beneath the *ibaCapture* interface.

1. Enter the general module information like name and time base in the "General" tab. Choose a name which makes it easy to identify the connected video server. The default setting for the number of cameras is 16. If you have more cameras you may change the value up to 64.
2. On the "Video server" tab you can configure the connection to the video server.
3. Click on the <Search> button or enter the IP address or the host name of the video server, i.e. the computer where *ibaCapture* server is running, into the "Address" field. When the requested video server appears in the table, select it and click on <Connect>.
4. Eventually, you can see a live view of the connected camera(s).
5. Click on the <4x4> tool button in the toolbar of the camera pane in order to get an overview of up to 16 cameras. If you have more than 16 cameras click on the up-arrow of the spinner in order to switch to the next group of 16 cameras.
6. in the "Analog" tab you can adjust a lag time for each camera in order to compensate latencies or network running times of the synchronization messages.
7. After applying the configuration, the *ibaCapture* server is shown as a module and the cameras as "signals" in the signal tree.

---

**Other documentation**

See the manual of *ibaCapture* for more information.

---

#### 4.1.1.2 ibaVision input module

An *ibaVision* input module is always linked to an *ibaVision* process. An *ibaVision* input module can send a video stream as well as analog and digital signals towards *ibaPDA*. The configuration of video stream and signals must be done in the *ibaVision* application.

The signals received from *ibaVision* can be displayed, processed and recorded in *ibaPDA* just like usual measuring signals.

**Principle**

In order to establish a connection with an *ibaVision* process, you should add an *ibaVision* module beneath the *ibaCapture* interface.

1. Enter the general module information like name and timebase on the *General* tab. Choose a name, which makes it easy to identify the *ibaVision* process. The default setting for the number of signals is 32. If necessary, you can change the number of signals (max. 1000).
2. On the *ibaVision process* tab you can configure the connection to the *ibaVision* application.
3. Click on the <Search> button or enter the IP-address or the host name of the *ibaVision* server, i.e. the computer where the *ibaVision* process is running, into the *Server address* field.
4. When the requested *ibaVision* server appears in the table, mark it and select the desired *ibaVision* application in the *Program* field.
5. In the *Module* field, select the *ibaPDA* output module, which has been configured in the *ibaVision* application and whose data should be received.
6. Click on the <Update signals> button. The signals which have been configured in the *ibaPDA* output module of *ibaVision* are automatically mapped to the signals of the *ibaVision* input module. You can find them in the *Analog* and *Digital* tabs.
7. After applying the configuration, the *ibaVision* process is shown as a module in the signal tree.

---

**Other documentation**

See the manual of *ibaVision* for more information.

---

## 4.2 OPC interface

The OPC interface enables data exchange between *ibaPDA* and one or several OPC servers via OPC variables. Supported standards are DA 1.0 and DA 2.0.

### Note



The OPC UA interface is a licensed interface and is described further below in the Ethernet-based interfaces chapter.

In order to make the measured signals available to other users in the network, *ibaPDA* serves also as an OPC server. Generally, all active signals are available via OPC.

Moreover, there are OPC server modules available containing signals, which can be written by other OPC clients. A maximum of up to 1024 modules are supported per interface.

The OPC interface of *ibaPDA* supports redundant OPC servers.

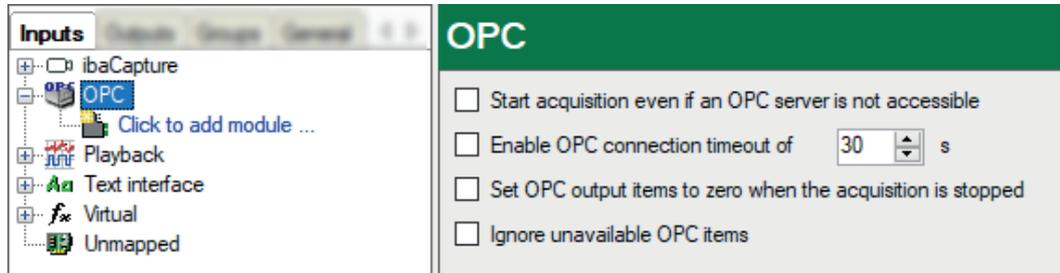
### Note



It is recommended to install .NET Framework 4.5 for a reliable function of the OPC connections.

### Basic settings for the OPC interface

Click on the OPC interface in the I/O Manager tree to select several options.



#### Start acquisition even if an OPC server is not accessible

Enable this option in order to prevent blocking of acquisition when OPC client modules are configured for measurement but not available at start of acquisition. After starting the acquisition, *ibaPDA* will continuously try to reach the OPC server. As soon as there is access to the OPC server, *ibaPDA* will establish the connection during the measurement.

#### Enable OPC connection timeout of ... s

Enable this option and enter a time value if you want to specify the time allowed to elapse for enforcing a connection to the OPC server. After this time elapsed, *ibaPDA* will not attempt to connect to the OPC server until next start of acquisition.

#### Set OPC output items to zero when the acquisition is stopped

If you enable this option, then the values of OPC output items will be set to zero when the acquisition is stopped. If this option is disabled, the OPC output items keep the last actual value they have had at the moment of acquisition stop.

### Ignore unavailable OPC items

If you enable this option, OPC items, which are not available at start of acquisition will be ignored and the acquisition still starts. If this option is disabled, the acquisition will not start unless all OPC items are available.

### General properties

You can define any number of OPC modules in the OPC interface in the tree of the I/O Manager. Each client module can communicate with a different OPC server.

The data transmission from an OPC server to the *ibaPDA* system requires some preconditions:

- Both computers, the one running the *ibaPDA* server and the one running the OPC server, should be logged on to the same domain or work group.
- On both computers, the same user account with the same user name and password should be configured. Administrator rights are required for both machines.
- Furthermore some of the DCOM settings must be checked and configured to guarantee for an adequate OPC communication. These settings are only relevant for the *ibaPDA* server PC, not for the *ibaPDA* client PC.

For the selection of the OPC signals to be measured, an OPC browser is available in the OPC module dialog.

### Available modules

- Module type OPC client, see ↗ *OPC Client module*, page 35
- Module type OPC server, see ↗ *OPC Server module*, page 41
- Module type redundant OPC client, see ↗ *Redundant OPC client module*, page 40

For more information about superordinate OPC settings, see part 2, *OPC Server*.

---

### Other documentation



For additional information about security settings and DCOM configuration, please refer to the OPC connection configurations guide.

This document is available on the "iba Software & Manuals" data medium and in the download area of the iba website.

---

## 4.2.1 OPC Client module

Each OPC Client module represents an OPC client, which may be connected to an OPC server. Each module may connect individually to a different or to the same OPC server.

### 4.2.1.1 OPC – General tab

#### Basic settings

For a description of the basic settings see ↗ *Common and general module settings*, page 20.

## OPC

### Computer name

This is the name of the computer running the OPC server. Enter the computer name here, either by browsing the network (  ) or entering the name or IP address manually.

### OPC server

In this field, select the OPC server which should be used as the data source. The system automatically detects all available OPC server processes running on the above-selected computer. You can select the OPC server process from the selection list in this field.

Just click into the input field and open the selection list.

---

### Note



If, contrary to expectations, no OPC server is shown in the list, this might be caused by a conflict in the Windows security settings of the computer. Please check whether the *ibaPDA* server computer has the necessary access rights on the OPC server system (OPCEnum DCOM settings).

---

### User account

Enter here your logon data for accessing the OPC server computer. To help you, you can open an input dialog with the  button.

If you leave this field empty, the system automatically uses the user account currently running *ibaPDA*. You may also enter a different user account (user name and password) with appropriate access rights on the OPC server computer.

Well-configured security settings on both computers are very important in order to guarantee a proper communication between OPC server and *ibaPDA*. The user account is also used to generate the list of the available OPC servers.

### Group name

This is the name of the OPC group which *ibaPDA* uses when connecting to the OPC server. If this field is left empty, *ibaPDA* uses a unique automatically generated group name.

### Update time

The update time determines how fast data is requested from the OPC server. The update time can be specified independently of the module time base.

### Fastest update time

This value is transmitted from the connected OPC server to *ibaPDA*. It shows the rate at which OPC variables are updated. If the time base of *ibaPDA* is less than the refresh rate of the OPC server, multiple measuring points with the same value will be recorded per update cycle of the OPC server. In this case, a warning message appears when starting the measurement or exiting the I/O Manager.

### Force data type

If an OPC server supports a dynamic change of data types (e.g., *ibaLogic-V4*), this option should be set to TRUE in order to avoid conflicts with file types. If this option is enabled (TRUE), *ibaPDA* forces the OPC server to return to the data type specified in the *ibaPDA* configuration. If this option is disabled (FALSE), *ibaPDA* requests the native data type from the OPC server.

**Do initial read**

If you set this option to TRUE, then *ibaPDA* performs a complete reading process of the OPC items after starting the measurement for this OPC client.

**Add item attempts**

By setting this value, you can specify the number of attempts that *ibaPDA* makes to add the requested OPC items to a group. The default setting is 1.

For some OPC servers, one attempt may not be enough because they do not accept items until they have completely finished loading their configuration. There is a one-second pause between two attempts.

**"Connect/disconnect", "Add signals" and "Clear" hyperlinks**

Click on the *Connect* hyperlink in order to establish a connection to the OPC server. Click on the *Add signals* hyperlink in order to add signals to your OPC module. An OPC browser assists you when searching for OPC tags. In order to delete signals from the signal table, select the corresponding signals and click the *Delete* hyperlink.

**Further information about OPC connections**

Both computers, the one running the *ibaPDA* server, and the one running the OPC server, should be logged on to the same domain or work group.

On both computers, the same user account with the same user name and password should be configured. Administrator rights are required for both machines.

There are sometimes significant differences in the settings between different Windows versions. You can find more information about this in a separate document.

---

**Other documentation**

For additional information about security settings and DCOM configuration, please refer to the OPC connection configurations guide.

This document is available on the "iba Software & Manuals" data storage medium and on our website in the download section.

---

### 4.2.1.2 Adding Signals from an OPC Server

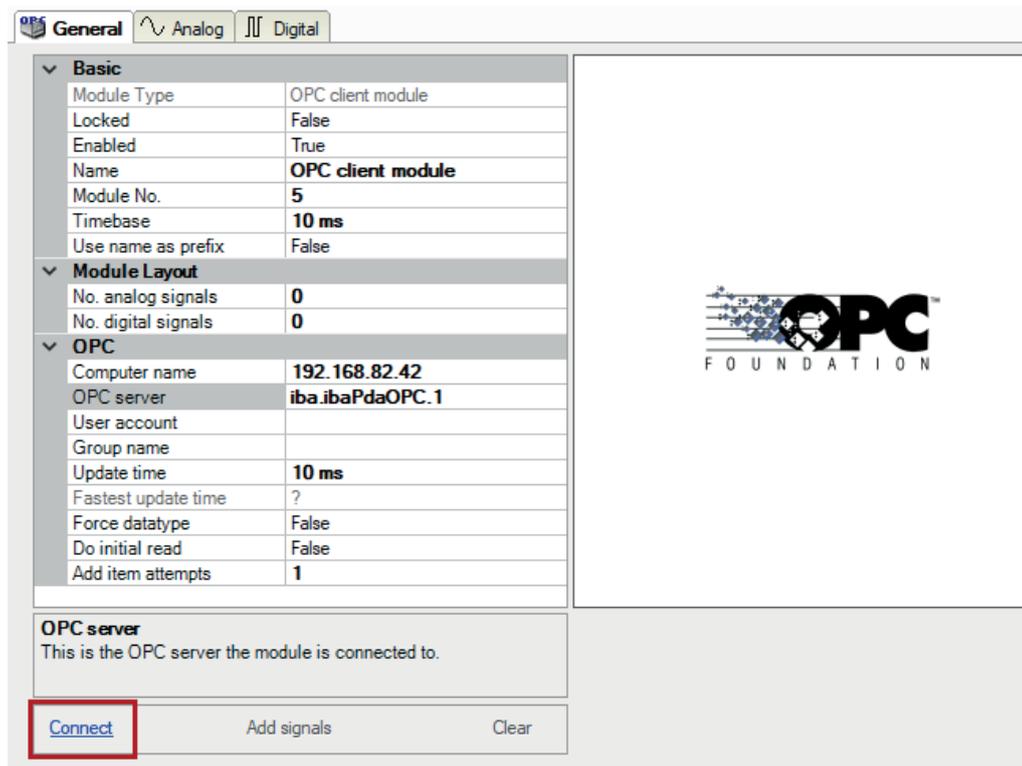
At the bottom of the *General* tab, you can find three hyperlinks:

"Connect", "Add signals" and "Clear".

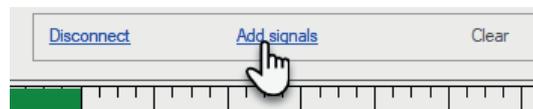
As long as there are not entries in the signal tables, the hyperlinks "Add signals" and "Clear" are disabled.

In order to add OPC variables to the signal tables, follow these steps:

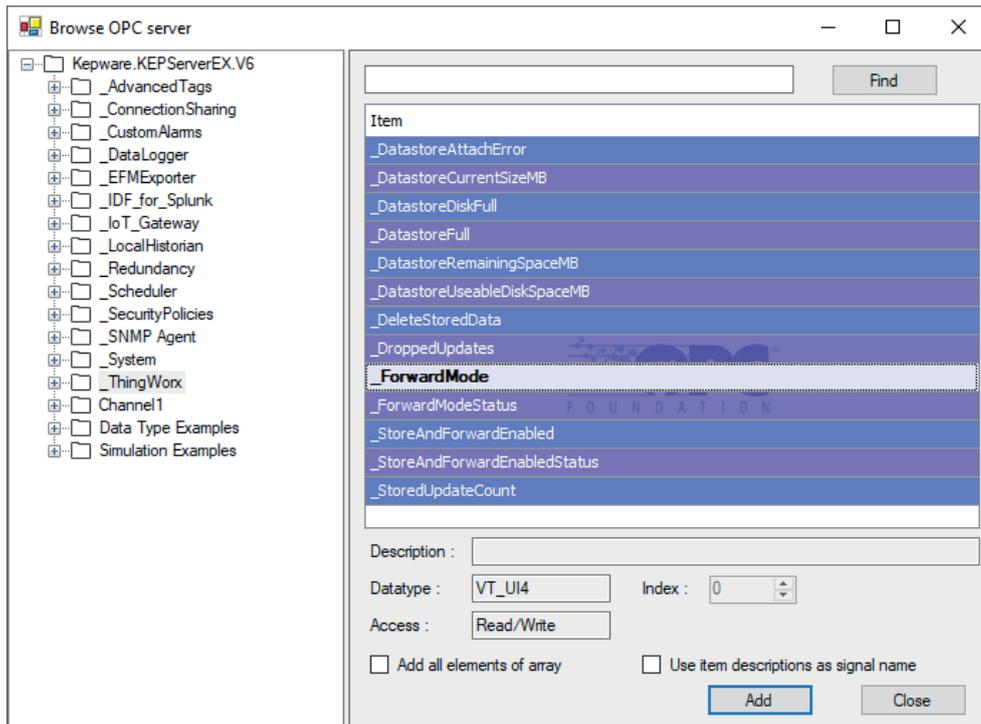
1. After you have made the settings for the OPC server in the *General* tab, click on "Connect".



2. If the settings are correct, the connection physically installed and the OPC server active, the "Add signals" hyperlink should become active shortly after. Click on the "Add signals" hyperlink.



3. The OPC browser window opens showing all OPC variables of the corresponding OPC server. Select the OPC tags you want to be taken into the *ibaPDA* signal tables for measurement. You can select single or multiple signals (<Ctrl>+mouse click) or even complete sets (<Shift>+mouse click).



4. When you have selected the signals, click <Add>. The signals are automatically taken into the next free table rows. Analog and digital tags are recognized and separated automatically.
5. If all signals have been added to the signal tables, close the dialog with click on the <Close> button.
6. Disconnect the OPC server with the "Disconnect" hyperlink if you want to continue working in the I/O manager. If you want to leave the I/O Manager anyway, just click <OK> in the I/O Manager dialog. The OPC server is then disconnected automatically.

### 4.2.1.3 OPC – Analog and Digital tab

#### General columns in the signal table

For a description of the general signal table columns see [Columns in tables with analog and digital signals](#), page 22.

#### Item ID

The entries in this column are filled-in automatically when adding OPC variables with the OPC browser. If you want to change an item ID, just click in the corresponding field and then on the browser button . The OPC browser opens again and you can select another signal.

Of course, you can directly enter the item ID as well. When moving the mouse over the signal, you will get information in the tooltip on whether or not the signal is valid.

**Tip**

Always use the OPC browser. Beside an easier selection and avoiding typing errors, there is an automatic data type verification.

E.g., if you select a binary OPC variable for input in the analog signal table, the OPC browser marks this data type conflict with a red-colored field in the data type display and disables the <OK> button.

This ensures that only data of the appropriate type is entered in the signal tables.

## 4.2.2 Redundant OPC client module

The redundant OPC client module can be used in conjunction with redundant OPC servers. A redundant OPC server configuration consists of two OPC servers using the same data model. Only one OPC server – the "primary" OPC server – provides data at a certain time. In case of failure, stop or shutdown of the primary OPC server (failover), the secondary OPC server assumes the data provision. This procedure also works vice versa.

All OPC tags to be measured are configured only once, as if there was only one OPC server as data source. *ibaPDA* uses a special control element to select the respective OPC server for the provision of valid data.

### 4.2.2.1 Redundant OPC client – General tab

#### Basic settings

For a description of the basic settings see [↗ Common and general module settings, page 20](#).

In general, these settings correspond to those of the default OPC client module. Only two OPC servers have to be configured.

#### Primary OPC server

This OPC server is addressed after the start of the acquisition first and regarded as an OPC server for normal operation.

#### Secondary OPC server

This OPC Server takes over the data supply in the event that the primary OPC server is disrupted or turned off. This OPC server runs on a different computer than the primary OPC server for logical reasons.

#### Redundancy

##### Failover item ID

This setting enables *ibaPDA* to monitor the correct OPC server.

*ibaPDA* monitors this item on both primary and secondary OPC servers. If the item value on a server is 1 (TRUE), then *ibaPDA* will switch to that server. If the item value on both servers is 0, *ibaPDA* will monitor the primary server.

After the connection to at least one OPC server has been established, select the failover item by using the OPC browser.

#### 4.2.2.2 Redundant OPC client – Analog and Digital tab

##### General columns in the signal table

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals](#), page 22.

#### 4.2.3 OPC Server module

In order to make the measured signals available to other users in the network, *ibaPDA* serves also as an OPC server.

All active signals are available over OPC, even without the configuration of an OPC server module.

But you also may add an OPC server module for the following purpose.

In the signal tables (analog and digital) of an OPC server, you can define the signals hosted by *ibaPDA* as OPC server, but can be written by other OPC clients. This means that OPC clients can write values on signals that are provided by the OPC server module of *ibaPDA*.

Up to 32 analog and 32 digital signals per module are supported.

##### 4.2.3.1 OPC server module – General tab

###### Basic settings

For a description of the basic settings see [↗ Common and general module settings](#), page 20.

##### 4.2.3.2 OPC server module – Analog and Digital tab

###### General columns in the signal table

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals](#), page 22.

## 4.3 Playback

This interface is used to replay an iba data file (\*.dat). Every data file recorded in *ibaPDA*, *ibaQDR* or *ibaLogic*, even from older program versions, can be replayed.

The playback interface is one of the few interfaces, which are always available even without a license key (dongle). Thus, it serves as a data source for the demo mode of *ibaPDA* if no real signals are available.

With a regularly licensed *ibaPDA* system, signals from a data file can be replayed and recorded along with real, current measurement signals.

Possible applications are, for example:

- Training of personnel in using *ibaPDA*
- Anonymizing recorded data (removing customer-specific terms and names)
- Creating multilingual data files with the same contents but in different languages
- Sharing experience in the process
- Simulation and test

### Using playback

1. Open the *ibaPDA* I/O Manager.
2. Click on the "Playback" branch in the I/O Manager tree.
3. Check the *Active* option.
4. Select the data file.  
Either enter the name of the data file that you want to play or look for it in the browser. If necessary, enter the name and password to gain access to the network drive.  
If the data file has been protected with a file password, enter the password in the field *File password*.
5. Click button <Read dat file>.  
This instructs the *ibaPDA* server to read out the data file and generate the modules and signals that are available in the data file.
6. Now choose how to proceed.  
There are multiple options available, depending on whether or not playback modules already exist in the configuration:  
If no Playback modules are available, choose from:
  - Add playback modules  
If there already are modules in your I/O configuration, this option keeps the existing modules and adds the data file modules as playback modules. If a module number is already used by another module, which is no Playback module, then the Playback module is assigned a new number.  
This is useful if you also want to monitor the actual values beside the replayed data.
  - Remove all modules  
If there already are modules in your I/O configuration, this option removes all modules

currently available from the configuration. Only modules from the data file will still be available as Playback modules, using the same order and with numbering as in the data file.

If there already are Playback modules, choose from:

- No changes  
The data file is read only. Existing modules and module structure remain unchanged.
- Replace playback module  
Existing playback modules are removed and replaced completely with modules from the data file.
- Remove all modules  
See above

Example of a configuration dialog of the playback interface with signals

After loading the data file, a signal tree and a preview of the first signal of the data file appear in the lower part of the dialog. You can select any signal from the signal tree for preview.

### Note



The playback interface skips length-based signals and signals with a 3-level channel number like [0.0.0] (e.g. dig512 modules or QDR data files).

## 7. Setting up the data area

By default, *ibaPDA* continuously replays the file from start to end. You can change the start and stop position by moving the green start and stop marker or by entering the start and stop position manually.

The range between the two markers will be replayed continuously without interruption as long as the acquisition runs.

During acquisition, a third marker appears on the graph. This is the red progress marker. It is positioned at the current playback time.

As for all other modules, you can change all properties and signals for the playback module.

8. Set the playback speed

You can also define the playback speed in the input field on top of the dialog. By default, the speed is set to 1.00, which is the original speed. The original speed is determined by the timebase of the data file at its original recording.

Irrespective of the timebase currently set in *ibaPDA*, the timebase of the playback module is specified according to the timebase of the original data file.

The following figure shows the differences between varying playback speed settings.

9. End the configuration by clicking <OK>.

The acquisition is automatically restarted; you can now see the playback module and the signals of the data file in the signal tree of *ibaPDA*. Proceed with the playback signals as with any other signal.

**Example: Results of differently set playback speeds**

An extract of a data file with a length of exactly 10 seconds (red signal) is replayed continuously at the following speeds:

- a) original speed
- b) double original speed
- c) half original speed

The blue signal is a real signal of 1 Hz, sampled with 1 ms.

The timebase of the data file is 20 ms. The current timebase of *ibaPDA* is 1 ms.

<p>a)</p>		<p>Speed: 1.0</p> <p>Red curve: 1 sample every 20 ms</p> <p>Blue curve: 1 sample every ms</p>
<p>b)</p>		<p>Speed: 2.0</p> <p>Red curve: 1 sample every 10 ms</p> <p>Blue curve: 1 sample every ms</p>
<p>c)</p>		<p>Speed: 0.5</p> <p>Red curve: 1 sample every 20 ms, yet twice (value changes every 40 ms)</p> <p>Blue curve: 1 sample every ms</p>

## 4.4 Signal mapper

The signal mapper interface is hosting so called signal mapper modules.

Signal mapper modules basically create an abstraction layer between the input layer and the processing or visualization layer. Input signals entering the *ibaPDA* system over various interfaces, can be mapped to internal signals.

The signals of a signal mapper module are used internally as input or source signals for further processing applications or visualization (e.g. for QPanel displays).

The signal mapper interface and modules do not require any license. Hence, you can create views and QPanel layouts even without an *ibaPDA* license if those are supplied by signals of signal mapper modules.

Use signal mapper modules in order to test the function of layout elements, even if the real input signals are not available yet. Therefore, create an I/O configuration which contains only signal mapper modules and no input signals. You can use the default values of the signal mapper signals for testing, while it is possible to change the values in the watch window.

If using multiple *ibaPDA* systems, apply signal mapper modules to achieve a certain standardization in terms of data processing and visualization.

The following module types are available for this interface:

Module Type	Function
Signal mapper ↗ <i>Signal mapper module</i> , page 46	Mapping of an arbitrary number of input signals or virtual signals to internal signals for further processing or visualization.

### 4.4.1 Signal mapper module

A signal mapper module serves as a translator between input signals and internal, standardized signals. Similar to a terminal bar of a device, you can "pre-wire" the required signals for configuring your application such as a QPanel visualization. Later, you only have to assign the real input signals to the corresponding signals of the module.

The signal mapper module supports the use of profiles. Consequently, the signals of a module are automatically created by assigning a profile to it. A profile can be used as often as you want by multiple modules. The number of profiles is unlimited.

With a profile you define which signals shall be used for further processing or visualization. Name these signals in a general but adequate way with regard to the intended application.

Because signal mapper modules are not licensed, the number of modules is unlimited too.

The signals of a signal mapper module are not taken into account in terms of the licensed size of your *ibaPDA* system.

### 4.4.1.1 Signal mapper - General tab

#### Basic settings

For a description of the basic settings see ↗ *Common and general module settings*, page 20.

In contrast to other modules there is no timebase setting for a signal mapper module. The reason is that a signal mapper module does not actually process signals on its own but just maps the signals. The timebases of the input signals are always adopted even if the signals come from different sources with different timebases.

#### Profile

Select here the profile which should configure this module.

If no profile is available yet, you can switch to the profile configuration dialog by clicking on <New profile...> or <Edit profile...>.

Without a profile a signal mapper module has no signals.

The *Analog* and *Digital* tabs are not visible before a profile has been applied.

#### Source signals

This option determines whether the source signals which are linked to the signal mapper modules are still available in the signal tree or not. Enabling this option (True) prevents the same signal to be recorded twice (first source signal, second mapped signal).

#### "Configure profiles" link

Clicking on this link opens the configuration dialog for profiles.

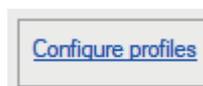
Always when you create a module, you can configure a profile. But you can also configure the required profiles first and create the modules later.

### 4.4.1.2 Signal mapper - Analog/Digital tabs

If no profile has been assigned to a signal mapper module yet the *Analog* and *Digital* tabs are missing. The tabs appear if a profile, which contains the corresponding analog and/or digital signals is assigned to the module.

## 4.4.2 Add and configure signal mapper profiles

To add a profile to a module, click the blue *Configure profiles* link at the bottom in the *General* tab.



Alternatively, you can open the drop down list under *Profile* in the *General* tab and click on <New profile> or <Edit profile>.

In both cases, the *Configure profiles* dialog opens.

Below the left window of the dialog you will find a set of buttons with the following functions:

	Add a profile
	Copy selected profile
	Delete selected profile
	Import profile(s) from a <i>*.signalMapperProfile</i> file
	Export selected profile to a <i>*.signalMapperProfile</i> file

Add a profile and rename it in a way that all relevant information for choosing a profile later is noticeable, e.g. intended application or visualization element.

Make your settings in the three tabs.

### General tab

#### Selection of metadata sources of the signals

Here, you can determine if and which metadata should be adopted from the source signal. Moreover, you can allow (default) or prohibit further editing of the data in the signal mapper module.

The metadata of a signal are name, comment 1 and comment 2, complemented by unit and datatype for analog values.

In the *Analog* and *Digital* tabs you define the signals which will be provided by the signal mapper module. There you can enter the metadata.

The source signals which will be assigned to the signal mapper module have their own metadata, at least a signal name.

#### Analog and Digital tab

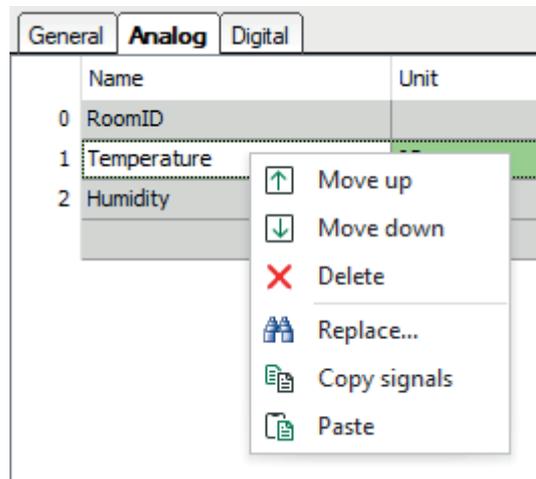
Here, define the analog and digital signals which any signal mapper module will provide after having assigned this profile. If you have filled one row (at least the signal name) and click in the empty space below, a new row will be added.

The datatype of analog values only distinguishes between *Text* and *Numeric*. Neither the length of text nor the datatype have to be specified in detail because they will be adopted from the source signal later.

If you create multiple rows you can organize the rows by using the buttons on the right side.

	Move selected row(s) up
	Move selected row(s) down
	Delete selected row(s)
	Copy all rows
	Paste from clipboard, starting from the selected row

You will find the same functions in the context menu after a right mouse click on the table.

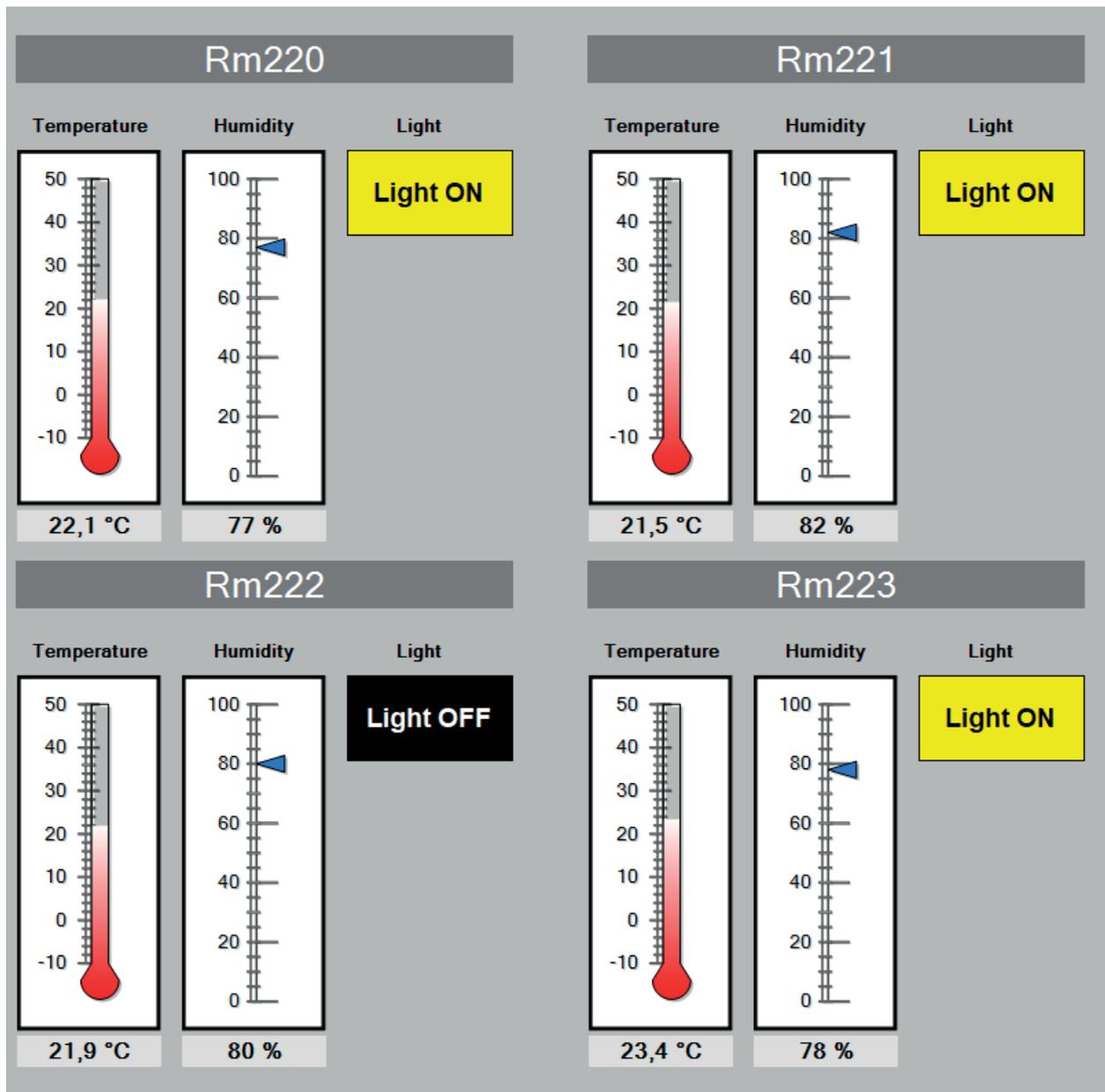


### 4.4.3 Signal mapper Example

#### Task

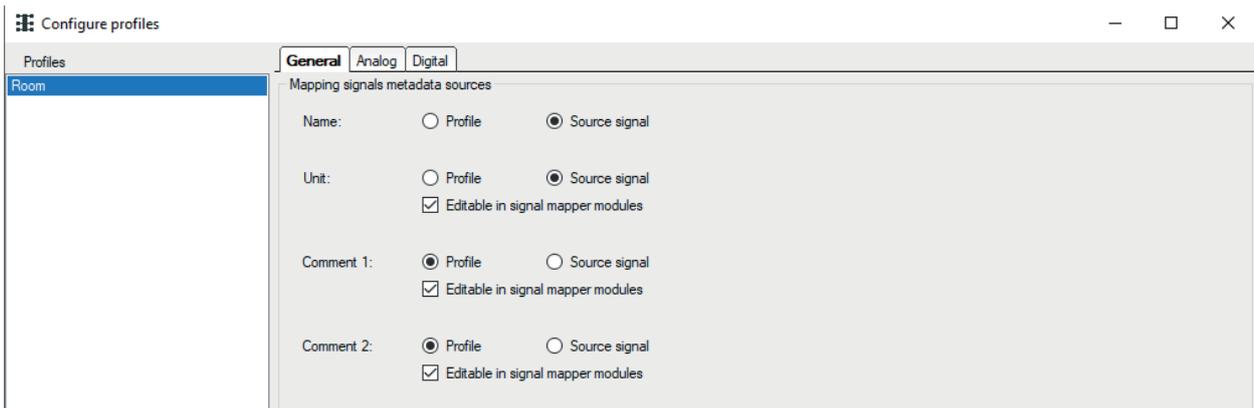
The measured values of temperature and humidity as well as the status of the lighting in four rooms in an office building should be acquired and visualized with ibaQPanel.

The QPanel layout should be designed before the real input signals are available. Hence, signal mapper modules will be used for the data supply of the QPanel elements.

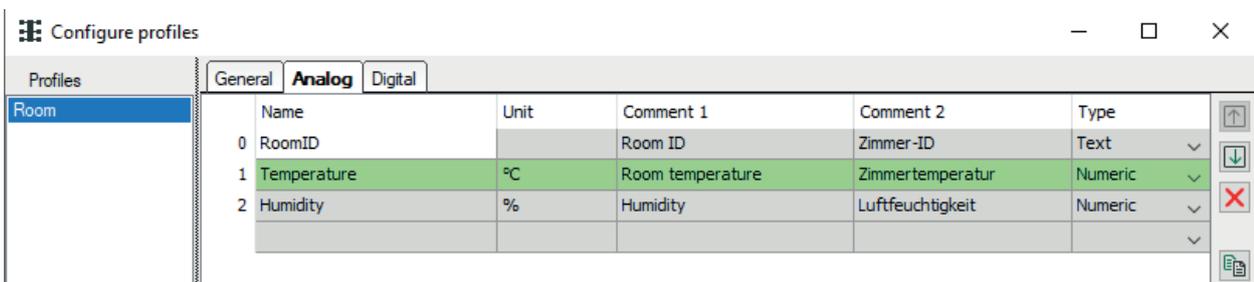


### Implementation

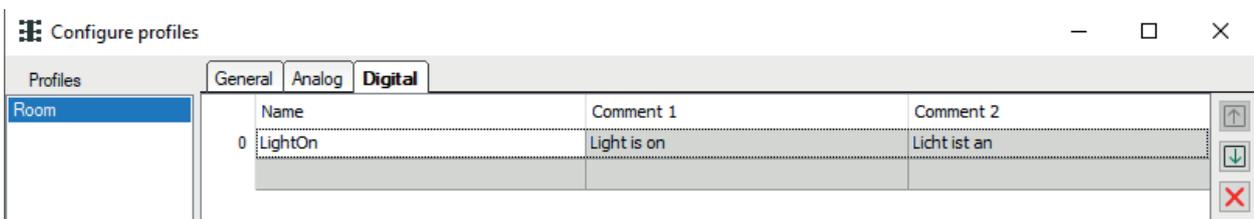
1. Add a first signal mapper module and rename it (module name "Room\_01").
2. In the general module settings click on the link *Configure profiles*.
3. Add a profile and rename it ("Room")
4. On the General tab of the profile decide which metadata should be adopted from the source signal and which should be retained by the profile. In this case, signal name and unit will be adopted from the source signal.



5. On the *Analog* tab define the analog values, which are needed for the visualization (RoomID, Temperature, Humidity).

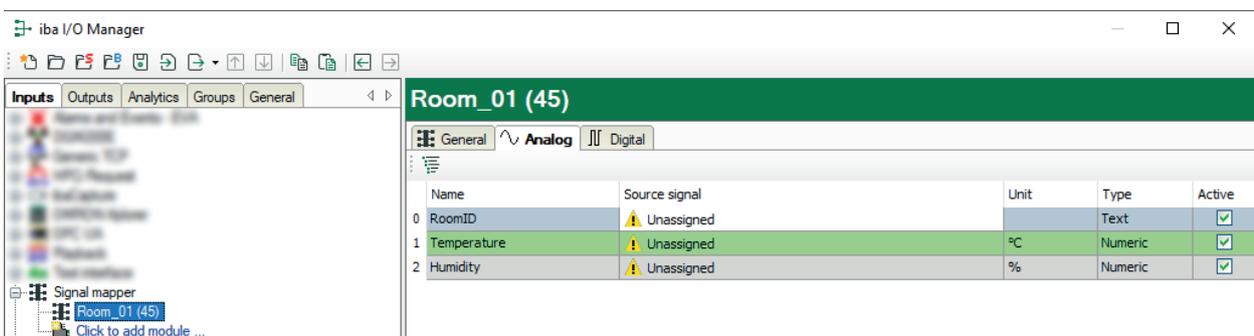


6. On the *Digital* tab define the digital signals, which are needed for the visualization (LightON).



7. Close the *Configure profiles* dialog by <OK> and select the profile "Room" under *Profile* in the general module settings.

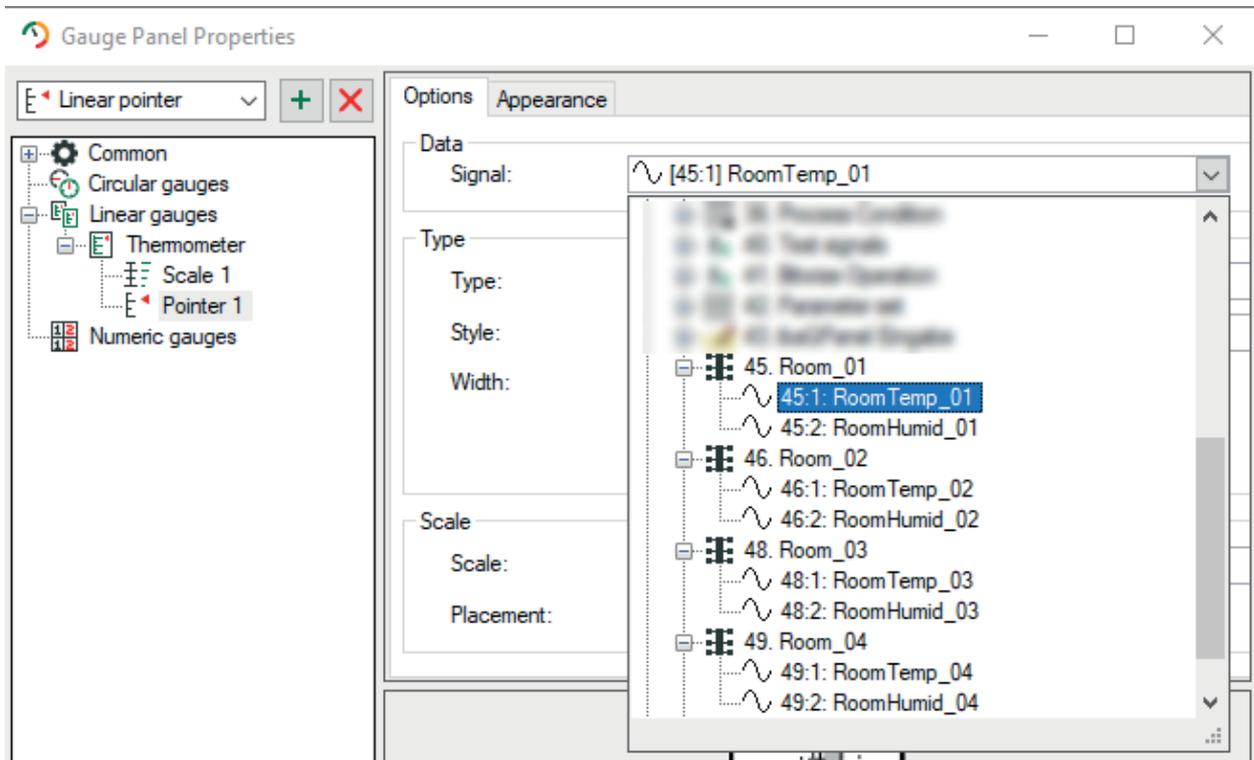
8. Together with the profile the module automatically gets its *Analog* and *Digital* tabs containing the defined signals. The source signals are not assigned yet.



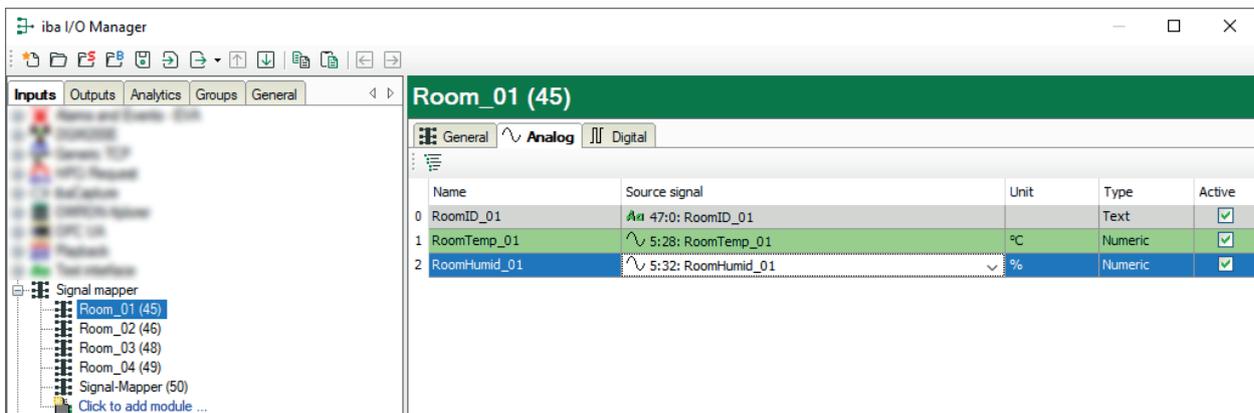
Copy the signal mapper module three times and rename each of them, resulting in four modules "Room\_01"... "Room\_04". All four modules still have the same signal names.

10. Design the QPanel elements for the display and group them for one room. Then copy this group three times.

Link the QPanel elements of each room with their respective signals of the signal mapper modules "Room\_01" to "Room\_04". The input signals of the QPanel elements still have the same name (e.g. "Temperature") although they come from different modules.



10. If the source signals are available you can assign them to the signal mapper modules on the *Analog* and *Digital* tabs. Because we have configured in the profile that the signal name should be adopted from the source signal, the signal names of the signal mapper modules change correspondingly.



## 4.5 Text interface

Text signals can be received via different PLC or manufacturer-specific interfaces, where they are typically processed as analog signals with the STRING[x] data type.

The text interface also offers the possibility to receive text information in other ways. For this purpose, the following modules are available via this interface:

Module type	Function
TCP text	<i>ibaPDA</i> receives the text as a TCP/IP telegram via the network interface.
UDP text	<i>ibaPDA</i> receives the text as a UDP datagram via the network interface.
Serial text	<i>ibaPDA</i> receives the text via a serial interface (COM1...COM4). Converters are supported, e.g., from RS232/RS485 to USB.
File text	The text is available in a text file (*.txt, *.csv, *.json), which is opened and read by <i>ibaPDA</i> .

All these module types can be adapted to the source text in order to read out the desired information and convert it into text signals.

Detailed information about the handling of text signals as well as the descriptions of the interface and modules can be found in part 2, *Text signals and text processing*.

## 4.6 Virtual

The virtual interface can be found in the Analytics tab of the I/O Manager. When updating from older *ibaPDA* versions to v8.4.0 or higher this interface will be moved automatically to the *Analytics* tab.

The *virtual* interface is a special feature, which enables the user to create virtual signals, i.e. generate or calculate signals and to use them like real input signals for acquisition. Calculation results in virtual modules can be stored with high accuracy (64 bit).

A group of different module types is available for this interface. Each module offers predefined functions for particular purposes. A maximum of 4096 modules are supported at the *Virtual* data interface.

Module types	Function	Application fields
➤ <i>16 bit decoder</i> , page 57	Splits a 16 bit integer (e.g. analog input) into 16 single bits (digital signals)	Using analog values as container for digital signals (e.g. status words)
➤ <i>32 bit decoder</i> , page 59	Splits a 32 bit integer or real (e. g. analog input) into 32 single bits (digital signals)	Using analog values as container for digital signals (e.g. status words)
➤ <i>Multidecoder module</i> , page 60	Collection of multiple decoders (8, 16 or 32 bit) in one module; decoding and naming of the discrete bits are determined by profiles.	Using analog values as container for digital signals (e.g. status words)
➤ <i>16-bit encoder</i> , page 66	Packs multiple digital signals into one 16-bit integer analog signal	Transmission of multiple binary data in one analog value (e.g. status word)
➤ <i>32-bit encoder</i> , page 67	Packs multiple digital signals into one 32-bit integer analog signal	Transmission of multiple binary data in one analog value (e.g. status word)
➤ <i>Electric modules</i> , page 68	Provide effective values for voltage and current as well as power factor, reactive, real and apparent power in a standard delta, single-phase, two-phase or star system.	Power generation and distribution.
➤ <i>Trigger module</i> , page 75	Provides customized digital trigger signals based on complex trigger conditions.	Event-driven control of data storage and/or creation of events in HD event store
➤ <i>Virtual</i> , page 80	Provides customized virtual signals created with the expression builder.	Any mathematical or logical calculation
➤ <i>Virtual retentive</i> , page 83	Provides analog virtual signals, which have been created with the expression editor.	When restarting the acquisition the signals in this module retain their last value or use a configured default value, e. g. for elapsed time meters.

Module types	Function	Application fields
➤ <i>Shift register</i> , page 84	Stacks actual values of a signal, controlled by a trigger.	Visualization of the last n values
➤ <i>ibaQPanel input</i> , page 72	Provision of the signals specified via the QPanel input element. The function is now integrated in the ibaQPanel text input.	Still available for compatibility with older QPanels; only the text input element is available for creating new QPanels.
➤ <i>ibaQPanel text input</i> , page 74	Provides text signals, which can be determined via the ibaQPanel text input control.	Interactive visualization of measurement; Adding manually entered texts to the data file
Text shift register See part 2, <i>Text shift register</i> .	Stacks actual values (texts) of a text signal, trigger-controlled.	Visualization of the last n values (texts)
Text creator See part 2, <i>Custom text (text creator)</i> .	Used to configure custom text signals	Creation of static or dynamic texts
Text splitter See part 2, <i>Text splitter module</i> .	Creates additional text signals or signals of other data types from a text signal	Splits long texts into smaller units, conversion of digits into numerical values
➤ <i>Computation module</i> , page 85	Provides arbitrary, self-generated signals similar to the virtual module with the difference that calculation operands can be occupied by placeholders. The calculation is saved in a profile.	Efficient configuration of extensive, complex, recurring and/or protection deserving calculations. Comparable to "Macros" in ibaAnalyzer.
➤ <i>Computation retentive module</i> , page 93	Basically, this module works the same way as the the computation module but it allows the use of retentive functions.	When restarting the acquisition the signals in this module retain their last value or use a configured default value, e. g. for elapsed time meters.
➤ <i>Lookup table</i> , page 94	Translation of different values based on a predefined profile via selection of key values	E.g., to translate error codes into understandable, meaningful error messages.
➤ <i>Parameter set</i> , page 105	Selection of different analog and digital parameters stored in a profile and selected via the value of a signal	Application of different parameters depending on an input signal, e.g. calculation of material-dependent quantities such as weight, heating curve, etc.
➤ <i>Process condition</i> , page 99	Definition of different process states that are stored in a profile.	Consideration of the influences of different process conditions during process analysis, such as material properties, geometries, temperatures, speeds, etc.

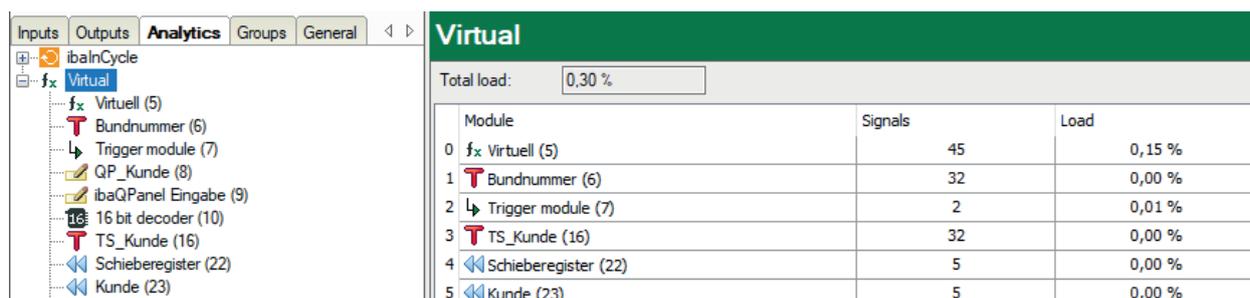
Module types	Function	Application fields
➤ <i>NMEA 0183 decoder</i> , page 109	Decoding a telegram according to NMEA 0183 standard, e.g. reading GPS information	Acquisition of GPS coordinates for tracking moving goods (ships, locomotives, trucks, etc.); visualization on map with ibaAnalyzer maps
➤ <i>Vector calculation module</i> , page 112	Calculation of statistical values on the crossprofile of a vector (min. max, median, average, sum, standard deviation)	E.g. determination of maximum and minimum values of a measured variable over the width of a web-shaped product over time

#### 4.6.1 Diagnosis of the processor load by virtual signals

Extensive calculations, many virtual signals or complex mathematical operations can put a heavy strain on the CPU of the *ibaPDA* computer.

So that you as the user can maintain control over how resource-intensive the calculation of your virtual signals is, the I/O Manager provides an overview of the processor load that each virtual module produces.

Select the *virtual* node in the interface tree to see the overview.



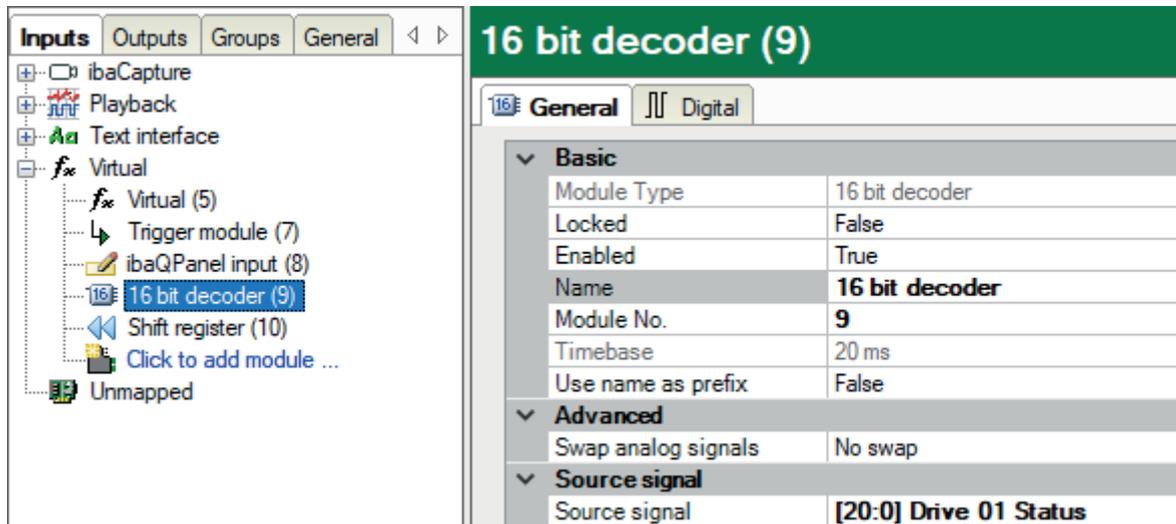
Virtual		
Total load: 0,30 %		
Module	Signals	Load
0 f <sub>x</sub> Virtuell (5)	45	0,15 %
1 T Bundnummer (6)	32	0,00 %
2 ↳ Trigger module (7)	2	0,01 %
3 T TS_Kunde (16)	32	0,00 %
4 ↳ Schieberegister (22)	5	0,00 %
5 ↳ Kunde (23)	5	0,00 %

The number of signals contained and the proportional load are displayed next to the name of the module.

The load value in the "Load" column indicates how much time *ibaPDA* takes to calculate the virtual signals. This time is specified here as load in %, i.e., as the percentage used in the last second on the calculation.

## 4.6.2 16 bit decoder

The *16 bit decoder* module can be used to create 16 digital signals out of one analog signal. The analog signal is converted to a 16 bit integer and each digital signal corresponds to one of the bits from 0 to 15. The source of the analog signals can be any other module.



### 4.6.2.1 16 bit decoder – General tab

#### Basic settings

For a description of the basic settings see ↗ *Common and general module settings*, page 20.

The time base of the decoder module is always the same as the time base of the source signal.

#### Swap mode

Set the swap mode according to the signal source.

You can choose between the following 4 options:

Mode	16 bit
No swap	AB
Depending on data type	BA
Swap 16 bit	AB
Swap 8 bit	BA

The swap mode to be selected depends on the swap mode of the signal source.

#### Source signal

Select the source signal to be decoded from the signal tree.

### 4.6.2.2 16 bit decoder – Digital tab

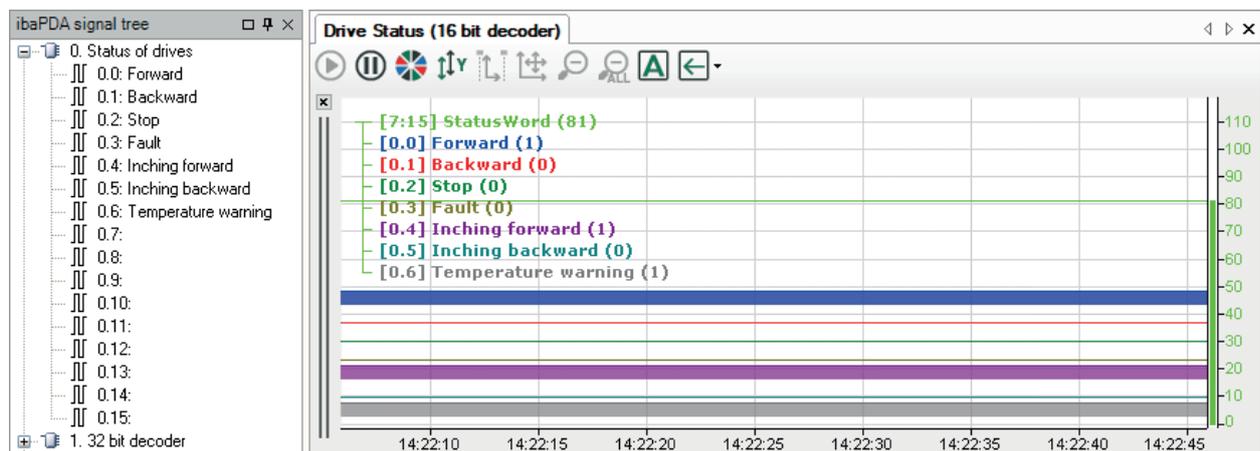
In the *Digital* tab, you can name the 16/32 digital signals.

#### Example

The source is a 16 bit integer with status information from a drive:

16 bit decoder (9)		
16 General Digital		
Name		Active
0	Forward	<input checked="" type="checkbox"/>
1	Backward	<input checked="" type="checkbox"/>
2	Stop	<input checked="" type="checkbox"/>
3	Fault	<input checked="" type="checkbox"/>
4	Inching forward	<input checked="" type="checkbox"/>
5	Inching backward	<input checked="" type="checkbox"/>
6	Temperature warning	<input checked="" type="checkbox"/>

Status-to-bits assignment in the 16 bit decoder module as known from the source.



Display of the individual bits of an integer. The integer value is 81 according to bits 0, 4 and 6 set to TRUE (1). Here, it can be seen that the drive is moving forward slowly and with temperature warning.

### 4.6.3 32 bit decoder

The *32 bit decoder* module is similar to the *16 bit decoder* module. The behavior of this module depends on the data type of the source signal and whether or not "Convert to integer" is enabled.

If the data type is floating-point and the "Convert to integer" option is set to FALSE, the value is treated as a bitmask and the bits are extracted. This is the same behavior as the GetFloatBit function in *ibaPDA* and the GetBitMask function in *ibaAnalyzer*.

If the data type is not floating-point or the "Convert to integer" option is set to TRUE, the value is rounded first to the nearest integer and subsequently the bits are extracted. This is the same behavior as the GetIntBit function in *ibaPDA* and the GetBit function in *ibaAnalyzer*.

#### Note



Certain interfaces automatically convert 32 bit integer values into floating point values during the data transfer between source system and *ibaPDA*. If this applies to the source signal, the original bit pattern will be broken. A warning is generated during the I/O configuration validation in such a situation. Currently affected interfaces are:

- OPC interface
- SIMOLINK module
- HPCi Request interface
- SIMADYN interface
- TDC interface
- PACO4 module
- X-Pact interface
- Playback interface

#### 4.6.3.1 32-bit decoder – General tab

##### Basic settings

For a description of the basic settings see [↗ Common and general module settings, page 20](#).

The time base of the decoder module is always the same as the time base of the source signal.

##### Swap mode

Set the swap mode according to the signal source.

You can choose between the following 4 options:

Mode	32 bit
No swap	ABCD
Depending on data type	DCBA
Swap 16 bit	CDAB
Swap 8 bit	BADC

The swap mode to be selected depends on the swap mode of the signal source.

### **Source signal**

Select the source signal to be decoded from the signal tree.

### **Convert to integer**

In case that the source signal is a floating point value, you can have the value converted into integer format.

If this option is enabled (TRUE), the floating point value is converted to integer format before the bits are extracted. Only the first 24 bits of the integer are valid.

If this option is enabled (TRUE) although the signal is already an integer, the format remains unchanged.

## **4.6.3.2 32 bit decoder – Digital tab**

In the *Digital* tab, you can name the 32 digital signals.

## **4.6.4 Multidecoder module**

With a Multidecoder module you can configure multiple decoders in one module, in order to decode analog signals into digital signals. The configuration of many decoders can be done easier and more clearly compared to single decoders.

One Multidecoder module can contain between 1 and 512 decoders. The default value is 32.

Each decoder can be configured for 8, 16 or 32 bits. You can mix decoders of different sizes in a Multidecoder module.

The Multidecoder module supports the use of profiles, which provide different decoder configurations to be assigned to the decoders individually. By means of profiles you can configure and reuse different decoder types for a wide range of applications.

### **4.6.4.1 Multidecoder – General tab**

#### **Basic settings**

For a description of the basic settings see [↗ Common and general module settings, page 20](#).

#### **Module layout**

Set up here the requested number of decoders to be used in this module.

Possible values: 1 to 512

You can add or remove decoders at any time later.

The discrete decoders can be found in the *Digital* tab.

#### **"Configure profiles" link**

Clicking on this link opens the configuration dialog for profiles.

It's a reasonable approach to configure the requested profiles before configuring the decoders. When configuring a decoder, you only have to select the right profile.

### 4.6.4.2 Multidecoder basic settings

According to the set number of decoders there is a corresponding number of signal groups available in the *Digital* tab, which can be extended or collapsed.

Decoder	Signal Id	Profile	DataType	Convert to integer	Active
0	⚠ Unassigned	<No profile>	WORD	<input checked="" type="checkbox"/>	<input type="checkbox"/>
1	⚠ Unassigned	<No profile>	WORD	<input checked="" type="checkbox"/>	<input type="checkbox"/>
2	⚠ Unassigned	<No profile>	WORD	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3	⚠ Unassigned	<No profile>	WORD	<input checked="" type="checkbox"/>	<input type="checkbox"/>
4	⚠ Unassigned	<No profile>	WORD	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Each group line stands for one decoder. You do the basic set up of the decoders in the columns of each group line.

#### Decoder

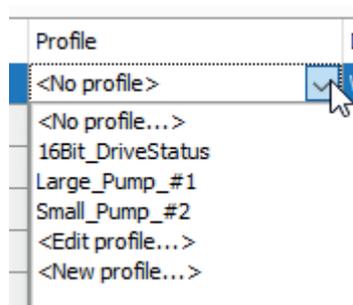
Enter here a comprehensible and unique name, which tells the purpose of this decoder, e.g. `MainDriveStatus_#1`

#### Signal Id

Select the analog signal which should be decoded in the *Signal Id* column. As long as no signal has been selected, the default setting is "Unassigned". If the signal id is unassigned the decoder cannot be activated.

#### Profile

In the *Profile* column select the profile for decoding.



If no profile is available yet, you can switch to the profile configuration dialog by clicking on `<New profile...>`.

#### Data type

The data type determines the endianness and the number of bits available in the current decoder. If you select a signal in *Signal Id* the data type will be adapted automatically to the data type of the signal.

- BYTE: Provides 8 bits.
- WORD: Provides 16 bits. WORD\_B is the Big Endian variant.
- DWORD: Provides 32 bits. DWORD\_B is the Big Endian variant.

#### Convert to integer

If you enable this option, the selected signal will be converted to an integer before being processed.

**Active**

You can activate or deactivate a decoder with this option. A decoder cannot be activated if the *Signal Id* is "Unassigned".

**4.6.4.3 Digital signals of the decoder**

If you extend the group line of a decoder you can see a number of lines which corresponds to the number of bits determined by the data type, e.g. 16 bits for an integer. Each line corresponds to a bit, i.e. to a digital signal.

If no profile has been assigned yet, the table is still empty.

You can enter a name and up to two comments for every bit.

If you have already assigned a profile to the decoder, the columns *Name*, *Comment 1* and *Comment 2* will be filled automatically.

If a profile is assigned, the bits cannot be changed.

**Example**

**Multidecoder (44)**

General Digital

Decoder	Signal Id	Profile	DataType	Conv...	Ac...
0 <input type="checkbox"/> Main drive #1	^ 5:27: DriveStatus_MD01	16Bit_DriveStatus	DWORD	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Name	Active
ON	<input checked="" type="checkbox"/>
Forward	<input checked="" type="checkbox"/>
Backward	<input checked="" type="checkbox"/>
Inching	<input checked="" type="checkbox"/>
Speed >	<input checked="" type="checkbox"/>
Warning_Temp_Stator	<input checked="" type="checkbox"/>
Alarm_Temp_Stator	<input checked="" type="checkbox"/>
Warning_Temp_Rotor	<input checked="" type="checkbox"/>
Alarm_Temp_Rotor	<input checked="" type="checkbox"/>
OverCurrent	<input checked="" type="checkbox"/>
Spare_10	<input checked="" type="checkbox"/>
Spare_11	<input checked="" type="checkbox"/>
Spare_12	<input checked="" type="checkbox"/>
Spare_13	<input checked="" type="checkbox"/>
Spare_14	<input checked="" type="checkbox"/>
Spare_15	<input checked="" type="checkbox"/>
	<input type="checkbox"/>
	<input type="checkbox"/>
	<input type="checkbox"/>

#### 4.6.4.4 Add and configure profiles

To add a profile for a Multidecoder module, click the blue *Configure profiles* link in the *General* tab at the bottom of the Multidecoder module.



Alternatively, you can open the drop down list on any decoder in the *Profile* column and click on *<New profile>* or *<Edit profile>*.

In both cases, the *Configure profiles* dialog opens.

Below the left window of the dialog you will find a set of buttons with the following functions:

	Adding profile
	Copy selected profile
	Delete selected profile
	Import profile(s) from a <i>*.multiDecoderProfile</i> file
	Export selected profile to a <i>*.multiDecoderProfile</i> file

Add a profile and rename it in a way that all relevant information for choosing a profile later is noticeable, e.g. number of bits and intended application.

Each profile consists of 32 bits. If a decoder assigned to a profile has less than 32 bits then only the used bits in profile are considered.

Now, enter a name for each bit in the table on the right side. Bits without a name will not be considered for decoding later.

These names will be the signal names of the digital signals. If needed, add some comments. The bit number in the first column is for information only and cannot be changed.

#### Example

Each of the main drives of a production plant provides an integer value, which encodes up to 16 status bits for transmitting the status of the drive. As there are many of these drives in the plant, a Multidecoder profile should be created, which can be assigned to any matching drive.

Name of the profile: 16Bit\_DriveStatus

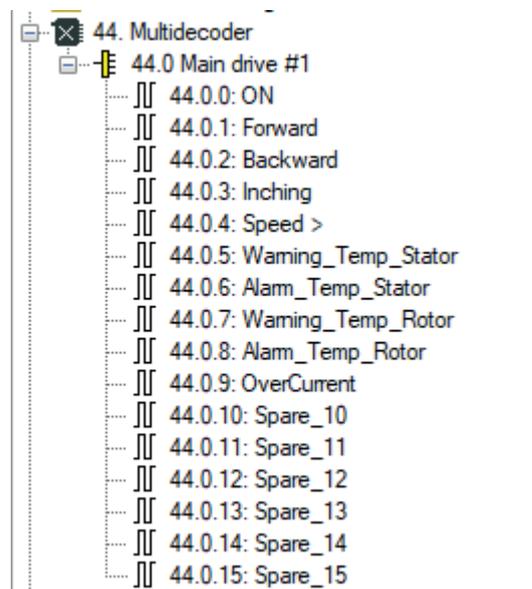
Application: Drive status with 16 bit

Profiles	Bit no.	Name	Comment 1	Comment 2
16Bit_DriveStatus	0	ON	Drive is switched on	
Large_Pump_#1	1	Forward	Drive running forward	
Small_Pump_#2	2	Backward	Drive running backward	
	3	Inching	Low speed for inching	
	4	Speed >	Speed limit exceeded	
	5	Warning_Temp_Stator	Overtemperature in stator	
	6	Alarm_Temp_Stator	Overtemperature in stator	
	7	Warning_Temp_Rotor	Overtemperature in rotor	
	8	Alarm_Temp_Rotor	Overtemperature in rotor	
	9	OverCurrent	Max. current exceeded	
	10	Spare_10	Spare for future use	
	11	Spare_11	Spare for future use	
	12	Spare_12	Spare for future use	
	13	Spare_13	Spare for future use	
	14	Spare_14	Spare for future use	
	15	Spare_15	Spare for future use	
	16			
	17			
	18			
	19			
	20			

#### 4.6.4.5 Representation of Multidecoders

The signal tree shows the Multidecoder module as a main node. One level below, there are the decoders and one level further down the digital signals of the decoders.

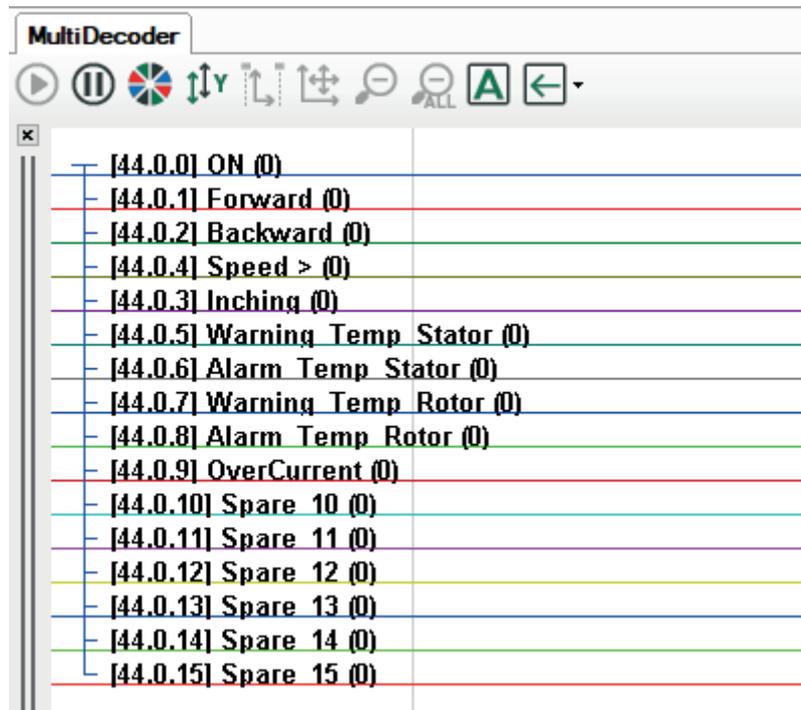
The following figure shows the Multidecoder module of our example (drive status).



You can drag and drop the signals into a trend graph.

You can also drag and drop a complete decoder into a trend graph. If you simply drag a decoder into a trend graph, then each signal will get its own signal strip.

If you press <Shift> at the same time, then all digital signals of the decoder will be shown in the same signal strip.



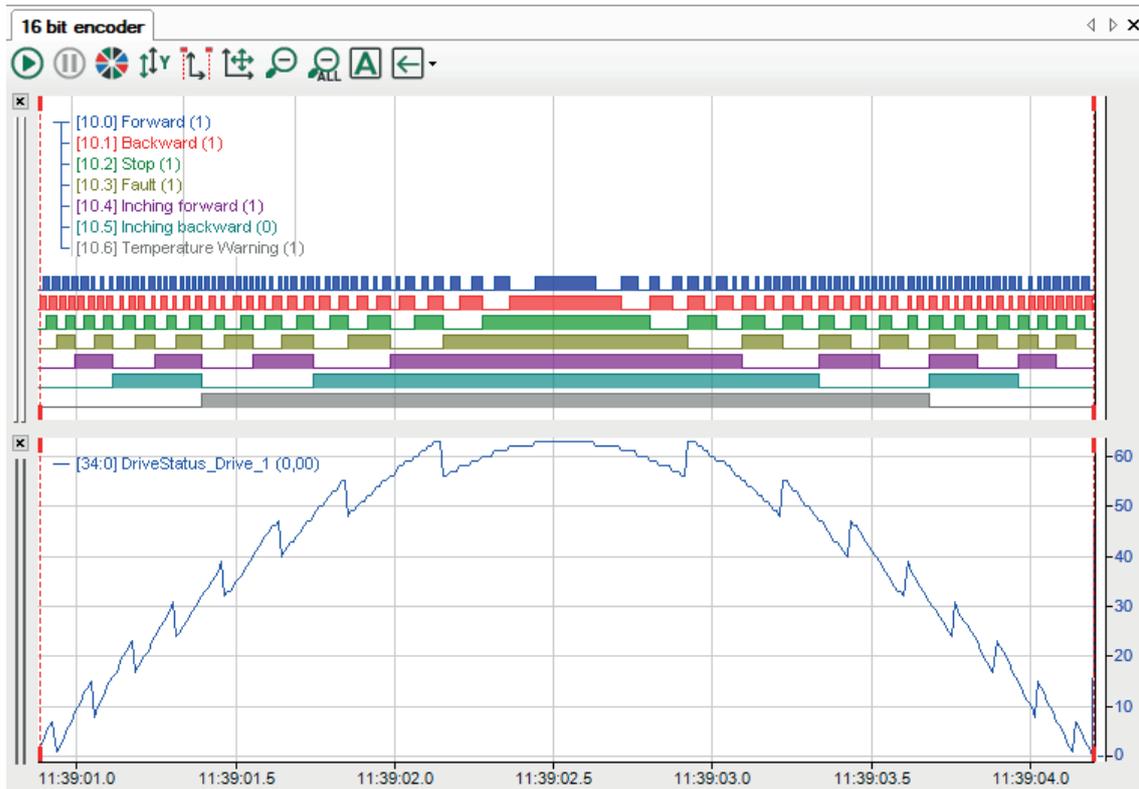
### 4.6.5 16-bit encoder

The 16-bit encoder module basically does exactly the opposite of the 16-bit decoder module.

Up to 16 digital signals are packed into a 16-bit integer analog signal.

You can configure 1 to 1000 analog signals per module.

the following figure shows an example of 7 digital signals (top) that are packed into one analog signal (bottom).



#### 4.6.5.1 16-bit encoder – General tab

##### General module settings

For a description of the basic settings see [↗ Common and general module settings, page 20.](#)

##### Module layout

Set the desired number of analog signals that this module should supply here.

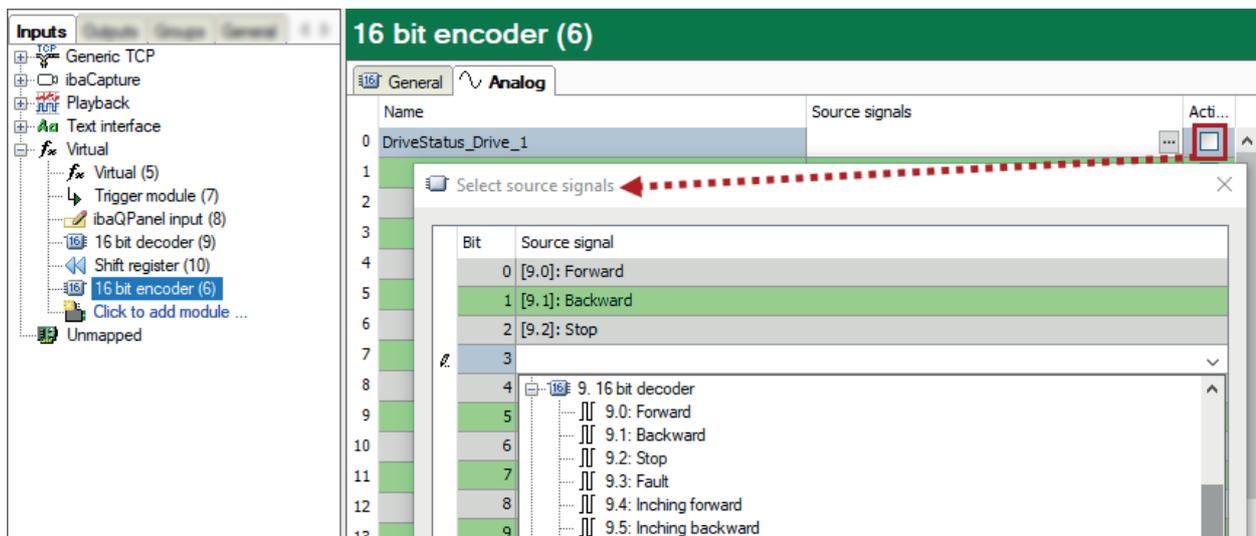
Value range: 1 ... 1000 (default: 32)

### 4.6.5.2 16-bit encoder – Analog tab

You can configure the analog signals in this tab. The signal table contains as many lines as you set under *Number of analog signals* in the *General* tab.

To select the digital signals to be packed into an analog signal, click the Browse button <...> in the *Source signals* column. The *Select source signals* dialog opens.

Enter the max. 16 digital signals in the specified bit sequence. For easier input, you can also click in the *Source signal* column and select the desired signal from the signal tree that opens.



There is another convenient function in this dialog.

If the digital source signals are already present in their module in the correct order, then you only need to select the first digital signal specifically for the desired line and then click on the *Source signal* column header. Starting from the current line, all further digital signals from the source module are then entered automatically.

If you know the signal numbers or have them in text form ([Module number.Signal number]), you can alternatively enter the signals directly in the *Source signals* column in the *Analog* tab or paste them using Copy & Paste, separated by commas in each case.

### 4.6.6 32-bit encoder

The 32-bit encoder module basically does exactly the opposite of the 32-bit decoder module.

Up to 32 digital signals are packed into a 32-bit integer analog signal.

You can configure 1 to 1000 analog signals per module.

It is configured in the same way as the 16-bit encoder module.

## 4.6.7 Electric modules

To calculate RMS values, electrical power and electrical power factors, the electrical modules are available for different types of networks:

- Delta module
- Star module
- Single-phase module
- Two-phase module

*ibaPDA* uses the same formulas as *ibaAnalyzer* to calculate the different electric values. The formulas of the single-line module are the same as the formulas of the star module, with  $u_2$ ,  $u_3$  and  $i_2$ ,  $i_3$ ,  $i_4$  equal to zero. The following figure shows an example for a delta module.

The screenshot shows the 'iba I/O Manager' window with the 'Electric delta module (7)' configuration. The configuration is divided into 'Basic' and 'Electric' tabs.

Basic	
Module Type	Electric delta module
Locked	False
Enabled	True
Name	Electric delta module
Module No.	7
Timebase	10 ms
Use name as prefix	False

Electric	
U12	[6:0] Voltage_U12
U31	[6:1] Voltage_U31
U23	[6:2] Voltage_U23
i1	[5:0] Current_L1
i2	[5:1] Current_L2
i3	[5:2] Current_L3
Use dynamic base frequen	False
Base frequency	50

The schematic diagram on the right shows a delta network with three nodes labeled 1, 2, and 3. Node 1 is at the top, node 2 is at the bottom right, and node 3 is at the bottom left. A neutral point 'N' is indicated by a dashed line. Input currents  $i_1$ ,  $i_2$ , and  $i_3$  are shown entering nodes 1, 2, and 3 respectively. Output voltages  $U_{31}$ ,  $U_{12}$ , and  $U_{23}$  are shown across the branches connecting nodes 3-1, 1-2, and 2-3.

### 4.6.7.1 Electric modules – General tab

#### Basic settings

For a description of the basic settings see [↗ Common and general module settings](#), page 20.

The input voltages and currents are to be entered in the *General* tab. Also the fundamental frequency given in Hz needs to be specified.

#### Time base

The time base of the module can be set to  $1 / \text{base frequency}$ , as only one new value will be available in each period.

#### Voltages and currents

With regard to the electrical diagram, select the corresponding voltage and current signals from the selection list in the signal tree.

### Fundamental DB first harmonic

If a certain base frequency has been specified, set the “Use dynamic base frequency” option to FALSE and enter the base frequency in the field below.

If the base frequency varies and the frequency value should be a separate signal, set the “Use dynamic base frequency” option to TRUE and select the appropriate frequency signal from the selection list in the field below. The following figure shows the assignment of the first voltage signal.

△ General
⤿ Analog

**Basic**

Module Type	Electric delta module
Locked	False
Enabled	True
Name	<b>Electric delta module</b>
Module No.	<b>7</b>
Timebase	<b>10 ms</b>
Use name as prefix	False

**Electric**

U12 [6:0] Voltage\_U12

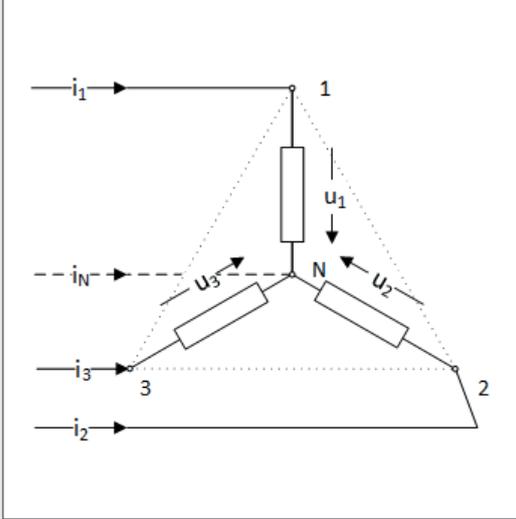
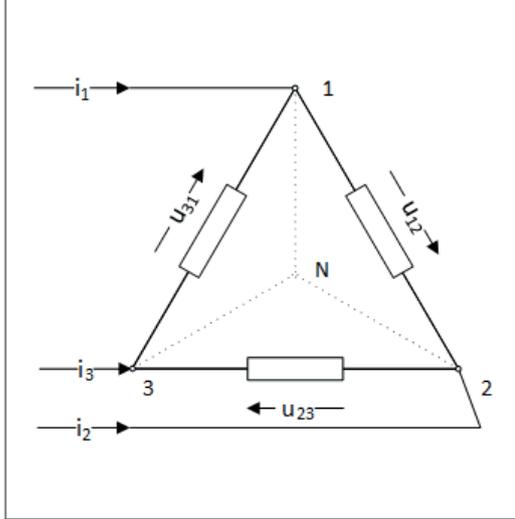
Unassigned

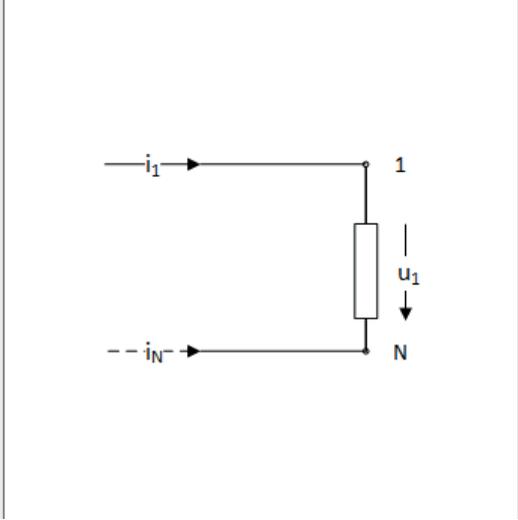
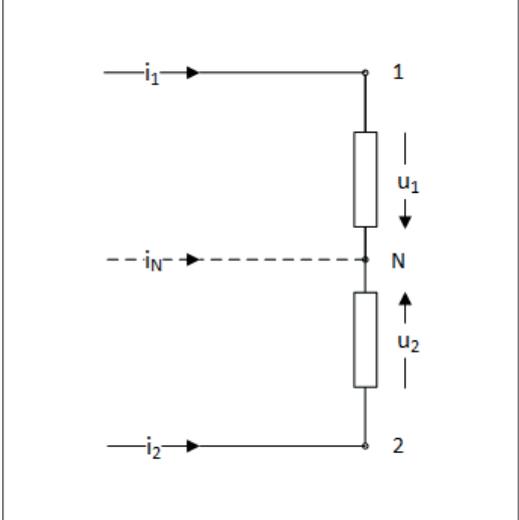
- 0. Hydr. Adjustment
- 1. Shear / RSF / S1-S6
- 2. Stands 1-7 a roll forces
- 3. IBA-Logic
- 4. Shear
- 5. ibaMS3xAI-1A
- 6. ibaMS4xAI-380VAC
- 6:0: Voltage\_U12
- 6:1: Voltage\_U31
- 6:2: Voltage\_U23

The diagram shows a delta network with three nodes labeled 1, 2, and 3. Node 1 is at the top, node 2 is at the bottom right, and node 3 is at the bottom left. A neutral point N is indicated by dashed lines connecting to the midpoints of each side. Three current sources enter the nodes:  $i_1$  at node 1,  $i_2$  at node 2, and  $i_3$  at node 3. Three voltage sources are connected across the branches:  $U_{31}$  between nodes 3 and 1,  $U_{12}$  between nodes 1 and 2, and  $U_{23}$  between nodes 2 and 3.

### 4.6.7.2 Electric modules – Analog tab

In the *Analog* tab, you will find the predefined results, which will then be available in the signal tree for display and recording. The number and type of the results may vary, depending on the module type.

Star module	Delta module
	
<ul style="list-style-type: none"> <li>effective voltage <math>U_1</math></li> <li>effective voltage <math>U_2</math></li> <li>effective voltage <math>U_3</math></li> <li>effective current <math>I_1</math></li> <li>effective current <math>I_2</math></li> <li>effective current <math>I_3</math></li> <li>effective current <math>I_N</math></li> <li>total effective voltage</li> <li>total effective current</li> <li>real power</li> <li>apparent power</li> <li>reactive power</li> <li>reactive power, signed</li> <li>power factor</li> <li>reactive power factor</li> <li>reactive power factor, signed</li> </ul>	<ul style="list-style-type: none"> <li>effective voltage <math>U_{12}</math></li> <li>effective voltage <math>U_{31}</math></li> <li>effective voltage <math>U_{23}</math></li> <li>effective current <math>I_1</math></li> <li>effective current <math>I_2</math></li> <li>effective current <math>I_3</math></li> <li>total effective voltage</li> <li>total effective current</li> <li>real power</li> <li>apparent power</li> <li>reactive power</li> <li>reactive power, signed</li> <li>power factor</li> <li>reactive power factor</li> <li>reactive power factor, signed</li> </ul>

Single-phase module	Two-phase module
	
<p>effective voltage <math>U_1</math></p> <p>effective current <math>I_1</math></p> <p>real power</p> <p>apparent power</p> <p>reactive power</p> <p>power factor</p> <p>signed reactive power</p> <p>reactive power factor</p> <p>signed reactive power factor</p>	<p>effective voltage <math>U_1</math></p> <p>effective voltage <math>U_2</math></p> <p>effective current <math>I_1</math></p> <p>effective current <math>I_2</math></p> <p>effective current <math>I_N</math></p> <p>total effective voltage</p> <p>total effective current</p> <p>real power</p> <p>apparent power</p> <p>reactive power</p> <p>power factor</p> <p>signed reactive power</p> <p>reactive power factor</p> <p>signed reactive power factor</p>

## 4.6.8 ibaQPanel input

The *ibaQPanel Input* module type can be added to the virtual interface.

The signals for this module type are writable using the *text input control* and *Button* tools in *ibaQPanel*.

Ideally, in *ibaQPanel*, use the *text input control* for analog signals and the *Button* for digital signals.

However, the *text input control* can also be used to write digital signals:

Value <> 0: TRUE

Value = 0: FALSE

To input texts (strings) use the module type *ibaQPanel text input* in conjunction with the *text input control*.

### 4.6.8.1 ibaQPanel input – General tab

#### Basic settings

For a description of the basic settings see [↗ Common and general module settings](#), page 20.

#### ibaQPanel input

##### Remember last values

If you enable this option, when applying a new I/O configuration the last known signal values are used instead of the default values.

#### Module layout

##### Number of analog and digital signals

Here, you can increase or decrease the number of signals in the module. By default, 32 signals are preset. You may enter any value between 0 and 1000. The signal tables will be adjusted accordingly.

### 4.6.8.2 ibaQPanel input – Analog and Digital tabs

#### Default value

The default value of a signal can be specified in the signal tables (analog/digital) of the module. The signal will have the default value each time the acquisition starts, unless the option *Remember last values* is enabled in the module settings.

Name	Unit	Initial value	Active	Actual	+
0 My analog value		0	<input checked="" type="checkbox"/>	0	^
1		0	<input checked="" type="checkbox"/>	0	
2		0	<input checked="" type="checkbox"/>	0	
3		0	<input checked="" type="checkbox"/>	0	
4		0	<input checked="" type="checkbox"/>	0	
5		0	<input checked="" type="checkbox"/>	0	
6		0	<input checked="" type="checkbox"/>	0	

#### Example

The *text input control* in *ibaQPanel* uses ‘My analog value’ as destination signal.

The “Pulse My digital value 1” button uses ‘My digital value 1’ as digital signal for the button’s “Pulse digital signal” command.

### 4.6.9 ibaQPanel text input

The *ibaQPanel text input* module type can be added to the virtual interface.

The signals for such a module are writable using the *ibaQPanel tool text input control*.

The general module settings correspond to those of the *ibaQPanel input* module.

The text signals are configured in the *Analog* tab.

General		Analog			
	Name	Initial value	Length	Active	Actual
0	Input_Quality		32	<input checked="" type="checkbox"/>	sehr gut ^
1	Input_Class		32	<input checked="" type="checkbox"/>	A
2			32	<input checked="" type="checkbox"/>	

You can specify a text in the *Initial value* column that is displayed in the *ibaQPanel text input control* as long as no text has been entered yet, e.g., after the acquisition has started.

If the *Remember last values* option is enabled in the module settings, then the initial value is replaced by the last known text.

In the *Length* column you can set the length of the text via the number of characters. Default value: 32 characters

To input analog or digital values use the module type *ibaQPanel input* in conjunction with the *text input control*.

#### Other documentation



For more information about handling text signals and other text modules, please refer to the *ibaPDA* manual, part 2, *Text signals and text processing*.

## 4.6.10 Trigger module

The trigger module is part of the trigger pool function in *ibaPDA*. Triggers are used, for instance, as start or stop triggers to control Data storages, for indication and alerting, or for displaying complex process events.

In addition, trigger releases can be entered as events in an *ibaHD-Server* recording. If several *ibaPDA* servers are used in a multistation network, one trigger module in an *ibaPDA* system can be configured as a "global trigger module". This can then be used to trigger data storage on the other systems in the multistation network.

The trigger module is a virtual module with digital signals only. Each of these digital signals is a trigger. Instead of the normal expression builder, the trigger module uses a special trigger editor dialog to create the trigger expressions.

### 4.6.10.1 Trigger module – General tab

#### Basic settings

For a description of the basic settings see [↗ Common and general module settings](#), page 20.

#### ibaHD store

If you are using an *ibaHD-Server*, the trigger releases can be written as events into an ibaHD event store.

#### ibaHD store

To enable this function, select the desired event store here.

If the desired store is not available, click on *Add new ibaHD event store*. Then select the desired ibaHD-Server and the event store and exit the *ibaHD-Server selection* dialog with <OK>.

#### Folder

Here, you can determine the folder in the store in which the trigger events for this module will be stored. If no folders are available, you can create a new one.

This is advisable for the sake of clarity.

#### Message

Select which information will be entered in the HD store via the corresponding digital signal (in the *Digital* tab). Available for selection:

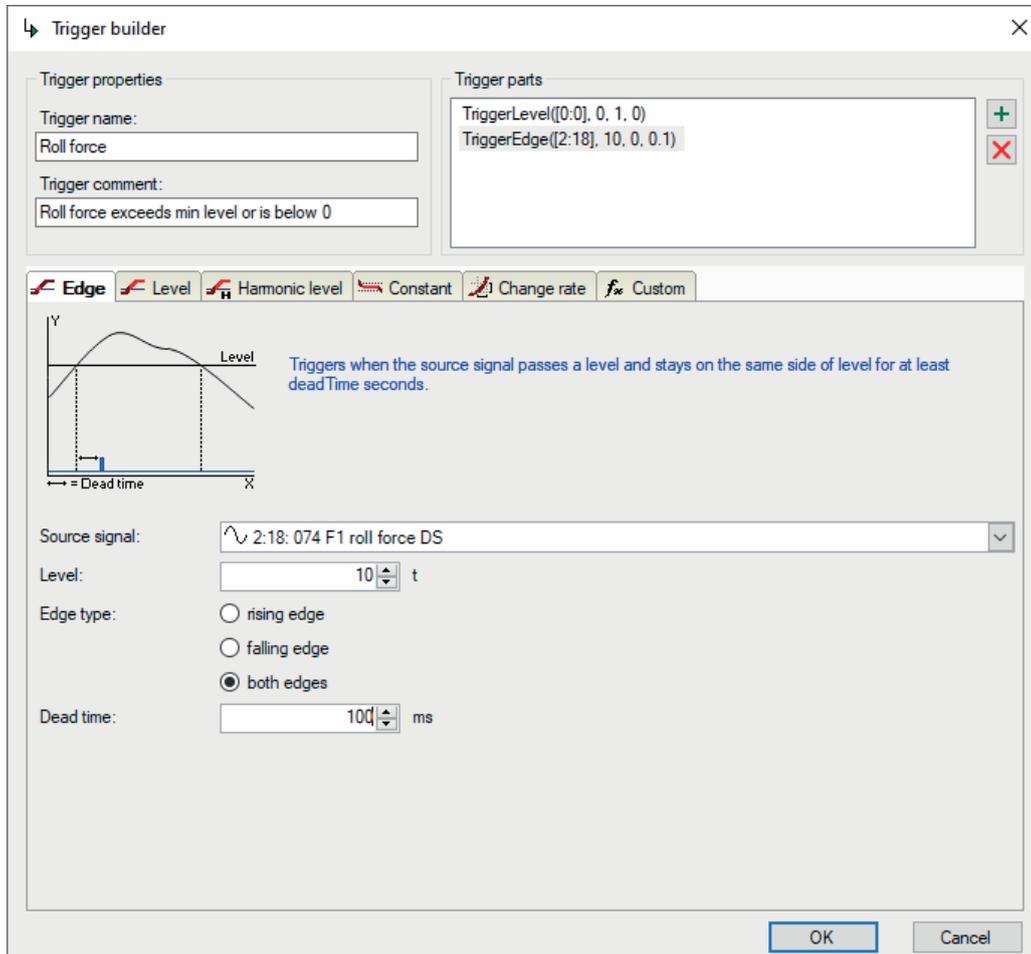
- Signal or trigger name
- Comment 1
- Comment 2

This information also appears as the event text in the message shown in the *ibaPDA* client's event display.

## 4.6.10.2 Trigger modules – Digital tab

### Expression

Enter the expression for the computation of the trigger signal or create the expression in the “Trigger builder” dialog. Open the expression dialog via the <fx> button in the “Expression” field.



### Trigger properties

#### Trigger name and comment

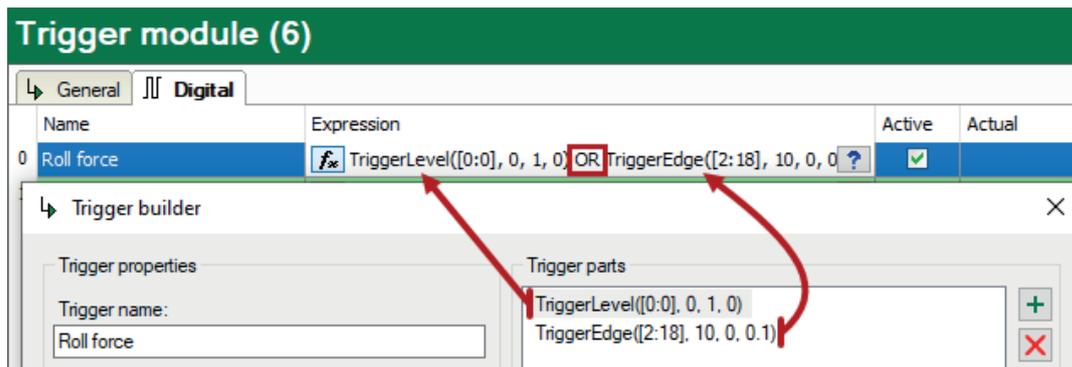
In the trigger properties, you may enter a trigger name and comment.

#### Trigger parts

A trigger expression consists of one or several parts. Each part is an expression itself. With the OR function, these parts can be combined. Therefore, the trigger is 1 (TRUE) if one of its parts is 1 (TRUE).

The example in the figure above shows 2 parts: `TriggerEdge([02:18],10,0,0.1)` and `TriggerLevel([0: 0],0.1,0)`.

The result:



If you want to add another trigger part, click the button showing the green plus symbol.

If you want to edit a trigger part, make sure that the part in question is selected. Consequently, all changes made in the settings below apply to this part.

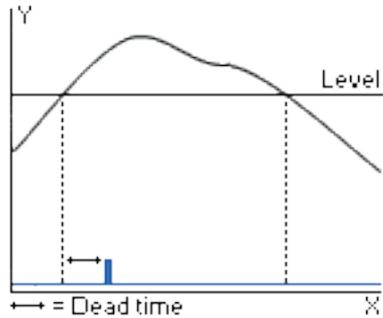
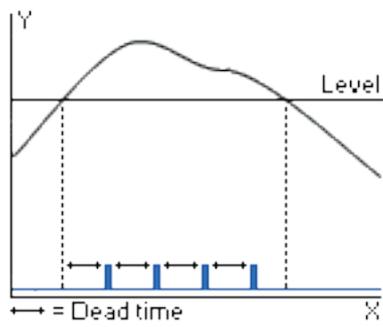
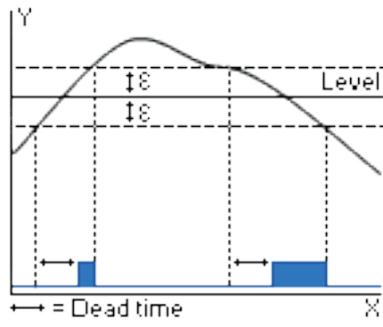
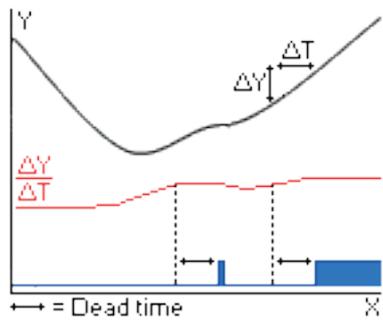
If you want to remove a trigger part, click the button showing the red X symbol.

### Trigger configuration tabs

Those expressions can be more or less complex, depending on the nature of the process or machinery. Five different trigger types are available for the most common situations.

- Edge
- Level
- Constant
- Change rate
- Custom

The characteristics of these trigger types are described in the table below. You will find the small drawings also in the dialog itself.

Trigger type	Explanation	Comment
Edge		Short trigger pulse duration after dead time.
Level		Short trigger pulse duration after dead time.
Constant		Duration of the trigger after dead time as long as the condition is fulfilled.
Change rate		Duration of the trigger after dead time as long as the condition is fulfilled.
Custom	Any logical expression or combination of trigger functions.	Full range of functions of the expression editor available.  Note that the result must be a digital signal.

Example of the use of an expression for a user-defined trigger.

Edge
Level
Harmonic level
Constant
Change rate
**fx Custom**

**Input signals**

- 0. Hydr. Adjustment
- 1. Shear / RSF / S1-S6
- 2. Stands 1-7 a roll forces
- 3. IBA-Logic
- 4. Shear
- 5. ibaQPanel text input
- 6. Trigger module
- 9. File Text
- 11. Serial Text
- 12. TCP Text

**Functions**

- fx Int
- fx InterruptCycleTime
- fx InterruptTime
- fx KurtosisInTime
- fx LimitAlarm
- fx Log
- fx Log10
- fx LP
- fx MAvg
- fx MAvgOnTrigger
- fx Max

**LimitAlarm('Expression', 'Limit', 'DeadBand', 'Time',**

returns true when 'Expression' is greater than 'Limit' for at least 'Time' seconds.  
 Returns false again when 'Expression' is smaller than 'Limit' - 'DeadBand'.  
 The optional 'Reset' parameter can be used to stop and restart the calculation:  
 'Reset' = 0: Do calculation  
 'Reset' = 1: Stop calculation and set result to 0  
 'Reset' = 2: Stop calculation and keep result

Expression

**LimitAlarm('0:30', '50', '10', '1', 'Reset=0')**

**Limit Alarm**

For information about how to use triggers as start or stop triggers for Data Storage refer to part 5, *Trigger setting*.

### 4.6.11 Virtual

This module type is always available. No license is needed.

You can find a description of the creation of virtual signals in part 4, chapter *Expression builder (virtual signals)*.

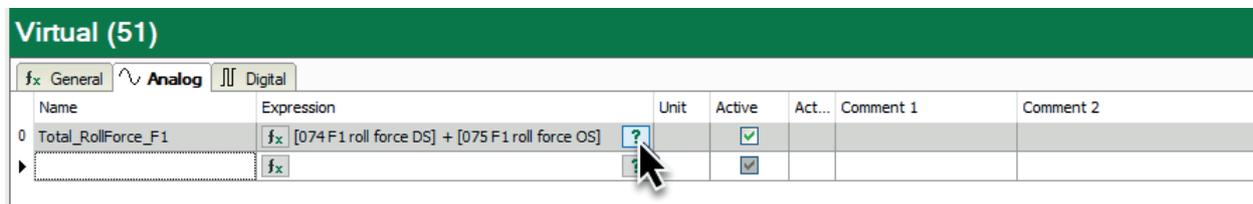
The number of signals (analog or binary) per module is not limited.

#### 4.6.11.1 Cross reference between expression and signal

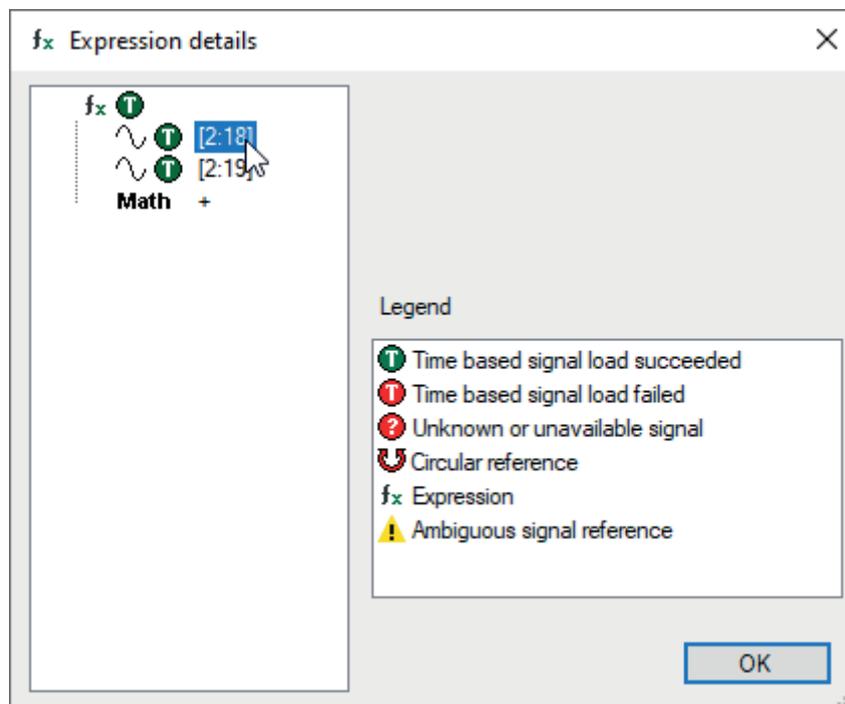
You can swiftly find a source signal of any expression in the virtual signals by using the cross reference function. Particularly, when using the signal name as reference in the expressions instead of the signal number the conclusion on the origin of the signal is not easy.

In order to find the signal source follow these steps:

1. Click on the little question mark button in the row with your expression.



2. The diagnostic window opens, showing the details of the expression. Double-click on the signal whose origin you are looking for.



3. The view switches immediately to the interface and module in the I/O Manager. The signal in question is highlighted.

The screenshot shows the 'iba I/O Manager' application. The left-hand pane displays a tree view of various input modules. The 'Stands 1-7 a roll forces (2)' module is selected and highlighted in blue. The right-hand pane shows the configuration for this module, with the 'General' tab active. Below the tab, there is a table with 21 rows, each representing a signal. The table has a 'Name' column and a corresponding signal name.

	Name
10	066 F4 RPM
11	067 F4 current
12	068 F5 RPM
13	069 F5 current
14	070 F6 RPM
15	071 F6 current
16	072 F7 RPM
17	073 F7 current
18	074 F1 roll force DS
19	075 F1 roll force OS
20	076 F2 roll force DS
21	077 F2 roll force OS

#### 4.6.11.2 Virtual – General tab

##### Basic settings

For a description of the basic settings see [➤ Common and general module settings](#), page 20.

#### 4.6.11.3 Virtual – Analog tab

##### General columns in the signal table

For a description of the general signal table columns see [➤ Columns in tables with analog and digital signals](#), page 22.

##### Expression

This column contains the definition of a virtual signal (or alarm). You may enter simple numerical or Boolean equations or complex mathematical expressions. Mathematical expressions are created with the expression editor. If you know the syntax, you may also manually enter the expressions in the table.

Click the <fx> button to open the expression builder.

For more information, see part 4, *Expression builder*.

Clicking on the <?> button executes a syntax check for the expression in the same row. A dialog window with diagnostic information about the expression opens. This is a good help when looking for errors in long and complex expressions.

Virtual analog signals are always REAL type values. In order to get a virtual digital signal or alarm, the result of the expression must be a Boolean quantity (TRUE or FALSE).

**Adding signals**

For virtual signals, there is always an empty row in the signal table. As soon as you start typing in a row, a new empty row will be added automatically.

## 4.6.12 Virtual retentive

This module type is always available. No license is needed.

You can find a description of the creation of virtual signals in part 4, chapter *Expression builder (virtual signals)*.

The number of signals per module is not restricted. Compared to the normal virtual module, this module provides for a retentive behavior. It means that the signals, which are defined in this module, retain their last valid value they had before the acquisition was stopped or restarted. This can be used to realize elapsed time meters or consumption meters. Alternatively, you can set an initial value, which should be used after restart of acquisition.

The module only offers analog values. You can store binary information by using analog values (real)  $0 \dots <0.5$  for FALSE or  $\geq 0.5$  for TRUE.

### 4.6.12.1 Virtual retentive – General tab

#### Basic settings

For a description of the basic settings see [↗ Common and general module settings](#), page 20.

### 4.6.12.2 Virtual retentive – Analog tab

#### General columns in the signal table

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals](#), page 22.

#### Expression

This column contains the definition of a virtual signal (or alarm). You may enter simple numerical or Boolean equations or complex mathematical expressions. Mathematical expressions are created with the expression editor. If you know the syntax, you may also manually enter the expressions in the table.

Click the <fx> button to open the expression builder.

For more information, see part 4, *Expression builder*.

Clicking on the <?> button executes a syntax check for the expression in the same row. A dialog window with diagnostic information about the expression opens. This is a good help when looking for errors in long and complex expressions.

#### Initial value

If the signal should use a defined initial value instead of retaining the last value at restart of acquisition, then enter the initial value here.

For virtual signals, there is always an empty row in the signal table. As soon as you start typing in a row, a new empty row will be added automatically.

#### Adding signals

For virtual signals, there is always an empty row in the signal table. As soon as you start typing in a row, a new empty row will be added automatically.

### 4.6.13 Shift register

This module type is always available. No license is needed.

With this module actual values of a signal can be stored trigger-driven in virtual signals. With every rising edge of the trigger signal, the actual value of the signal to be stored is written into the first virtual signal. All values which have been stored before are shifted by one index down. The depth of the stack is freely adjustable. If the number of stored values has reached the depth of the stack, then the oldest value will be discarded. Using this module, e. g. enables the visualization of recent values of a signal in *ibaQPanel*.

The number of signals provided by this module (depth) is not restricted.

#### 4.6.13.1 Shift register – General tab

##### Basic settings

For a description of the basic settings see [↗ Common and general module settings, page 20](#).

##### Shift register

###### Source signal

Select the signal whose values should be stored in the shift register.

###### Trigger signal

Select the signal, which should be used as trigger signal for the shift register. On each rising edge of the trigger signal the actual value of the source signal is stored in the first virtual signal in the *Analog* tab. Previous values are shifted by one index down.

###### Depth

The depth of the shift register determines how many values of the source signal are stored. The most recent value is always stored in the signal with index 0. If the number of stored values has reached the depth of the stack, then the oldest value is discarded.

###### Remember last values

If you set this option on TRUE, then the last stored values are used after application of a new IO-configuration and restart of acquisition instead of using default values. If you set this option on FALSE, the default values are used.

#### 4.6.13.2 Shift register – Analog tab

##### General columns in the signal table

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals, page 22](#).

The number of signals in this register is determined by the "Depth" value in the *General* register.

##### Default

Here, enter the values which should be used in case of applying a new I/O-configuration and restarting the acquisition, if the option "Remember last values" in the *General* register is set on FALSE.

#### 4.6.14 Computation module

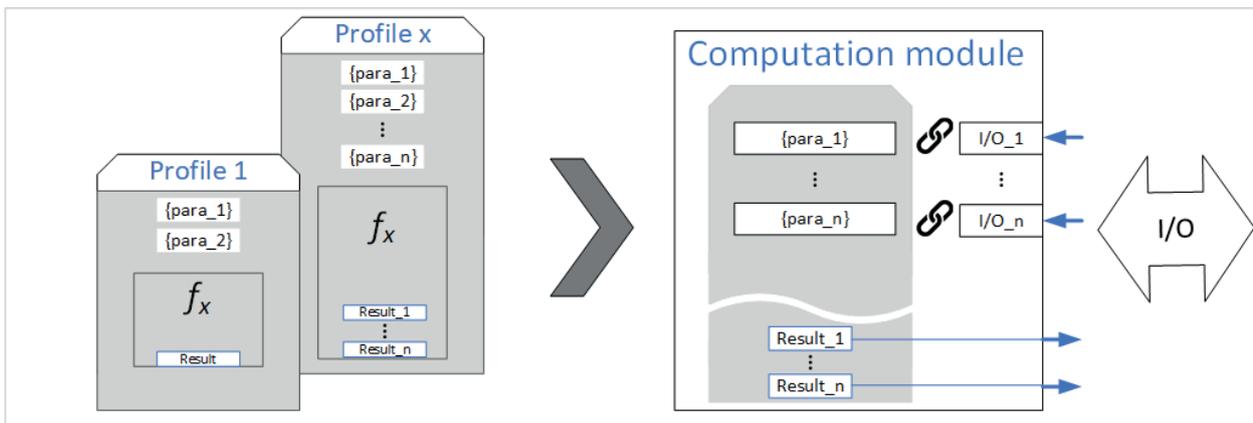
This module type is always available. No license is needed.

With this module, you can perform any calculations on the basis of the signals and expressions available in *ibaPDA*, similar to the "Virtual" module.

The difference to the "Virtual" module is that placeholders are used as operands in the calculation formulas instead of the signals themselves. The link to the real signals is therefore not created in the formulas themselves, but rather in the general module settings. The calculation functions are saved as so-called *profiles* and can thus be used several times. A profile forms the core with input and output parameters, so to speak, and the calculation module forms the shell as the interface to the input and output signals.

Once a calculation has been created, you thus have the possibility to use it several times, either within one *ibaPDA* system or – after export and import – on different *ibaPDA* systems. Both in the case of complicated calculations that you need several times, and in general for calculations that are often repeated, this saves you a lot of time and reduces the risk of errors.

The calculation functions can be arbitrarily complex and work with any number of input and output parameters. You can project as many profiles as you like, which are then available for selection when configuring a computation module. You configure one profile per calculation task. You can use exactly one profile per computation module. The following figure shows the principle of the profile-based calculation module.



If a calculation needs to be changed, then only the corresponding profile needs to be changed and the change is effective immediately wherever the profile in question is in use.

The profile-based concept also offers the possibility of know-how protection and license-controlled use.

For example, if your calculation contains technological knowledge that deserves protection, you can prevent it from being viewed or modified as the profiles are supported by the know-how protection feature in *ibaPDA*'s I/O Manager. Furthermore, the execution of a calculation can be bound to certain license numbers (dongles) by means of the profile property.

To protect the profiles, see also ↗ *Protecting profiles*, page 92.

### 4.6.14.1 Configure computation module

#### Procedure

1. Consider which calculations you want to perform and set the desired input and output parameters for the calculation.
2. Add a computation module under the "Virtual" interface.
3. Configure a profile:
  1. Define the placeholders.
  2. Configure the analog and/or digital expressions (your calculation) using the expression builder.
4. Save the profile and then select it in the general module settings.
5. Assign the real signals to the now-visible placeholders (input parameters) in the general module settings.
6. Click on <OK> or <Apply>.
7. After restarting the acquisition, you will see the computation module with the result variables in the signal tree.
8. You can display or process the results like any other signal.

### 4.6.14.2 Computation module – General tab

#### Basic settings

For a description of the basic settings see [↗ Common and general module settings](#), page 20.

#### Advanced

##### High accuracy

If you enable this option (True), then the calculation results are saved as 64-bit floating point values. If you do not enable "High accuracy", then 32-bit floating point values are used.

##### Profile

Select the appropriate profile that contains the calculation formula(s) for this module. If no profile is available yet, you must create one first.

To do this, see [↗ Add and configure profiles](#), page 86.

After you select the profile, additional lines containing the input parameters for the calculation appear directly below it.

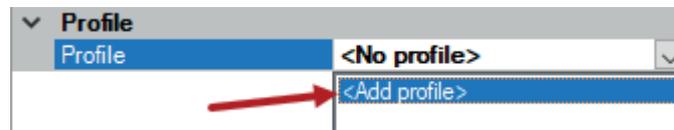
Assign the appropriate signal to the input parameters. If you click inside a field, the signal tree opens and you can select the appropriate signal.

### 4.6.14.3 Add and configure profiles

To add a profile for a computation module, click the blue *Configure profiles* link in the *General* tab at the bottom of the computation module.



Alternatively, you can open the drop-down list in the *Profile* row and click on *<Add profile>*.



In both cases, the *Configure profiles* dialog opens.

Below the left window of the dialog you will find a set of buttons with the following functions:

	Add profile
	Copy selected profile
	Delete selected profile
	Import profile(s) from a <i>*.computationProfile</i> file
	Export selected profile to a <i>*.computationProfile</i> file

Add a profile and name it so that the function of the calculation is recognizable.

Then switch to the *Placeholder* tab.

#### 4.6.14.3.1 Configure profiles – placeholders

To formulate your calculation, you must first define the placeholders for the input and output variables of the calculation.

In the *Placeholders* tab, enter the placeholders in order. Each time you complete a line, the next free line is automatically created.

- Name: Name of the placeholder as it will subsequently be used in the calculation function.
- Default constant value: Enter a value here that the placeholder should have if no signal is linked or the signal is invalid. Note that this value must match the value type.
- Value type: Select the value type from the drop-down list here. Choose the type that matches the signal or constant. The following types are available for selection:
  - Numeric... for a numerical value
  - Digital... for a purely digital signal (True/False, 0/1)
  - Text... for a text signal
  - Any (numeric default)... The value type is detected automatically
  - Any (digital default)... The value type is detected automatically
  - Any (text default)... The value type is detected automatically

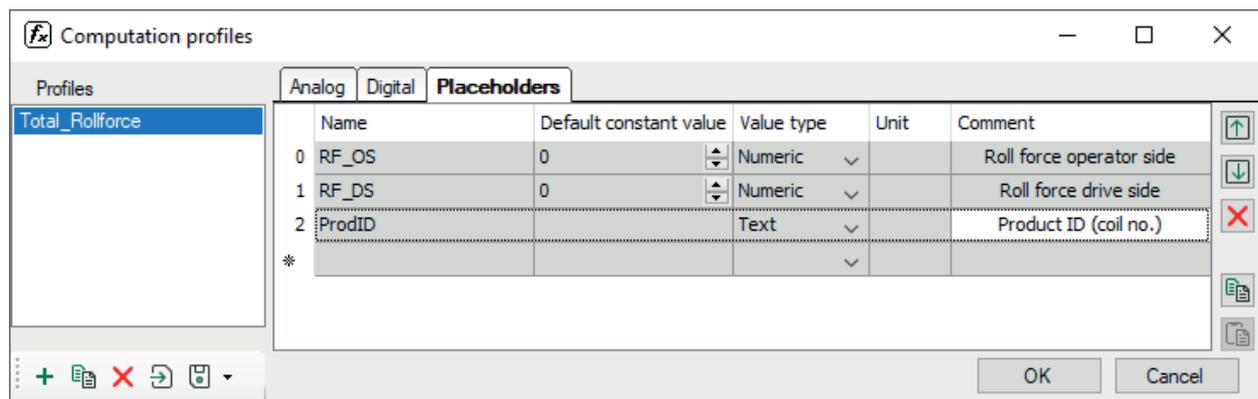
- Comment: You can enter a short description of the placeholder here. This text will then be displayed later in the *General* tab of the computation module that you use to link the placeholders to the signals.

In the next step, you can formulate the expressions in the *Analog* and/or *Digital* tabs.

### Example

Calculation of total and differential rolling force with limit-value monitoring

1. Add computation module and then configure profiles.
2. Define placeholders for the two measured input values "rolling force, operating side" and "rolling force, drive side", as well as the product ID.



#### 4.6.14.3.2 Configure profiles – analog and digital expressions

In the *Analog* and *Digital* tabs you can now formulate the desired calculations.

As with the "Virtual" module, all the functions in the expression builder are available when you click the <fx> button in the *Expression* column.

In the formulas you can use the placeholders as well as the results of other expressions in the same profile, or constant values.

If you want to use the result of one line in another expression, use the predefined placeholders {analog:X} or {digital:Y}, where X and Y stand for the name of the expression in the *Analog* or *Digital* tab of the same profile.

If you populate several lines, you can manipulate the lines with the buttons on the right margin. The order of the lines is not relevant for the calculation of the expressions.

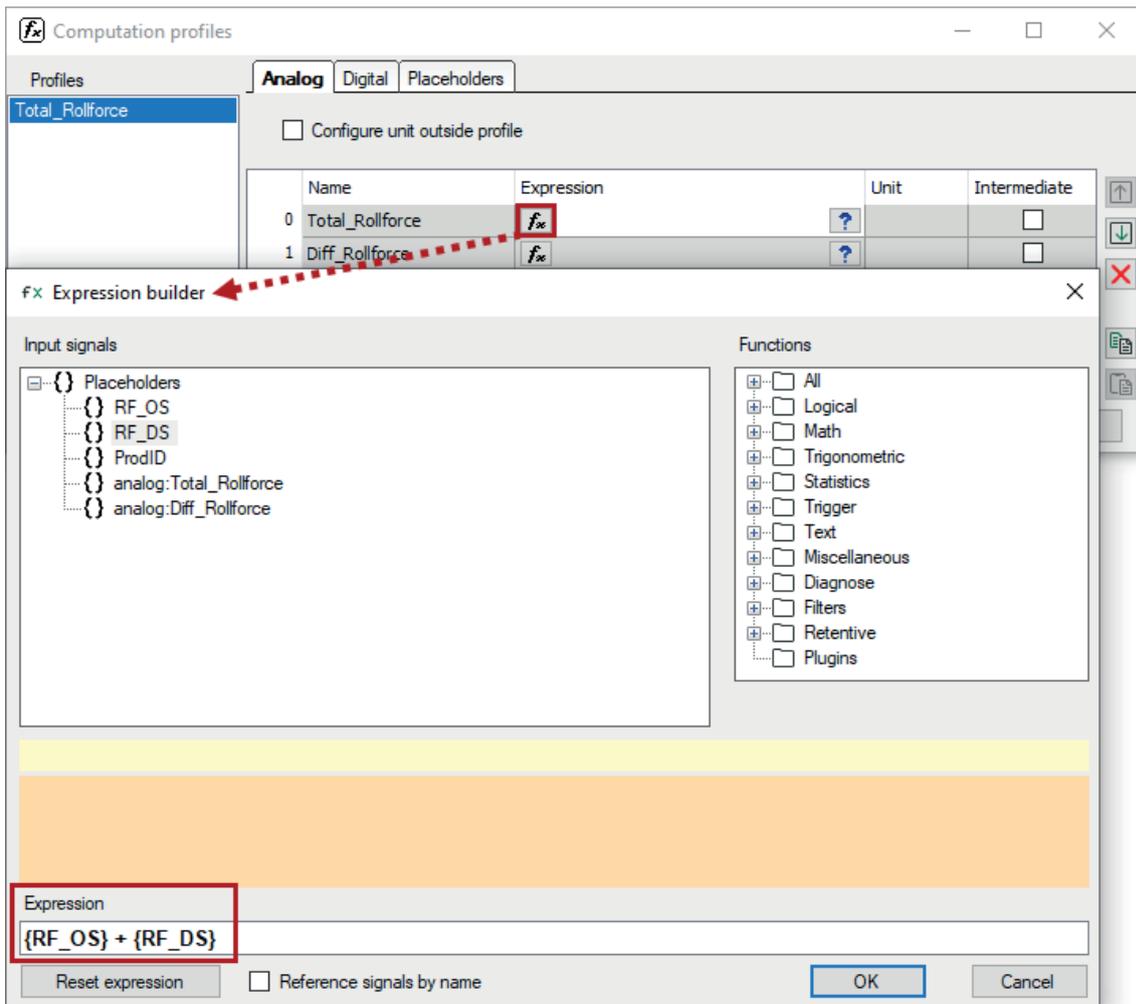
	Move selected row(s) up
	Move selected row(s) down
	Delete selected row(s)
	Copy all rows
	Paste from clipboard, starting from the selected row

In the *Intermediate* column, you have the option to classify each expression as an internal intermediate value. These expressions or signals are then not displayed in the signal table and are not available to be displayed in the *ibaPDA* client.

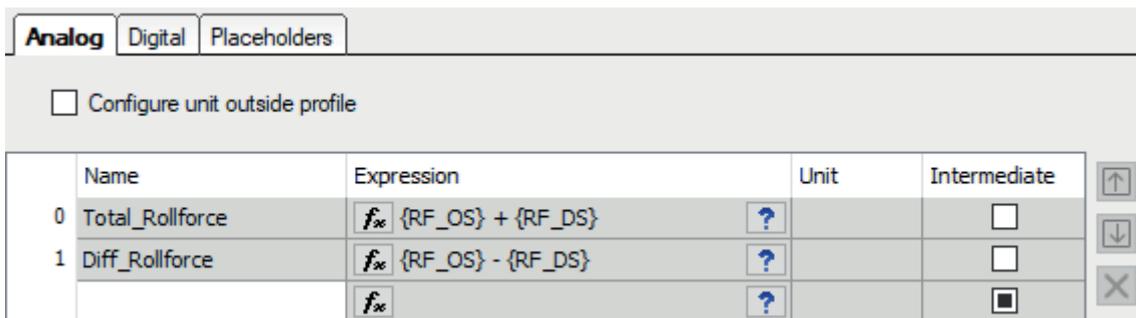
**Example**

**Analog**

Calculation of the total rolling force as the sum of the rolling force on the operating side and the rolling force on the drive side. Calculation of the differential rolling force as the difference between the two rolling forces.



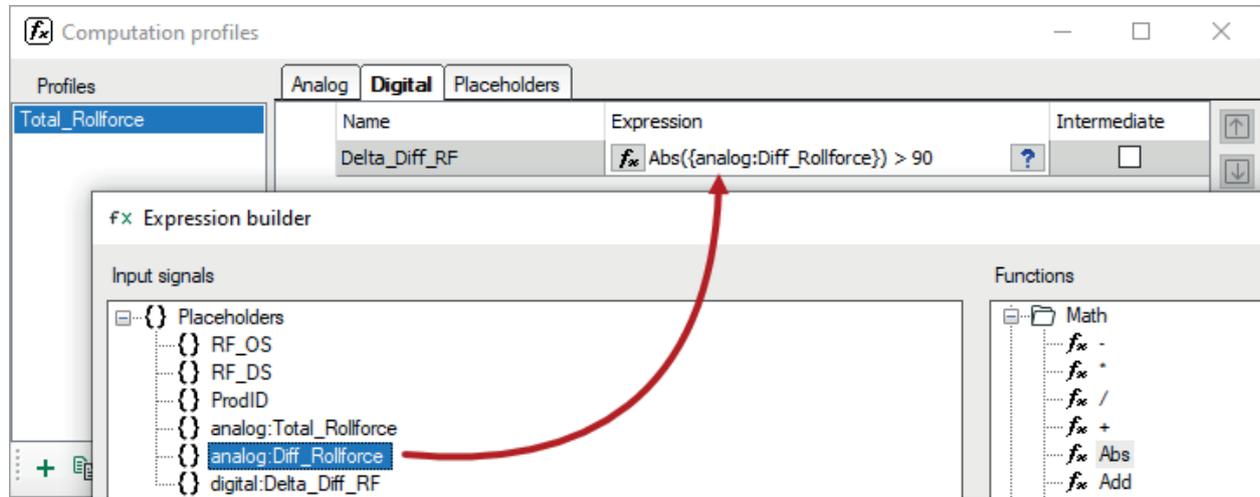
In the expression builder, only the defined placeholders and placeholders for the internal calculation results are available for the calculation. I/O signals cannot be used here.



### Digital

A digital signal should indicate when the absolute value of the differential rolling force exceeds a certain level (90 t).

In the following figure an internal calculation result is used for another expression.

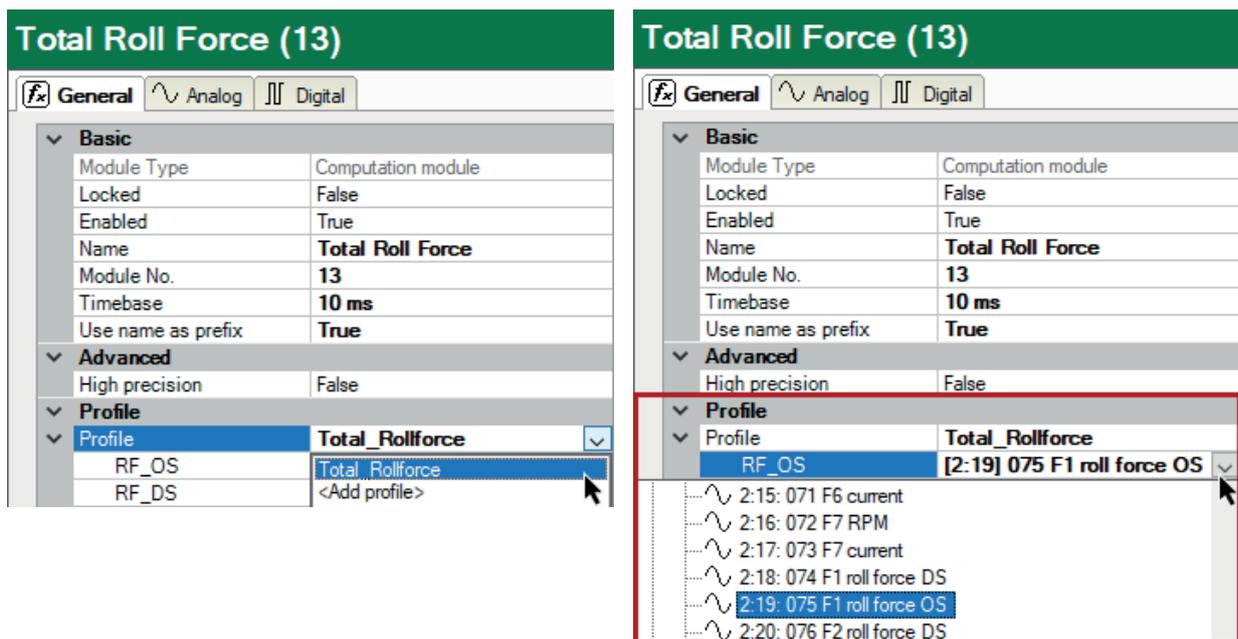


The limit value of 90 t is permanently entered here. If this value were variable or process-dependent, then a placeholder would also have to be defined for it, which would then be linked to a corresponding signal.

#### 4.6.14.4 Computation module – assign signals

When the desired profile is ready and available, you can select it in the general module settings for the computation module and then assign the signals to the placeholders.

The following figures show how to select the profile (left) and assign input signals to the placeholders (right).



The comment from the placeholder definition is displayed here.

▼ Profile

▼ Profile	Total_Rollforce
RF_OS	[2:19] 075 F1 roll force OS
RF_DS	[2:18] 074 F1 roll force DS
ProdID	[6:0] CoilNo_TS

**ProdID**  
Product ID (coil no.)

You can assign a signal or constant value to this placeholder.

The completed computation module appears in the signal tree. You can use its signals like signals of other modules, e.g. in a trendgraph.

Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[2:19] 075 F1 roll force OS	17:57:31.41	17:57:31.41	0,00 s	652,53 t	652,53 t	0,00 t
[2:18] 074 F1 roll force DS	17:57:31.41	17:57:31.41	0,00 s	572,45 t	572,45 t	0,00 t
[13:1] Total Roll Force\Diff_Rollforce	17:57:31.41	17:57:31.41	0,00 s	80,08	80,08	0,00
[13:0] Total Roll Force\Delta_Diff_RF	17:57:31.41	17:57:31.41	0,00 s	0	0	0
[13:0] Total Roll Force\Total_Rollforce	17:57:31.41	17:57:31.41	0,00 s	1226,64	1226,64	0,00

In this way, you can now add other computation modules that perform the same calculation by working with the same profile. In our example of a 7-stand rolling mill, you can now create one computation module per stand and thus project the calculations very efficiently.

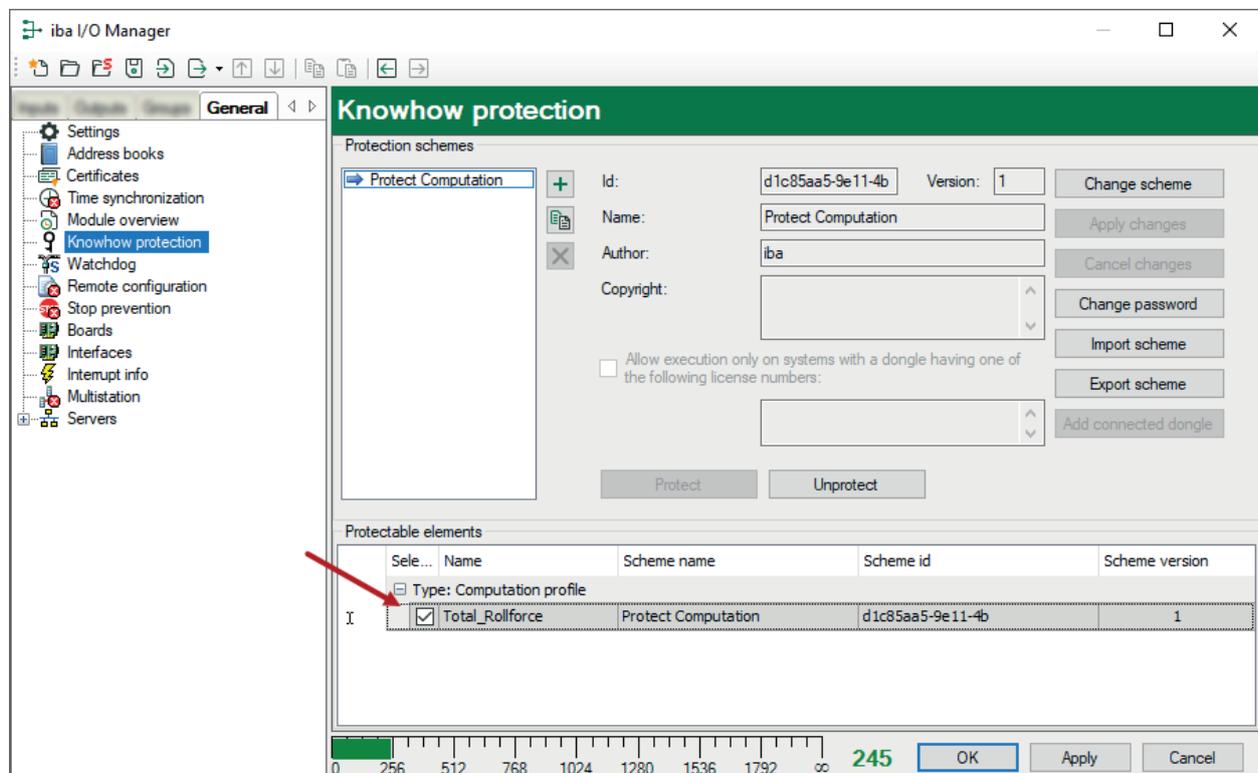
If the calculations change or more calculation variables are added, only the profile needs to be changed and all instances of the computation module will be automatically updated. The signal assignments must only be completed in each module if additional placeholders are added.

#### 4.6.14.5 Protecting profiles

You can prevent all profiles that provide the calculation basis for certain modules from being viewed or modified via the know-how protection feature. The profiles can be part of the following modules:

- ibalnSpectra
- ibalnCycle
- Computation module
- Lookup table
- Process condition
- Parameter set

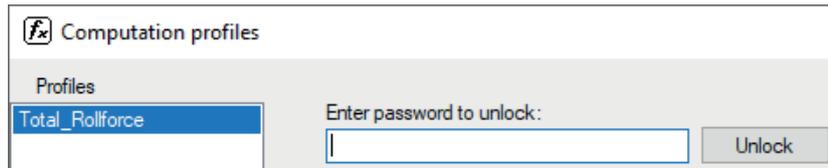
All defined profiles are listed in the know-how protection dialog. The following figure shows the example for a profile of a computation module.



### Protecting a profile

1. If you have not already done so, define a protection scheme and select it in the "Protection schemes" list.
2. Then select the desired profile (add a checkmark).
3. Click <Protect> and enter the password for the protection scheme.
4. Confirm your choice by clicking on <OK>.

If you then click *Configure profiles* in the module where this profile is used, you must first enter the protection scheme password to see the calculation.



### Other documentation



More information about the topic of know-how protection can be found in the *ibaPDA* manual, part 2, *Know-how protection*.

### 4.6.15 Computation retentive module

The retentive computation module has the same settings and configuration method like the standard computation module.

The only difference is the fact, that the functions used in the module should be retentive functions as well. If you use non-retentive functions, then a note will be output on the validation at start of acquisition.

#### Tip



In the expression builder, you can find the retentive functions in the *Retentive* folder among the *Functions*.

## 4.6.16 Lookup table

This module type is always available. No license is needed.

The function of this module type is to read and output values from a 2-column table depending on a key signal. For example, one possible application is the translation of error codes into easily understandable error descriptions. Numerical quantities or texts can be processed.

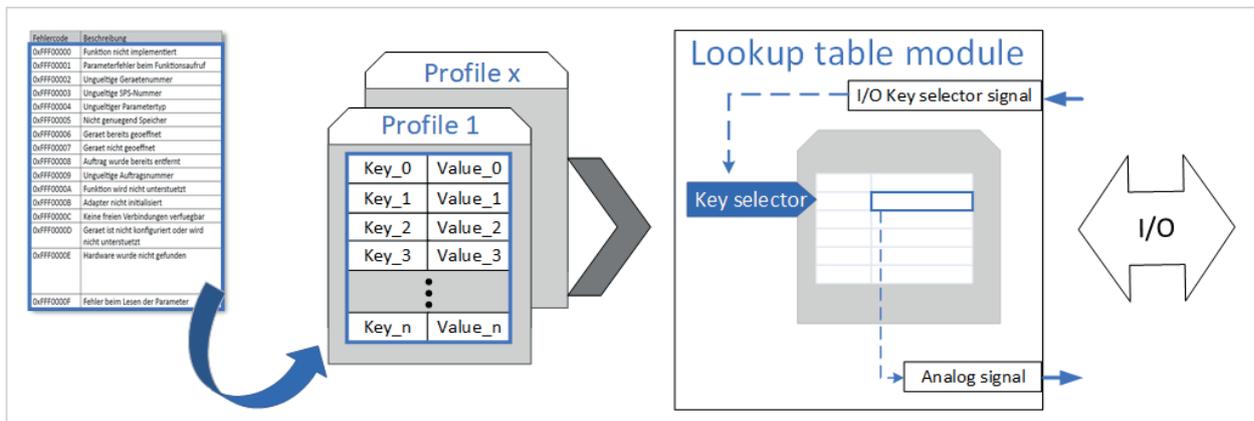
The table contents are transferred to a so-called *profile* and saved. The entry into the profile can be done line by line manually or via copy & paste from a table, e.g., in MS Excel.

A profile always contains only one table.

A key selector signal is used to select the row of the table by comparing the value of this signal with the entries in the *Key* table column. The value from the *Value* column is output as an analog value (numeric or text). If the key selector signal's value does not exist in the table, then a default value is output.

Exactly one profile can and must be selected per lookup table module.

The figure below shows the principle of the profile-based lookup table module.



The module offers 10 output signals as standard, each of which can be individually linked to a key selector signal. You can increase or decrease this number. This allows you to serve multiple applications with one lookup table module.

If, for example, a plant contains 10 drives, each of which sends an error message with an error code, then create a module with 10 analog signals, one for each drive. You can then select the signal with the error message from the corresponding drive as the key selector signal. This way, you can visualize the error messages in plain text for all 10 drives, e.g., in ibaQPanel.

The profile-based concept also offers the possibility of know-how protection and license-controlled use. With the know-how protection feature in the I/O Manager of *ibaPDA*, you can prevent the lookup tables from being viewed or modified. Furthermore, the use of a lookup table can be bound to certain license numbers (dongles) by means of the profile property.

To protect the profiles, see also [Protecting profiles](#), page 92.

### 4.6.16.1 Configure lookup table module

#### Procedure

1. Prepare a table with the keys or codes in the left column and the matching values in the right column. It is easier if you already have the table in digital form.
2. Add a "Lookup Table" module under the "Virtual" interface.
3. Configure a profile:
  - Select the data type for the keys.
  - Set a default text for unknown keys.
  - Transfer the table contents to the profile.
4. Save the profile and then select it in the general module settings under *Table profile*.
5. In the general module settings, adjust the number of analog signals depending on how many applications you want to use this lookup table for.
6. Select the *Analog* tab and give the analog signals a name.
7. In the Key selector column, select the appropriate signal with which the code (key) is transmitted in each case.
8. Click on <OK> or <Apply>.
9. After restarting the acquisition, you see the lookup table module with the analog signals in the signal tree.
10. You can display or process the results like any other signal.

### 4.6.16.2 Lookup table – General tab

#### Basic settings

For a description of the basic settings see ↗ *Common and general module settings*, page 20.

#### Lookup tables

##### Table profile

From the drop-down list, select the appropriate profile that describes the table for this module. Without a profile, the module is invalid. If no profile is available yet, you must create one first.

See ↗ *Add and configure profiles*, page 96.

#### Module layout

##### Number of analog signals

Default value: 10

You can decrease or increase the number, up to 1000. Set the value to exactly match the number of applications that will access this module.

### 4.6.16.3 Add and configure profiles

To add a profile for a lookup table, click the blue *Configure profiles* link in the *General* tab at the bottom of the lookup table module.



Alternatively, you can open the drop-down list in the *Table profile* row and click on *<Add lookup table profile>*.



In both cases, the *Lookup tables* dialog opens. In the second variant, a profile is created immediately.

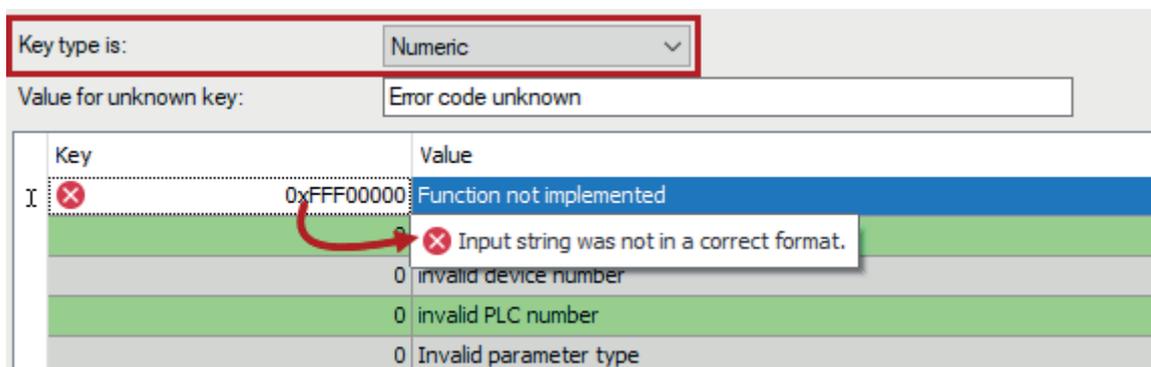
Below the left window of the dialog you will find a set of buttons with the following functions:

	Add profile
	Copy selected profile
	Delete selected profile
	Import profile(s) from a <i>*.lookupTableProfile</i> file
	Export selected profile to a <i>*.lookupTableProfile</i> file

Add a profile and name it so that the function or table content is recognizable.

Select whether the key (contents of the left table column) has a numeric data type or is a text signal. This type usually corresponds to the data type of the signal that transmits the key value and is later configured as a key selector signal.

If the set type and key do not match, an error message is displayed.



Enter another value that will be output if no matching key is found in the table, e.g., "Key unknown".

Now populate the table. You can either enter the data pairs (key-value pairs) manually in each row or copy a ready-made table, e.g., from Excel, to the clipboard and paste it from the clipboard.

To do this, use the button  on the right edge of the dialog.

**Note**



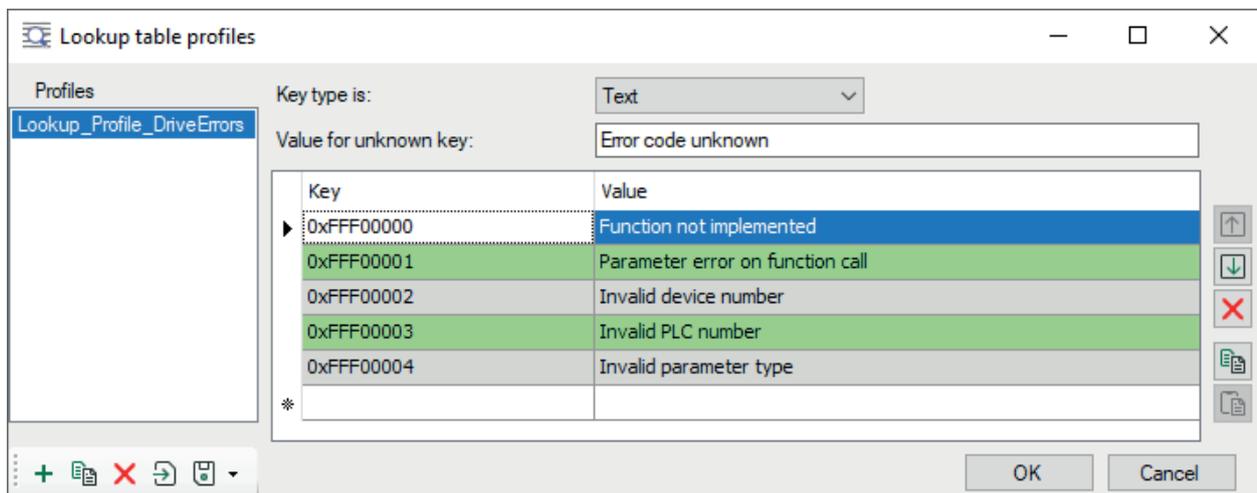
Make sure that the keys are always unique. If there are two or more identical keys in the table, only the result value of the first of these keys is output.

The buttons at the edge have the following functions:

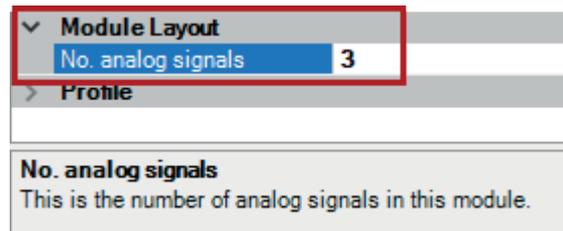
	Move selected row(s) up
	Move selected row(s) down
	Delete selected row(s)
	Copy all rows
	Paste from clipboard, starting from the selected row

**Example**

The following figure shows a profile with five error messages from drives. The error codes (keys) and values are entered as text.



This lookup table is used for 3 drives. Therefore, the lookup table module is configured for 3 analog signals.



The 3 output signals of the module (1), key signals (2) and current values (3) are displayed in the module's *Analog* tab.

Lookup table (14)			
General Analog			
Name	Key selector	Active	Actual
0 Error_Drive1	Aa 15:0: ErrorCode_Input_Drv1	<input checked="" type="checkbox"/>	Error code unknown
1 Error_Drive2	Aa 15:1: ErrorCode_Input_Drv2	<input checked="" type="checkbox"/>	Error code unknown
2 Error_Drive3	Aa 15:2: ErrorCode_Input_Drv3	<input checked="" type="checkbox"/>	Error code unknown

Red brackets below the table indicate: (1) under the Name column, (2) under the Key selector column, and (3) under the Actual column.

In the example, the key signals were implemented with ibaQPanel text inputs for the purpose of easier explanation. In a real application, these can of course be signals from a PLC or other systems that transmit the error code.

After completing the module configuration and restarting the acquisition, the analog signals from the lookup table module are available in the signal tree (1) and can be displayed and recorded. In this example, they were placed on labels in a QPanel (bottom row).

The error code (= key) is entered manually here. If, as with drive 1, a key is entered that is not in the table (profile), then the default text (value for unknown key) appears.

If correct keys are entered, then the appropriate error texts appear in the red fields.

#### 4.6.17 Process condition

This module type is always available. No license is needed.

The function of this module type is to describe different states that a process/plant can have and to identify them via a characteristic value, the so-called condition signal. Depending on the characteristics of a process and the planned application, the various process states are determined by means of so-called process conditions. These process conditions, in turn, result from a combination of different process signals, such as

- Driving mode/operating status
- Speeds, rotational speeds, temperatures, pressures
- Material type
- Product features
- Tool type (material, diameter)

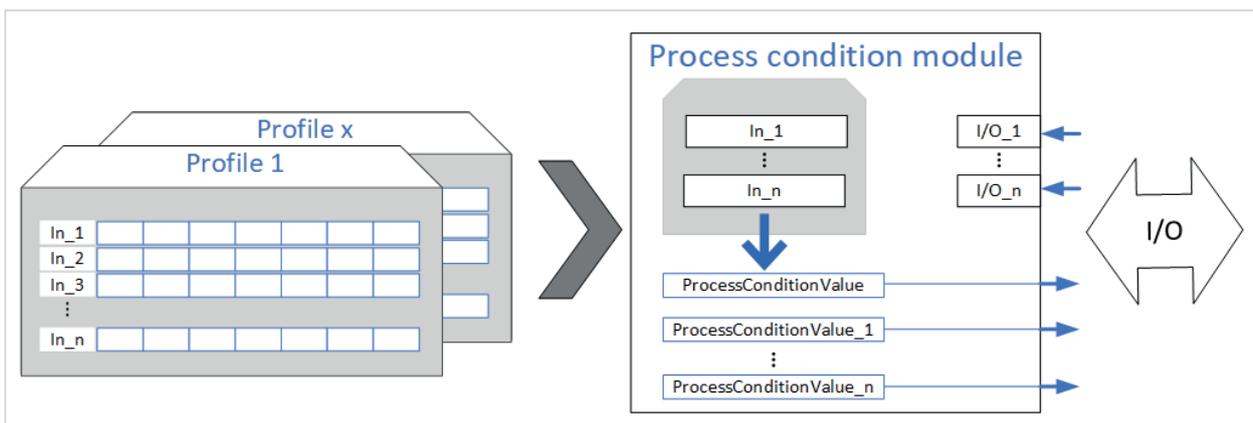
Each process condition is defined and stored in the form of a profile. When configuring a "Process condition" module, the appropriate profile is then loaded, which then allows the module to output the appropriate signals.

Process conditions are mainly used to enable the self-learning modules to communicate the current process condition both when learning the reference curves and during the current calculation. The following modules are so called self-learning modules:

- ibaInSpectra Auto-Adapting
- ibaInCycle Auto-Adapting

During configuration of the auto-adapting modules, this corresponds to the "status signal" parameter in the general module settings.

The figure below shows the principle of the profile-based process condition.



The profile-based concept also offers the possibility of know-how protection and license-controlled use. For example, if your process condition contains information that deserves protection, you can prevent it from being viewed or modified as the profiles are supported by the know-how protection feature in *ibaPDA*'s I/O Manager. Furthermore, the use of a process condition can be bound to certain license numbers (dongles) by means of the profile property.

To protect the profiles, see also [➤ Protecting profiles](#), page 92.

### 4.6.17.1 Configure process condition

#### Procedure

1. Consider which factors and parameters could have an influence on the process with regard to the planned analysis.  
For example, wear detection for saw blades with *ibaInCycle*:
  - Material type of the material to be sawn
  - Saw blade diameter
2. Check whether the appropriate signals for mapping these factors are present in the I/O configuration. If not, create them. These can be analog, digital and/or text signals.
3. Define reasonable values or value ranges for these signals.  
For example, wear detection for saw blades with *ibaInCycle*:
  - 3 material types: 1, 2 and 3
  - Saw blade diameter: 100 ... 300 mm
4. Add a *Process condition* module.
5. Create and configure a profile by entering the previously determined process variables in the *Inputs* table.
6. Select the desired profile in the general module settings.
7. Assign the signals to the now-visible input parameters in the general module settings.
8. To use the process condition, select the desired output signal for the *Process condition* module as the status signal in the module settings, e.g., for the *InCycle Auto-Adapting* module.

### 4.6.17.2 Process condition – General tab

#### Basic settings

For a description of the basic settings see [↗ Common and general module settings](#), page 20.

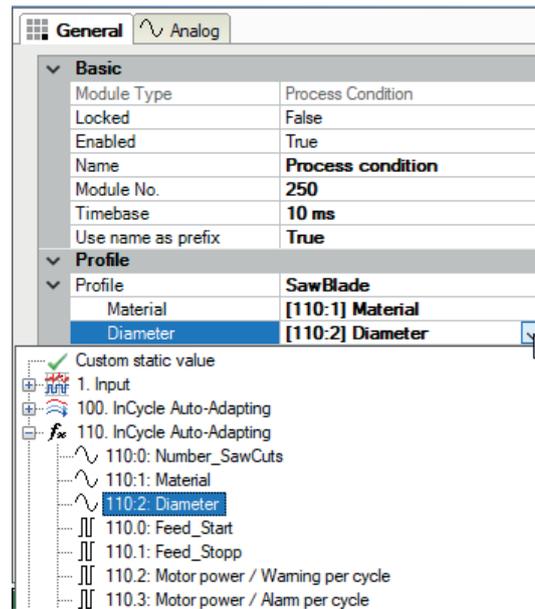
#### Profile

From the drop-down list, select the appropriate profile that describes the process condition for this module. Without a profile, the module is invalid. If no profile is available yet, you must create one first.

See [↗ Add profiles](#), page 101.

If you have created and selected a profile, then the parameters as defined in the profile will automatically appear here.

Next, select the signals that provide the values for the parameters.

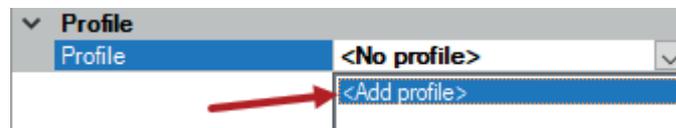


### 4.6.17.3 Add profiles

To add a profile for a process condition, click the blue *Configure profiles* link in the *General* tab at the bottom of the process condition module.



Alternatively, you can open the drop-down list in the *Profile* row and click on *<Add profile>*.



In both cases, the *Process condition profiles* dialog opens.

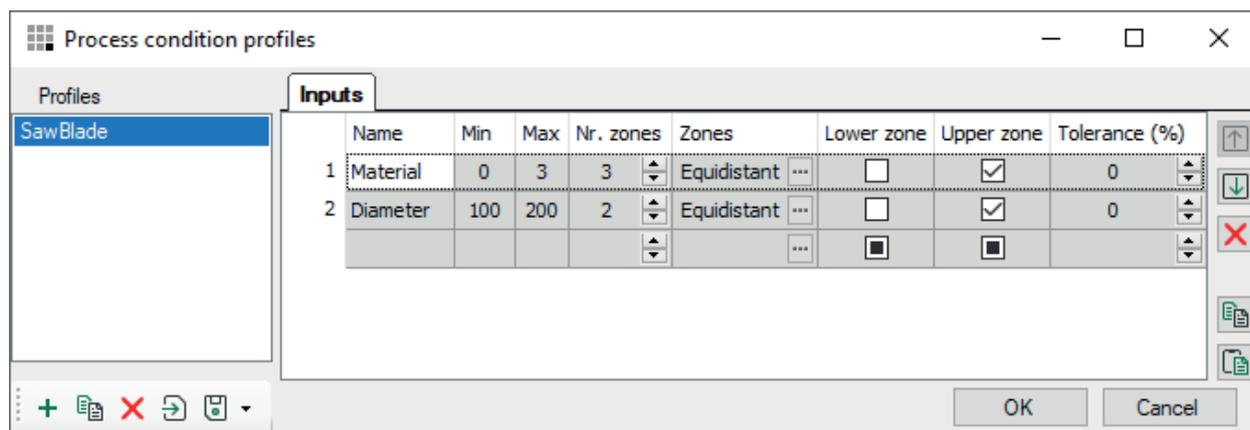
Below the left window of the dialog you will find a set of buttons with the following functions:

	Add profile
	Copy selected profile
	Delete selected profile
	Import profile(s) from a *.processCondProfile file
	Export selected profile to a *.processCondProfile file

Add a profile and name it so that the process condition is recognizable.

Then perform the configuration in the *Inputs* tab.

#### 4.6.17.4 Configure profiles – Inputs



In the *Inputs* tab, enter the placeholders for the parameters in order. Each time you complete a line, the next free line is automatically created.

Process condition profiles can be used for example to monitor the wear of a saw blade as shown in the figure above.

##### Name

Enter the name of the parameter here. This name is then displayed in the module settings for the assignment of the corresponding signal.

##### Min, Max

Enter here the lowest and highest value that the parameter can accept. Only numeric values are allowed.

If parameters are available as text signals in *ibaPDA*, you will need to convert them into numbers first using other virtual functions, e.g., via the *Switch* or *ConvertFromText* functions.

##### No. Zones

Select the number of subranges into which the total value range (between Min and Max) should be divided. Choose the number appropriately and according to the variance of your parameter values.

For example, it makes no sense to define more than three zones if there are only three material types.

##### Zones

If you click on the <...> button in this column, a dialog opens in which the zone boundaries are displayed.

Here you can select whether the total value range should be subdivided automatically into equal zones (equidistant) or according to your own specifications (custom).

##### Lower, upper zone

By applying a checkmark you can determine whether values below Min (Lower Zone) and/or above Max (Upper Zone) should also be considered in the process condition.

### Tolerance (%)

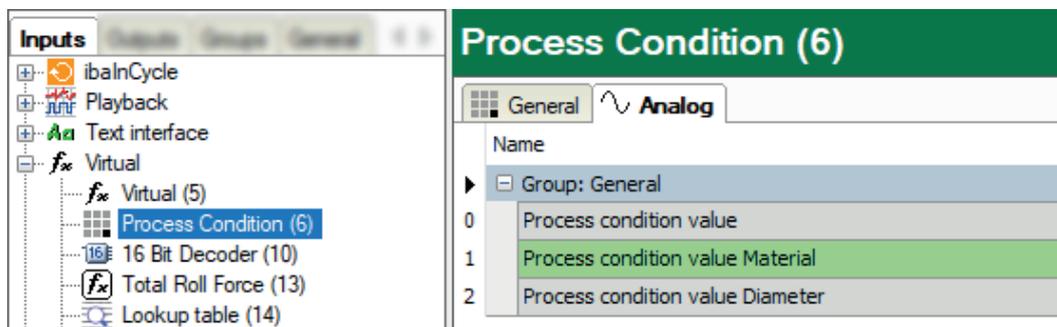
Here, you can set a percentage tolerance value with respect to the zone boundaries. You can thus avoid frequent switching of the zones if, for example, the parameter value fluctuates around the zone limit value.

Example: Saw blade diameter between 100 mm and 300 mm

An equidistant division into 2 zones results in the limit values 100, 200 and 300. If, for example, the diameter value often fluctuates between 199.5 mm and 202, then the zone would be switched with each change, which is undesirable in this case. If you set a tolerance of 5 %, then the zone would only switch at values below 190 mm or above 210 mm.

### 4.6.17.5 Process condition – Analog tab

In the *Analog* tab you will find the output values for the module, the so-called status signals.



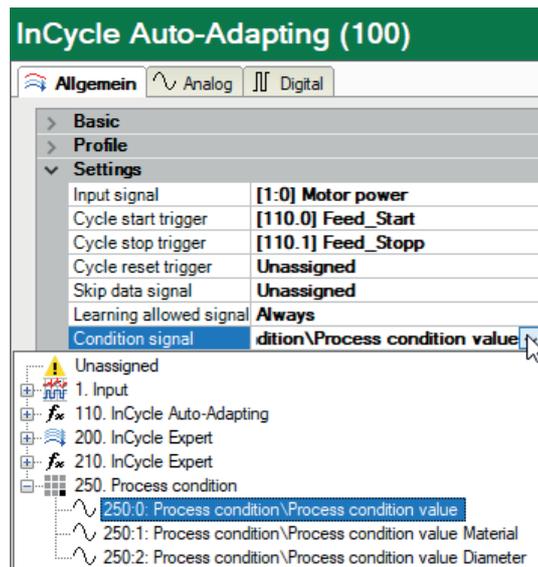
The first signal applies to the entire module, i.e., to the combination of all parameters included in the profile. In addition, there is a separate status signal for each parameter.

The values of these signals (integer) are determined internally in the module and do not have any external function. They are only used to index and manage the various process conditions for use by the Auto-Adapting modules.

Depending on which parameter is relevant for the learning process, you can then choose which signals to enter as a status signal for the Auto-Adapting modules.

### 4.6.17.6 Using process conditions

Once you have finished configuring the "Process condition" module, then you can select the desired condition signal in an Auto-Adapting module in the general module settings.



For each value that this status signal can have, you can teach in any number of measurement curves in order to use these later as comparison curves for the various process statuses.

### Other documentation



A detailed description of the Auto-Adapting modules and the teach-in process can be found in the manuals for the products *ibaInSpectra* and *ibaInCycle*.

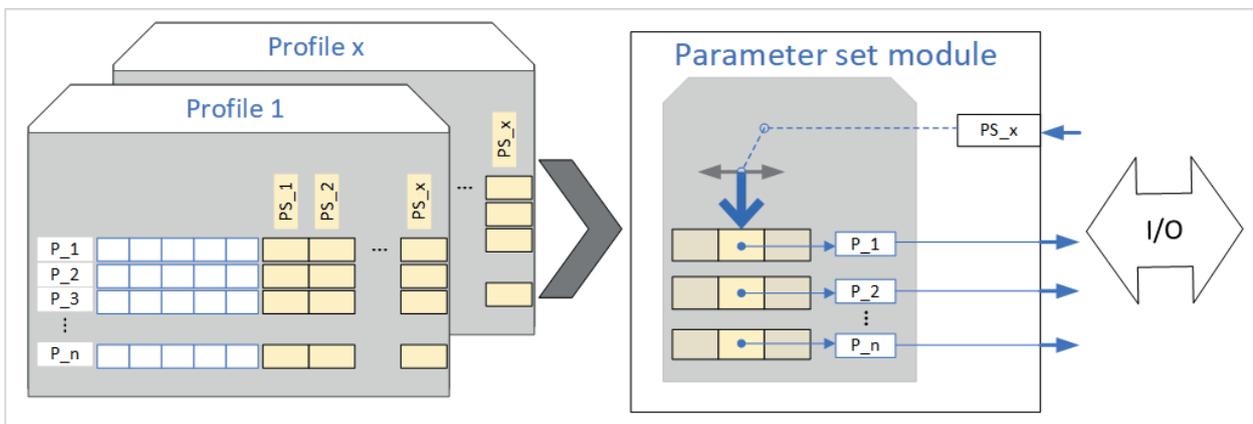
#### 4.6.18 Parameter set

This module type is always available. No license is needed.

The function of this module type is to read and output parameter sets depending on a key signal from stored tables. The value of each key signal is used to select one parameter set, which can contain any number of parameters. One possible application is, for example, the selection of a set of machine setting values depending on the product to be manufactured.

The parameter sets as well as the parameters themselves are defined in so-called profiles. In the profiles, you define the parameter sets with a name and selection value (numeric or as text) for the key signal. For the parameters you define the names, comments, units and data types as well as the values per parameter set.

The figure below shows the principle of the profile-based parameter set.



All parameter sets in a profile have the same parameters.

The parameter data can be entered into the profile line by line manually or via copy & paste from a table, e.g., in MS Excel.

A profile always contains a table with the parameter set definitions and two further tables with the analog and digital parameters. A key selector signal is used to select the parameter set by comparing the value of this signal with the entries in the parameter set definition.

If the value of the key signal matches the defined identifier for the parameter set, then the parameter values belonging to this parameter set are transferred to the module's analog and digital signals.

Subsequently, these signals can be displayed and recorded and used for other purposes, e.g., as input signals for a *Process condition* module.

The profile-based concept also offers the possibility of know-how protection and license-controlled use. With the know-how protection feature in the I/O Manager of *ibaPDA*, you can prevent the parameter sets from being viewed or modified. Furthermore, the use of a parameter set can be bound to certain license numbers (dongles) by means of the profile property.

To protect the profiles, see also ➤ *Protecting profiles*, page 92.

### 4.6.18.1 Configure parameter set module

#### Procedure

1. Consider which parameters are relevant for the planned application and plant/machine, and how many different parameter sets there are.  
Example for a milling machine:
  - Length, width and height of the finished part
  - Part number, customer ID, country, material
  - For a product range of 3 different parts, 3 parameter sets are required.
2. Consider which signal (analog or text) you want to use to select the desired parameter set. This could be a signal from a production planning system or PLC, or a manual input from the operator.
3. Check whether the appropriate signals for mapping these factors are present in the I/O configuration. If not, create them. These can be analog, digital and/or text signals.
4. Decide whether a numeric or a text signal should be used to select the parameter set. Make a note of the values of this key signal for each parameter set ...
  - ... for a numeric signal, this is one value or value range (min-max)
  - ... for a text signal, this is one character string (filter)
5. Add a "Parameter set" module.
6. Create and configure a profile by entering the previously defined parameter sets and the values or texts for the key signal in the *Parameter sets* table.
7. Then enter the desired parameters into the Analog and/or Digital tabs, including at least the name, data type and parameter value, if necessary with additional comments.
8. Select the profile in the general module settings.
9. Select the signal for set selection in the general module settings.

### 4.6.18.2 Parameter set – General tab

#### Basic settings

For a description of the basic settings see [↗ Common and general module settings, page 20](#).

#### Profile

From the drop-down list, select the appropriate profile that describes the parameter set for this module. Without a profile, the module is invalid. If no profile is available yet, you must create one first.

To do this, see [↗ Add profiles, page 107](#).

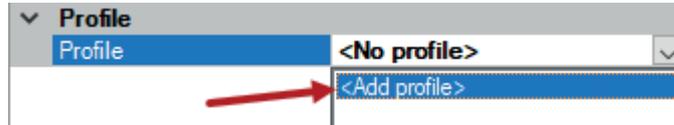
If you have created and selected a profile, then choose the selector signal for the set.

### 4.6.18.3 Add profiles

To add a profile for a parameter set module, click the blue *Configure profiles* link in the *General* tab at the bottom of the module.



Alternatively, you can open the drop-down list in the *Profile* row and click on *<Add profile>*.



In both cases, the *Configure profiles* dialog opens.

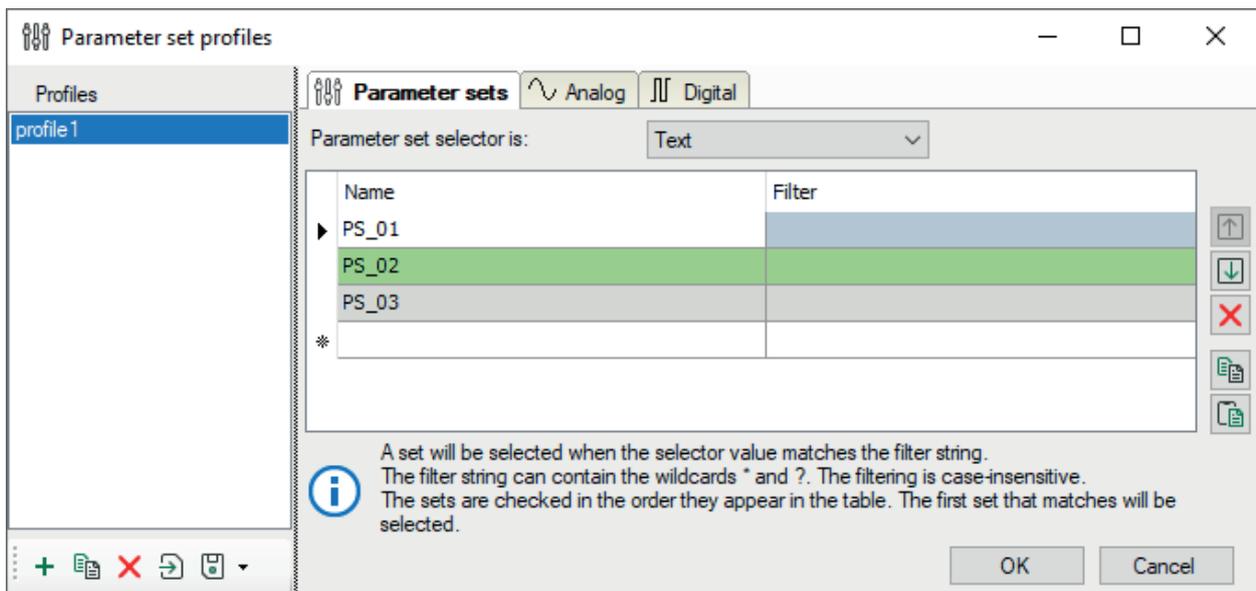
Below the left window of the dialog you will find a set of buttons with the following functions:

	Add profile
	Copy selected profile
	Delete selected profile
	Import profile(s) from a *.paramSetProfile file
	Export selected profile to a *.paramSetProfile file

Add a profile and name it so that the parameter set is recognizable.

Then carry out the configuration in the *Parameter sets*, *Analog* and *Digital* tabs.

### 4.6.18.4 Configure profiles – Parameter sets



In the *Parameter set selector* field, set whether the parameter set is selected via a numeric or a text signal. This selection then applies to all parameter sets in this profile.

Depending on the selection, the table below shows the columns *Name* and *Filter* (text) or *Name*, *Min* and *Max* (numeric).

In the *Name* column enter a comprehensive name for the parameter set. Each time you complete a line, the next free line is automatically created. If you have several parameter sets, enter them line by line. If you have many parameter sets, you can also prepare the table in another program (e.g., MS Excel) and then paste it via the Windows clipboard.

### Numeric set selection

For a numeric set selection, enter the limit values for the set selection signal in the *Min* and *Max* columns.

A set will be selected if the selection value is  $\geq$ Min and  $<$ Max. If Min and Max are equal, the set will be selected where the selection value is equal to Min. The sets are checked in the order in which they appear in the table. The first set that fits is selected.

### Text set selection

A set will be selected if the selection value is equal to the filter string. The filter string can contain the wildcards \* and ?. The filtering is not case-sensitive. The sets are checked in the order in which they appear in the table. The first set that fits is selected.

You can manipulate the lines using the icon buttons on the right margin. Note that the order of the parameter sets can be relevant for the selection.

	Move selected row(s) up
	Move selected row(s) down
	Delete selected row(s)
	Copy all rows
	Paste from clipboard, starting from the selected row

## 4.6.18.5 Configure profiles – analog and digital parameters

Parameter sets		Analog		Digital		Parameter set values			
Name	Comment 1	Comment 2	Unit	Type	Default	PS_01	PS_02	PS_03	
Parameter_1	Length		mm	FLOAT	0	0	0	0	
Parameter_2	Width		mm	FLOAT	0	0	0	0	
Parameter_3	Height		mm	FLOAT	0	0	0	0	
Parameter_4	Part no.			STRING					
Parameter_5	Material			STRING					
*									

In these tabs you can define the parameters and provide the necessary values for each of the previously defined parameter sets. Each parameter set gets its own column to the right of the *Default* column.

Enter a meaningful name for the parameter in the *Name* column and select the appropriate data type in the *Type* column. If required, you can also specify comments and a physical unit. Finally, enter a value in the *Default* column that will be applied to the signal if no valid parameter set has been selected or no parameter set value is available.

Each time you complete a line, the next free line is automatically created. If you have several parameters, enter them line by line. If you have many parameters, you can also prepare the table in another program (e.g., MS Excel) and then paste it via the Windows clipboard.

The same applies for the digital signals, except that you do not need to enter a unit and a data type.

#### 4.6.18.6 Parameter set – Analog and Digital tabs

In the *Analog* and *Digital* tabs you will find the output values for the module. If a parameter set is currently selected while data acquisition is in progress, the actual values are also displayed.

The following figure shows the module's output signals with the signal values according to parameter set PS\_02 from the previous example.

Name	Unit	DataType	Active	Actual
0 Parameter_1	mm	FLOAT	<input checked="" type="checkbox"/>	
1 Parameter_2	mm	FLOAT	<input checked="" type="checkbox"/>	
2 Parameter_3	mm	FLOAT	<input checked="" type="checkbox"/>	
3 Parameter_4		STRING	<input checked="" type="checkbox"/>	
4 Parameter_5		STRING	<input checked="" type="checkbox"/>	

#### 4.6.19 NMEA 0183 decoder

NMEA is a marine standard protocol for the exchange of maritime data, such as e.g. wind, water depth and also GPS data. The virtual *ibaPDA* module NMEA 0183 decoder enables you to evaluate these data. The *NMEA 0183 decoder* module analyzes (decodes) received NMEA 0183 data sets and converts the signals so that *ibaPDA* can use them.

To set up the *NMEA 0183 decoder* module, you must first set up an interface in *ibaPDA* that receives the NMEA 0183 data sets. This interface can be e.g. a module *Serial Text* or *TCP/UDP Text*. In the module *NMEA 0183 decoder*, you set the corresponding signal of the interface as source signal.

A NMEA 0183 message consists of a string of data fields that are separated by commas. In the tabs *Analog* and *Digital* in the *NMEA 0183 decoder* module, you can set these data fields as signals. You define how *ibaPDA* interprets the individual signals.

### 4.6.19.1 NMEA 0183 decoder – General tab

#### Basic settings

For a description of the basic settings see [↗ Common and general module settings, page 20.](#)

#### Source

##### Source signal

Select the source signal to be decoded from the signal tree.

#### Module layout

##### Number of analog and digital signals

Set the number of analog and digital signals for this module (min. 0, max. 1000).

If you enter "0", no *Analog* or *Digital* tab appears. Also, you cannot select the corresponding signals in the symbol browser.

#### NMEA

##### Valid checksum only

Select whether *ibaPDA* ignores NMEA records with invalid checksums (True) or processes all records (False). You can also select the validity of the checksum as a digital signal in the symbol browser.

##### Link "Select NMEA variables"

Use this link to open the symbol browser for selecting the signals to be measured.

### 4.6.19.2 NMEA 0183 Decoder – Register Analog und Digital

In these tabs you configure the analog signals or the digital signals. The signal table has as many lines as specified in the *General* tab at *Number of analog signals* or *Number of digital signals*.

**NMEA 0183 decoder (5)**

	Name	Unit	Sentence	Field	NMEA data type	Acti...	Actual	
5	GGA: Altitude		GGA	9	Real	<input checked="" type="checkbox"/>		^
6	GLL: Latitude		GLL	1	PositionConverted	<input checked="" type="checkbox"/>	51,0266421	
7	GLL: Longitude		GLL	3	PositionConverted	<input checked="" type="checkbox"/>	3,4511332	
8	GLL: Time of day(UTC)		GLL	5	TimeConverted	<input checked="" type="checkbox"/>	62478	
9	GLL: Data status		GLL	6	Text	<input checked="" type="checkbox"/>	A	
10	GGA: CRC Valid		GGA	-1	Bool	<input type="checkbox"/>		v

The yellow part highlights the selected part.

```

$ G N G L L
5 1 0 2 . 6 6 4 2 1
N
0 0 3 4 5 . 1 1 3 3 2
E
1 7 2 1 1 8 . 0 0
A
D

```

### General columns in the signal table

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals](#), page 22.

#### Sentence

Specify the type of the NMEA sentence for the signal.

#### Field

Specify the position of the signal in the NMEA sentence. The fields in the sentence are separated by commas.

#### NMEA data type

Select the data type of the signal.

#### Actual

The *Actual* column shows you the sentence of the source signal as current values. Each line corresponds to one field of the sentence. In the *Field* column in the signal table, you can assign the record fields to the signals.

The different colors refer to the following states in the preview area:

Color	Description
Green (1)	Green lines are assigned to one text signal in the <i>Field</i> column.
Red (2)	Red lines are assigned to multiple text signals in the <i>Field</i> column.
Yellow (3)	When you select lines in the signal table, they are highlighted in yellow.

(1) = least dominant, (3) = most dominant color

### Adding signals to the signal table

#### Via the symbol browser

You can select the signals of the NMEA 0183 sentences via the symbol browser. To open the symbol browser, you have two options:

- Click the link in the *General* tab of the module.

The signals you select in the symbol browser automatically fill the next free row of the signal table. Multiselection is possible. This is helpful if you fill the signal table for the first time or if you want to fill up a signal table.

- In the signal table, select the line where you want to insert the signal. Click the <...> button in the *Sentence* column. The symbol browser opens.

With this method, you determine exactly the position where a symbol is to be entered in the table. Multiselection is not possible.

#### Manual input

If you cannot find the appropriate signals in the symbol browser, you can also enter the signals manually into the table.

1. Enter a name and sentence type in the corresponding columns of a row.
2. Select the appropriate NMEA data type.
3. Activate the signal.

### 4.6.19.3 NMEA 0183 decoder – Diagnostics tab

In the *Diagnostics* tab, all configured signals are listed in a table with their data type and actual value.

### 4.6.20 Vector calculation module

The vector calculation module can be used for calculation of statistical values of a vector. In this case, vectors are multi-dimensional signals (arrays), which should be defined in the *Groups* section in the I/O Manager. Typical vectors are, for example, the signal groups of temperature scanners, flatness or thickness gauges.

The following calculations are available:

- Minimum
- Maximum
- Sum
- Average
- Median
- Standard deviation
- Skewness
- Kurtosis

The calculation of the values works in cross-profile manner over all vector elements for each point of time and not on each vector element over time.

You can use one vector calculation module to do calculations on one or more vectors. For each calculated value you define the vector, the range of vector elements and the requested calculation function. Up to 1000 calculations can be configured in one module. The default setting is 32.

The module creates virtual signals, which are then available for visualization and further processing.

#### 4.6.20.1 Vector calculation module - General tab

##### Basic settings

For a description of the basic settings see [↗ Common and general module settings, page 20](#).

---

##### Note



The time base of the module determines the update time of the calculations. The default setting is 10 ms. Please note, that using smaller timebases in combination with large vectors can cause a high load on the *ibaPDA* server.

---

## Advanced

### High accuracy

If you enable this option (True), then the calculation results are saved as 64-bit floating point values. If you do not enable "High accuracy", then 32-bit floating point values are used.

### Module Layout

Set the desired number of analog signals that this module should supply here. Value range: 1 ... 1000 (Default: 32)

## 4.6.20.2 Vector calculation module - Analog tab

In the *Analog* tab you can define the signals to be calculated. Proceed as follows:

1. Enter a meaningful name for each signal in the *Name* column, which refers to the source vector and the calculated value type.
2. In the *Vector* column select the desired vector signal. Only vector signals are offered, which have been defined in the *Groups* section of the I/O Manager.
3. Define the range of vector elements supposed to be subject to calculation in the columns *First index* and *Last index*. There, you have different options:
  1. Select "First" and "Last", if you want to include all vector elements in the calculation (default).
  2. Enter static index values (integer numbers from 0 to x), if you only want to process a part of the vector, i.e. apply the calculation beginning from and/or up to a specific vector element.
  3. Select analog signals which are dedicated to providing index values. Thus, the range of vector elements for calculation can be altered dynamically.
4. Finally, select the requested calculation function in the *Function* column.
5. Make sure, that the signals are set "active" and close the dialog with <OK>.

The following example explains the application of the module.

### 4.6.20.3 Vector calculation module - Example

A temperature scanner with 12 measuring zones detects the temperature profile of a web-shaped product. The scanner supplies 12 analog signals, which form the vector *VectorDemoSmall* in the *Groups* section of the I/O Manager.

#### Module configuration

General		Analog					
Name	Vector	First index	Last index	Function	Active	Actual	
0	Vector_Min	VectorDemoSmall	First	Last	Minimum	<input checked="" type="checkbox"/>	
1	Vector_Max	VectorDemoSmall	First	Last	Maximum	<input checked="" type="checkbox"/>	
2	Vector_Avg	VectorDemoSmall	First	Last	Average	<input checked="" type="checkbox"/>	
3	Vector_Med	VectorDemoSmall	First	Last	Median	<input checked="" type="checkbox"/>	
4	Vector_Sum	Unassigned	Last	Last	Sum	<input checked="" type="checkbox"/>	
5	Vector_StdDev	Vectors	Last	Last	Standard deviation	<input checked="" type="checkbox"/>	
6	Vector_Skewness	Contour	Last	Last	Skewness	<input checked="" type="checkbox"/>	
7	Vector_Kurtosis	VectorDemo	Last	Last	Kurtosis	<input checked="" type="checkbox"/>	
8		VectorDemoSmall	Last	Last	Minimum	<input type="checkbox"/>	

For this example one signal has been configured for each calculation function, e.g. *Vector\_Min* for minimum, *Vector\_Avg* for average.

The vector *VectorDemoSmall* is selected in the *Vector* column for each signal.

*First index* = First and *Last index* = Last for including all 12 measuring zones in the calculation.

The setting *First index* = 0 and *Last index* = 11 would lead to the same result, in this example.

First index	Last index
First	Last

If you only want to consider the zones around the center, e.g. because the lateral zones often deliver invalid values, you may enter 2 and 9.

Selecting the calculation function for each signal.

Function	Active	Actual
Minimum	<input checked="" type="checkbox"/>	
Maximum	<input checked="" type="checkbox"/>	
Average	<input checked="" type="checkbox"/>	
Median	<input checked="" type="checkbox"/>	
Minimum	<input checked="" type="checkbox"/>	
Maximum	<input checked="" type="checkbox"/>	
Sum	<input checked="" type="checkbox"/>	
Average	<input checked="" type="checkbox"/>	
Median	<input checked="" type="checkbox"/>	
Standard deviation	<input type="checkbox"/>	
Skewness	<input type="checkbox"/>	
Kurtosis	<input type="checkbox"/>	

### Result

The following figure shows in the upper part the trend chart of the temperature scanner in 2D false color representation. The Y-axis shows the values 0 to 11, corresponding to the 12 temperature measuring zones. The temperature values are represented by the colors. For a better display the color scale is adapted to the temperature range from approx. 2200 °C to 2300 °C.

Below, there are four trend graphs showing the calculated values of the vector calculation module for maximum, average, median and standard deviation. The block-shaped trend graphs result from an acquisition timebase of 1 second for the temperature values.



## 4.7 Unmapped

Branch for "parking" temporarily unused modules and signals.

If, for example, you want to disconnect an interface card or input device and thus have to change the I/O configuration, the corresponding modules and signals would get lost because *ibaPDA* will not detect the devices after driver restart. In order to avoid this, you should move via drag & drop the modules from the original interface to the "Unmapped" branch. The configuration of modules and signals will stay there as configured so you can move them back to the original interface after the I/O configuration has been reinstalled.

When you select the "Unmapped" interface in the tree, the following button appears on the right side:

### **Map modules to available interfaces**

When you press this button, *ibaPDA* will attempt to allocate the modules of the "Unmapped" interface to the appropriate "real" interfaces.

## 5 ibaNet data interfaces (for iba devices)

The interfaces of the ibaFOB cards are only visible in the I/O Manager of *ibaPDA* if one or more cards are installed in the computer. iba devices with an ibaNet connection (FO) can only be connected to *ibaPDA* via an ibaFOB card.

ibaNet-E is an Ethernet-based interface with a special protocol supported by compatible devices (e.g., ibaW-750).

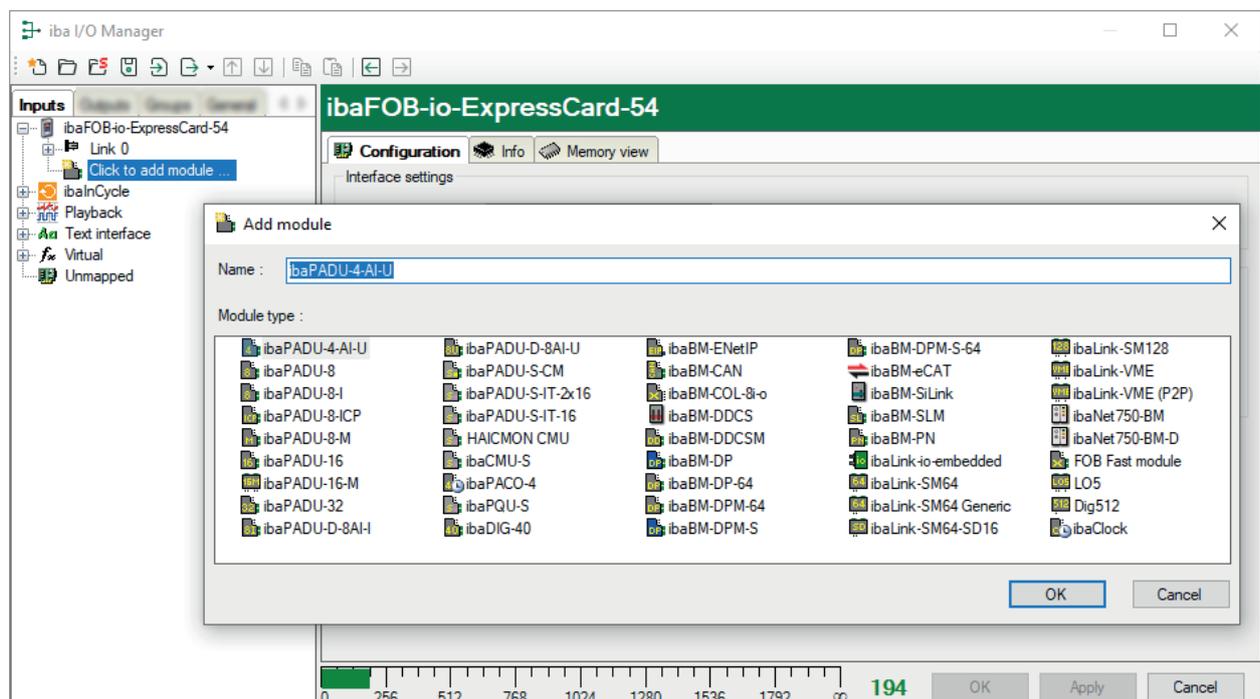
Name	Type	Connection to ...	Comment	Link
ibaFOB-4io <sup>1)</sup>	PCI cards FOB-io-F, 4i-F, 4o	ibaPADU, SM64, SM128, ibaNet750, DPM-64	Supports all devices with 2 Mbit/s, 3.3 Mbit/s and 5 Mbit/s	<a href="#">↗ ibaFOB-io-, 2io-, 4io- etc., page 122</a>
ibaFOB-4io-D, -Dexp	PCI cards ibaFOB-io-D, -2i-D, -4i-D, -4o-D, -2io-D  PCIe cards ibaFOB-io-Dexp, -2i-Dexp, 4i-Dexp, -2io-Dexp	ibaPADU, SM64, SM128, ibaNet750, ibaBM..., ibaPADU-S family ibaLink-VME	Supports all devices and FOB protocols incl. 32 Mbit flex	<a href="#">↗ ibaFOB-io-, 2io-, 4io- etc., page 122</a>
ibaFOB-4io-S <sup>1)</sup>	PCI cards FOB-io-S, -4i-S, -4o	ibaPADU, SM64, SM128, ibaNet750, DPM-64	No devices with 32 Mbit/s or 32 Mbit flex	<a href="#">↗ ibaFOB-io-, 2io-, 4io- etc., page 122</a>
ibaFOB-4i-X <sup>1)</sup>	PCI cards FOB-io-X, -4i-X, -4o	ibaPADU, SM64, SM128, ibaNet750, ibaBM..., ibaPADU-S	Supports 32 Mbit/s, yet not PADU-8-ICP, -M!	<a href="#">↗ ibaFOB-io-, 2io-, 4io- etc., page 122</a>
ibaFOB-io-ExpressCard	ExpressCard FOB-io-Express Card /54, /34	ibaPADU, SM64, SM128, ibaNet750, ibaBM..., ibaPADU-S family ibaLink-VME	for notebooks with an ExpressCard slot of 34 mm or 54 mm; supports all devices (compatible with ibaFOB-io-D)	<a href="#">↗ ibaFOB-io-ExpressCard, page 134</a>

Name	Type	Connection to ...	Comment	Link
ibaFOB-io-USB	USB adapter	like ibaFOB-io-ExpressCard	for notebooks and PCs with USB 2.0 or higher; cannot be combined with other ibaFOB-io cards.	
ibaNet-E	Ethernet-based (NIC)	ibaW-750	The protocol can also be implemented on third-party devices (OEM)	➤ <i>ibaNet-E</i> , page 135

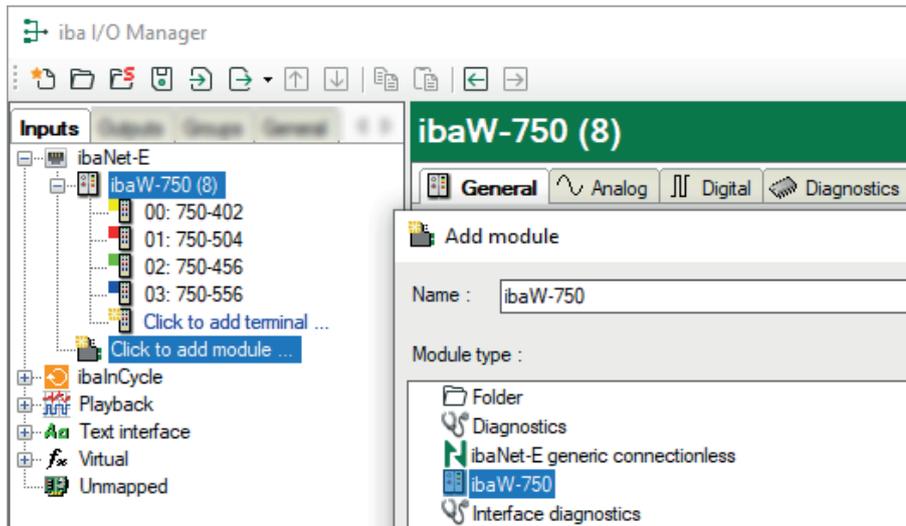
<sup>1)</sup> Only to support backward compatibility; components are no longer available

The devices themselves are configured as modules. Depending on the type of device, the measurement signals are configured in the directly connected device module or in other, subordinate device modules (e.g., *ibaPADU-S* modular system, *ibaNet750*) or software modules (e.g., bus modules such as *ibaBM-DP*).

You can see which devices or modules you can use on an interface in the I/O Manager if you click on "Click to add module" under the interface. *ibaPDA* always shows you the possible combinations. The figure below shows an example for selecting possible devices on a free link of an ibaFOB-io-ExpressCard.



The figure below shows an example for ibaW-750 on ibaNet-E.



## 5.1 ibaNet protocols

### ibaNet (FO)

Different ibaNet communication protocols were also developed with the different generations of the ibaFOB boards. The cards of the *ibaFOB-D* series support all previously realized protocols, guaranteeing a complete downward compatibility.

The following table shows an overview of the available protocols with an indication of transmission speeds, signal number, data acquisition times (sampling time) and typical devices:

Protocol	Maximum signal count per fiber optic	Sampling time	Typical devices
One-way operating modes, inputs only			
2 Mbit	32 INT + 32 digital	≥1 ms	ibaPADU16/32 (old, S/N <1000)
3 Mbit	64 INT + 64 digital	≥1 ms	ibaPADU8/16/32
	64 REAL+ 64 digital	≥1 ms	ibaLink-SM-64-i-o
5 Mbit	8 INT + 8 digital	≥50 μs	SIMATIC TDC LO5
32 Mbit	64 INT + 64 digital	≥50 μs	SIMATIC TDC LO6
	128 INT + 128 digital	≥100 μs	SIMATIC TDC LO6
	512 REAL + 512 digital	≥800 μs	ABB AC 800PEC (1 ms)
	DPM-S mode	≥800 μs	ibaBM-DPM-S (1 ms)
	8 x (64 INT + 64 digital)	≥1 ms	ibaBM-COL-8i-o (1 ms)
Two-way operating modes, inputs and outputs (output link required)			
5 Mbit	8 INT + 8 digital	≥40 μs	ibaPADU-8-M ibaPADU-8-ICP
32 Mbit Flex	variable*	≥10 μs	ibaPADU-S-CM
One-way operating modes, only outputs (output link required)			
3 Mbit	64 REAL + 64 digital	≥1 ms	ibaNet750-BM
32 Mbit	Does not yet support software-side.		

Table 1: All protocols of the ibaNet technology in comparison

\*Example: User data transmission of 64 bytes at 25 μs sampling time or 3100 bytes at 1 ms.

## ibaNet-E

ibaNet-E is the protocol of the interface with the same name.

ibaNet-E enables fast, efficient and deterministic communication between the acquisition computer and other involved components. Standard Ethernet cabling and standard network infrastructure can be used for data communication. This supports the usage in a virtualized environment, as no special boards in the *ibaPDA* computer are needed for connecting appropriate peripheral I/O-devices (e. g. ibaW-750).

With ibaNet-E, different applications, i.e. data acquisition from several data sources as well as controlling through outputs, can be realized. Not every ibaNet-E device supports the full ibaNet-E scope of functions. Therefore, some functionalities may not be available for all ibaNet-E devices.

Feature	ibaNet 32Mbit Flex	ibaNet-E
Automatic detection of devices	Yes	Yes
Module configuration out of ibaPDA	Yes	Yes
Deterministic transmission of measurement data	Yes	Yes
Synchronized sampling	10 $\mu$ s ... 1.4 ms	$\geq 1$ ms
Use of standard (industrial grade) infrastructure	No	Yes
Connection to ibaPDA via...	ibaFOB board	Standard Ethernet board
Bandwidth	Max. 32 Mbit/s	Whatever is supported by infrastructure
Bandwidth reservation	Guaranteed	Not guaranteed
Transmission	Isosynchronous	Buffered (ACQ) or isosynchronous (PLC)
Cable	FO	Standard Ethernet
Topology	Ring	Line and/or star
Number of devices	$\leq 15$	no limitation

Table 2: Features of ibaNet-E compared to ibaNet 32Mbit Flex

The protocol ibaNet-E can be implemented as an OEM component in 3rd party devices.

## 5.2 ibaFOB-io-, 2io-, 4io- etc.

This section describes the configuration of the latest generation of *ibaFOB* cards, the *ibaFOB-D* interface card. An FOB (fiber optical board) is an interface card for fiber optical connections.

It applies to all versions of the card, such as *ibaFOB-io-D*, *ibaFOB-2io-D* and *ibaFOB-4io-D* (*FO-B4i-D + FOB-4o-D*), as well as to PCI and PCIe types. *ibaFOB-...-Dexp* is the PCI Express variant of the card.

Beside the new 32 Mbit flex protocol, the generation of *ibaFOB-D* cards can also process all previous ibaNet data transfer rates and modes of measurement.

These descriptions generally also apply to previous card generations such as -PCI, -S and -X. Some functions, however, could have been removed, replaced or added. Furthermore, previous cards had only limited options regarding data transfer rate and modes of measurements.

All modules that may previously have been assigned in *ibaPDA* to *ibaFOB-F*, *ibaFOB-S* and *ibaFOB-X* cards, can also be assigned to an *ibaFOB-D* card / *ibaFOB-Dexp* card.

The system automatically detects which card(s) are installed and shows them in the tree structure. You may use cards of different generations simultaneously in the same computer. You will see different names in the tree structure depending on the card model installed.

Cards with output channels ("...io...") or with an output module ("ibaFOB-4o...") can be used to emit signals from *ibaPDA*. The card then also appears in the tree structure in the outputs area of the I/O Manager. Suitable modules for the outputs can then be added there.

For more information see part 2, *Outputs*.

---

### Notes



- Please note that the *ibaFOB-F*, *ibaFOB-S* and *ibaFOB-X* cards are only supported by 32-bit Windows operating systems (x86).
- Up to 8 *ibaFOB-4i-D* cards are supported in an ibaPDA computer.
- When using cards of the type *ibaFOB-D*, *ibaFOB-SDexp* or *ibaFOB-TDCexp*, only these cards can be configured as the interrupt master. This serves to protect against a DMA memory overflow.

---

### Other documentation

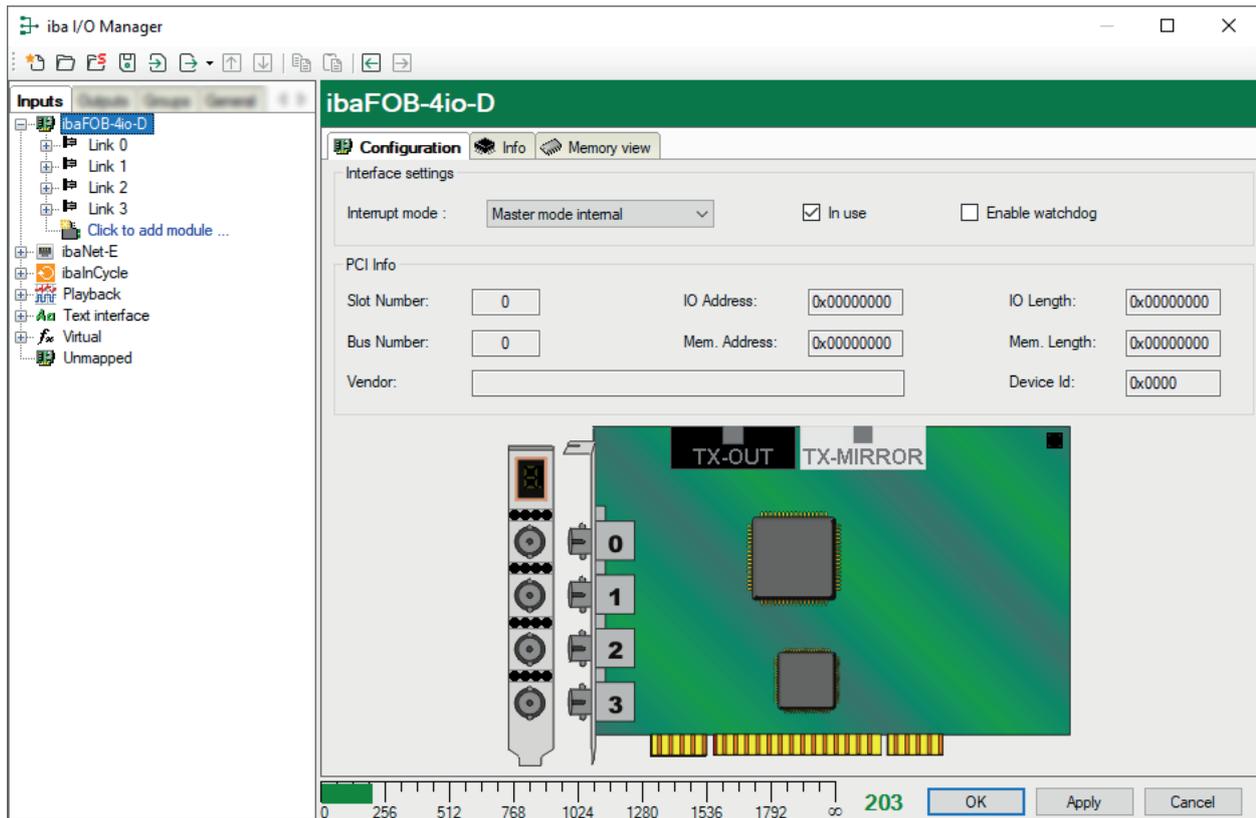


For more information about the hardware features of the cards, see the relevant manual for the respective card.

Information on previous *ibaFOB-io* cards can be found in the corresponding card manual or in previous editions of the *ibaPDA* manual.

---

If you select the *ibaFOB-...-D* interface in the tree structure of the *ibaPDA* I/O Manager, then a simple presentation of the card is displayed in the *Configuration* tab in the right part of the dialog.



The dialog window provides you with the information described in the following chapters.

## 5.2.1 Interface settings

### Interrupt mode

Use the drop-down list in this field in order to select the interrupt mode of the card.

Here, you decide whether the card generates the interrupt for other iba PCI card installed (interrupt master) or receives an interrupt, i. e. working as interrupt slave. This is the interrupt which synchronizes the iba PCI/PCIe cards over the flat ribbon cable.

Interrupt master here means synchronization master.

- Master mode internal
  - This mode may be set for only *one* card, no matter if internal or external. Select this mode if the card should generate the interrupt for all other iba PCI/PCIe cards.
- Master mode external
  - This mode may be set for only *one* card, no matter if internal or external. Select this mode if the *ibaPDA* system is to be synchronized from an external source, such *ibaBM-SLM*. This function is required, among others, for bus-synchronous measurements using the SIMOLINK Monitor (*ibaBM-SLM*). The PC is then synchronized with the input data stream on channel 0 of the relevant board. The recorder feed stops if no data is received.
- Slave mode
  - This mode must be set for all the other boards besides the master board.

### "In use" selection box

If this option is disabled, *ibaPDA* ignores the corresponding board. This is, for example, necessary if *ibaPDA* and *ibaLogic* are active in a hybrid configuration on the same computer, with each program requiring its own board. One card can only be used by one application at a time.

### "Enable watchdog" checkbox

The board watchdog can be enabled for the purpose of monitoring the proper operation of *ibaPDA* by another system. If the watchdog is enabled, the board will generate an alarm message in case the acquisition is not running for more than 2 seconds. The alarm message can only be used by the FO output channel (I/O Manager: "Alarms"). Therefore, an extension module *ibaFOB-4o* or an *ibaFOB-io-D* card is required. In case of an alarm, all output values will be set to 0 (zero) in the alarm message.

The alarm is also set during the booting of the computer. If an alarm occurs, then all red LEDs light up on the FO inputs. The LED for the debug on the card then also lights up red.

## 5.2.2 PCI Info

In the PCI Info area of the dialog, you find the following information:

- Slot Number  
Number of the PCI slot where the card is plugged in.
- Bus Number  
PCI bus connected with this slot
- IO Address  
Start address of the I/O address range of the card (hex)
- Mem. Address  
Start address of the memory region (hex)
- IO Length  
Size of the I/O address range (hex), free / reserved
- Memory Length  
Size of the memory region (hex), free / reserved
- Manufacturer  
Name of the card manufacturer
- card ID:  
PCI card ID (hex); also listed in the PCI table when system is booting.

## 5.2.3 Card display

The graphical representation of the card shows the animated displays and indicators of the board. The 7-segment display shows the actual board number whereas the LEDs indicate the actual status of the connections.

The displays and their meaning are summarized in the table below:

**FO input status display**

LED	Status	Description
Run (green)	blinking	Power is on and the channel is functioning properly.
	OFF	Controller stopped (hardware failure)
Slow link (yellow)	ON	Receiving messages on this channel with 2 Mbit/s, 3.3 Mbit/s or 5 Mbit/s, link correctly configured.
	blinking	Receiving messages on this channel with 2 Mbit/s, 3.3 Mbit/s or 5 Mbit/s, yet connection was configured for another protocol.
	OFF	No 2 Mbit/s, 3.3 Mbit/s or 5 Mbit/s messages found. Fiber optics not connected?
Fast link (white)	ON	Receiving messages on this channel with 32 Mbit/s, connection is configured correctly for 32 Mbit/s.
	blinking	Receiving messages on this channel with 32 Mbit/s, yet link configured for another protocol.
	OFF	No 32 Mbit/s telegrams found. Fiber optics not connected?
Error (red)	ON	Watchdog alarm output switch opened
	blinking	Running the "Golden FPGA Flash Rescue mode"
	OFF	Normal state

**7-segment display**

The 7-segment display shows the following information:

- Horizontal segment only: Card not initialized
- Numbers 0 to 7: Board ID after board was initialized
- The decimal point in the display indicates whether the board is configured to be:
  - An internal interrupt master (dot is on) or
  - An external interrupt master (dot is blinking) or
  - An interrupt slave (dot is off)

**"Debug" LED**

This multicolor LED on the board (upper right corner) is for debug and service purposes only.

LED	Status	Description
Trouble-shooting	green:	Board is active and ok.
	red:	Watchdog alarm or different error

### Connector for ibaFOB-4o



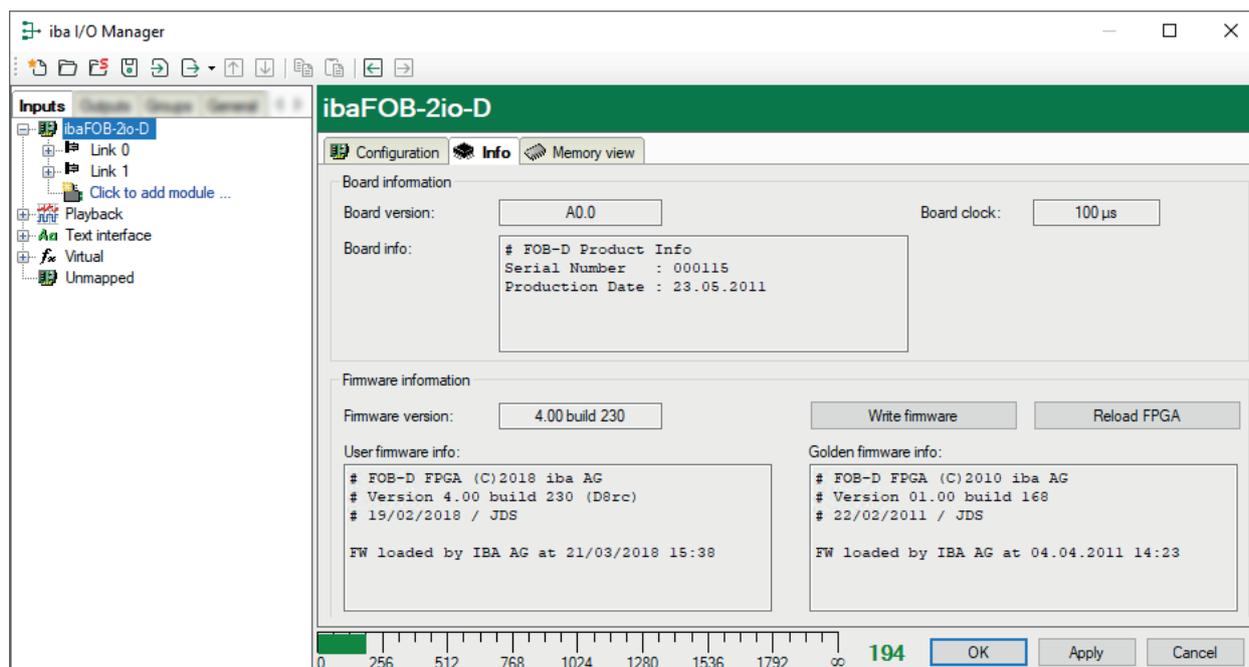
The *ibaFOB-4o* module can be connected to the "TX-OUT" or "TX-MIRROR" connector of an iba-FOB-...-D card and can thus fulfill different tasks.

- TX-OUT: For use as an output module for output signals that *ibaPDA* or *ibaLogic* is to output to external devices (*ibaPADU-8-O*, *ibaNet750-BM-D* with output terminals, etc.)
- TX-MIRROR: Used to mirror the FO input channels; the incoming FO data streams are output again on the hardware side, i.e., with no detour via the software, e.g., for parallel data supply of an *ibaQDR* system.

The graphic shows where an *ibaFOB-4o* module is connected via a green square.

### 5.2.4 Info tab (card level)

On the *Info* tab, you can see information about the board and the loaded firmware. Service and support functions, such as reloading the FPGA and updating the firmware, are available on this tab.



#### Note



The firmware should first be updated after consulting with the Service and Support department of iba AG.

For detailed information on loading the firmware, refer to the manual of the card.

For older card types *ibaFOB-F*, *ibaFOB-S* and *ibaFOB-X* as well as for *ibaFOB-TDC*, *ibaFOB-SD* and *ibaCom-L2B*, there is an Info tab at the card level, which looks a little different. The currently

loaded firmware version is shown on the different processors. It is also possible to load the firmware into the card. The button <Write firmware> opens in the “Load firmware” dialog.

### 5.2.5 Memory view (card level)

This view provides very detailed information about memory use for service purposes.

### 5.2.6 ibaFOB-io on link level

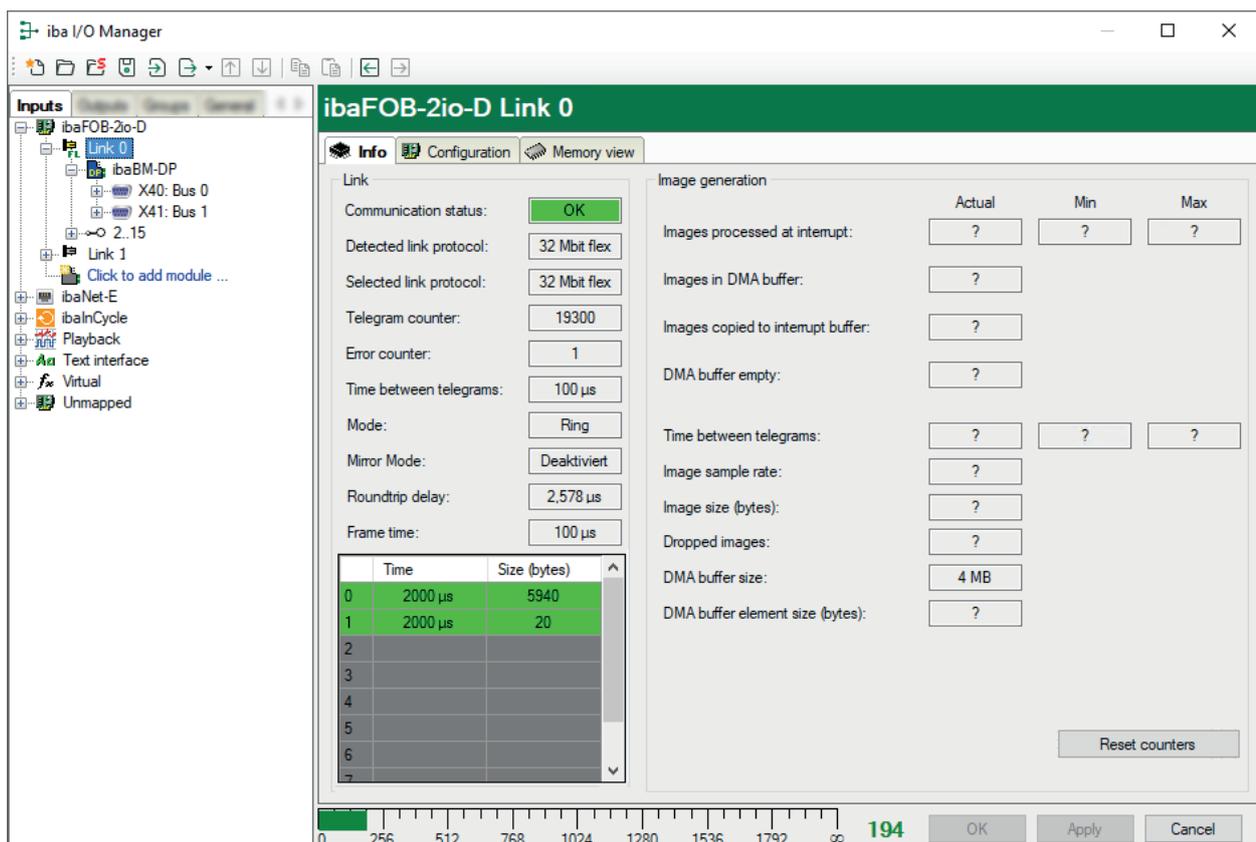
When the Link level is highlighted in the tree, you will find some information about communication and processors of the card.

For this purpose, there are also the *Info*, *Configuration* and *Memory view* tabs.

In the tree view, the link icon indicates whether devices are configured on the link and which protocol is set for the link.

### 5.2.7 Info tab (link level)

Though being very detailed and specialized, some of the information under the *Info* tab may also be useful for the average user.



Displayed properties and parameters vary depending on the operation mode the link is running in.

**Permanent information in the Link area****Communication status**

OK if the FO communication works. This means that the messages received correspond with the mode that is configured on the link. The connection mode is determined by the module connected to the link, for example if the connection mode is set to 3.3 Mbit, a *ibaPADU-8* module is set up. If an *ibaPADU-8-ICP* module is configured, the link will be set to 5.0 Mbit/s mode.

**Detected link protocol**

This is the link protocol detected by the card. This can either be 2.0 Mbit/s, 3.3 Mbit/s, 5.0 Mbit/s, 32 Mbit/s or "?" (no connected device).

**Selected link protocol**

This is the link protocol in which the connection is configured. It is determined by the module connected to the link.

**Telegram counter**

Counter of messages correctly received.

**Error counter**

Counter of received messages containing errors (e.g. incorrect checksum)  
If this counter is changing, this means the FO communication is not working correctly.

**Time between telegrams**

The time span between the two latest correctly received messages.

**Additionally available in the Link area when running in 3.3/2.0 Mbit/s mode****FO signal strength**

This is the difference between the maximum value and the minimum value received from the FO unit. The maximum value is 255. The higher this value, the stronger the FO input signal.

**Device ID**

This is the device ID of the last device in the FO chain, attached to the link.

**Telegram format**

This is the format of the analog data transferred in the telegram. The possible values are integer, real and S5 real.

**Additionally available in the Link area when running in 5.0 Mbit/s mode****Status of the device firmware**

Status of the firmware of the connected device.

**Table gain (amplification) and filter**

Gains and filters configured in the device. This applies only to the *ibaPADU-8-ICP* device.

**"Image generation" area**

The information on the right side of the dialog describes the image generation. An image is a collection of bytes that the board writes into the computer system memory via DMA. This image contains all the data of the measurement signals from that link.

Here is a short description of the image generation information:

**Images processed at interrupt**

These counters show how many images were available in the DMA buffer when the last interrupt occurred. This value should normally correspond with the interrupt time divided by the image sampling rate.

**Images in the DMA buffer**

This is the number of images that are in the DMA buffer. This number should remain constant. If this number starts increasing, an error occurred. This can be the case if e. g. an interrupt is missed.

**Images copied to interrupt buffer**

This counter shows how many images have been captured from the DMA buffer and processed by *ibaPDA*. This counter should increase constantly.

**DMA buffer empty**

This counter increments each time when the interrupt fires while the DMA buffer is empty. If this is the case, the driver will use the value 0 (zero) for all signals that are on this link. This happens if the FO connection is interrupted.

**Time between telegrams**

The time between the last two messages received correctly

This corresponds to the time in the FO communication information; however, the driver maintains the minimum and maximum values. There should not be much difference between the minimum and maximum values.

**Image sample rate**

The rate at which the board writes images to the DMA buffer. This should be at least equal to or faster than the fastest time base of the modules connected to this link.

**Image size**

The image size given in bytes. Multiplying the image size by the image recording rate gives the number of bytes/sec transferred through this link over the PCI bus.

**Dropped images**

This counter increments when the card's DMA FIFO is full and additional images are added. If this is the case, a severe error occurred. This means that the card cannot transfer any images via the PCI bus.

**DMA buffer size**

Size of the DMA buffer for this interface

**DMA buffer element size**

Size of the elements in the DMA buffer (in bytes)

**Available in 32Mbit Flex mode**

By using the 32Mbit Flex mode protocol you can connect up to 15 devices in a ring topology. The link numbers 1 to 15 in the signal tree below the *ibaFOB-D* board correspond to the address, which is set on the rotary switch of the connected device.

Additional information is available as follows:

### In the link area

#### Mode

Status of the connection mode:

- Ring: one or more devices (cascade) are bidirectionally connected and the FO ring is closed.
- Open chain: Only the fiber optic input is connected to a device. The output is not connected or the FO ring is interrupted

#### Time between telegrams

Time between two telegrams measured by the *ibaFOB-D* board. It should be equal to the configured frame time.

#### Mirror mode

Indication whether the mirror mode is enabled or not.

In mirror mode multiple *ibaPDA* systems can receive the data from the same *32Mbit Flex*-devices at the same time.

For detailed information on the mirror mode, refer to the manual of the card.

#### Roundtrip delay

Telegram cycle in the closed FO ring. The time depends on the number of the connected devices in the ring (approx. 2  $\mu$ s per device).

Due to the roundtrip delay the data of the connected devices might be captured asynchronously (up to one telegram cycle).

#### Frame time

Fixed cycle time the data frames are being sent. (Smallest set time base of the connected devices or 100  $\mu$ s, if this time base is an integer multiple of 100  $\mu$ s). The time base of all devices must be a multiple of the smallest time base.)

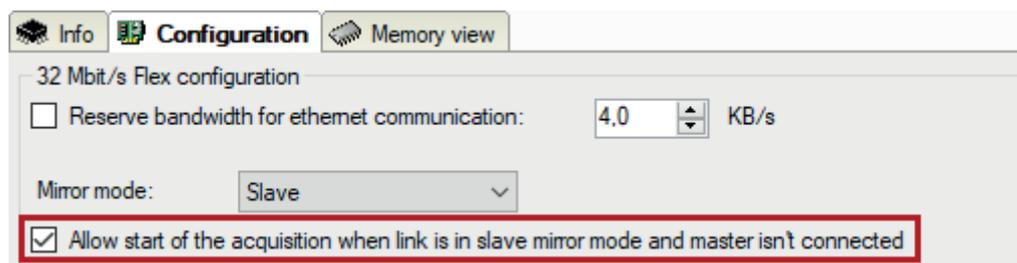
#### Table

The table shows the cycle time and the data size of the respective channel:

- Row 0: Ethernet channel
- Rows 1-15: connected devices with the respective address 1-15

## 5.2.8 Configuration tab (link level)

In the *Configuration* tab, there are additional settings for operation with 32Mbit Flex.



**Reserve bandwidth for Ethernet communication.**

The Ethernet channel (address 0 in the Flex configuration) is used on the fiber optic link to transmit configuration data, to communicate with a web interface if necessary and, in particular, to display the Profibus diagnostics in the case of ibaBM-DP. If many devices are configured with a lot of signals, it may happen that only the minimum size of 1 kB/s is reserved for the Ethernet channel. This is often insufficient and the Profibus diagnosis may not be displayed as a result or communication with the web interface may become very slow.

With the "Reserve bandwidth for Ethernet communication" option, a fixed bandwidth can be reserved for Ethernet communication for the relevant link so that the devices are always accessible. The default value of 4 kB/s is usually sufficient for configuration data and Profibus diagnosis.

**Mirror mode**

3 settings are available for mirror mode:

- Disabled: The data are not mirrored so that this *ibaPDA* system is the only one capable of configuring the devices and acquiring data.
- Master: This *ibaPDA* system configures the Flex devices on this link: The data and the device configurations are mirrored so that other *ibaPDA* systems can also capture the data.
- Slave: This *ibaPDA* system receives the device configuration from the *ibaPDA* master so that it can capture the data configured by the *ibaPDA* master.

For detailed information on the mirror mode, refer to the manual of the card.

**Allow start of acquisition when link is in slave mirror mode and the master is not connected**

If this option is enabled, the acquisition is started even if the slave does not receive a configuration from the master within 6 s. It then starts the acquisition with the last valid configuration.

**32 Mbit/s Flex frame simulation**

The data size per participant is dynamically allocated in a 32Mbit Flex ring and is calculated by *ibaPDA*. The amount of data depends on the number of analog and digital signals configured in *ibaPDA* and the smallest timebase configured in the ring.

An integrated simulation function calculates the amounts of data that can be transferred per participant over the FO connection with the 32Mbit Flex protocol.

The data size in bytes of each device on the link and the timebase of the data acquisition on the link (in  $\mu\text{s}$ ) is needed for the calculation.

The values can be entered manually or retrieved automatically from the current configuration either by clicking the <Estimate values from current configuration> button or by selecting the corresponding link of the ibaFOB card in the module tree.

The devices in the Flex ring and the corresponding data sizes are listed in the table on the left. Address 0 is reserved for the Ethernet channel and is not editable.

The "Flex frame utilization" section shows how much bandwidth is still available. The color of the section changes with the utilization rate.

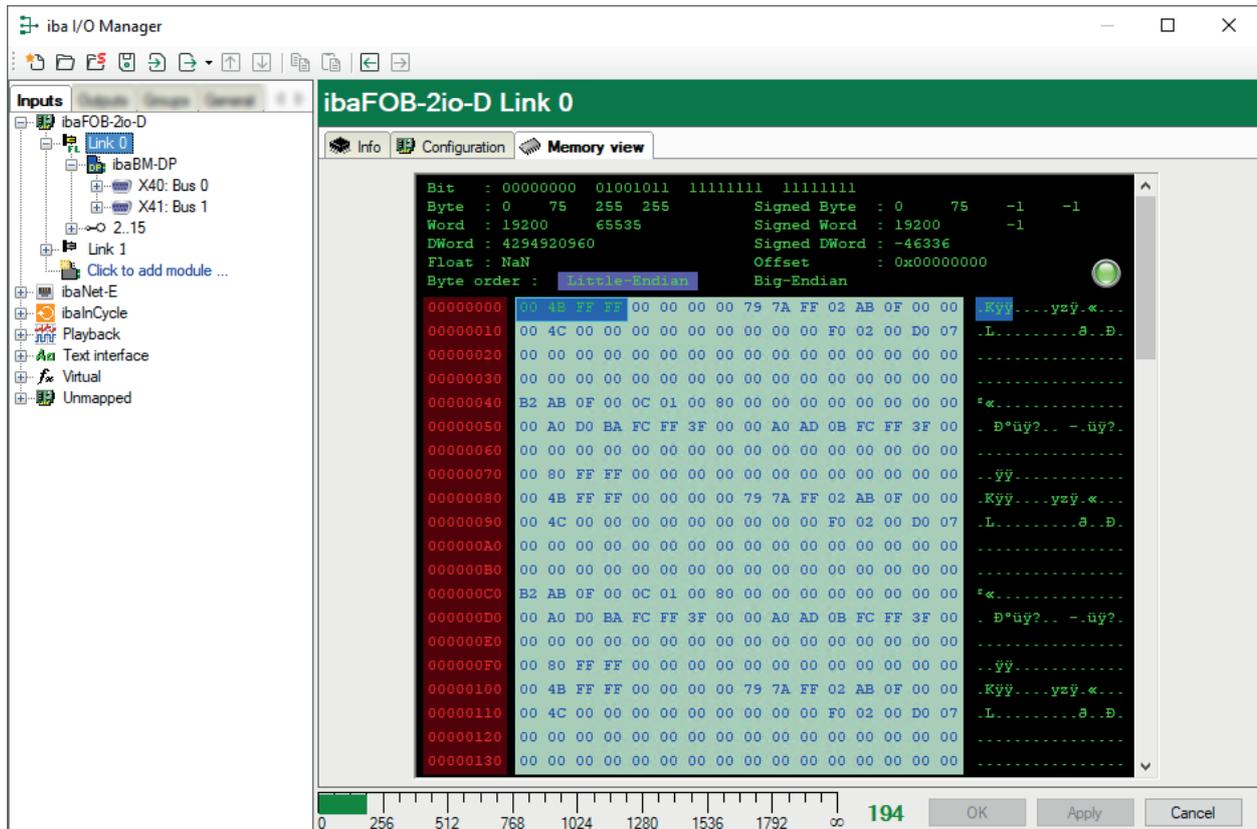
- Green: OK
- Orange: Band width for the Ethernet channel < 3 kB/s
- Red: Too much data configured.

The data values obtained automatically are estimated first. When the configuration is applied by clicking <OK> or <Apply>, the actual data sizes are shown on the *Info* tab.

If too much data is configured, you can either reduce the number of signals to be measured or increase the timebase.

### 5.2.9 Memory view tab (link level)

This view provides very detailed information about memory use for service purposes.



This pane provides very detailed information about memory use for service purposes.

Usually there is no need to access this dialog. The blinking green light indicates a running system. The offset addresses equal the address entries in the signal tables of the data modules. You can identify the format of the incoming data (swap mode). Make a right mouse click in order to switch the address mode from hexadecimal to decimal (or vice versa) and to freeze the display.

### 5.3 ibaFOB-io-ExpressCard

This card is part of the *ibaFOB-D* card family and should be used for measuring purposes on mobile computers. It can also be used for connecting a notebook computer with iba field devices such as *ibaPADU* analog-digital converter units, *ibaNet750* devices, *ibaLink* system connections and iba bus monitors.

Compared with the former *ibaPCMCIA-F* card, with its integrated FO adapter the *ibaFOB-io-ExpressCard* is suitable for higher data transmission rates of up to 32 Mbit/s, including 32 Mbit Flex, and exhibits a performance similar to the *ibaFOB-io-D* card. It can also be used under the current 64-bit Windows operating systems.

*ibaPDA*, version 6.24 or higher, is required for this card to be used.

The configuration dialogs of the *ibaFOB-io-ExpressCard* correspond to those of the *ibaFOB-io-D* card described above.

The card is available in two versions:

- ibaFOB-io-ExpressCard-34 for ExpressCard slots 34 mm wide
- ibaFOB-io-ExpressCard-54 for ExpressCard slots 54 mm wide

If you right-click on the board's node in the interface tree in the I/O Manager then the context menu allows you to export the configuration file of the board. There must be a running measurement for an export. This function is only used for diagnostic purposes for our support.

You can insert the card into the ExpressCard slot of your notebook or remove it at any time. If the card is inserted and connected while the *ibaPDA* service is running, the automatic detection starts immediately.

---

#### Other documentation



For more information on the hardware features of the cards, see the card manual for the *ibaFOB-io-ExpressCard*.

---

## 5.4 ibaNet-E

### Description

The ibaNet-E interface is used for data acquisition from iba or also third-party devices with the ibaNet-E transmission protocol in *ibaPDA*, hereafter referred to as ibaNet-E devices.

ibaNet-E enables fast, efficient and deterministic communication between the acquisition computer and other components involved. Standard Ethernet cabling and standard network infrastructure can be used for data communication.

With ibaNet-E, different applications, i.e. data acquisition from several data sources as well as controlling through outputs, can be realized. Not every ibaNet-E device supports the full ibaNet-E scope of functions. Therefore, some functionalities may not be available for all ibaNet-E devices.

### Interface configuration

The configuration of the devices is performed in the *ibaPDA* software. The ibaNet-E interface is available in the interface tree by default. Device-specific modules are added to the ibaNet-E interface to acquire data from iba devices. General modules can be added to the interface to acquire data from third-party devices.

### Port no.

This is the port which is used for communication via ibaNet-E. If necessary, you can change the setting. Default setting is 7082.

### <Reset port to default>

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

### Allow ports through firewall

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

### Network interfaces

Using this drop-down list, you can select which network adapters on your computer should be 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 of the network adapters, an error message will be produced when validating the I/O configuration. By default the option *All* is selected.

### Available modules

- ibaW-750  
The ibaW-750 module is used to acquire data from ibaW-750 devices. For further information, please refer to the ibaW-750 manual. No *ibaPDA-Interface-ibaNet-E* license is required to connect iba devices.
- ibaNet-E generic connectionless  
This module is used to acquire data from third-party devices via ibaNet-E. The module *ibaNet-E generic connectionless* is only available if the license *ibaPDA-Interface-ibaNet-E* is present.

■ S7 Request/S7 Request Decoder

These two modules providing direct access on operands and symbols in a SIMATIC S7 PLC are also supported on the ibaNet-E interface. The modules are only available if the license *ibaPDA-Request-S7-DP/PN/ibaNet-E* is present.

■ HiPAC Request

This module for direct access on data in a Danieli HiPAC PLC is only available if the license *ibaPDA-Request-HiPAC* is present.

■ Interface diagnostics

By using this module you can acquire some status information from the ibaNet-E interface.

**Product name**

ibaPDA-Interface-ibaNet-E (Art. no. 31.001006), basic license for 2 connections

one-step-up-Interface-ibaNet-E (Art. no. 31.101006), extension license for 2 connections

These interface licenses are required if third-party devices will communicate with ibaPDA via ibaNet-E. The extension can be added up to 126 times, i.e., a maximum of 254 connections can be used in total.

**Other documentation**



There is a separate manual for the ibaNet-E interface, which is applicable for the licensed product *ibaPDA-Interface-ibaNet-E*.

## 5.5 Modules for ibaPADU-series devices

These module types are used for acquiring signals through parallel analog-to-digital units (PADU). The connection to the *ibaPDA* PC is established with optical fibers via *ibaFOB* cards. The modules correspond exactly to the devices, i.e., the signals to be measured are configured in the device module.

**Other documentation**



A detailed description of the modules and their configuration can be found in the corresponding device manual.

Module name	Device	ibaFOB	Comment	Manual
ibaPADU-8	ibaPADU-8	-S, -X, -D	8/16/32 AE ±10 V and	ibaPADU-8-16-32-32-R
ibaPADU-16	ibaPADU-16		8/16/32 DE ±24 V	
ibaPADU-32	ibaPADU-32		Sampling rate 1 kHz	
ibaPADU-32	ibaPADU-32R			
ibaPADU-8	ibaPADU-8AI-U			ibaPADU-8AI-U/-8AI-I

Module name	Device	ibaFOB	Comment	Manual
ibaPADU-8	ibaPADU-8-I ibaPADU-8-AI-I	-S, -X, -D	8 AE $\pm 20$ mA and 8 DE $\pm 24$ V acquisition rate 1 kHz	ibaPADU-8-I ibaPADU-8AI-U/- 8AI-I
ibaPADU-8-M	ibaPADU-8-M	-S, -D	8 AE $\pm 10$ V and 8 DE $\pm 24$ V acquisition rate 25 kHz	ibaPADU-8-M
ibaPADU-8-ICP	ibaPADU-8-ICP	-S, -D	8 AE ICP sensor and 8 DE $\pm 24$ V acquisition rate 25 kHz	ibaPADU-8-ICP
ibaPADU-16-M	ibaPADU-16-M	-S, -D	Modular design: 16 AE depending on the equipment and 16 DE $\pm 24$ V acquisition rate 25 kHz	ibaPADU-16-M
ibaDig-40	ibaDIG-40	-S, -D	40 DE $\pm 48$ V acquisition rate 25 kHz	ibaDig-40
ibaPA- DU-D-8AI-U	ibaPA- DU-D-8AI-U	-D	8 AE $\pm 2.5$ V, $\pm 10$ V, $\pm 24$ V, $\pm 60$ V and 8 DE $\pm 24$ V acquisition rate up to 40 kHz	ibaPADU-D-8AI-U/- 8AI-I
ibaPADU-D-8AI-I	ibaPA- DU-D-8AI-I	-D	8 AE $\pm 20$ mA, 0...20 mA, 4...20 mA and 8 DE $\pm 24$ V acquisition rate up to 40 kHz	ibaPADU-D-8AI-U/- 8AI-I
ibaPADU-4-AI-U	ibaPA- DU-4-AI-U	-D	4 AE $\pm 250$ mV, $\pm 500$ mV, $\pm 1$ V, $\pm 2.5$ V, $\pm 5$ V, $\pm 10$ V, $\pm 24$ V acquisition rate up to 100 kHz	ibaPADU-4-AI-U
ibaPACO-4	ibaPACO-4	-S, -X, -D	4 counter inputs 8 DE $\pm 24$ V acquisition rate 1 kHz	ibaPACO-4

### 5.5.1 Example: ibaPADU-8

#### 5.5.1.1 PADU – General tab

##### Basic settings

For a description of the basic settings see [↗ Common and general module settings](#), page 20.

## Input range

### Min / Max

This setting is available for PADU 8 and PADU 8-I modules. It allows ibaPADU-8 and ibaPADU-8-I devices to be available for different input voltage levels. If known, enter the low end of the input area in "MIN" and the high end of the input range in "MAX". Changing these settings will also change the default values in the analog signal table of the module ("Min" / "Max" columns) accordingly.

### 5.5.1.2 PADU – Analog and Digital tab

#### General columns in the signal table

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals, page 22.](#)

## 5.6 Modules for ibaPADU-S-series devices (modular system)

The module configuration follows a hierarchical structure according to the device design.

The module of a central unit must first be created below the interface (FOB-D-card). The lower-level device modules can then be created in order to configure the signals there. The structure of these modules corresponds to the device. The number of signals cannot be increased.

Module name	Device	Comment	Manual
<b>Central units</b>			
ibaPADU-S-CM	ibaPADU-S-CM	Data acquisition without preprocessing	ibaPADU-S-CM
ibaPADU-S-IT-2x16	ibaPADU-S-IT-2x16	Data acquisition and preprocessing (ibaLogic)	ibaPADU-S-IT-2x16
ibaPADU-S-IT	ibaPADU-S-IT	Data acquisition and preprocessing (ibaLogic)	ibaPADU-S-IT
ibaPQU-S	ibaPQU-S	Measured value acquisition and calculation of characteristic values for the electrical power quality	ibaPQU-S
ibaCMU-S	ibaCMU-S	Measured value acquisition and calculations for the vibration analysis (condition monitoring)	ibaCMU-S
<b>Input modules</b>			
ibaMS3xAI-1A/100A ibaMS3xAI-1A ibaMS3xAI-5A	ibaMS3xAI-1A/100A ibaMS3xAI-1A ibaMS3xAI-5A	3 AI	ibaMS3xAI-1A-5A-1A100A
ibaMS4x-AI-380VAC	ibaMS4x-AI-380VAC	4 AI	ibaMS4xAI-380VAC
ibaMS8x-AI-110VAC	ibaMS8x-AI-110VAC	8 AI	ibaMS8xAI-110VAC

Module name	Device	Comment	Manual
ibaMS16xAI-10V ibaMS16xAI-10V-HI ibaMS16xAI-24V ibaMS16x-AI-20mA	ibaMS16x-AI-10V ibaMS16xAI-10V-HI ibaMS16x-AI-24V ibaMS16x-AI-20mA	16 AI	ibaMS16xAI-10V/-10V-HI/-24V/-24V-HI/-20mA
ibaMS16x-DI-220V ibaMS16xDI-24V	ibaMS16x-DI-220V ibaMS16x-DI-24V	16 DI	ibaMS16xDI-24V/-220V
ibaMS32xDI-24V	ibaMS32x-DI-24V	32 DI	ibaMS32xDI-24V
ibaMS8xICP ibaMS8xIEPE	ibaMS8xICP ibaMS8xIEPE	8 AI for ICP vibration sensors 8 AI for IEPE vibration sensors (additional setting options)	ibaMS8xICP ibaMS8xIEPE
ibaMS-4xUCO	ibaMS-4xUCO	Pulse counter and frequency meter (digital)	ibaMS-4xUCO
<b>Output modules</b>			
ibaMS16x-AO-10V ibaMS16x-AO-20mA	ibaMS16x AO-10V ibaMS16x AO-20mA	16 DO	ibaMS16xAO-10V/-20mA
ibaMS16xDO-2A		16 DO	ibaMS16xDO-2A
ibaMS32x-DO-24V		32 DO	ibaMS32xDO-24V
<b>Input and output modules</b>			
ibaMS4xADIO	ibaMS4xADIO	4 AE ( $\pm 10$ V oder $\pm 20$ mA) + 4 DE (24 V) + 4 AA ( $\pm 10$ V) + 4 DA	ibaMS4xADIO
ibaMS16x-DIO-24V	ibaMS16x-DIO-24V	16 DI + 16 DO	ibaMS16xDIO-24V

**Note**

You need an FO card with input and output links of the *ibaFOB-D* type with a firmware version V2.00 (build 172) or higher. Otherwise, you need to do a Firmware update. You can find a description (in the *ibaFOB-D* manual) and the latest firmware on the "iba Software & Manuals" data medium included in the delivery.

### 5.6.1 Example: ibaPADU-S-CM

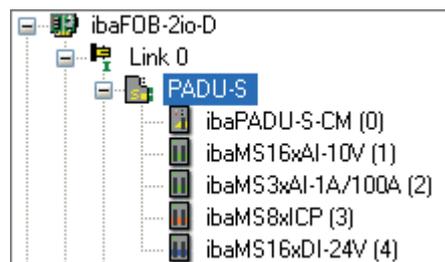
The structure of the devices and modules in the ibaPADU-S system is hierarchically structured.

The first module, which can be found below the interface, is a general "PADU-S" module, in which the parent settings for the entire ibaPADU-S unit are made and some diagnostic functions are provided.

Then, below it, are the modules for the individual devices according to the 5 slots on the ibaPADU-S-unit.

You must first configure the central unit and then the individual input/output modules.

The figure below shows an example of a central ibaPADU-S-CM unit with 4 input components.



#### 5.6.1.1 PADU-S – General tab

##### Basic settings

For a description of the basic settings see [↗ Common and general module settings, page 20](#).

##### Connection

##### IP address

The IP address or host name of the ibaPADU-S-CM device (read only).

##### Enable/disable automatically

If TRUE then the acquisition starts even if a device is missing. For Flex enabled devices, *ibaPDA* periodically attempts to reestablish the connection to the device while the acquisition is ongoing. If a connection can be restored, *ibaPDA* automatically restarts the acquisition.

##### More functions

##### Write configuration to device

Writes the current configuration to the device

##### Read configuration from device

Reads the configuration stored most recently from the device.

Click <OK> or <Apply> to apply the modified settings.

#### 5.6.1.2 PADU-S – Analog and Digital tab

##### Note



The *Analog* and *Digital* tabs are only displayed when the acquisition has been started with analog or digital input modules.

In the list, you can see the configured analog or digital signals and the current values.

### 5.6.1.3 PADU-S – Diagnostics tab

The *Diagnostics* tab contains information on the hardware, firmware and FPGA version as well as the serial number of the central unit and of the connected modules.

#### Write firmware

With this button, it is possible to perform firmware updates. Select the update file `padusc-m_v[xx.yy.zzz].iba` in the browser and start the update with <OK>.

---

#### Note



This process may take several minutes and must not be interrupted. After an update, the device will restart automatically.

---

#### Reset to factory settings

Click this button to reset all settings to the factory defaults after confirming the following prompt with <Yes>.

You will then receive the following message and the device will automatically perform a restart after completion.

### 5.6.1.4 ibaPADU-S-CM – General tab

#### Basic settings

For a description of the basic settings see [↗ Common and general module settings](#), page 20.

### 5.6.1.5 ibaPADU-S-CM – Digital tab

You can enter a name for the signal and two additional comments here if you click on the  symbol in the signal name field.

#### Debounce filter

In the dropdown menu, you can choose the operating mode for the debounce filter. The following settings are available: Off, Stretch rising edge, Stretch falling edge, Stretch both edges, Delay both edges.

#### Debounce time (µs)

You can set the debounce time here in µs.

#### Active

Enabling/disabling the signal

---

#### Other documentation



Configure the plugged analog and digital modules. The description can be found in the module manuals.

## 5.7 Modules for ibaBM series devices (bus modules)

For devices from the bus modules series, such as *ibaBM-DP*, the measurement signals are configured in software modules, which are subordinate to the device module and differ in the form of data processing. The number of signals and the data type can usually be varied.

### Other documentation



A detailed description of the modules and their configuration can be found in the corresponding device manual.

### Bus modules (1st level devices)

Module name	Device	ibaFOB	Comment	Manual
ibaBM-CAN	ibaBM-CAN	-S, -X, -D/-Dexp	CanOpen;	ibaBM-CAN
ibaBM-COL-8i-o	ibaBM-COL-8i-o	-X, -D/-Dexp	Data concentrator for ibaNet; combines up to 8 links with 3.3 Mbit/s (F-mode) to a 32 Mbit link	ibaBM-COL-8i-o
ibaBM-DDCS	ibaBM-DDCS	-D/-Dexp	ABB DDCCS Drive Bus; 32Mbit Flex	ibaBM-DDCS
ibaBM-DDCSM	ibaBM-DDCSM	-S, -X, -D/-Dexp	ABB DDCCS Drive bus; device no longer available	ibaBM-DDCSM
ibaBM-DP	ibaBM-DP	-S, -X, -D/-Dexp	Profibus; with FOB-S and -X compatibility mode only (such as ibaBM-DPM-S-64, -DPM-S)	ibaBM-DP
ibaBM-DPM-64	ibaBM-DP	-S, -X, -D/-Dexp	Profibus; module for compatibility mode, such as ibaBM-DPM-S-64)	ibaBM-DP
ibaBM-DPM-64	ib-aBM-DPM-64	-S, -X, -D/-Dexp	Profibus; device no longer available	ibaBM-DPM-64
ibaBM-DPM-S	ibaBM-DPM-S	-S, -X, -D/-Dexp	Profibus; device no longer available	ibaBM-DPM-S
ib-aBM-DPM-S-64	ibaBM-DPM-S	-S, -X, -D/-Dexp	Profibus; device no longer available	ibaBM-DPM-S-64
ibaBM-eCAT	ibaBM-eCAT	-D/-Dexp	EtherCAT;	ibaBM-eCAT
ibaBM-ENetIP	ibaBM-ENetIP	-D/-Dexp	EtherNet/IP	ibaBM-ENetIP
ibaBM-SiLink	ibaBM-SiLink	-S, -X, -D/-Dexp	Sinamics Link;	ibaBM-SiLink
ibaBM-SLM	ibaBM-SLM	-S, -X, -D/-Dexp	Simolink; device no longer available	ibaBM-SLM
ibaBM-PN	ibaBM-PN	-D/-Dexp	Profinet;	ibaBM-PN

**Submodules (2nd level data)**

Module name	Device	Comment	Manual
Sniffer	ibaBM-CAN	CanOpen;	ibaBM-CAN
Connection...	ibaBM-COL-8i-o	Modules can be added to each of the 8 connections for devices in the 3.3 Mbit mode, e.g., up to 8x ibaPA-DU-8 in series.	ibaBM-COL-8i-o
Data Set	ibaBM-DDCS	ABB DDCS Drive Bus; 32Mbit Flex	ibaBM-DDCS
Parameters			
Diagnostics			
Data set telegram counter			
Active slave	ibaBM-DP	Module for an active DP slave	ibaBM-DP
Sniffer			
Active slave decoder			
Sniffer decoder			
S7 Request		Module for request to S7 (optional access to symbols) <sup>2)</sup>	ibaPDA-Request-S7-DP/PN
S7 request (iba-Com-L2B-compatible)			
S7 request Dig512 (iba-Com-L2B-compatible)			
S7 request decoder			
FM458 Request		Module for request to S7-FM458 <sup>2)</sup>	ibaPDA-Request-FM458/TDC
TDC request			
		Module for request to SIMATIC TDC <sup>2)</sup>	

<sup>2)</sup> only available with license

Module name	Device	Comment	Manual
Standard	ibaBM-eCAT	Module for an EtherCAT slave	ibaBM-eCAT
EtherCAT decoder		Module for an EtherCAT slave (packed digital signals)	
TwinCAT request		Module for request to Beckhoff TwinCAT <sup>3)</sup>	
EtherNet/IP sniffer	ibaBM-ENetIP	Module for listening to data traffic between scanner and adapter	ibaBM-ENetIP
EtherNet/IP sniffer decoder		Module for listening to data traffic between scanner and adapter for packed digital signals	
Generic	ibaBM-SiLink	Module for Sinamics link controller with freely definable signals, various data types for analog signals	ibaBM-SiLink
Control unit		Module for Sinamics link controller with 16 data words and 1 digital signal	
Device slot	ibaBM-PN (device)	Module for a PROFINET-IO device	ibaBM-PN
Device slot decoder		Module for a PROFINET-IO device (packed digital signals)	
S7 Request		Module for request to S7 (only available with license)	ibaPDA-Request-S7-DP/PN
S7 request decoder		Module for request to S7 (packed digital signals) (only available with license)	

<sup>3)</sup> only available with license

Module name	Device	Comment	Manual
Sniffer	ibaBM-PN (TAP)	Module for listening to data traffic	ibaBM-PN
Sniffer decoder		Module for listening to data traffic (packed digital signals)	
Sniffer SiLink		Module for monitoring data traffic at a SINAMICS Control Unit	
Bus diagnostics		Module for acquisition of predefined diagnostic signals from the connected PROFINET network	
Device diagnostics		Module for acquisition of predefined diagnostic signals from a specific PROFINET network	

### 5.7.1 Example: ibaBM-DP with active slave

To be able to use *ibaBM-DP* with *ibaPDA*, the device must be set up in the I/O Manager of *ibaPDA*. To do this, you must have established the communication link between the device and *ibaPDA* as described in the device manual.

The "ibaBM-DP" device module as well as an example of the "Active slave" submodule are described below.

For detailed information and a description of the other submodules, see the manual for the device.

The "Active slave" submodule can also be used to output data via the Profibus. Devices and submodules are therefore also displayed in the *Outputs* area of the I/O Manager.

For more information see part 2, *Outputs*.

#### 5.7.1.1 ibaBM-DP device module

The device module of type ibaBM-DP has 5 different tabs. The *General*, *Diagnostics* and *PROFIBUS Browser* tabs are always available. The *Analog* and *Digital* tabs contain dynamic online views on the analog and digital signals captured by the device. Both of these tabs are therefore only visible after the submodules have been added and the configuration has been transmitted to the device.

### 5.7.1.1.1 ibaBM-DP – General tab

The screenshot shows the 'iba I/O Manager' window with the 'General' tab selected for the 'ibaBM-DP' module. The left sidebar shows a tree view of the system components, including 'Link 0' through 'Link 3' and various input modules. The main configuration area is divided into sections: Basic, Connection, Bus 0, and Bus 1. The 'Basic' section includes fields for Module Type (ibaBM-DP), Locked (False), Enabled (True), Name (ibaBM-DP), Timebase (10 ms), and Use name as prefix (False). The 'Connection' section includes Mode (Flex mode), IP Address (172.29.0.101), Auto write configuration (True), Auto enable/disable (False), and Redundancy mode (False). The 'Bus 0' and 'Bus 1' sections each have fields for Enable default values (True), Default analog value (0), Default digital value (0), and Active slaves. A 'Name' field is also present with the instruction 'The name of the module.' and a 'Read configuration from device' button. At the bottom, a progress bar shows a value of 194, and there are 'OK', 'Apply', and 'Cancel' buttons.

#### Basic settings

For a description of the basic settings see [↗ Common and general module settings](#), page 20.

#### Time base

Acquisition timebase in ms used for this device. With 32Mbit Flex, cycles up to 0.5 ms are possible (depending on the number of signals). The smallest timebase in compatibility mode is 1 ms.

#### Connection

##### Mode

Choose between Flex mode (the 32Mbit Flex protocol is used on the fiber optic link) and compatibility mode (the fixed 32Mbit protocol is used). More signals can be measured in Flex mode and additional outputs can be sent.

This value must correspond to the settings of switches S1 and S2 on the device:

- S1 = 1 and S2 = 0: Compatibility mode
- S1 = 1 and S2 = 1...F: Flex mode

**IP address**

IP address of the device

- The IP address cannot be changed in Flex mode. Please refer to the device manual for information on the structure of the automatically created IP address.
- Here, the device name or IP address is entered in compatibility mode. When automatic detection is performed, the device name of the connected device is displayed.

**Auto write configuration**

The configuration is transmitted to the device every time *ibaPDA* is started. In Flex mode, this setting is TRUE and cannot be changed. In compatibility mode, this option can be set to FALSE if no constant online connection to *ibaBM-DP* is available and the configuration should not always be transferred.

**Enable/disable automatically**

If TRUE then the acquisition starts even if a device is missing. The missing device is temporarily disabled in the configuration. During the measurement, *ibaPDA* attempts to restore the connection to the missing device. If successful, the measurement is automatically restarted including the previously missing device.

If FALSE, the measurement does not start if *ibaPDA* cannot establish a connection to the device.

**Redundancy mode**

Redundancy mode is activated here. The device then treats both Profibus strings as one redundant Profibus string. More information on the operation of *ibaBM-DP* on the redundant Profibus is provided in the device manual.

**Bus 0/1****Enable default values**

With TRUE, the default values (see below) are sent by the device in the case of a slave not supplied with data (e.g., a break in the Profibus cable or Master in STOP).

With FALSE, the last received data are repeated in this case.

**Default analog value**

If the default values are enabled (see option above), all the analog signals of a disconnected slave are set to this default analog value.

**Default digital value**

If the default values are enabled (see option above), all the digital signals of a disconnected slave are set to this default digital value.

---

**Note**

Should analog and digital access overlap and access the same addresses, the default analog value is overwritten by the default digital value at these points.

---

**Active slaves (display only)**

Numbers of the active slaves that are configured on the respective bus.

## Commands for reading/writing the configuration

### Read configuration from device / write configuration to device

With these commands it is possible to directly write a configuration for *ibaBM-DP* directly to the device or read it from the device.

Redundancy mode	raise
<b>Bus 0</b>	
Enable default values	True
Default analog value	0
Default digital value	0
Active slaves	
<b>Bus 1</b>	
Enable default values	True
Default analog value	0
Default digital value	0
Active slaves	

**Name**  
The name of the module.

[Read configuration from device](#)  
[Write configuration to device](#)

### Notes

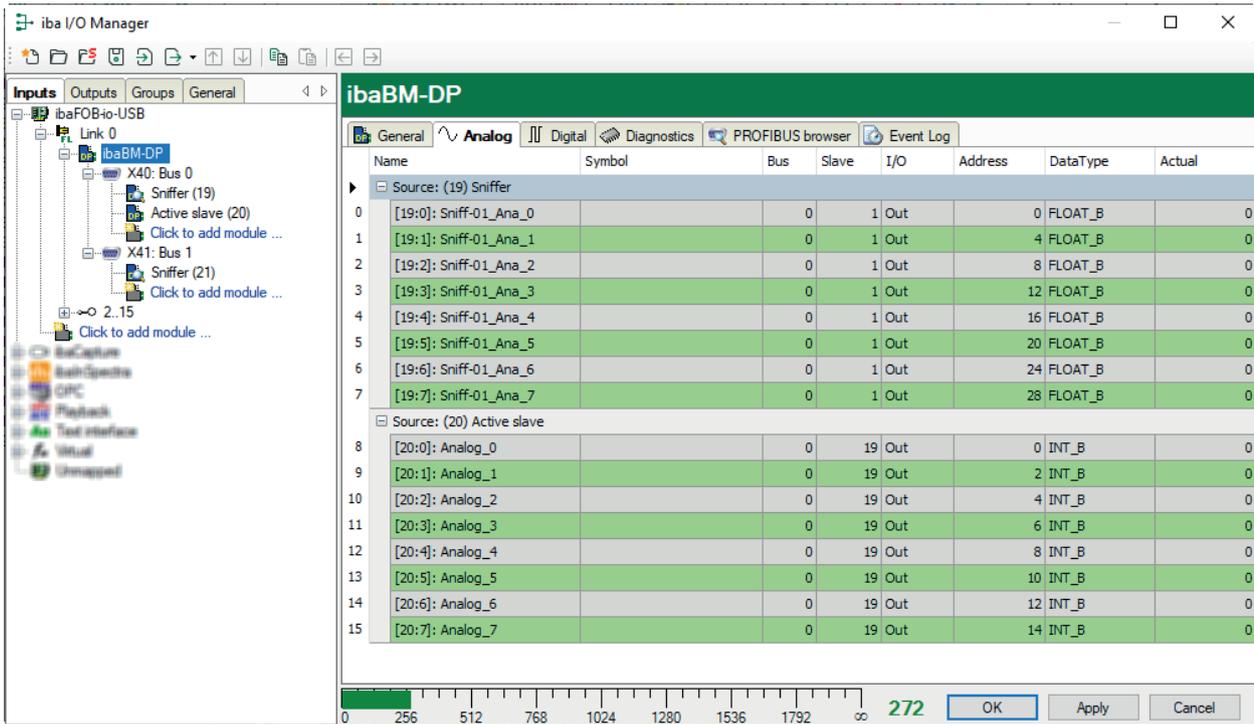


In compatibility mode, an Ethernet connection is always required for reading/writing the configuration. It is not generally possible to read the configuration in compatibility mode via this link with the *ibaBM-DP* device module.

If these commands are used, no validation of the configuration is carried out in the I/O Manager (as is the case when clicking on <OK> or <Apply>). This is why iba always recommends performing the configuration using the <OK> or <Apply> operating buttons.

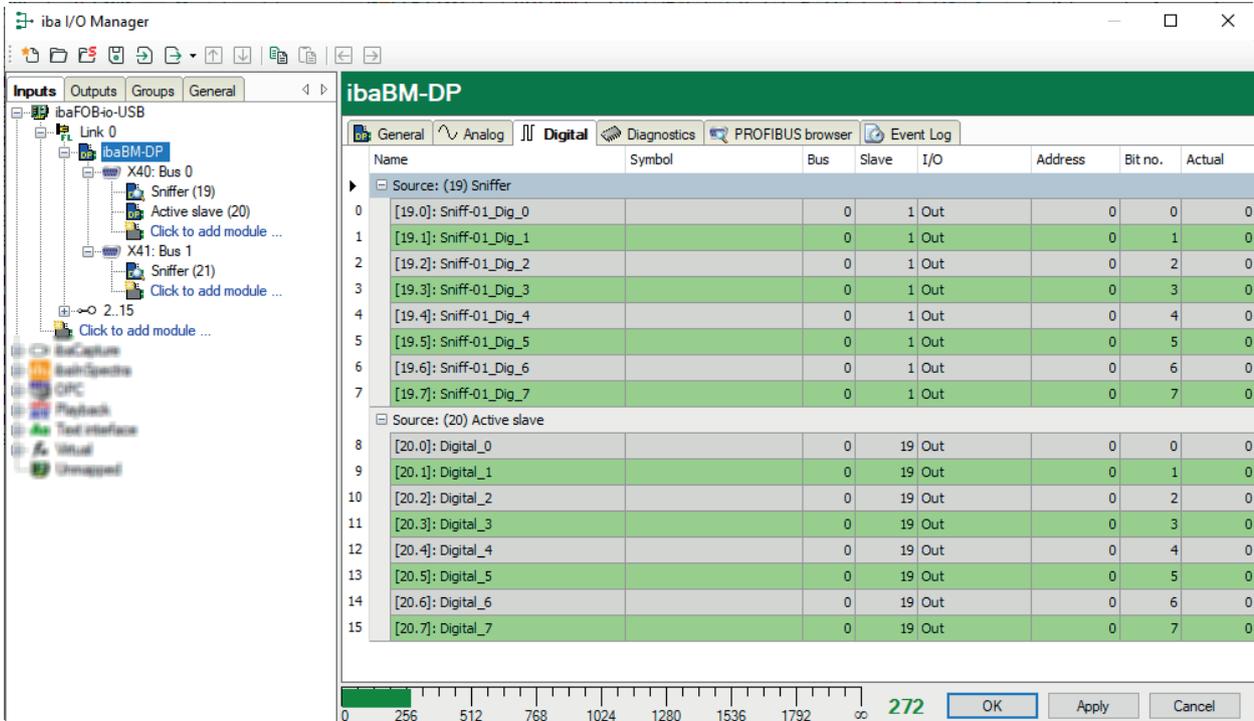
### 5.7.1.1.2 ibaBM-DP – Analog tab

If analog signals are configured in the submodules and the configuration has been transmitted to *ibaBM-DP*, an overview of all the captured analog signals is displayed here with an online presentation of the currently captured values.



### 5.7.1.1.3 ibaBM-DP – Digital tab

If digital signals are configured in the submodules and the configuration has been transmitted to *ibaBM-DP*, an overview of all the captured digital signals is displayed here with an online presentation of the currently captured values.



### 5.7.1.1.4 ibaBM-DP – Diagnostics tab

The detected masters and slaves as well as their respective status for the two PROFIBUS systems are displayed here.

### 5.7.1.1.5 ibaBM-DP – PROFIBUS browser tab

The *PROFIBUS Browser* tab belongs to the diagnostic functions and displays detailed information about both PROFIBUS systems (e. g. bus cycle time) and about the available input and output areas of the individual slaves.

### 5.7.1.2 Bus module X40: Bus 0 / X41: Bus 1

The status and diagnostics information for each connected PROFIBUS string is displayed here:

The screenshot shows the PROFIBUS browser interface for 'X40: Bus 0'. On the left, a tree view shows the device hierarchy: ibaFOB-4io-D, Link 0, ibaBM-DP, X40: Bus 0 (selected), Sniffer (19), Active slave (20), Active slave (22), Active slave (23), Active slave (24), X41: Bus 1, Sniffer (21), Active slave (25), and Link 3. The main panel displays the following information:

- Status: Running (12)
- Baudrate: 12 MBit/s
- Cycle time: 1033 µs
- Masters: 1
- Online slaves: 5
- Active slaves: 4
- Offline slaves: 2
- Phantom slaves: 2
- Collision slaves: 0
- Corrupt frames: (empty)
- Highest address on bus: (empty)

A 12x16 grid of slave status indicators is shown, with columns 0-15 and rows 16-31. The grid contains colored boxes representing the status of each slave address. A 'Reset min/max voltage windows' button is located at the bottom right.

### 5.7.1.3 Active slave submodule

The "Active slave" submodule can be added to an *ibaBM-DP* device module. You create a single slave on *ibaBM-DP* with the "Active slave" submodule. A master can send data directly to this slave for recording.

### 5.7.1.3.1 General tab

The screenshot shows the 'iba I/O Manager' window. On the left is a tree view of the device hierarchy, with 'Active slave (46)' selected under 'X40: Bus 0'. The main panel is titled 'Active slave (46)' and has three tabs: 'General' (selected), 'Analog', and 'Digital'. The 'General' tab contains the following settings:

<b>Basic</b>	
Module Type	ibaBM-DP\Active slave
Locked	False
Enabled	True
Name	<b>Active slave</b>
Module No.	<b>46</b>
Timebase	10 ms
Use name as prefix	False
<b>Module Layout</b>	
No. analog signals	<b>64</b>
No. digital signals	<b>64</b>
<b>PROFIBUS</b>	
Bus number	<b>0</b>
Slave number	<b>1</b>

Below the settings is a 'Module No.' section with a text box containing the number 531 and a 'Browse PROFIBUS' button. At the bottom of the window, there is a progress bar, a numerical display showing '531', and 'OK', 'Apply', and 'Cancel' buttons.

#### Basic settings

For a description of the basic settings see [Common and general module settings](#), page 20.

#### Advanced

##### Number of analog signals

Set the number of analog signals for this module (min. 0, max. 512).

##### Number of digital signals

Set the number of digital signals for this module (min. 0, max. 512).

#### PROFIBUS

##### Bus number

Determine on which bus system (Bus 0: X40, Bus 1: X41) the active slave should be created

##### Slave number

Set the address of the active slave of ibaBM-DP.

#### Command for browsing the PROFIBUS

##### Browse PROFIBUS

This command opens the PROFIBUS browser with which the signals from the input and output data range of slaves can be interactively added to the analog and digital signals.

**Caution!****Connection of the PROFIBUS cable**

A conflict of several slaves with the same number can cause a complete communication failure in PROFIBUS and ultimately a system downtime as well.

- Only connect the PROFIBUS cable after the "active slaves" have been properly configured so that it is ensured that there are no duplicate slave numbers.

**Note**

By adding more "Active slave" type submodules, you create more slaves on *ibaBM-DP*.

The number of active slaves is limited to a total of 8 in the basic version. Configuring more active slaves will result in an error. Please contact iba Support if you need more than 8 active slaves. You can increase the number of active slaves to 16 using an additional license.

**5.7.1.3.2 Analog tab**

Name	Unit	Gain	Offset	I/O	Address	DataType	Active
0 Analog 0		1	0	Out	0	INT_B	<input type="checkbox"/>
1 Analog 1		1	0	Out	2	INT_B	<input type="checkbox"/>
2 Analog 2		1	0	Out	4	INT_B	<input type="checkbox"/>
3 Analog 3		1	0	Out	6	INT_B	<input type="checkbox"/>
4 Analog 4		1	0	Out	8	INT_B	<input type="checkbox"/>

Enter the analog signals that are to be recorded in order.

For a description of the general signal table columns see [Columns in tables with analog and digital signals](#), page 22.

**I/O**

Select the I/O type of the signal:

- In: Input signal from the master view
- Out: Output signal from the master view
- Service: Only for service purposes in support cases

**Address**

The byte address of the signal within the input or output data range of the slave. In each case, the address range begins with the 0 address.

**Data type**

Data type of the signal. Available data types:

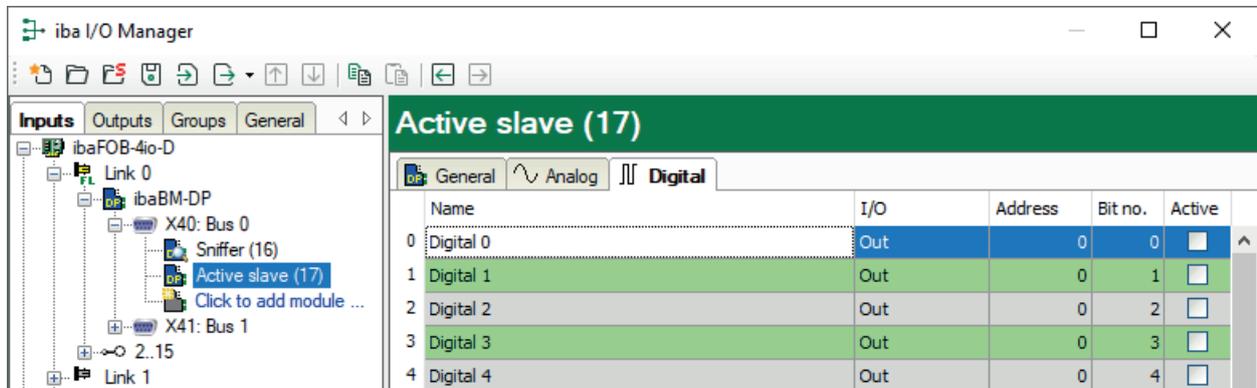
Data type		Description	Values Range
Big Endian	Little Endian		
BYTE	BYTE	8 Bit unsigned	0 ... 255
INT_B	INT	16 Bit signed	-32768 ... 32767
WORD_B	WORD	16 Bit unsigned	0 ... 65535
DINT_B	DINT	32 Bit signed	-2147483647 ... 2147483647
DWORD_B	DWORD	32 Bit unsigned	0 ... 4294967295
FLOAT_B	FLOAT	IEEE754; Single Precision; 32 bit floating point	±3.402823 · E+38...±1.175495· E-38
S5_FLOAT_B	S5_FLOAT	Simatic S5 Float Format, 32 bit	±0.1701412 E+39...±0.1469368 E-38

**Tip**



If you enter the signals of a slave consecutively, only the data types for all signals have to be set in order for the byte addresses of the signals to be calculated automatically. To do this, enter the correct byte address in the address column only for the first signal of the slave in question, and then click on the column header. Starting from the first address (where the cursor is) and taking into account the data types, the addresses of the other signals for this slave are entered automatically.

**5.7.1.3.3 Digital tab**



Enter the digital signals that are to be recorded in order.

For a description of the general signal table columns see [Columns in tables with analog and digital signals](#), page 22.

### I/O

Select the I/O type of the signal:

- In: Input signal from the master view
- Out: Output signal from the master view
- Status: Specifies the status of the slave defined with "Slave"  
(TRUE: Slave is OK, FALSE: Slave is not OK).
- Active bus: Only relevant in the redundant mode
- Service: Only for service purposes in support cases

### Address

The byte address of the signal within the input or output data range of the slave. In each case, the address range begins with the 0 address.

### Bit no.

Enter the bit number within the byte specified with "Address" here.

## 5.8 Modules for ibaLink-series devices (system interfaces)

The ibaLink-series devices are system components that can be plugged into third party systems, e. g., in VME racks. Because the ibaLink devices also have an input channel, signals to the connected target system can also be issued in conjunction with an ibaFOB-io card. The corresponding card module appears in the interface tree in the *Outputs* area of the I/O Manager.

For more information see part 2, *Outputs*.

### Other documentation



A detailed description of the modules and their configuration can be found in the corresponding device manual.

Module name	Device	ibaFOB	Comment	Manual
ibaLink-SM64	ibaLink-SM-64-io	-S, -X, -D	SIMATIC S5 and MMC 64 analog + 64 digital output data and 64 analog + 64 digital input data  Analog data types: Real, integer, S5 real	ibaLink-SM-64-io
ibaLink-SM64 generic	ibaLink-SM-64-io	-S, -X, -D	such as SM64, but signal count can be set between 0 and 64 in each case, no S5 Real	ibaLink-SM-64-io

Module name	Device	ibaFOB	Comment	Manual
ibaLink-SM64-SD16	ibaLink-SM-64-SD16	-S, -X, -D	Simadyn D 64 analog + 64 digital output data and 64 analog + 64 digital input data Data types N2, N4, NF	ibaLink-SM-64-SD16
ibaLink-SM128	ibaLink-SM-128-i-2o	-S, -X, -D	VME systems such as Simatic TDC, GE HPCi, SMS X-Pact 2 x 64 analog + 2 x 64 digital output data and 64 analog + 64 digital input data Module is no longer manufactured, substitute: ibaLink-VME	ibaLink-SM-128-i-2o
ibaLink-VME	ibaLink-VME	-S, -X, -D	VME systems such as Simatic TDC, GE HPCi, SMS X-Pact with ibaFOB-S and -X only compatibility mode such as SM128V, with ibaFOB-D also 32Mbit Flex up to 2000 analog + 2000 digital output data and 1000 analog + 1000 digital input data	ibaLink-VME
ibaLink-VME (P2P)	ibaLink-VME	-S, -D	Special operating mode for 2 cross-connected cards. ibaPDA listens to a free port. Data transfer adjustable between 50 ms and 1400 ms	ibaLink-VME

### 5.8.1 Example: ibaLink-VME in 32Mbit Flex mode

In the 32 Mbit Flex mode, the ibaLink-VME card can be connected to an ibaFOB-io channel both alone and as a participant in a ring.

In this example, a single ibaLink-VME card is connected to *ibaPDA*.

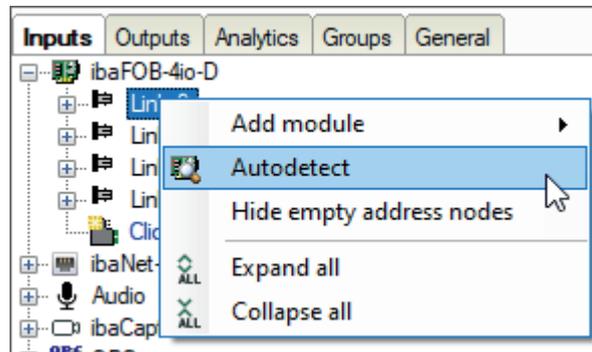
Information about the card's other applications and operating modes can be found in the card manual.

For more information see part 2, *Outputs*.

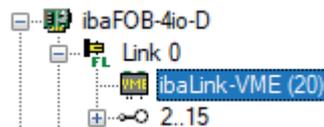
### 5.8.1.1 Add ibaLink VME module

In the 32Mbit Flex mode, the number of signals can be set flexibly in *ibaPDA*.

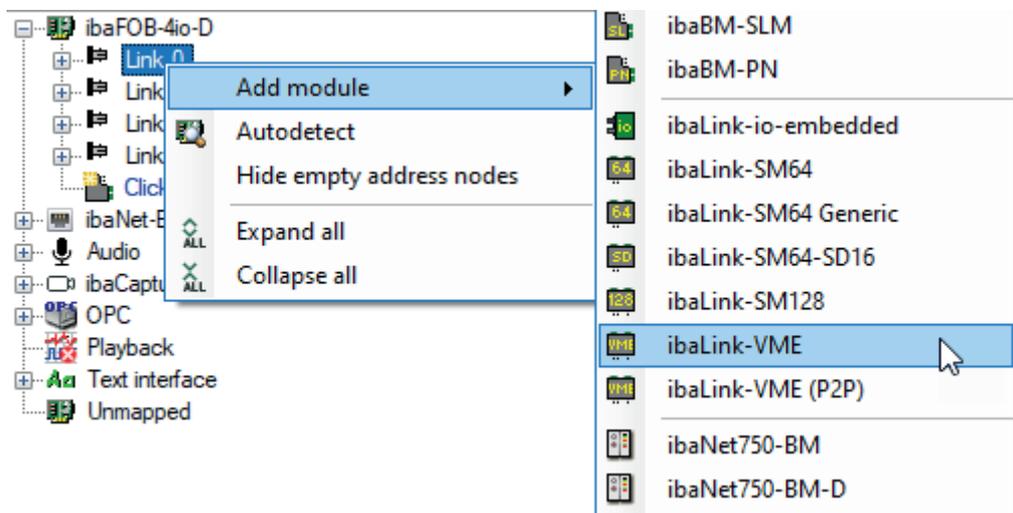
1. Start *ibaPDA* and open the I/O Manager.
2. In the signal tree (left), mark the link of the *ibaFOB-D* card to which *ibaLink-VME* is connected. Right-click the link to open a sub-menu. Select "Autodetect".



*ibaPDA* detects the component automatically and displays it in the signal tree.



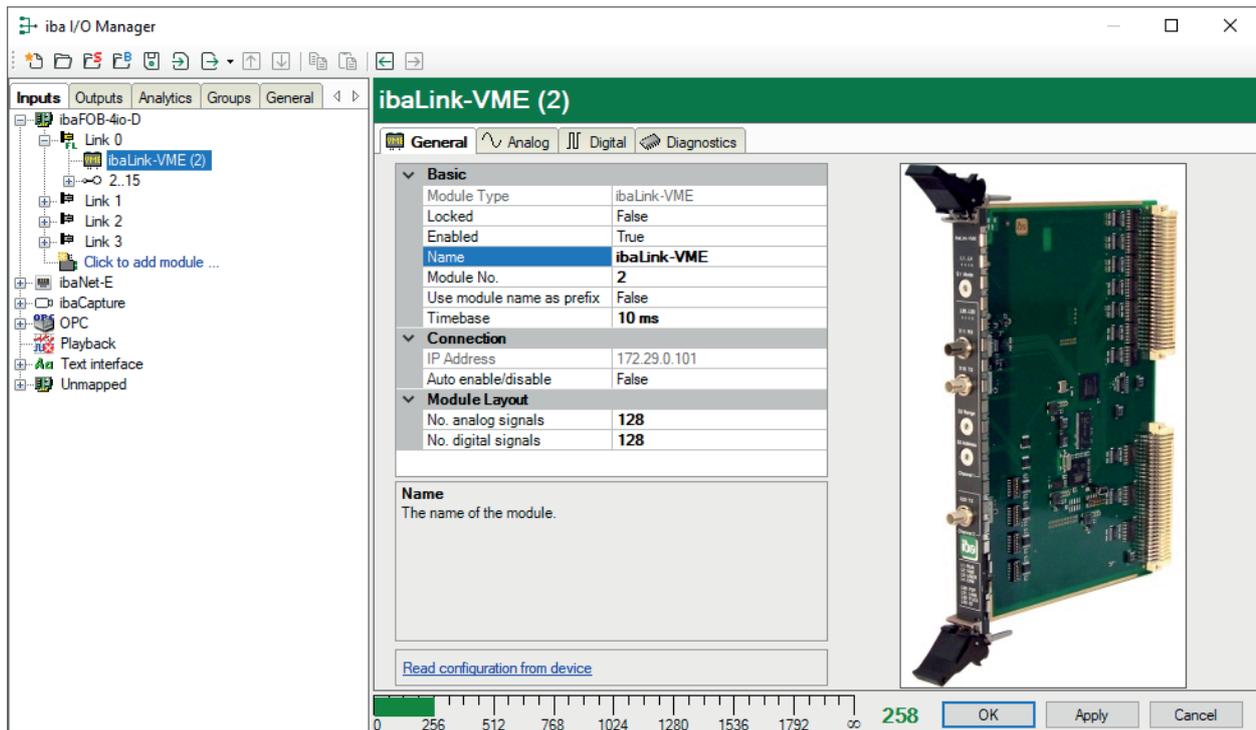
3. You can also add the component manually. Right-click on the link of the *ibaFOB-D* card to which the component should be connected and select "Add module" and "ibaLink-VME" from the list displayed.



The component is then displayed in the signal tree.

4. Hold the mouse button down and drag the component to the address (link 1 to 15 under the device), which is set with the S1 rotary switch on the device. Position 1 to F corresponds to address 1 to 15.
5. Configure the *ibaLink-VME* module in the I/O Manager.

### 5.8.1.1.1 ibaLink-VME – General tab



#### Basic settings

For a description of the basic settings see ↗ *Common and general module settings*, page 20.

#### Time base

Specifies the acquisition timebase used for *ibaLink-VME*: You can set smaller times here as the general acquisition timebase; cycles up to 25  $\mu$ s (depending on the number of signals) are possible. All signals of the module will be sampled on this time base.

#### Module structure

##### Number of analog signals

Set the number of analog signals for this module.

##### Number of digital signals

Set the number of digital signals for this module.

#### Connection

##### IP address

IP address for the ibaFlex communication of the device (not changeable)

##### Enable/disable automatically

If TRUE then the acquisition starts even if a device is missing.

#### More functions

##### Write configuration to device

Transmits the current configuration to the device.

## Read configuration from device

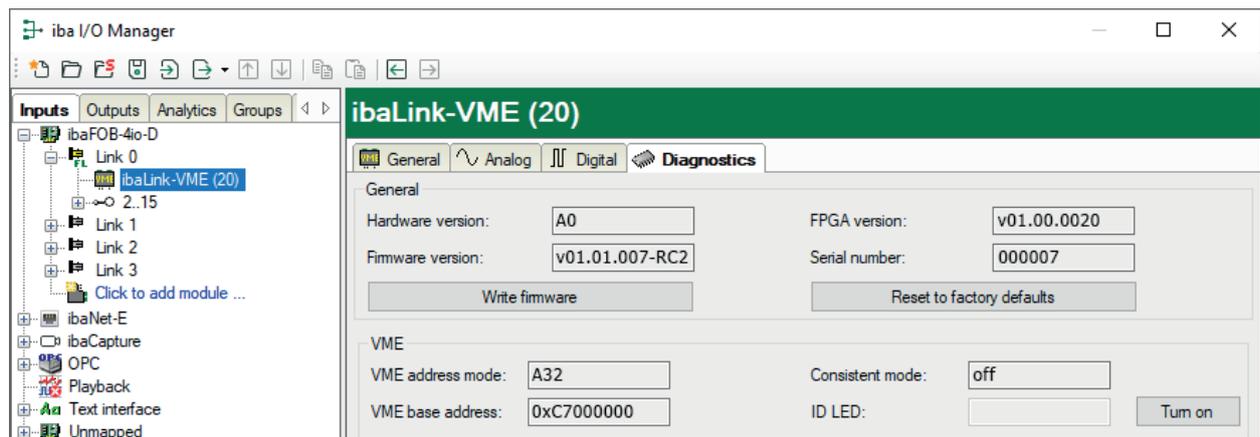
Reads the most recently stored configuration from the device.

Modified settings are applied by clicking on <OK> or <Apply>.

### 5.8.1.1.2 ibaLink-VME – Analog and Digital tab

For a description of the general signal table columns see ↗ *Columns in tables with analog and digital signals*, page 22.

### 5.8.1.1.3 ibaLink-VME – Diagnostics tab



#### "General" area

The "General" area provides information about the version and serial number of the connected *ibaLink-VME* board.

- <Write firmware>

Click here for a browser window where you can choose the new firmware. It takes several minutes for the firmware to load. After it is loaded, you will be prompted to reset *ibaLink-VME*, i.e., the rack in which the board is inserted.

- <Reset to default settings>

The configuration data is deleted.

#### VME area

The "VME" area provides information about the card's selected addressing mode, the VME base address and whether or not consistency mode is enabled. The ID LED can also be controlled.

## 5.9 ibaNet750/ibaNet750-D module type

Devices of type ibaNet750-BM and ibaNet750-BM-D are used for connecting to WAGO I/O system 750. With the help of the device developed by iba and WAGO/Beckhoff, you can access a wide range of input/output terminals from WAGO/Beckhoff. The ibaNet750 device is connected to the K bus of the WAGO system as a head module and transmits the measured data to *ibaPDA* via a fiber optic connection and the *ibaFOB* card. A variety of different input and output terminals can be connected.

The output signals are configured in the outputs tab.

For more information see part 2, *Outputs*.

### Other documentation



A detailed description of the modules and their configuration can be found in the corresponding device manual.

### Head modules

Module name	Device	ibaFOB	Comment	Manual
ibaNet750-BM	ibaNet750-BM	-S, -X, -D	Max. 32 DI+32DO+32AI+32AO 3 Mbit Acquisition rate 1 kHz Device no longer available	ibaNet750-BM
ibaNet750-BM-D	ibaNet750-BM-D	-S, -X, -D	Max. 255 WAGO terminals Max. 2048 bytes via K bus 3Mbit, 32Mbit, 32Mbit Flex Acquisition rate up to 40 kHz Automatic detection of connected terminals Can be configured as an ibaNet750-BM module (backward compatible)	ibaNet750-BM-D

### Note



If you want to operate an ibaNet750-BM-D device in 3 Mbit mode, then you have to select the ibaNet750-BM module type in the I/O Manager.

## Terminals

After an ibaNet750 module has been added to the configuration, the required terminals should be selected in the next step. You have several options for adding terminals.

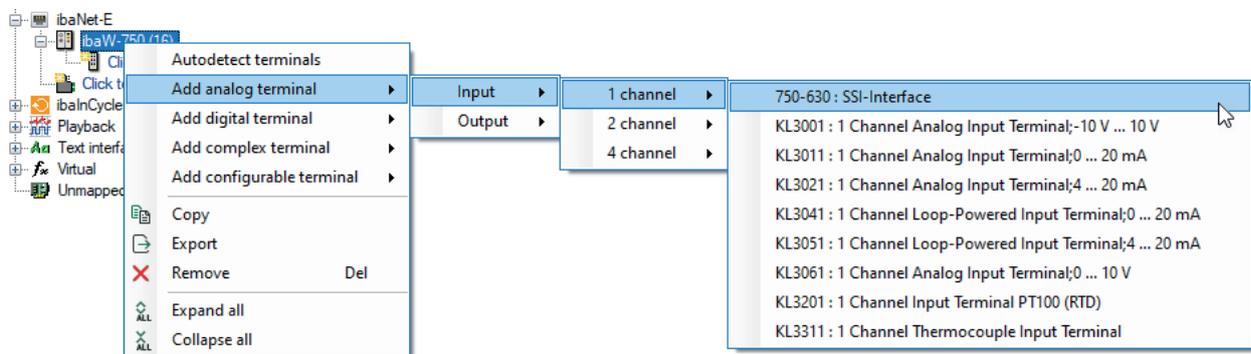
### Adding connected devices

If you have connected device to the *ibaPDA* interface, right-click on the corresponding link (connector) in the tree structure beneath the data interface. Select *Autodetect* from the context menu.

→ The system automatically detects the new device and adds the corresponding modules and signals.

### Right-click on module in the tree structure of the I/O Manager

Open the context menu via right-click on the ibaNet750 module in the tree structure. A multi-level submenu helps you selecting the available terminals.



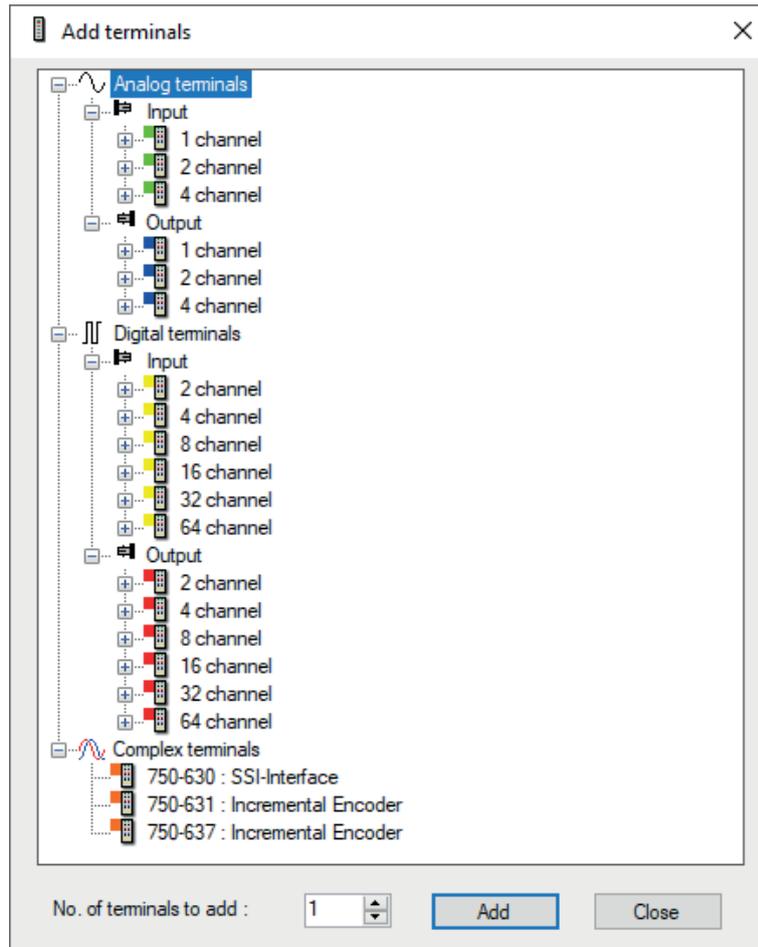
### Via the Click to add terminal... command in the tree structure of the I/O Manager

1. To add further terminals to the *ibaNet750-BM/-BM-D* module, click on the *Click to add terminal* command below the module.
2. Add the desired terminals, either by double-clicking on the terminal or by clicking the <Add> button.

If you want to add several terminals of the same type, you can first enter the number of terminals to be added and then click <Add>.

→ The corresponding signals are added to the signal lists of the module according to the terminal type.

3. Close the dialog with <Close>.



Not all of the WAGO terminal range can be used. Contact iba if in doubt. This also applies to the compatible, in principle, modules of Beckhoff.

## 5.10 FOB Fast module

This module type should be used if *ibaPDA* measurements are to be performed with an *iba-FOB-X*, or *ibaFOB-D* interface card or a device with a high data rate (32 Mbit/s).

This module type is only available for links of *ibaFOB-X* or *ibaFOB-D* cards. FobFast modules can only be connected to links that are suitable for measurement operations in high data rate mode. The high data rate mode of a link is indicated by a small X at the symbol in the signal tree. 

Generally, iba devices with a data transmission rate of 32 Mbit/s are available in the I/O Manager with their own modules. However, some older devices, like the 1st generation of *ibaPA-DU-S-IT*, can be connected only over a FOB Fast module.

Today, only external systems, such as the ABB AC 800PEC, need to be connected over a FOB Fast module.

## 5.10.1 FOB Fast – General tab

### Basic settings

For a description of the basic settings see [↗ Common and general module settings, page 20.](#)

#### Note



As for a FOB Fast module, the time base must be an integer multiple of the telegram of the time base (see below).

### Swap mode

Set the swap mode according to the signal source. You can choose between the following 4 options:

Mode	16 bit	32 bit
No swap	AB	ABCD
Depending on data type	BA	DCBA
Swap 16 bit	AB	CDAB
Swap 8 bit	BA	BADC

The swap mode to be selected depends on the swap mode of the signal source.

## FOB

### Connection mode

The connection mode is to be selected from the selection list in this field. A variety of modes or data types and volumes is available. Due to the limited capacity of the different components, there is a relation between the number of signals, data type and shortest telegram time base. In addition, the connection mode to be selected depends also on the connected system, i.e. on the data source.

With the selection of the connection mode, i.e. the data types, the signal tables of the analog signals will be adjusted automatically.

(Integer: Min / Max; Real: Gain / Offset)

### Telegram time base

The telegram time base is to be entered here. It determines how fast the data is actually being transmitted. The received samples are resampled according to this telegram timebase in order to ensure an equidistant measured values visualization required by *ibaAnalyzer*.

The telegram time base should be equal or greater than the minimum possible time base that is permissible according to the connection mode (value in brackets). Smaller values for the telegram time base are corrected automatically.

### "Advanced mode" selection box

If you activate the advanced mode, more options for data configuration will become available:

### Number of analog/digital signals

Only available in advanced mode!

Here, you can manually enter the number of the analog and digital signals to be transmitted, in the same way, the data source composes the telegrams.

The signal tables for the analog and digital signals are adjusted automatically.

Furthermore, you may determine the position in the telegram and the data type of each signal in the signal table.

### 5.10.2 FOB Fast – Analog tab

Depending on the connection mode setting and/or the activation of the advanced mode in the *General* tab, the signal tables may look different (number of rows).

#### General columns in the signal table

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals](#), page 22.

#### Only for normal mode and connection modes using integer data:

##### Min

Lower limit of measuring range

The analog voltage standard level of -10 V is assigned to a physical variable of -10 °C, for example. The value can be entered directly or set with two-point scaling.

##### Max

Upper limit of measuring range

The technological upper limit of the measuring range is to be entered here.

The analog voltage standard level of +10 V is assigned to a physical variable of 43 °C, for example. The value can be entered directly or set with two-point scaling.

#### For advanced mode or with connection modes using real data:

##### Gain and offset

The values for gain and offset describe a linear characteristic curve for scaling. If incoming values are given in physical units, gain and offset can be ignored, i.e. set gain = 1 and offset = 0. When using physical values, gain and offset can be ignored, i.e. set gain = 1 and offset = 0.

However, control applications of the connected automation systems often use normalized values for calculation; so, analog values only range, for example, between 0 and 1 or -1 and +1. In order to get a correct scale for display in terms of physical units, a scaling factor needs to be specified. This factor can be evaluated from the gain and offset parameters.

Gain and offset can be entered directly or set by means of the two-point scaling with two pairs of applicable values.

You can open the dialog for two-point scaling with a click on the small button  in the fields "Gain" or "Offset." (Cursor must be in the fields to see the button.)

**For advanced mode only:****Address**

In this additional column (together with the "Data type" column), the user can specify the address, i.e. the byte-offset of the different channels in the Fob telegram. The offset can be entered as hexadecimal or decimal values by selecting the appropriate option from the context menu. Default setting is 0x40 for analog values and 0xC0 for digital values. If you want to change the values for an entire column, you just need to change the first value and then click on the column header. Based on the address offset of the first signal, all other values below will be automatically set according to data and value type:

- Analog signals in FLOAT format: in 4 byte steps
- Analog signals in INT16 format: in 2 byte steps
- Analog signals in BYTE format: in 1 byte steps
- Digital values in 32-bit groups: increment of the bit no. by 1 up to 31. Then increment of the address by 4.

For the digital signals, you may use a DINT to transmit 32 single bits. You can unpack these by specifying the address and bit no. Except for the first one, it is not necessary to enter a new address for the bits in a DINT.

**Data type**

In each field of this column, you can select the data type in use.

Click in a table cell and open the drop-down list.

Available for selection are:

Data type		Description	Values Range
Big Endian	Little Endian		
BYTE	BYTE	8 bit without positive or negative sign	0 ... 255
INT_B	INT	16 bit with positive or negative sign	-32768 ... 32767
WORD_B	WORD	16 bit without positive or negative sign	0 ... 65535
DINT_B	DINT	32 bit with positive or negative sign	-2147483647 ... 2147483647
DWORD_B	DWORD	32 bit without positive or negative sign	0 ... 4294967295
FLOAT_B	FLOAT	IEEE754; single precision; 32 bit floating point value	±3.402823·E+38 ... ±1.175495·E-38

As the set memory addresses change according to the data type, it might be necessary to adjust the addresses afterwards.

### 5.10.3 FOB Fast – Digital tab

#### General columns in the signal table

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals](#), page 22.

#### For advanced mode only:

##### Address

In this additional column (together with the "Data type" column), the user can specify the address, i.e. the byte-offset of the different channels in the Fob telegram. The offset can be entered as hexadecimal or decimal values by selecting the appropriate option from the context menu. Default setting is 0x40 for analog values and 0xC0 for digital values. If you want to change the values for an entire column, you just need to change the first value and then click on the column header. Based on the address offset of the first signal, all other values below will be automatically set according to data and value type:

##### **Digital values in 32-bit groups: increment of the bit no. by 1 up to 31. Then increment of the address by 4.**

For the digital signals, you may use a DINT to transmit 32 single bits. You can unpack them by specifying the address and bit no. Except for the first one, it is not necessary to enter a new address for the bits in a DINT.

##### Bit no.

The values in this column, range 0...31 (with 4 byte-packages), specifies the position of the digital signal in the Fob telegram, relative to the offset address.

## 6 Ethernet-based interfaces

Ethernet-based interfaces have the advantage that they do not require any special hardware, with very few exceptions. The interfaces simply operate via the computer's network interface.

You need the relevant license for each interface (except OPC) and activation in the dongle. For most interfaces, you can set up and use up to 64 connections per license. The number of connections can be increased via extension licenses.

Name	Type	Connection to ...	Comment	Link
AN-X-DCSNet	NIC	RELIANCE DCS Network	via AN-X-DCSNet / Ethernet Gateway	<a href="#">↗ AN-X-DCS-Net, page 170</a>
EGD	NIC	EGD network		<a href="#">↗ EGD (Ethernet Global Data), page 171</a>
EtherNet/IP	NIC	Allen-Bradley, Ethernet-based	MicroLogix, ControlLogix, etc.	<a href="#">↗ EtherNet/IP, page 172</a>
GCOM	NIC	GCOM multidrop bus	special modules for flatness available	<a href="#">↗ GCOM, page 174</a>
Generic TCP	NIC	TCP/IP network		<a href="#">↗ Generic TCP, page 175</a>
Generic UDP	NIC	UDP/IP network	optionally with HiPAC Request or TwinCAT Request	<a href="#">↗ Generic UDP, page 176</a>
ibaLogic TCP	NIC	ibaLogic		<a href="#">↗ ibaLogic TCP, page 178</a>
ibaNet-E	NIC	ibaW-750	Also 3rd party devices that support the ibaNNet-E protocol.	<a href="#">↗ ibaNNet-E, page 135</a>
IEC 61850 Client	NIC	Compatible devices for protection and control technology	Station automation with MMS or GOOSE	<a href="#">↗ IEC 61850 Client, page 179</a>
IEC 61850-9-2	NIC	Data sources with sampled values streams in accordance with IEC61850-9-2		<a href="#">↗ IEC 61850-9-2, page 180</a>
LANDSCAN	NIC	Temperature scanner from LAND (Ametek)		<a href="#">↗ LANDSCAN, page 181</a>
LMI Gocator	NIC	Laser scanners of the Gocator family	2D/3D profile acquisition	<a href="#">↗ LMI-Gocator, page 182</a>
Micro-Epsilon	NIC	Laser scanners of the scanCONTROL family	2D/3D profile acquisition	<a href="#">↗ Micro-Epsilon, page 183</a>

Name	Type	Connection to ...	Comment	Link
Modbus TCP Client	NIC	MODBUS over TCP, Modicon Quantum	ibaPDA is MODBUS client (master)	<a href="#">↗ Modbus TCP client, page 185</a>
Modbus TCP Server	NIC	MODBUS over TCP, Modicon Quantum	ibaPDA is MODBUS server (slave)	<a href="#">↗ Modbus TCP Server, page 186</a>
MQTT	NIC	MQTT broker	subscription only	<a href="#">↗ MQTT interface, page 273</a>
OPC	NIC	OPC server (also redundant), OPC client	License-free standard interface OPC DA	<a href="#">↗ OPC interface, page 34</a>
OPC UA	NIC	OPC UA server		<a href="#">↗ OPC UA, page 184</a>
Raw Ethernet		Raw Ethernet Multicast		<a href="#">↗ Raw Ethernet, page 188</a>
Raytek	NIC	Temperature scanner from Raytek (Fluke Process Instruments)		<a href="#">↗ Raytek, page 189</a>
S7 TCP/UDP	NIC	SIMATIC S7		<a href="#">↗ S7 TCP/UDP, page 189</a>
Sisteam TCP	NIC		Also SIMATIC TDC over CP5100	<a href="#">↗ Sisteam TCP, page 191</a>
TDC TCP/UDP	NIC	SIMATIC TDC	via CP51M1	<a href="#">↗ TDC TCP/UDP, page 192</a>
VIP TCP/UDP	NIC	ABB AC450 RMC, AC 800 M, AC 800 PEC	and Simatic S7	<a href="#">↗ VIP TCP/UDP, page 194</a>

NIC = Network Interface Card

## 6.1 General and common settings

All interfaces of this group basically use the same technical bases and procedures. The default network connector of the *ibaPDA* computer or a supplementary network adapter is used for establishing the connections.

Due to the variety of the ISO/OSI layer model and/or the TCP/IP model, many different protocols can be realized when it comes to special communication requirements and/or automation systems. Some protocols were customized to meet the specific requirements of manufacturers of automation systems, others comply with international standards.

Some features are available for several Ethernet-based interfaces, while others are unique, such as port numbers, swap modes etc. All features can be found in the corresponding configuration dialog.

Common characteristics:

- All Ethernet-based interfaces are activated with the software (dongle license).
- Up to 64 connections can be established per interface and base license, except AN-X-DCSNet (60) and Raw Ethernet, OPC and PLC Xplorer interfaces.  
For some interfaces the number of connections can be increased up to 256 connections, when additional licenses would be purchased.
- Every connection refers to a module.
- For each connection of an interface, the connection status is displayed in a table.
- Active connections are detected automatically by the system.

## 6.2 Connection table

All Ethernet-based interfaces offer a table in the I/O Manager that shows the status of each connection. Each line represents one connection.

The columns contain different values and information depending on the type of interface. The possible options and buttons above the table are also interface-specific.

Typically, the target systems connected are identified in the first column (left) by its name or its IP address.

The table shows the cycle times and error counters of the individual connections during data acquisition. Click the <Reset counter> button to reset the error counter 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.
Orange	The connection is OK, however the data update is slower than the configured update time.
Red	The connection has failed or been interrupted.
Gray	No connection configured.

The figure below shows the connection table of the Codesys-Xplorer interface as an example.

The screenshot shows the Iba I/O Manager window with the Codesys-Xplorer interface. The connection table is as follows:

	Name	Error count	Update time Actual	Response time Actual	Response time Average	Response time Min	Response time Max
0	Codesys V2...	0	1,0 ms	0,0 ms	0,0 ms	0,0 ms	14,0 ms
1	Codesys V3...	2	1,4 ms	0,0 ms	0,5 ms	0,0 ms	145,0 ms
2	?	?	?	?	?	?	?

## 6.3 AN-X-DCSNet

### Description

iba has implemented an AN-X-DCSNet interface to perform measurements in a RELIANCE DCS network.

AN-X-DCSNet is the name of a device, manufactured by Quest Technical Solutions. This device establishes a connection to the DCS network via a passive tap. It can be configured as a master or as a slave on the DCS network. It monitors the input and output data of all participants (drops) in the network. *ibaPDA* establishes a connection to an AN-X-DCSNet device via Ethernet and configures it so that it cyclically sends the requested drop data to *ibaPDA*. *ibaPDA* supports up to 4 devices. Each device can host up to 10 connections.

### Interface configuration

#### Port number

This is the UDP port number that *ibaPDA* listens to for messages from an AN-X-DCSNet device. Usually, the default port number 47920 can remain unchanged.

#### Allow ports through firewall

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

#### <Reset port to default>

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

#### Network interfaces

Using this drop-down list, you can select which network adapters on your computer should be 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 of the network adapters, an error message will be produced when validating the I/O configuration. By default the option *All* is selected.

### Connection table

#### Source IP address

This is the IP address of an AN-X-DCSNet device.

#### Destination IP address,

This is the Multicast IP address of the target controller.

#### Configuration ID

This custom 32-bit configuration number determines the data structure.

## 6.4 EGD (Ethernet Global Data)

### Description

EGD is a protocol for exchanging PLC and computer data, developed by GE Fanuc Automation and GE Drive Systems. It is used for interfacing with controllers, e. g., GE Fanuc 9030/9070, GE Energy Power Conversion HPCi or Convertteam Alspa 8035. EGD uses UDP or datagram messages for fast data transfer of up to 1400 bytes of data from a producer to one or more consumers. EGD supports a set of commands for accessing data and protocol information over EGD nodes. EGD protocol messages are categorized as command, data, or configuration messages. The following *ibaPDA*-specific restrictions apply:

- Only EGD data protocols are supported
- *ibaPDA* supports both groups addressing (IP multicast) as well as individual destination addresses (IP unicast).
- *ibaPDA* acts solely as a consumer.
- Up to 64 connections can be used per interface license. By purchasing up to 3 one-step-up licenses, the number of connections can be increased up to 256.

### Interface configuration

#### EGD port number

This is the port number that *ibaPDA* listens to for messages from EGD producers. Usually, the default port no. 18246 can remain unchanged.

#### Allow ports through firewall

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

#### <Reset port to default>

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

#### Network interfaces

Using this drop-down list, you can select which network adapters on your computer should be 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 of the network adapters, an error message will be produced when validating the I/O configuration. By default the option *All* is selected.

#### Connection table

##### IP address

This is the IP address of the connected controller, i. e., of the producing node.

##### Producer ID

The producer ID should match the IP address

**Exchange ID**

The exchange ID should correspond to the specified exchange ID on the producing node.

**Available modules**

- EGD
- EGD Multicast

**Product name**

ibaPDA-Interface-EGD (Art. no. 31.001070)

---

**Other documentation**

A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-EGD*.

---

## 6.5 EtherNet/IP

**Description**

Ethernet Industrial Protocol (EtherNet/IP) is an open industrial networking standard introduced by Rockwell Automation that supports both real-time I/O transfer and message exchange. EtherNet/IP uses TCP/IP for general messaging / information exchange services and UDP/IP for I/O messaging services for control applications.

The EtherNet/IP interface is a driver which enables *ibaPDA* to read data from Rockwell controllers over TCP/IP. *ibaPDA* acts as a server and awaits connections from the client. The controller (Rockwell PLC) acts as a client and requests connections to the *ibaPDA* driver.

The EtherNet/IP interface can also be used for output signals. The interface is also available in the outputs area of the I/O configuration.

EtherNet/IP can be connected to the following controllers:

- Rockwell/Allen-Bradley
  - CompactLogix
  - FlexLogix
  - ControlLogix
  - SoftLogix 5800
- Schneider Electric
  - M580 ePAC

Up to 64 connections can be used per interface license. By purchasing up to 3 one-step-up licenses, the number of connections can be increased up to 256.

## Interface configuration

### Network interfaces

Using this drop-down list, you can select which network adapters on your computer should be 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 of the network adapters, an error message will be produced when validating the I/O configuration. By default the option *All* is selected.

### Base multicast address

This setting only applies to multicast communication. The base multicast address is used as the destination address to which *ibaPDA* will respond. The last part of the address is replaced by the assembly instance number automatically. In general, you can keep the default address given by *ibaPDA* unless it is already being used or changes are required due to router or firewall settings.

### Multicast TTL

The TTL (Time to Live) parameter is set to 1 by default. Each router between the PLC and *ibaPDA* gradually reduces the TTL value by 1 when a multicast package arrives. A router discards a package when the TTL value reaches 0 (zero). You only need to set a value greater than 1 when the PLC is behind one or more routers.

In addition to Multicast, *ibaPDA* also supports Unicast connections. A Unicast connection is automatically detected by *ibaPDA* when arranging the connection with the PLC.

### Allow ports through firewall

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

## Connection table

### IP address

This is the IP address of the connected Rockwell controller.

### Available modules

#### EtherNet/IP I/O module

Can be used in the input and output direction.

### Product name

ibaPDA-Interface-Ethernet-IP (Art. no. 31.001005)

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product, *ibaPDA-Interface-EthernetIP*.

---

## 6.6 GCOM

### Description

On the ABB Master side, the GCOM communication software is implemented on a microprocessor-based communication card (DSCS 150 in MG 230/1 and MP 200/1, SC530 in AC 450). The GCOM subnet (bus) can serve up to 4 external computers on the same bus with a transmission rate of up to 10 Mbit/s. Tests with MP200-based Stressometer system (hardware release 3.0) have been approved. The GCOM interface supports up to 4 links. Several modules can be configured per link.

### Interface configuration

#### Network interface

With this option you decide which NIC (network interface card) the GCOM connection should run on. The drop-down list contains all of the registered network interfaces.

#### Mode

With this option, you decide whether a connection should run in active or passive mode:

- *Active mode*: The *ibaPDA* system functions as an active GCOM network node, and sends ACK messages as well as "I am here" messages on the network in cyclical intervals.
- *Passive mode*: The *ibaPDA* system behaves completely passively on the network. It collects data in this mode, but does not send any messages in the network. This mode can be used in order to be able to use *ibaPDA* in parallel to an existing flatness logger.

#### Network ID

Enter the ABB master subnet that is used for the communication with the ABB master system.

#### ABB master node ID

Enter the ABB master node that is used in your network.

#### Logger node ID

Specify the node ID of the *ibaPDA* system in the ABB master subnet.

#### Logger node MAC

Network address to which the data messages are sent from the ABB master. Please note that this address may differ from the current MAC address of the NIC that is connected to the ABB network. The *ibaPDA* driver only evaluates messages that are intended for this MAC address.

#### Message grouping

Define the messages to be synchronized here. Use the following characters for the definition:

- ";" to separate two groups
- "," to separate two message IDs within a group

Define a range of message IDs in a group:

A maximum of 256 groups can be defined, where each group can contain 32 messages.

Default setting: 80-83;85-87;89;90

The link level in the tree structure provides diagnostic information about the network adapter, GCOM configuration and message counters.

### Available modules

- GCOM generic

### Product name

ibaPDA-Interface-GCOM (Art. no. 31.001080)

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-GCOM*.

---

## 6.7 Generic TCP

### Description

The Generic TCP interface can be used by every controller able to send TCP/IP messages. Transmission Control Protocol (TCP) is one of the core protocols of the Internet protocol suite. The Generic TCP messages are IP Unicast protocols sent from one or several controllers to the *ibaPDA* system using a defined port number.

Each TCP connection is precisely identified within *ibaPDA* by the destination "port number" and "source IP address." The telegram between *ibaPDA* and the target system must have a fixed structure. The maximum length of the TCP/IP message is limited to 4096 bytes. Up to 64 connections can be used per interface license. By purchasing up to 3 one-step-up licenses, the number of connections can be increased up to 256.

The Generic *ibaPDA* interface can also be used for output signals. The interface is also available in the outputs area of the I/O configuration.

For more information see part 2, *Outputs*.

Text signals are supported among the analog signals with data type STRING[32].

### Interface configuration

#### TCP port list

Setting the destination port range. As shown in the example, the range is defined from 5010 to 5017. The *ibaPDA* driver will therefore monitor port 5010 to port 5017. One controller can send multiple messages to *ibaPDA* using multiple ports.

#### <Allow ports through firewall>

When installing *ibaPDA*, the default port numbers of the protocols used are automatically entered in the firewall. If the port number is changed here or the interface has subsequently been activated, it is necessary to release this port in the firewall.

**Network interfaces**

Using this drop-down list, you can select which network adapters on your computer should be 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 of the network adapters, an error message will be produced when validating the I/O configuration. By default the option *All* is selected.

**Connection table****IP address source**

This is the source IP address of a connected controller.

**Destination port**

The destination port number shows which port is used by the controller for sending data to *ibaPDA*.

**Available modules**

- Generic TCP
- Generic TCP Output

**Product name**

ibaPDA-Interface-Generic-TCP (Art. no. 31.001076)

---

**Other documentation**

A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-Generic-TCP*.

---

## 6.8 Generic UDP

**Description**

The Generic UDP interface can be used by every controller able to send UDP/IP messages. User Datagram Protocol (UDP) is one of the core protocols of the Internet protocol suite. The Generic UDP messages are IP Unicast messages sent from one or several controllers to the *ibaPDA* system using a defined port number. *ibaPDA* also supports UDP/IP multicast messages. Each UDP connection is precisely identified in *ibaPDA* by the destination "Port" and source of the "Address".

Up to 64 connections can be used per interface license. By purchasing up to 3 one-step-up licenses, the number of connections can be increased up to 256.

Text signals are supported among the analog signals with data type STRING[32].

## Interface configuration

### UDP port list

Setup the destination "port" range. As shown in the example, the range is defined from 5010 to 5017. The *ibaPDA* driver will therefore monitor port 5010 to port 5017. One controller can send multiple messages to *ibaPDA* by using multiple ports. You may enter multiple ranges that must be separated by comma.

### Allow ports through firewall

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

### Network interfaces

Using this drop-down list, you can select which network adapters on your computer should be 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 of the network adapters, an error message will be produced when validating the I/O configuration. By default the option *All* is selected.

## Connection table

### Address

This is the source IP address of a connected controller.

### Port

The destination port number shows which port is used by the controller for sending data to *ibaPDA*.

The connections in the table are sorted by address and port. Double-click on a row to navigate to the corresponding address offset in the memory view.

## Available modules

- Generic multicast UDP
- Generic unicast UDP
- HiPAC Request (only if license available)
- TwinCAT Request (only if license available)

## Product name

ibaPDA-Interface-Generic-UDP (Art. no. 31.001075)

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-Generic-UDP*.

The modules HiPAC Request and TwinCAT Request are described in the documentation of the corresponding products *ibaPDA-Request-HiPAC* and *ibaPDA-Request-TwinCAT*.

---

## 6.9 ibaLogic TCP

### Description

ibaLogic TCP is a special connection between *ibaLogic* and *ibaPDA*. Data can be transmitted directly from the *ibaLogic* application program to *ibaPDA* for the measurement.

This interface is only available if licensed in the dongle. Up to 64 connections or modules respectively are supported.

### Interface configuration

#### Port number

Default setting: 40000 This port number can be changed but must be the same in both *ibaLogic* and *ibaPDA* systems in order to establish a connection.

#### <Reset port to default>

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

#### Allow ports through firewall

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

#### Network interfaces

Using this drop-down list, you can select which network adapters on your computer should be 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 of the network adapters, an error message will be produced when validating the I/O configuration. By default the option *All* is selected.

### Connection table

#### Address

This is the IP address of the source computer from which *ibaPDA* receives data.

#### Module index

The module index indicates which module number in *ibaPDA* should be assigned to this connection. The module index must be configured in the source system, e.g. ABB, *ibaLogic*. This module index must correspond to the entry in the field, Module index, in the corresponding module

of the I/O Manager (module level, General tab). If necessary, adjust the module index on the source page or in *ibaPDA*.

### Available modules

- ibaLogic

### Product name

ibaPDA-Interface-ibaLogic-TCP (Art. no. 31.001015)

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product, *ibaLogic* (version 3.xx), *ibaLogic-V4* or *ibaLogic-V5*.

---

## 6.10 IEC 61850 Client

### Description

The *ibaPDA-Interface-IEC61850-Client* data interface is suitable for the measurement data acquisition of an IEC 61850-compliant server via standard network cards.

The following services are supported:

- Manufacturing Messaging Specification (MMS)
- Generic Object Oriented Substation Events (GOOSE)
- IEC 61850-7-2 COMTRADE File Support

You can conveniently make a selection using the symbolic names supported by the IEC 61850 symbol browser. This enables access to all measurable symbols based on the imported server object list of the IEC 61850 device. *ibaPDA* can generate addressbooks for the IEDs offline out of the SCL files of the servers.

Up to 64 connections can be used per interface license. By purchasing further one-step-up licenses, the number of connections can be increased as needed.

### Interface configuration

#### Set all values to zero when the connection to a device is lost

If this option is enabled, all the measured values of a device are set to zero as soon as the connection is lost. If this option is disabled, *ibaPDA* will keep last valid measured value in the memory at the time the connection was lost.

#### Start acquisition even if a device is not accessible

If this option is enabled, the acquisition will start even if a device is not accessible. Instead of an error, a warning is indicated in the validation dialog. If the system has been started without a connection to the device, *ibaPDA* will try to connect to the device at regular intervals.

### Accept unaccessible attributes

Enable this option to start the acquisition, even if no attributes are accessible. The inaccessible symbols are issued as warnings in the validation dialog. This can only occur if the address book is not up to date!

If you do not enable this option then the measurement does not start if there are inaccessible symbols.

### Connection table

For information about the connection table, see [➤ Connection table, page 169](#).

The update time is only displayed for modules that automatically request data from the IEC server with the set update time. This only applies for MMS modules without report control blocks.

The response time for MMS modules without Report Control Block describes the time that passes between the data request from *ibaPDA* and the response being received.

For all other module types, the IEC server automatically sends data according to different criteria. In this case, the response time describes the time span between two received telegrams.

### Available modules

- IEC 61850 Device
  - GOOSE module
  - MMS module
  - File module

### Product name

ibaPDA-Interface-IEC61850-Client (Art. no. 31.001090)

---

#### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product, *ibaPDA-Interface-IEC61850-Client*.

---

## 6.11 IEC 61850-9-2

### Description

The standard IEC 61850 of the International Electrotechnical Commission (IEC) describes a general transmission protocol for protection and control technology in electrical switchgear of medium and high-voltage technology (station automation).

Part 9-2 describes the so-called sampled values. These are currents and voltages measured in real time and sent via Ethernet frames. The acquisition of up to two streams is supported with one interface license. In total, a maximum of 4 licenses (=8 streams) can be used.

Sampled Values streams are sent by IEC 61850 devices as multicast ISO messages. *ibaPDA* can receive them via the computer's standard Ethernet interfaces.

## Interface configuration

At the interface level, no settings need to be configured.

## Connection table

The table shows different diagnostic values of the individual connections during data acquisition. To reset the update time, the message counter and the sequence errors to zero, click on the <Reset counters> button. The message counter is a continuous counter that is incremented by one with each received message. A message can contain several samples. The update time shows the measured time between samples. Sequence errors indicate that no continuous sequence counter was detected in successive messages that were received. The diagnostic data from this connection table can also be acquired via a diagnostic module. Each diagnostic module can be coupled to a Sampled Values stream via the *Target module* property.

## Available modules

- IEC61850-9-2 module

## Product name

ibaPDA-Interface-IEC61850-9-2 (Art. no. 31.001400)

## Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product, *ibaPDA-Interface-IEC61850-9-2*.

## 6.12 LANDSCAN

### Description

The LANDSCAN interface can be used to measure data from LAND (Ametek) temperature line scanners. Up to 2 devices or connections are supported with an interface license. In total, a maximum of 8 licenses (=16 devices) can be used. The scanners generate 1000 samples per line and they can scan at up to 150 lines per second. The scanners can send their data in ASCII or binary mode. Both modes are supported by *ibaPDA*, while the binary mode is more efficient and is therefore recommended if the scanner supports it. Older versions of the scanners only support ASCII mode.

### Interface configuration

The interface itself has the following functions and configuration possibilities:

Name	Address	Error Count	Mode	Update time Actual	Update time Average	Update time Min	Update time Max
0	?	?	?	?	?	?	?
1	?	?	?	?	?	?	?

**Set all values to zero when the connection to a device is lost**

If enabled, all measured values of the LANDSCAN-device are set to zero as soon as the connection is lost. If this option is disabled, *ibaPDA* keeps the last valid measured value in the memory at the time the connection was lost.

**Start acquisition even if a device is not accessible**

If this option is enabled, the acquisition will start even if the LANDSCAN device is not accessible. Instead of an error message, an alert is prompted in the validation dialog. If the system had been started without a connection to the device, *ibaPDA* periodically tries to connect to the device.

**Connection table**

The table shows the cycle times and error counters of the individual connections during data measurement. To reset the calculated times and error counters to zero, simply click on the <Reset counters> button.

**Available modules**

- LSP (BINARY)
- LSP (ASCII)

**Product name**

ibaPDA-Interface-LANDSCAN (Art. no. 31.001011)

---

**Other documentation**

A detailed description of this interface and its configuration can be found in the corresponding manual for the product, *ibaPDA-Interface-LANDSCAN*.

---

## 6.13 LMI-Gocator

**Description**

The LMI-Gocator interface is suitable for measurement data acquisition from Gocator<sup>®</sup> sensors (LMI Technologies Inc.). It allows data from multiple, adjacent sensors to be collected and combined into one profile. Beside the module for data acquisition, the interface offers a status module which delivers a bunch of status signals of the measuring device.

The connections to the sensors and *ibaPDA* can be established via standard Ethernet interfaces of the computer. No further software is necessary for operation.

**Interface configuration****Set all values to zero when the connection to a device is lost**

If enabled, all measured values of the sensor are set to zero as soon as the connection is lost. If this option is disabled, *ibaPDA* will keep the last valid measured value at the time the connection was lost in the memory.

**Start acquisition even if a device is not accessible**

If this option is enabled, the acquisition will start even if the sensor is not accessible. Instead of an error, an alert is indicated in the validation dialog. If the system has been started without a connection to the sensor, *ibaPDA* will periodically try to connect to the sensor.

**Available modules**

- LMI-Gocator
- LMI-Gocator Status

**Product name**

ibaPDA-Interface-LMI-Gocator (Art. no. 31.001012)

---

**Other documentation**

A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-LMI-Gocator*.

---

## 6.14 Micro-Epsilon

**Description**

The Micro-Epsilon interface is suitable for measurement data acquisition from laser profile scanners of the Micro-Epsilon scanCONTROL family. Up to 2 devices or connections are supported with an interface license. In total, a maximum of 8 licenses (=16 devices) can be used.

The connections to the devices can be established via the computer's standard Ethernet interfaces. No further software is necessary for operation.

**Interface configuration****Set all values to zero when the connection to a device is lost**

If enabled, all measured values of the scanCONTROL device are set to zero as soon as the connection is lost. If this option is disabled, *ibaPDA* will keep the last valid measured value at the time the connection was lost in the memory.

**Start acquisition even if a device is not accessible**

If this option is enabled, the acquisition will start even if the scanCONTROL device is not accessible. Instead of an error, an alert is indicated in the validation dialog. If the system has been started without a connection to the device, *ibaPDA* will try to connect to the device at regular intervals.

**Available modules**

- scanCONTROL

**Product name**

ibaPDA-Interface-Micro-Epsilon (Art. no. 31.001016)

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-Micro-Epsilon*.

---

## 6.15 OPC UA

### Description

Both OPC UA client and OPC UA server modules can be configured on the OPC UA interface.

In order to use OPC UA client modules, you need an *ibaPDA-Interface-OPC-UA-Client* license.

To use OPC UA server modules, you need an *ibaPDA-OPC-UA-Server+* license.

With an OPC UA client module, *ibaPDA* connects to an OPC UA server to receive data. *ibaPDA* is not cyclically polling for new data. Instead, *ibaPDA* will be notified by the OPC UA server whenever at least one of the values to be measured has changed. *ibaPDA* can only read and not write the variables provided by the OPC UA server.

An OPC UA server module can be used to write the variables provided by an OPC UA server.

With an OPC UA client interface license, up to 16 connections can be configured per license. A total of a maximum of 256 connections can be implemented by the additional purchase of up to 15 further one-step-up OPC UA client licenses.

You can import or generate the certificates required for the communication between OPC UA client (*ibaPDA*) and an OPC UA server in *ibaPDA*.

The signals to be measured can be conveniently selected using their symbolic names with support from the OPC UA Symbol Browser. This allows access to all measurable symbols, which are defined in the OPC UA server.

The acquisition of current values is supported.

### Interface configuration

The interface itself has the following features and configuration options:

#### **Set all values to zero when the connection to an OPC UA server is broken.**

If this option is enabled, all the measured values of an OPC UA server affected are set to zero as soon as the connection is lost. If this option is disabled, *ibaPDA* keeps the last valid measured value in the memory at the time the connection was lost in the memory.

#### **Start acquisition even if an OPC UA server is not accessible.**

If this option is enabled, the acquisition will start even if an OPC UA server is not accessible. In case of an error, a warning is indicated in the validation dialog. If the system has been started without a connection to a configured OPC UA server, then *ibaPDA* tries to connect to this server at regular intervals. As long as the OPC UA server is not connected, the measured values stay at zero.

**Allow inaccessible symbols.**

Enable this option to start the acquisition even if no symbols are accessible. The inaccessible symbols are issued as warnings in the validation dialog box.

This can only occur if a symbol, whose address is requested by *ibaPDA* from the OPC UA server, is no longer available on the server. The OPC UA server then issues an error.

If you enable this option, *ibaPDA* ignores this error message and starts the acquisition anyway.

Measurement will not start when inaccessible symbols are present if you do not enable this option.

**Connection table**

The table shows the error counters as well as the response times (actual value, average, min and max) of each connection during the data measurement. To reset the calculated times and error counters to zero, simply click on the <Reset counters> button.

**Button <Open log file>**

If connections to OPC UA servers have been established, all connection-specific actions are logged in a text file. Using this button, you can open and see this file. In the file system on the hard disk, you will find the log files in the program path of the *ibaPDA* server (...\\Programs\\iba\\ibaPDA\\Server\\Log\\). The file name of the current log file is `OpcUAClientLog.txt`, the name of the archived log files is `OpcUAClientLog_yyyy_mm_dd_hh_mm_ss.txt`.

**Available modules**

- OPC UA Client (ibaPDA-Interface-OPC-UA-Client license required)
- OPC UA Server (ibaPDA-OPC-UA-Server+ license required)

**Product name**

ibaPDA-Interface-OPC-UA-Client (Art. no. 31.001111)

ibaPDA-OPC-UA-Server+ (Art. no. 30.670051)

---

**Other documentation**

A detailed description of this interface and its configuration can be found in the corresponding manual for the product, *ibaPDA-Interface-OPC-UA-Client*. You will find more information about the OPC UA server module in the product manual for *ibaPDA-OPC-UA-Server+*.

---

## 6.16 Modbus TCP client

**Description**

The Modbus TCP client interface lets *ibaPDA* act as a Modbus client (master) and requests data from the connected Modbus servers.

Up to 64 connections can be used per interface license. By purchasing up to 3 one-step-up licenses, the number of connections can be increased up to 256.

Each connection corresponds to a Modbus client module in *ibaPDA*. Each connection has a separate thread so that all connections are independent of each other. The cycle time mentioned in the connection table is the time it takes to read out all connection data configured. The <Reset counters> button can be used to reset the error counter and the average, minimum and maximum cycle times. When problems occur, check all communication via Modbus TCP/IP by pressing the <Open log file> button.

### Interface configuration

#### Set all values to zero...

You can optionally set all values to zero when the connection to a Modbus server is lost. Therefore, this option needs to be enabled.

#### Start acquisition even if...

By default, the acquisition cannot be started when the Modbus server is not accessible. However, enabling this checkbox allows you to force the start anyway.

### Connection table

#### IP address

This is the IP address of the connected Modbus controller.

#### Cycle time

The cycle time values refer to the communication cycle.

### Available modules

- Modbus client (protocols Modbus TCP and serial)

### Product name

ibaPDA-Interface-Modbus-TCP-Client (Art. no. 31.001022)

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product, *ibaPDA-Interface-Modbus-TCP-Client*.

---

## 6.17 Modbus TCP Server

### Description

The Modbus TCP Server interface lets *ibaPDA* act as a server (slave) that expects a client connection. The PLC acts as client (master) and requests a connection to *ibaPDA*. *ibaPDA* will not request information; it only listens to information that is sent over the network on port 502.

Up to 64 connections can be used per interface license. By purchasing up to 3 one-step-up licenses, the number of connections can be increased up to 256.

## Interface configuration

### Network interfaces

Using this drop-down list, you can select which network adapters on your computer should be 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 of the network adapters, an error message will be produced when validating the I/O configuration. By default the option *All* is selected.

### Use the network byte order on network

This option is enabled by default because the default byte order for data transfer over the Modbus TCP/IP network is Big Endian.

- Swap float values on word base

When using the default settings, you can optionally change the swap mode for float values to the Real module type. The telegram header is not affected.

### Use little-endian byte order on network

This option should be used if the client is based on a little-endian machine.

### Swap digital signals

If you enable this option, then the bytes are swapped only in the digital grids of the real and integer module types. The module types, Dig512 and generic, are not affected by this.

### Send response to Modbus master

If you enable this option, then each telegram is acknowledged with a response telegram. The response message mirrors the MBAP header, function code, starting address and data volume.

### Ignore sequence counter

If *ibaPDA* detects that the sequence counter (Transaction Identifier) is not continuously incremented, it displays "Sequence error" in the connection table.

Enabling this option removes the sequence counter columns from the connection grid.

### Allow ports through firewall

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

## Connection table

### Address

This is the IP address of the source computer from which *ibaPDA* receives data.

### Module index

The module index indicates which module number in *ibaPDA* should be assigned to this connection. The module index must be configured in the source system, e.g. ABB, *ibaLogic*. This module index must correspond to the entry in the field, Module index, in the corresponding module of the I/O Manager (module level, General tab). If necessary, adjust the module index on the source page or in *ibaPDA*.

**Available modules**

- Modbus Generic
- Modbus Dig512
- Modbus Integer
- Modbus Real

**Product name**

ibaPDA-Interface-Modbus-TCP-Server (Art. no. 31.001020)

---

**Other documentation**

A detailed description of this interface and its configuration can be found in the corresponding manual for the product, *ibaPDA-Interface-Modbus-TCP-Server*.

---

## 6.18 Raw Ethernet

**Description**

The Raw Ethernet communication uses IEEE 802.3 multicast frames. Up to 4 links are supported for data acquisition. Each link can be defined on a different NIC (network interface card). For each connection, the data sent must have a fixed arrangement.

If 2 links are defined for the same NIC, the multicast addresses must be different.

A maximum of up to 1024 modules are supported per interface.

**Interface configuration****Multicast address**

Enter different Multicast addresses for different connections for the same NIC.

**Network interface**

For each individual connection, select the network interface card (NIC) which is used for Raw Ethernet communication.

**Available modules**

- Raw Ethernet

**Product name**

ibaPDA-Interface-Raw-Ethernet (Art.- no. 31.001030)

---

**Other documentation.**

A detailed description of this interface and its configuration can be found in the corresponding manual for the product, *ibaPDA-Interface-Raw-Ethernet*.

---

## 6.19 Raytek

### Description

The Raytek interface is suitable for recording measurement data from Raytek MP40, MP50 and MP150 type (Fluke Process Instruments) temperature linescanners.

The scanners generate 1024 measurement points per line and they can scan up to 150 lines per second. The scanners automatically send their data to *ibaPDA* via an Ethernet TCP/IP connection. *ibaPDA* does not need to request the measuring data.

The scanners only support one connection per device. Two connections are supported per interface license.

The connections to the devices can be established via the computer's standard Ethernet interfaces. No further software is necessary for operation.

### Interface configuration

#### Set all values to zero when the connection to a device is lost

If this option is enabled, all measured values of a Raytek device are set to zero as soon as the connection is lost. If this option is disabled, *ibaPDA* will keep the last valid measured value at the time the connection was lost in its memory.

#### Start acquisition even if an device is not accessible

If this option is enabled, the acquisition will start even if the Raytek device is not accessible. Instead of an error, a warning is indicated in the validation dialog. If the system has been started without a connection to the device, *ibaPDA* will periodically try to connect to the device.

### Available modules

- Raytek MPx linescanner

### Product name

ibaPDA-Interface-Raytek

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-Raytek*.

---

## 6.20 S7 TCP/UDP

### Description

This interface provides *ibaPDA* with access to a SIMATIC S7 PLC via TCP/IP or UDP. The connection should be established over CP 443-1 for S7-400 CPUs or CP 343-1 for S7-300 CPUs.

The data to be measured will be moved by the S7 application program into data blocks (DBs) and stored in a fixed data structure. This structure is derived from the *ibaPDA* module types. The DBs are then sent as a message to the *ibaPDA* computer via S7 communication blocks (FCs). A specific connection must be configured on the S7-CPU for each message.

The maximum length of the message is limited to 4096 bytes.

Up to 64 connections can be used per interface license. By purchasing up to 3 one-step-up licenses, the number of connections can be increased up to 256.

### Interface configuration

#### Port number

The port no. must be exactly the same as previously configured when setting up the S7 connection.

#### <Reset port to default>

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

#### Allow ports through firewall

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

#### Network interfaces

Using this drop-down list, you can select which network adapters on your computer should be 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 of the network adapters, an error message will be produced when validating the I/O configuration. By default the option *All* is selected.

#### TCP/UDP port

"OK" is displayed here if the socket can be opened on this port. ERROR is displayed if conflicts occur, e.g., if the port is already busy.

#### <Reset statistics> button

By clicking on this button you can reset all the counters in the table.

### Connection table

#### Address

This is the IP address of the source computer from which *ibaPDA* receives data.

#### Mode

TCP or UDP communication mode display

#### Module index

The module index indicates which module number in *ibaPDA* should be assigned to this connection. The module index must be configured in the source system, e.g. ABB, *ibaLogic*. This module index must correspond to the entry in the field, Module index, in the corresponding module of the I/O Manager (module level, General tab). If necessary, adjust the module index on the source page or in *ibaPDA*.

### Available modules

- S7 TCP/UDP Generic
- S7 TCP/UDP Integer

- S7 TCP/UDP Real
- S7 UDP Request
- S7 UDP Request Decoder

**Product name**

ibaPDA-Interface-S7-TCP/UDP (Art. no. 31.001040)

**Other documentation**

A detailed description of this interface and its configuration can be found in the corresponding manual for the product, *ibaPDA-Interface-S7-TCP-UDP*.

## 6.21 Sisteam TCP

**Description**

The Sisteam TCP interface can be used by any controller capable of sending messages using the Sisteam protocols, in particular the Sisteam controllers of INGELECTRIC S.A.

Sisteam over TCP/IP is limited to sending data packets to destination port 8738 (hex: 0x2222). This means that the *ibaPDA* driver can only receive messages over port 8738. The source port is generated randomly as expected by any TCP/IP application capable to establish multiple connections (links). With the TCP/IP driver, *ibaPDA* acts as a connection server and the controllers act as clients. This means that the *ibaPDA* station monitors port 8738 for sender information and connection requests.

Up to 64 connections or modules respectively are supported.

**Interface configuration****Allow ports through firewall**

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

**Network interfaces**

Using this drop-down list, you can select which network adapters on your computer should be 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 of the network adapters, an error message will be produced when validating the I/O configuration. By default the option *All* is selected.

**Connection table****Address**

This is the IP address of the source computer from which *ibaPDA* receives data.

### Module index

The module index indicates which module number in *ibaPDA* should be assigned to this connection. The module index must be configured in the source system, e.g. ABB, *ibaLogic*. This module index must correspond to the entry in the field, Module index, in the corresponding module of the I/O Manager (module level, General tab). If necessary, adjust the module index on the source page or in *ibaPDA*.

### Available modules

- Sisteam TCP generic
- Sisteam TCP integer
- Sisteam TCP real

### Product name

ibaPDA-Interface-Sisteam-TCP (Art. no. 31.001055)

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product, *ibaPDA-Interface-Sisteam-TCP*.

---

## 6.22 TDC TCP/UDP

### Description

This interface provides *ibaPDA* with access to a SIMATIC TDC via a CP51M1 card. The interfaces, TDC TCP/UDP and S7 TCP/UDP, generally use the same protocol but a different port number. SIMATIC TDC must be configured as a TCP/IP client, i. e., the connection is established on the TDC side. For this reason, address level 2 has to be configured in the AT connection of the transmission module. The TCP/IP "Delayed acknowledgment" properties must be deactivated on the *ibaPDA* computer.

For TDC systems with a CP5100 board, we recommend using the Sisteam TCP interface.

The maximum length of the message is limited to 4096 bytes.

Up to 64 connections can be used per interface license. By purchasing up to 3 one-step-up licenses, the number of connections can be increased up to 256.

### Interface configuration

#### Port number

The default port number is 4171. This port number can be changed but must be the same in both systems, SIMATIC TDC and *ibaPDA*, in order to establish a connection.

#### <Reset port to default>

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

### Allow ports through firewall

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

### Network interfaces

Using this drop-down list, you can select which network adapters on your computer should be 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 of the network adapters, an error message will be produced when validating the I/O configuration. By default the option *All* is selected.

### TCP/UDP port

"OK" is displayed here if the socket can be opened on this port. "ERROR" is displayed if conflicts occur, e.g., if the port is already busy.

### <Reset statistics>

By clicking on this button you can reset all the counters in the table.

### Connection table

#### Address

This is the IP address of the source computer from which *ibaPDA* receives data.

#### Mode

TCP or UDP communication mode display

#### Module index

The module index indicates which module number in *ibaPDA* should be assigned to this connection. The module index must be configured in the source system, e.g. ABB or *ibaLogic*. This module index must correspond with the entry in the field, Module index, in the corresponding module of the I/O Manager (module level, General tab). If necessary, adjust the module index on the source page or in *ibaPDA*.

#### Available modules

- TDC TCP/UDP generic
- TDC TCP/UDP Integer
- TDC TCP/UDP real

#### Product name

ibaPDA-Interface-SIMATIC TDC TCP/UDP (Art. no. 31.001056)

---

#### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual of the product, *ibaPDA-Interface-TDC-TCP-UDP*.

---

## 6.23 VIP TCP/UDP

### Description

The Vendor Internet Protocol (VIP) function is used for communication between ABB AC450RMC controllers. In the context of this manual, the VIP protocol is used to capture data from various ABB controllers with *ibaPDA*. The following ABB controllers are suitable:

- AC450 RMC
- AC800M
- AC80
- AC800 PEC

*ibaPDA* acts as a server continuously monitoring port 5001 for any VIP clients requesting a connection or sending data.

The maximum length of the message is limited to 4096 bytes.

Up to 64 connections can be used per interface license. By purchasing up to 3 one-step-up licenses, the number of connections can be increased up to 256.

### Interface configuration

#### Port number

The default port number is 5001. This port number can be changed but must be the same in both systems, SIMATIC TDC and *ibaPDA*, in order to establish a connection.

#### <Reset port to default>

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

#### Allow ports through firewall

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

#### Network interfaces

Using this drop-down list, you can select which network adapters on your computer should be 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 of the network adapters, an error message will be produced when validating the I/O configuration. By default the option *All* is selected.

#### TCP/UDP port

"OK" is displayed here if the socket can be opened on this port. "ERROR" is displayed if conflicts occur, e.g., if the port is already occupied.

#### <Reset statistics>

By clicking on this button you can reset all the counters in the table.

### Connection table

- Address  
This is the IP address of the source computer from which *ibaPDA* receives data.
- Module index  
The module index indicates which module number in *ibaPDA* should be assigned to this connection. The module index is to be configured in the source system, e. g., ABB, ibaLogic. This module index must equal the entry in the field *Module index* in the corresponding module of the I/O Manager (module level, *General* tab ). If necessary, adjust the module index on the source page or in *ibaPDA*.

### Available modules

- VIP TCP/UDP generic
- VIP TCP/UDP integer
- VIP TCP/UDP real

### Product name

ibaPDA-Interface-VIP-TCP/UDP (Art. no. 31.001065)

---

#### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product, *ibaPDA-Interface-VIP-TCP-UDP*.

---

## 6.24 Modules for Ethernet-based interfaces

The modules available for Ethernet-based data interfaces are mentioned in the description of the interfaces.

See chapter ↗ *Ethernet-based interfaces*, page 166

Some Ethernet-based interfaces also offer the possibility of outputting data from *ibaPDA* to other participants ("Outp." column in the "Overview" table).

For more information see part 2, *Outputs*.

---

#### Other documentation



A detailed description of the modules and their configuration can be found in the corresponding interface manual.

---

## Overview

Module	Interface	Outp.	Comment	Manual
Generic DCS	AN-X-DCSNet		Submodule for the AN-X-DCSNet device	ibaPDA-Interface-AN-X-DCSNet
Symbolic DCS				
PLC-5	AB-Ethernet AB-Xplorer			ibaPDA-Interface-AB-Xplorer
SLC-500/MicroLogix				
EGD	EGD		Ethernet global data	ibaPDA-Interface-EGD
EGD Multicast				
EtherNet/IP I/O Module	EtherNet/IP	yes		ibaPDA-Interface-EtherNet-IP
EtherNet/IP I/O Scanner		yes		
EtherNet/IP Produced Tag				
GCOM generic	GCOM		Submodule on GCOM link	ibaPDA-Interface-GCOM
Generic TCP	Generic TCP			ibaPDA-Interface-Generic-TCP
Generic TCP Output		yes		
Generic unicast UDP	Generic UDP			ibaPDA-Interface-Generic-UDP
Generic multicast UDP				
Generic UDP Output		yes	Outputs only	
HiPAC Request			visible only with HiPAC request license	
Codesys Request V2				
Codesys Request V3				
TwinCAT-Request			visible only with TwinCAT request license	ibaPDA-Request-Twin-CAT
ibaLogic	ibaLogic TCP			ibaLogic
IEC 61850 device (GOOSE module, MMS module)	IEC 61850 Client			ibaPDA-Interface-IEC61850
IEC 61850-9-2 module	IEC 61850-9-2			ibaPDA-Interface-IEC61850-9-2
LSP (BINARY)	LANDSCAN			ibaPDA-Interface-LANDSCAN
LSP (ASCII)				

Module	Interface	Outp.	Comment	Manual
Modbus client	Modbus TCP Client	yes		ibaPDA-Interface-Modbus-TCP-Client
Modbus Dig512	Modbus TCP Server			ibaPDA-Interface-Modbus-TCP-Server
Modbus Generic				
Modbus Integer				
Modbus Real				
MQTT	MQTT	yes		ibaPDA-Interface-MQTT
OPC client module	OPC		OPC DA	➔ <i>OPC interface</i> , page 34
OPC server module			OPC DA	
Redundant OPC client			OPC DA	
OPC output module		yes	OPC DA	
OPC UA Client	OPC UA	yes		ibaPDA-Interface-OPC-UA-Client
OPC UA Server		yes	only usable with OPC UA Server + license	ibaPDA-OPC-UA-Server+
Raw Ethernet	Raw Ethernet		Submodule on the raw Ethernet link	ibaPDA-Interface-Raw-Ethernet
S7TCP/UDP Generic	S7 TCP/UDP			ibaPDA-Interface-S7-TCP-UDP
S7TCP/UDP Integer				
S7TCP/UDP Real				
S7 UDP Request				
S7 UDP Request Decoder				
Sisteam TCP Generic	Sisteam TCP		Analog values: Mixed data types	ibaPDA-Interface-Sisteam-TCP
Sisteam TCP integer			Analog values: integer only	
Sisteam TCP real			Analog values: real only	
TDC TCP/UDP Generic	TDC TCP/UDP		Analog values: Mixed data types	ibaPDA-Interface-TDC-TCP-UDP
TDC TCP/UDP Integer			Analog values: integer only	
TDC TCP/UDP real			Analog values: real only	

Module	Interface	Outp.	Comment	Manual
VIP TCP/UDP Generic	VIP TCP/UDP		Analog values: Mixed data types	ibaPDA-Interface-VIP- TDC-UDP
VIP TCP/UDP integer			Analog values: integer only	
VIP TCP/UDP real			Analog values: real only	

## 7 PLC-Xplorer interfaces

Name	Type	Connection to...	Comment	Link
AB-Xplorer	NIC	Allen-Bradley PLC-5, SLC-500		<a href="#">↗ AB-Xplorer, page 200</a>
ABB-Xplorer	NIC	ABB AC800M, AC800PEC		<a href="#">↗ ABB-Xplorer, page 201</a>
Bachmann-Xplorer	NIC	Bachmann M1		<a href="#">↗ Bachmann-Xplorer, page 202</a>
B&R-Xplorer	NIC	B&R X20 and others		<a href="#">↗ B&amp;R-Xplorer, page 203</a>
Codesys-Xplorer	NIC	Control with CODESYS V2 or CODESYS V3	e.g., 3S CODESYS SP, ABB AC500, Danieli HiPAC and many others	<a href="#">↗ Codesys-Xplorer, page 205</a>
Logix-Xplorer	NIC	Allen-Bradley ControlLogix, CompactLogix, MicroLogix		<a href="#">↗ Logix-Xplorer, page 206</a>
Melsec-Xplorer	NIC	Mitsubishi MELSEC		<a href="#">↗ MELSEC-Xplorer, page 207</a>
OMRON-Xplorer	NIC	OMRON PLC		<a href="#">↗ OMRON-Xplorer, page 208</a>
S7-Xplorer	NIC, MPI/Profibus	SIMATIC S7-200, -300, -400, -400H, -1200, -1500, LOGO! SIMATIC S5		<a href="#">↗ S7-Xplorer, page 209</a>
Sigmatex-Xplorer	NIC	Sigmatex-CPU's		<a href="#">↗ Sigma-tek-Xplorer, page 211</a>
TwinCAT-Xplorer	NIC	Beckhoff PLC		<a href="#">↗ TwinCAT-Xplorer, page 212</a>

NIC = Network Interface Card

## 7.1 AB-Xplorer

### Description

The AB-Xplorer interface is suitable for the acquisition of measured data with *ibaPDA* from Allen-Bradley/Rockwell PLC-5, SLC-500 and MicroLogix controller via an Ethernet connection.

The AB-Xplorer interface can use different ways:

1. Direct Ethernet connection: This can be used for controllers with Ethernet onboard like e.g. the SLC-5/05 and PLC-5 with 1785-ENET card installed.
2. Connection via a 1761-NET-ENI adapter: The adapter should be connected to the DF1 port of SLC-500, PLC-5 or MicroLogix controllers.
3. EtherNet/IP connection: The MicroLogix 1100 and 1400 controllers support EtherNet/IP directly.
4. Connection via DH+ gateway: An SLC-5/04 or PLC-5 controller can be connected via DH+ (Data Highway Plus) to a ControlLogix controller. *ibaPDA* will then communicate with the SLC-5/04 or PLC-5 via the ControlLogix controller.

Up to 16 connections can be used per interface license. By purchasing up to 14 licenses, the number of connections can be increased up to 240.

### Interface configuration

#### Set all values to zero when the connection to a PLC is lost.

If this option is enabled, all measured values of the PLC are set to zero as soon as the connection gets lost.

If this option is disabled, *ibaPDA* will keep the last valid measured value at the time the connection was lost in the memory.

#### Start acquisition even if a PLC is not accessible.

If this option is enabled, the acquisition will start even if the Allen-Bradley controller is not accessible. In case of an error, a warning is indicated in the validation dialog. If the system has been started without a connection to the Allen-Bradley controller, *ibaPDA* will periodically try to connect to the PLC.

### Connection table

For information about the connection table, see [↗ Connection table](#), page 169

### Available modules

- PLC-5
- SLC-500/MicroLogix

### Product name

ibaPDA-Interface-AB-Xplorer (Art. no. 31.000003)

---

**Note**

A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-AB-Xplorer*

---

## 7.2 ABB-Xplorer

### Description

The ABB-Xplorer interface is suitable for measurement data acquisition with *ibaPDA* to ABB AC800M and AC800PEC controllers.

The data is cyclically read by *ibaPDA* instead of being sent by the PLC. In the ABB AC800 controller, no programming work is necessary for establishing a connection between *ibaPDA* and a controller with defined IP address and for sending the respective signals.

For transmitting measurement data, no additional software from ABB is necessary. However, for AC800 controllers, variables that are accessible from *ibaPDA* must be defined as MMS access variables in the ABB Compact Control Builder.

### Interface configuration

#### Set all values to zero when the connection to a PLC is interrupted

If enabled, all measured values of the PLC are set to zero as soon as the connection is lost. If this option is disabled, *ibaPDA* will keep the last valid measured value at the time the connection was lost in the memory.

#### Start acquisition even if a PLC is not accessible

If this option is enabled, the acquisition will start even if an AC800 controller is not accessible. Instead of an error, a warning is indicated in the validation dialog. If the system has been started without a connection to the AC800 controller, *ibaPDA* will try to connect to the PLC at regular intervals. The measured values stay at zero as long as the PLC is disconnected.

#### Allow inaccessible parameters

Enable this option to start the acquisition even if no parameters are accessible. The inaccessible parameters are output as warnings in the validation dialog instead of errors.

This can only occur if the address book is not up-to-date.

If this option is not enabled and inaccessible parameters are present, then the acquisition will not start.

### Connection table

For information about the connection table, see [➤ Connection table](#), page 169.

### <Managing address books>

Clicking on the <Managing address books> button takes you to the address book management of *ibaPDA*.

The table shows a list of all of the address books currently present in the system with IP address of the PLC from which the address book was created, as well as the date of creation, size and

modules that were configured for the respective CPU. Use the <Delete select address books> button to delete selected address books.

### Available modules

ABB-Xplorer MMS

### Product name

ibaPDA-Interface-ABB-Xplorer (Art. no. 31.000009)

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-ABB-Xplorer*.

---

## 7.3 Bachmann-Xplorer

### Description

The Bachmann-Xplorer interface is suitable for measurement data acquisition with *ibaPDA* from a Bachmann M1 system via an Ethernet connection.

The data is cyclically read by *ibaPDA* instead of being sent by the PLC. In the M1 controller, no programming and configuration is necessary to establish a connection between *ibaPDA* and a controller with defined IP address and to transmit the respective signals.

In addition, no additional software from Bachmann is necessary to transmit measurement data. The M1 variables to be measured can be conveniently selected in the M1 address book browser.

### Interface settings

#### **Set all values to zero when the connection to a PLC is lost.**

If enabled, all measured values of the PLC are set to zero as soon as the connection is lost. If this option is disabled, *ibaPDA* will keep the last valid measured value at the time the connection was lost in the memory.

#### **Start acquisition even if a PLC is not accessible.**

If this option is enabled, the acquisition will start even if an M1 controller is not accessible. In case of an error, an alert is indicated in the validation dialog. If the system has been started without a connection to the M1 controller, *ibaPDA* will attempt to connect to the PLC at regular intervals. The measured values will remain at zero as long as the PLC is disconnected.

#### **Allow inaccessible parameters**

Enable this option to start the acquisition even if no parameters are accessible. The inaccessible parameters are output as warnings in the validation dialog instead of errors.

This can only occur if the address book is not up-to-date.

If this option is disabled and inaccessible parameters are present, then the acquisition will not start.

**<Manage address books>**

Clicking on the <Manage address books> button takes you to the address book management of *ibaPDA*.

The table shows a list of all of the address books currently present in the system with IP address of the PLC from which the address book was created, as well as the date of creation, size and modules that were configured for the respective CPU. Use the <Delete selected address books> button to delete selected address books.

**Connection table**

For information about the connection table, see [➤ Connection table, page 169](#).

**Available modules**

M1-Xplorer

**Product name**

ibaPDA-Interface-Bachmann-Xplorer (Art. no. 31.000034)

---

**Other documentation**

A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-Bachmann-Xplorer*.

---

## 7.4 B&R-Xplorer

**Description**

The B&R-Xplorer data interface is suitable for the recording of measured data on B&R Industrial PCs and control systems like the X20 system. The communication between *ibaPDA* and the B&R system is established over standard network cards. Access is transparent to the controller. It is not necessary to configure or program the controller especially.

Up to 16 connections can be configured with a B&R-Xplorer interface on each license. A total of a maximum of 240 connections can be implemented by the additional purchase of up to 14 further one-step-up-B&R-Xplorer licenses. One connection is required for each B&R PLC.

This is an Xplorer interface: This means that the data is not sent by the PLC, but rather is read cyclically by *ibaPDA*. To communicate with the PLC, *ibaPDA* uses the B&R PVI Library (PVI Manager). PVI Manager can be used locally or on a remote computer.

The signals to be measured can be conveniently selected using their symbolic names with support from the *ibaPDA* symbol browser. This allows access to all measurable symbols, which are stored in the PLC itself.

**Interface configuration**

The interface has the following features and configuration options:

**Set all values to zero when the connection to a PLC is lost.**

If enabled, all measured values of the PLC are set to zero as soon as the connection is lost. If this option is disabled, *ibaPDA* will keep the last valid measured value in memory at the time the connection was lost.

**Start acquisition even if a PLC is not accessible.**

If this option is enabled, the acquisition will start even if the controller is not accessible. A warning is issued in the validation dialog. If the system has been started without a connection to the controller, *ibaPDA* will try to connect to the CPU at regular intervals

**Allow inaccessible symbols**

Enable this option to start the acquisition even if no symbols are accessible. The inaccessible symbols are indicated as warnings in the validation dialog instead of errors.

This can only occur if the address book is not up-to-date.

If you do not enable this option, then the acquisition does not start if there are inaccessible symbols.

**<Open log file>**

If connections to B&R controllers have been established, all connection-specific actions are logged in a text file. You can open and view the file by clicking on this button. You will find the log file in the program path of the *ibaPDA* server in the file system on the hard disk (...\[ProgramData\iba\ibaPDA\Log\](#)). The file name of the current log file is [B&R.txt](#); the name of the archived log files is [B&RLog\\_yyyy\\_mm\\_dd\\_hh\\_mm\\_ss.txt](#).

**Connection table**

For information about the connection table, see [↗ Connection table](#), page 169.

**Available modules**

- B&R-Xplorer

**Product name**

ibaPDA-Interface-B&R-Xplorer (Art. no. 31.000006)

---

**Other documentation**

A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-B&R-Xplorer*.

---

## 7.5 Codesys-Xplorer

### Description

The Codesys-Xplorer data interface for CODESYS is appropriate for measurement data acquisition via TCP/IP using the standard NICs. Access is transparent for the controller. It is not necessary to separately configure or program the controller. The signals to be measured can conveniently be selected based on the symbolic names supported by the *ibaPDA* symbol browser. This enables the access to all defined symbols of the linked CODESYS project.

Up to 16 connections can be used per interface license. By purchasing up to 14 licenses, the number of connections can be increased up to 240.

### Interface configuration

#### Set all values to zero when the connection to a PLC is lost.

If this option is enabled, all the measured values of a CODESYS CPU are set to zero as soon as the connection is lost. If this option is disabled, *ibaPDA* will keep the last valid measured value at the time the connection was lost in the memory.

#### Start acquisition even when a PLC is not accessible.

If this option is enabled, the acquisition will start even if a CODESYS-CPU server is not accessible. A warning is issued in the validation dialog. If the system has been started without a connection to the CODESYS CPU, *ibaPDA* will try to connect to the CPU at regular intervals

#### Allow inaccessible symbols

Enable this option to start the acquisition even if no symbols are accessible. The inaccessible symbols are issued as warnings in the validation dialog. This can only occur if the address book is not up to date. If you do not enable this option then the measurement does not start if there are inaccessible symbols.

### Connection table

For information about the connection table, see [➤ Connection table](#), page 169

### Available modules

- Codesys V2
- Codesys V3

### Product name

ibaPDA-Interface-Codesys-Xplorer (Art. no. 31.000002)

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-Codesys-Xplorer*.

---

## 7.6 Logix-Xplorer

### Description

The Logix-Xplorer interface is suitable for the recording of measured data with *ibaPDA* on an Allen Bradley Controller, type ControlLogix, CompactLogix or MicroLogix using an EtherNet/IP connection. Beside digital signals and analog signals of various types it supports text signals too.

### Other documentation



For more information about the EtherNet/IP protocol, see the *ibaPDA-Interface-EtherNetIP* manual.

The data is cyclically read by *ibaPDA* instead of being sent by the PLC.

In the Logix controller, no programming and configuration is necessary for establishing a connection between *ibaPDA* and a controller with defined IP address and for sending the respective signals. For transferring measurement data, no additional software of Rockwell Automation is necessary.

### Interface configuration

The interface has the following features and configuration options:

#### **Set all values to zero when the connection to a PLC is lost.**

If enabled, all measured values of the PLC are set to zero as soon as the connection is lost. If this option is disabled, *ibaPDA* will keep the last valid measured value in memory at the time the connection was lost.

#### **Start acquisition even if a PLC is not accessible.**

If this option is enabled, the acquisition will start even if a Logix controller is not accessible. In case of an error, an alert is indicated in the validation dialog. If the system has been started without a connection to the Logix controller, *ibaPDA* will try to connect to the PLC at regular intervals. The measured values stay at zero as long as the PLC is disconnected.

#### **Allow inaccessible symbols**

Enable this option to start the acquisition even if no symbols are accessible. The inaccessible symbols are indicated as warnings in the validation dialog instead of errors. This can only occur if the address book is not up-to-date.

If you do not enable this option, then the acquisition does not start if there are inaccessible symbols.

#### **<Open log file>**

If connections to Logix controllers have been established, all connection-specific actions are logged in a text file. Using this button, you can open and see this file. In the file system on the hard disc, you will find the log file in the program path of the *ibaPDA* server (...\[ProgramData\iba\ibaPDA\Log\](#)). The file name of the current log file is [LogixLog.txt](#); the name of the archived log files is [LogixLog\\_yyyy\\_mm\\_dd\\_hh\\_mm\\_ss.txt](#).

## Connection table

For information about the connection table, see [➤ Connection table](#), page 169.

## Available modules

- Logix-Xplorer

## Product name

ibaPDA-Interface-Logix-Xplorer (Art. no. 31.000007)

---

## Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-Logix-Xplorer*.

---

## 7.7 MELSEC-Xplorer

### Description

The MELSEC-Xplorer interface is suitable for measurement data acquisition with *ibaPDA* from Mitsubishi MELSEC controllers. The data is cyclically read by *ibaPDA* instead of being sent by the PLC.

In the MELSEC controller, no programming and configuration is necessary for establishing a connection between *ibaPDA* and a controller with defined IP address and for sending the respective signals. Only the Mitsubishi MC protocol must be enabled in the PLC parameters on the controller (GX Works).

The connection between *ibaPDA* and MELSEC control can be established either over the network interface of the CPU or over an Ethernet communication module.

### Interface configuration

The interface has the following features and configuration options:

#### Set all values to zero when the connection to a PLC is lost.

If enabled, all measured values of the PLC are set to zero as soon as the connection is lost. If this option is disabled, *ibaPDA* will keep the last valid measured value at the time the connection was lost in the memory.

#### Start acquisition even if a PLC is not accessible

If this option is enabled, the acquisition will start even if a MELSEC controller is not accessible. In case of an error, an alert is indicated in the validation dialog. If the system has been started without a connection to the MELSEC controller, *ibaPDA* will try to connect to the PLC at regular intervals. The measured values stay at zero as long as the PLC is disconnected.

#### Allow inaccessible operands

Enable this option to start the acquisition even if no symbols are accessible. The inaccessible symbols are indicated as warnings in the validation dialog instead of errors. This can only occur if the address book is not up-to-date.

If you do not enable this option, then the acquisition does not start if there are inaccessible symbols.

#### <Open log file>

If connections to Logix controllers have been established, all connection-specific actions are logged in a text file. Using this button, you can open and see this file. In the file system on the hard disc, you will find the log file in the program path of the *ibaPDA* server (...\[ProgramData\iba\ibaPDA\Log\](#)). The file name of the current log file is `LMelsecLog.txt`; the name of the archived log files is `MelsecLog_yyyy_mm_dd_hh_mm_ss.txt`.

#### Connection table

For information about the connection table, see ↗ [Connection table](#), page 169.

#### Available modules

- MELSEC

#### Product name

ibaPDA-Interface-MELSEC-Xplorer (Art. no. 31.000008)

---

#### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-MELSEC-Xplorer*.

---

## 7.8 OMRON-Xplorer

The OMRON-Xplorer interface is suitable for the acquisition of measured data with *ibaPDA* from an OMRON controller via an Ethernet connection and using the FINS protocol.

The data is cyclically read by *ibaPDA* instead of being sent by the PLC.

#### Interface configuration

##### Start acquisition even if a CPU is not accessible

Enable this option to start the acquisition even if a CPU is not accessible. Instead of an error, a warning is prompted in the validation dialog.

If the system has been started without a connection to the CPU, *ibaPDA* tries to connect to the PLC at regular intervals. The measured values stay at zero as long as the PLC is disconnected.

##### Allow inaccessible operands

Enable this option to start the acquisition even if operands are not accessible. The inaccessible operands are prompted as warnings in the validation dialog instead of errors. If this option is disabled and inaccessible operands are present, then the acquisition will not start.

#### <Open log file>

If connections to controllers have been established, all connection specific actions are recorded in a text file. Using this button, you can open and check this file. In the file system on the hard disk, you find the log files of this interface in the path ...\[ProgramData\iba\ibaPDA\Log](#).

The file name of the current log file is `InterfaceLog.txt`; the name of the archived log files is `InterfaceLog_yyyy_mm_dd_hh_mm_ss.txt`.

#### <Reset statistics>

Click this button to reset the calculated times in the table to zero.

#### Connection table

For information about the connection table, see [↗ Connection table](#), page 169.

#### Available modules

- OMRON FINS Module

#### Product name

ibaPDA-Interface-OMRON-Xplorer (Art. no. 31.000035)

---

#### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-OMRON-Xplorer*.

---

## 7.9 S7-Xplorer

### Description

The S7-Xplorer interface is suitable for acquiring measured data via TCP/IP using the standard network interface cards as well as using the PPI, MPI, PROFIBUS, TCP/IP and ISO transport logs via SIMATIC NET interface cards. Though for the controller, access is transparent. It is not necessary to configure or program the controller especially.

The signals to be measured can conveniently be selected with the aid of the absolute operand addresses or the symbolic names supported by the *ibaPDA* address-book browser. This enables the access to all defined symbols of the linked STEP 7 project.

When using the SIMATIC S7 CFC editor (version V 6.0) on the same computer, the signals and connectors to be measured can be configured from the control program using drag & drop.

With an additional interface converter (ACCON-S5-LAN<sup>®</sup>, Deltalogic or IBH Link S5++, IBHsoftec) measurement data from a SIMATIC S5 control can also be acquired via its AS511 interface.

Up to 16 connections can be used per interface license. By purchasing up to 14 licenses, the number of connections can be increased up to 240.

### Interface configuration

#### Set all values to zero when the connection to a PLC is lost.

If this option is enabled, all the measured values of an S7 CPU are set to zero as soon as the connection is lost.

**Start acquisition even if a PLC is not accessible.**

If this option is enabled, the acquisition will start even if an S7 CPU server is not accessible. A warning is issued in the validation dialog. If the system has been started without a connection to the S7 CPU, *ibaPDA* will try to connect to the CPU at regular intervals

**Allow inaccessible symbols**

Enable this option to start the acquisition even if no S7 operands are accessible. The inaccessible operands are indicated as alerts in the validation dialog instead of errors.

**Enable S7-Xplorer outputs**

Enable this option to enable the output modules. With the S7 Xplorer outputs, it is possible to write directly on S7 operands and S7 symbols.

**<Manage address books>**

This switches you to the dialog of the *Address books* node in the *General* tab of the I/O Manager. You can then import, create or delete address books.

For more information about address books, see part 2, *Address books*.

**<Open log file>**

This opens a text file, in which all the connection-specific information is logged.

**<Reset counters>**

The error counters and response times in the connection table are reset to zero.

**Connection table**

For information about the connection table, see [➤ Connection table](#), page 169.

---

**Note**

When the use of the S7-Xplorer module is redundant, one row is displayed for each of the two connections. The one not active receives the "Standby" status and is displayed in red. This is not an error situation but the normal operating mode.

---

**Available modules**

- S7-Xplorer
- S7-Xplorer decoder
- S7-Xplorer redundant
- S7-Xplorer SINUMERIK
- S5 (LAN-Adapter)
- S7-200
- LOGO!

**Product name**

ibaPDA-Interface-S7-Xplorer (Art. no. 31.000001)

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-S7-Xplorer*.

---

## 7.10 Sigmatek-Xplorer

### Description

The Sigmatek-Xplorer data interface is suitable for measurement data acquisition on a SIGMATEK PLC via TCP/IP using the standard NICs. Access is transparent for the controller. It is not necessary to configure or program the controller especially. The signals to be measured can conveniently be selected based on the symbolic names supported by the *ibaPDA* symbol browser. This enables access to all measurable symbols (servers, clients, global variables), based on the imported server object list of the SIGMATEK LASAL project. LASAL SERVICE is the programming software of SIGMATEK.

Up to 16 connections can be configured per license with a Sigmatek-Xplorer interface. A total of 64 connections can be established by purchasing up to an additional 3 *one-step-up-Sigmathek-Xplorer* licenses. One connection is required per connected SIGMATEK PLC.

### Interface configuration

#### **Set all values to zero when the connection to a PLC is lost.**

If this option is enabled, all the measured values of the PLC are set to zero as soon as the connection is lost. If this option is disabled, *ibaPDA* will keep the last valid measured value at the time the connection was lost in the memory.

#### **Start acquisition even if a PLC is not accessible.**

If this option is enabled, the acquisition will start even if the PLC is not accessible. In case of an error, an alert is indicated in the validation dialog. If the system has been started without a connection to the PLC, *ibaPDA* will try to connect to the PLC at regular intervals.

#### **Allow inaccessible symbols**

Enable this option to start the acquisition even if no symbols are accessible. The inaccessible symbols are issued as warnings in the validation dialog. This can only occur if the address book is not up to date. If you do not activate this option, then the measurement will not start in the presence of inaccessible symbols.

### Connection table

For information about the connection table, see [➤ Connection table](#), page 169

### Available modules

- Sigmathek-Xplorer

### Product name

ibaPDA-Interface-Sigmathek-Xplorer (Art. no. 31.000004)

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-Sigmatek-Xplorer*.

---

## 7.11 TwinCAT-Xplorer

### Description

The TwinCAT-Xplorer data interface is suitable for the recording of measured data on a Beckhoff PLC using the Beckhoff ADS protocol over standard network cards. Access is transparent for the controller. It is not necessary to configure or program the controller especially. Up to 16 connections can be configured with a TwinCAT-Xplorer interface on each license. A total of a maximum of 240 connections can be implemented by the additional purchase of up to 14 further one-step-up-TwinCAT-Xplorer licenses. One connection is required for each Beckhoff PLC.

The interface supports TwinCAT versions 2 and 3 running on industrial PCs, embedded PCs (CX series) and bus controllers (BC/BX series).

The signals to be measured can be conveniently selected using their symbolic names with support from the *ibaPDASymbol* Browser. This allows access to all measurable symbols, which are stored in the PLC itself or which are available in a symbol file (.tpy).

### Interface configuration

The interface has the following features and configuration options:

#### **Set all values to zero when the connection to a PLC is lost.**

If enabled, all measured values of the PLC are set to zero as soon as the connection is lost. If this option is disabled, *ibaPDA* will keep the last valid measured value in memory at the time the connection was lost.

#### **Start acquisition even if a PLC is not accessible.**

If this option is enabled, the acquisition will start even if the controller is not accessible. A warning is issued in the validation dialog. If the system has been started without a connection to the controller, *ibaPDA* will try to connect to the PLC at regular intervals

#### **Allow inaccessible symbols**

Enable this option to start the acquisition even if no symbols are accessible. The inaccessible symbols are indicated as warnings in the validation dialog instead of errors.

This can only occur if the address book is not up-to-date.

If you do not enable this option, then the acquisition does not start if there are inaccessible symbols.

#### **<Open log file>**

If connections to TwinCAT controllers have been established, all connection-specific actions are logged in a text file. Using this button, you can open and see this file. In the file system on the hard disk, you will find the log file in the program path of the *ibaPDA* server (...\[ProgramData\iba\ibaPDA\Log\](#)). The file name of the current log file is [TwinCATLog.txt](#); the name of the archived log files is [TwinCATLog\\_yyyy\\_mm\\_dd\\_hh\\_mm\\_ss.txt](#).

**Connection table**

For information about the connection table, see [➤ Connection table](#), page 169.

**Available modules**

- TwinCAT PLC
- BC/BX Controller

**Product name**

ibaPDA-Interface-TwinCAT-Xplorer (Art. no. 31.000005)

---

**Other documentation**

A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-TwinCAT-Xplorer*.

---

## 8 Siemens-specific interfaces

Name	Type	Connection to...	Comment	Link
ibaFOB-SD iba- FOB-SDexp	PCI card PCIe card	SIMADYN D		<a href="#">↗ FOB-SD/- SDexp and FOB-TDC/-TD- Cexp, page 215</a>
ibaFOB-TDC ibaFOB-TD- Cexp	PCI card PCIe card	SIMATIC TDC		<a href="#">↗ FOB-SD/- SDexp and FOB-TDC/-TD- Cexp, page 215</a>
Simadyn Request	Software	SIMADYN D	ibaFOB-SD required	<a href="#">↗ TDC Request and Simadyn Re- quest, page 217</a>
TDC Request	Software	SIMATIC TDC	ibaFOB-TDC required	<a href="#">↗ TDC Request and Simadyn Re- quest, page 217</a>
ibaFOB- PlusControl	PCI card	PlusControl	Special card for Siemens Energy	
CP1616 CP1626	Siemens PCI-/PCIe card	PROFINET		<a href="#">↗ CP1616, page 218</a> <a href="#">↗ CP1626, page 219</a>
MMC request	Software	Simicro MMC216		<a href="#">↗ MMC request, page 221</a>
S7-Xplorer	Software	SIMATIC S7	ibaPDA-PLC-Xplor- er or ibaPDA-Inter- face-S7-Xplorer	<a href="#">↗ S7-Xplorer, page 209</a>
SINAM- ICS-Xplorer	Software	SINAMICS frequen- cy converters		<a href="#">↗ SIN- AMICS-Xplorer, page 223</a>
SIMO- TION-Xplor- er	Software	SIMOTION frequen- cy converters		<a href="#">↗ SIMOTI- ON-Xplorer, page 224</a>
SINUMER- IK-Xplorer	Software	SINUMERIK CNC controls, NCK		<a href="#">↗ SINUME- RIK-Xplorer, page 226</a>

Many combinations of hardware interfaces, software interfaces and features can be used by *ibaPDA* in the SIMATIC environment. Several options are available for the "Request" feature, in particular. The following table shows some combinations of target system, interfaces and licenses.

Target system	Interface	Transmission	iba interface	Manual	License
SIMADYN D	CS12/13/14	LWL	ibaFOB-SD	Request-SD-TDC	Request-SD
SIMATIC TDC	CP53		ibaFOB-SDexp		
	GDM		ibaFOB-TDC ibaFOB-TDC-exp		
S7-400	CP50	PROFIBUS	ibaBM-DP ibaBM-DPM-S	Request-FM458-TDC	Request-FM458/TDC
	FM458		ibaCom-L2B		
S7-300	S7-CPU CP x43	PROFINET	ibaBM-PN	Request-S7-DP/PN/ ibaNet-E	Request-S7-DP/PN/ ibaNet-E
		UDP	ibaNet-E		
S7-1200			Ethernet	Request-S7-UDP	Request-S7-UDP
S7-1500		TCP/IP, MPI, DP	Ethernet, MPI-Adapter	S7-Xplorer	S7-Xplorer

## 8.1 FOB-SD/-SDexp and FOB-TDC/-TDCexp

### Description

The *ibaFOB-TDC* (PCI) or *ibaFOB-TDCexp* (PCIe) board is used for connections between the SIMATIC TDC control system and *ibaPDA*. The board must be installed in the *ibaPDA* PC and connected to a free port on the CP52IO interface board in the Global Data Memory (GDM) of the SIMATIC TDC system.

The *ibaFOB-SD* (PCI) or *ibaFOB-SDexp* (PCIe) board connects the *ibaPDA* system or the Soft-SPS *ibaLogic* with the Siemens SIMADYN D control system. The board therefore needs to be connected to one of the free fiber optic links of the SIMADYN-D CS12/13 or 14.

As the SIMADYN D system is no longer supported by Siemens, the following description refers only to *ibaFOB-TDC*. All statements also apply to *ibaFOB-TDCexp* and *ibaFOB-SDexp*.

Except for some minor differences in the properties and settings, the *ibaFOB-SD* and *ibaFOB-TDC* cards are almost the same.

The newer PCI Express card models offer extended diagnostic capabilities compared to their predecessors.

### Note



These interface cards are also a prerequisite for using the Simadyn Request and TDC Request interfaces.

**Available modules**

- TDC Lite (ibaFOB-TDC)
- TDC Text (ibaFOB-TDC)
- Simadyn D Lite (ibaFOB-SD)
- Simadyn D Text (ibaFOB-SD)

**Product name**

ibaFOB-SD (Art. no. 11.112700), ibaFOB-SDexp (Art. no. 11.112701), ibaFOB-TDC (Art. no. 11.112600) or ibaFOB-TDCexp (Art. no. 11.112601)

---

**Other documentation**

A detailed description of this interface and its configuration can be found in the manuals for the following products:

- *ibaFOB-SD* or *ibaFOB-SDexp*
  - *ibaFOB-TDC* or *ibaFOB-TDCexp*
  - *ibaPDA-Request-SD-TDC*
-

## 8.2 TDC Request and Simadyn Request

The TDC Request data interface is based on the *ibaFOB-TDC* and *ibaFOB-TDCexp* component. The purchase of an additional license for this extension is required. With this interface, you can easily access all signals of the TDC stations connected to the GDM.

The use of this interface must be enabled in the dongle and an *ibaFOB-TDC* or *ibaFOB-TDCexp* card must be inserted into the *ibaPDA* computer.

The only connection required is a fiber optic link between *ibaPDA* and the GDM of SIMATIC TDC.

The following modules can be used in *ibaPDA* with the TDC Request interface:

- TDC Request
  - Access on all data of all TDC stations connected with the GDM, per request
  - max. 1024 modules per interface
  - max. 1000 analog + 1000 digital signals per module
  - Convenient signal selection via signal browser (address book)
  - Convenient signal selection using drag & drop from the SIMATIC Manager (CFC) to *ibaPDA*. (The *ibaPDA* client must be installed on the same PC as the SIMATIC Manager.)

The number of signals to be used is only limited by the *ibaPDA* license and the performance of the systems.

---

### Note



All of the items described here also apply to Simadyn Request (CFC programming required).

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual *ibaPDA-Request-SD-TDC*.

---

## 8.3 CP1616

### Description

The Siemens CP1616 board must be installed in the *ibaPDA* PC in order to obtain this interface in the I/O Manager. The card is required for connecting to a PROFINET communication network. Prior thereto, the board needs to be properly configured with SIMATIC system software (STEP 7, NetPro, etc.).

Up to 4 CP1616 cards are supported per *ibaPDA* computer.

If the card has been setup correctly and connected to an active PROFINET network, the status bar shows the name of the controller (CP1616) and its status (online/offline). Extensive information regarding interrupts is available for diagnostic purposes.

A maximum of up to 1024 modules are supported per interface.

---

### Note



A different and more flexible PROFINET connection option is the use of the *ibaBM-PN* device in conjunction with an *ibaFOB-io-D*-card.

---

### Interface configuration

In principle, no settings are required at this stage. The following buttons are available:

- <Reset>: resets the card.
- <Open log file>: opens the PROFINET log file.
- <Dump firmware trace buffer>: This button is used to issue the signal buffer of the firmware as a text file. *ibaPDA* can automatically detect CP1616 firmware errors. If such an error occurs, *ibaPDA* will output the firmware signal buffer automatically and record an error message in the event log.
- <Reset counters>: resets all diagnostic counters and backup times Resets all diagnostic counters and copy times.

### Connection table

The table gives an overview of all receivers configured on the card. With an established connection, the receiver has a green background. A red background shows that there is no connection. Each row, i.e., each receiver address, corresponds to a module configured under this interface.

### Available modules

- PROFINET controller IRT top
- PROFINET controller RT/IRT flex
- Simotion D

### Product name

ibaPDA-Interface-Profinet-CP (Art. no. 31.001350)

---

### Other documentation



A detailed description of this interface and its configuration as well as an example can be found in the documentation:

- ibaPDA-Interface-Profinet\_QuickGuide\_v1.0\_en.pdf
  - ibaPDA\_Interface\_SimotionAndSinamics\_v1.0\_de.pdf
- 

## 8.4 CP1626

### Description

The Siemens CP1626 module is the PCI Expression version of the CP1616 module and it must be installed in the *ibaPDA* computer in order to obtain this interface in the I/O Manager. The board is required for connecting to a Profinet communication network. Before this step, the board needs to be properly configured with SIMATIC system software (STEP 7, NetPro, etc.).

The module has the following properties:

- 2 PROFINET interface each with 2 ports
- Connection with *ibaPDA* to port X1 (PROFINET controller)
- PROFITNET must be configured so that CP1626, as the controller, reads the data from one or more PROFINET devices.
- The CP1626 module only works if DMA is enabled.

Up to 4 CP1626 boards are supported per *ibaPDA* computer.

If the board has been setup correctly and connected to an active PROFINET network, the status bar shows the name of the controller (CP1626) and its status (online/offline). Extensive information regarding interrupts is available for diagnostic purposes.

A maximum of up to 1024 modules are supported per interface.

---

### Note



A different and more flexible PROFINET connection option is the use of the *ibaBM-PN* device in conjunction with an *ibaFOB-io-D*-card.

---

### Interface configuration

In principle, no settings are required at this stage. The following buttons are available:

- <Reset>: resets the card.
- <Open log file>: opens the PROFINET log file.
- <Dump firmware trace buffer>: This button is used to issue the signal buffer of the firmware as a text file. *ibaPDA* can automatically detect CP1626 firmware errors. If such an error occurs, *ibaPDA* will output the firmware signal buffer automatically and record an error message in the event log.
- <Reset counters>: Resets all diagnostic counters and copy times.

**Connection table**

The table gives an overview of all receivers configured on the board. With an established connection, the receiver has a green background. A red background shows that there is no connection.

Each row, i.e., each receiver address, corresponds to a module configured under this interface.

**Available modules**

- PROFINETcontroller IRT top
- PROFINET controller RT/IRT flex
- Simotion D

**Product name**

ibaPDA-Interface-Profinet-CP (Art. no. 31.001350)

---

**Other documentation**

More information can be found in the documentation exemplified with the CP1616 board:

- ibaPDA-Interface-Profinet\_QuickGuide\_v1.0\_en.pdf
  - ibaPDA\_Interface\_SimotionAndSinamics\_v1.0\_de.pdf
-

## 8.5 MMC request

### Description

This interface was developed for a particular project and enables *ibaPDA* to access all internal variables of up to 4 MMC controls.

The C-nodes are used as gateways for communication between *ibaPDA* and individual virtual process environments.

### Interface configuration

#### Port no.:

Usually, the default port no. 6115 can remain unchanged. The same port no. has to be configured also in the serial port server.

#### Address book path

Enter the full path name of the folder in which the MMC LOC files are stored.

#### User name / password

Enter the correct user name and password here if the address book path refers to a network drive that requires user credentials.

#### Disable signals on non-responding VEs

If this option is enabled, the measurement channels for non-responding VEs will be temporarily deactivated. They are reactivated with the next request.

#### Enable VE reconnect detection

If this option is enabled, missing VEs are also checked during measurement. In case the VEs are available again, the measurement stops, a request will be executed and the measurement re-starts.

#### MMC VE response timeout [s]:

This setting defines the maximum period to wait for a response to commands to the MMC VEs. You may adjust the time according to the number of VEs and the MMC configuration.

### Connection table

When connected to active MMC controllers and with an address book loaded, the table beneath the properties shows the MMC-internal names of the VEs and the corresponding symbol names.

### Available modules

- MMC request

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product, `ibaPDAV6-Request_MMC_TKS_WW2_v1_de.pdf`.

---

## 8.6 ibaFOB-PlusControl

### Description

The *ibaFOB-PlusControl* PC card is used for the connection from Siemens PLUSCONTROL systems to the *ibaPDA* data acquisition system. To do this, the *ibaFOB-PlusControl* card is connected to a PLUSCONTROL CP.

The connection supports a data transmission rate of 1 Gbit/s. Up to 400 analog or digital signals can be transmitted

The parameters are set completely by the software. Jumper and jumper settings are not required.

### Interface configuration

As with the other ibaFOB cards, you should first select the card node in the interface tree in the I/O Manager.

The interrupt mode is set in the *Configuration* tab.

The interrupt mode is automatically determined by *ibaPDA*:

"Slave mode" is set as soon as other iba cards are inserted. Only in the event that no other card types are inserted besides several *ibaFOB PlusControl* cards you can determine which of the cards has the "Internal interrupt master" mode and thus generates the interrupt for the other cards. The interrupt is transmitted via the synchronization line to the other iba-PCIe cards (interrupt slaves) (with the supplied flat ribbon cables).

Check "In use" if the card should be used by *ibaPDA*.

### Test connection

Select the "Link" node of the card in the interface tree.

The link to the connected PLUSCONTROL system is established by clicking on the <Test connection> button in the *Connection* tab.

### Available modules

- PlusControl standard

### Product name

ibaFOB-PlusControl (Art. no. 11.112602)

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the card *ibaFOB-PlusControl*.

---

## 8.7 SINAMICS-Xplorer

### Description

The SINAMICS-Xplorer interface is suitable for measurement data acquisition from SINAMICS frequency inverters via TCP/IP using the standard network cards, as well as via Profibus using SIMATIC NET interface cards. Access is transparent for the drive controller. It is not necessary to specially configure or program it.

The signals to be measured are selected by entering the desired parameter indices and data types in the I/O Manager of *ibaPDA*. A parameter construction kit is available to allow the input of multiple parameters.

The connections to the SINAMICS drives can be established via standard interfaces of the computer or corresponding CP modules.

On the SINAMICS side, the following interfaces are supported:

- LAN X127 (TCP)
- PROFINET X150 P1 and X150 P2 (TCP)
- Communication Board Ethernet CBE20 X1400 (TCP)
- Profibus interface X126 (PROFIBUS)

Additional Siemens software (e.g., SIMATIC NET or SIMATIC STEP 7) is needed for operation, in case the connection to the control system is established via a SIMATIC NET communication card (CP) in the computer to an integrated Ethernet interface of the CPU (if available) or to a CP module in the PLC.

Generally, no particular configuration and programming is required on the drive side. When using the *PC/CP* connection mode, a suitable access point must be configured in the SIMATIC PG/PC interface of the *ibaPDA* computer.

Up to 32 drives can be connected per license. Extensions are possible.

### Interface settings

#### Set all values to zero if the connection to a drive...

If this option is enabled, all the measured values for a drive are set to zero as soon as the connection is lost.

#### Start acquisition even if a drive is not accessible

If this option is enabled, the acquisition will start even if a drive is not accessible. Instead of an error, a warning is indicated in the validation dialog. If the system has been started without a connection to the drive, *ibaPDA* will try to connect to the drive at regular intervals

#### Allow inaccessible parameters

Enable this option to start the acquisition even if no drive parameters are accessible. Inaccessible parameters are then issued as alerts in the validation dialog rather than errors.

#### <Open log file>

If connections to drives have been established, all connection-specific actions are recorded in a text file. Using this button, you can open and see this file. In the file system on the hard drive,

you will find the log files in the program path of the *ibaPDA* server (...\[ProgramData\iba\ibaPDA\Log\](#)). The file name of the current log file is [SINAMICSLog.txt](#), the name of the archived log files is [SINAMICSLog\\_yyyy\\_mm\\_dd\\_hh\\_mm\\_ss.txt](#).

#### <Reset counter>

Resets the error counter as well as the response times in the connection table to zero.

#### Connection table

The table shows the counters as well as the response times of the individual connections during data measurement.

#### Available modules

- SINAMICS

#### Product name

ibaPDA-Interface-SINAMICS-Xplorer (Art. no. 31.000030)

The SINAMICS-Xplorer interface license is also included in the ibaPDA-Drive-Xplorer product (Art. no. 31.001044).

---

#### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-SINAMICS-Xplorer*.

---

## 8.8 SIMOTION-Xplorer

### Description

The SIMOTION-Xplorer interface is suitable for measurement data acquisition from SIMOTION frequency inverters via Ethernet using the standard network cards, as well as via PROFIBUS using SIMATIC NET interface cards. Access is transparent for the drive controller. It is not necessary to specially configure or program it.

The signals to be measured can be selected on the basis of an address book via a symbol browser in the I/O Manager of *ibaPDA*.

The connections to the SIMOTION drives can be established via standard interfaces of the computer or corresponding CP modules.

Additional Siemens software (e.g., SIMATIC NET, SIMATIC STEP 7 or SIMOTION SCOUT) is required for operation if the connection to the PLC is established via a SIMATIC NET communication board (CP) in the computer to an integrated Ethernet interface of the CPU (if available), or to a corresponding CP module in the PLC.

Generally, no particular configuration and programming is required on the drive side. In order to subsequently configure the signals or symbols to be measured from the drives in *ibaPDA*, you will need to export a so-called STI file with the SIMOTION configuration software for each drive, e.g., Siemens SCOUT. *ibaPDA* needs the STI file to generate an address book with the signals for the drive.

When using the PC/CP connection mode, a suitable access point must be configured in the SIMATIC PG/PC interface of the *ibaPDA* computer.

Up to 32 drives can be connected per license. Extensions are possible.

### Interface settings

#### Set all values to zero if the connection to a drive...

If this option is enabled, all the measured values for a drive are set to zero as soon as the connection is lost.

#### Start acquisition even if a drive is not accessible

If this option is enabled, the acquisition will start even if a drive is not accessible. A warning is indicated in the validation dialog. If the system has been started without a connection to the drive, *ibaPDA* will try to connect to the drive at regular intervals

#### Allow inaccessible parameters

Enable this option to start the acquisition even if no drive parameters are accessible. Inaccessible parameters are then issued as alerts in the validation dialog rather than errors.

#### <Managing address books>

This button takes you to the dialog for creating and managing address books. A valid address book is required to configure measurement signals from the drives.

#### <Open log file>

If connections to drives have been established, all connection-specific actions are recorded in a text file. Using this button, you can open and see this file. In the file system on the hard drive, you will find the log files in the program path of the *ibaPDA* server (...\[ProgramData\iba\ibaPDA\Log\](#)). The file name of the current log file is [SIMOTIONLog.txt](#), the name of the archived log files is [SIMOTIONLog\\_yyyy\\_mm\\_dd\\_hh\\_mm\\_ss.txt](#).

#### <Reset counter>

Resets the error counter as well as the response times in the connection table to zero.

#### Connection table

The table shows the error counters as well as the response times of the individual connections during data measurement.

#### Available modules

- SIMOTION

#### Product name

ibaPDA-Interface-SIMOTION-Xplorer (Art. no. 31.000031)

The SIMOTION-Xplorer interface license is also included in the *ibaPDA-Drive-Xplorer* product (Art. no. 31.001044).

---

#### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-SIMOTION-Xplorer*.

## 8.9 SINUMERIK-Xplorer

### Description

The SINUMERIK-Xplorer interface is suitable for measurement data acquisition from the NCK part (Numerical Control Kernel) of Siemens SINUMERIK CNC controls via Ethernet using standard network cards as well as via Profibus using SIMATIC NET interface cards. Access is transparent for the SINUMERIK controller.

It is not necessary to specially configure or program it. The signals to be measured can be selected via the Sinumerik symbol browser in the I/O Manager of *ibaPDA*.

The connections to the SINUMERIK NCUs (Numerical Control Unit) can be established via standard interfaces of the computer or corresponding CP modules.

Additional Siemens software (e.g., SIMATIC NET or SIMATIC STEP 7) is needed for operation, in case the connection to the control system is established via a SIMATIC NET communication board (CP) in the computer to an integrated Ethernet interface of the CPU (if available) or to a CP module in the PLC.

Generally, no particular configuration and programming is required on the SINUMERIK side.

When using the PC/CP connection mode, a suitable access point must be configured in the SIMATIC PG/PC interface of the *ibaPDA* computer.

### Interface settings

#### **Set all values to zero when an NCK connection is lost.**

If this option is enabled, all the measured values of an NCK are set to zero as soon as the connection is lost.

#### **Start acquisition even if an NCK is not accessible.**

If this option is enabled, the acquisition will start even if an NCK is not accessible. A warning is indicated in the validation dialog. If the system was started without a connection to the NCK, *ibaPDA* will periodically attempt to connect to the NCK.

Enabling this option is recommended if several connections to a CPU have been configured. As the SINUMERIK CPU is not able to start multiple connections at the same time, this allows the acquisition to be started without problems.

#### **Allow inaccessible NCK variables**

Enable this option to start the acquisition even if NCK variables are not accessible or if they are configured with an incorrect size. The inaccessible variables are displayed as a warning in the validation dialog.

#### **Automatically restart ibaPDA server when NCK variable sizes automatically changed**

If this option is enabled, the *ibaPDA* server is restarted automatically if the NCK variable sizes have changed automatically.

#### **<Open log file>**

If connections to NCKs have been established, all connection-specific actions are logged in a text file. Using this button, you can open and see this file. In the file system on the hard drive, you will find the log files in the program path of the *ibaPDA* server (...\[ProgramData\iba\](#)

`ibaPDA\Log\`). The file name of the current log file is `SINUMERIKLog.txt`, the name of the archived log files is `SINUMERIKLog_yyyy_mm_dd_hh_mm_ss.txt`.

**<Reset counter>**

Resets the error counter as well as the response times in the connection table to zero.

**Connection table**

The table shows the error counters as well as the response times of the individual connections during data measurement.

**Available modules**

- SINUMERIK-Xplorer

**Product name**

ibaPDA-Interface-SINUMERIK-Xplorer (Art. no. 31.000033)

---

**Other documentation**

A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-SINUMERIK-Xplorer*.

---

## 9 Other manufacturer-specific interfaces

Name	Type	Connection to...	Comment	Link
DTBox-Request	Software license	DTBox	Via reflective memory or UDP	<a href="#">↗ DTBox-Request, page 230</a>
Hitachi MicroSigma	Software license	Hitachi $\mu\Sigma$ Network 1000-Network	special hardware IKS-LM-SN1G or -SN100 required	<a href="#">↗ Hitachi MicroSigma, page 244</a>
HPCi DGM200P	Software license	GE Energy Power Conversion HPCi, HPC LD2 (CC100, DGM 200)	DGM200P board (PCI) from GE Energy Power Conversion required	<a href="#">↗ DGM200P, page 233</a>
HPCi DGM200E	Software license	GE Energy Power Conversion HPCi, HPC LD2 (CC100, DGM 200)	DGM200E adapter from GE Energy Power Conversion required	<a href="#">↗ DGM200E, page 231</a>
HPCi Request	Software license	GE Energy Power Conversion HPCi	works with DGM200E, DGM200P, SM128V and Reflective Memory	<a href="#">↗ HPCi Request, page 244</a>
Modbus serial	S232 or similar	Modbus serial interface	Modbus Serial Slave or Modbus Serial Master	<a href="#">↗ Modbus serial, page 248</a>
PC Link	DCS-NET interface card for PCLink/Automax e.g. Reliance 5136-RE2-PCI	PCLink network		<a href="#">↗ PC Link, page 250</a>
Reflective Memory	VMIPCI 5565, 5576, 5579, 5588 etc.	VMIC counterparts, such as GE HPCi, SMS X-Pact		<a href="#">↗ Reflective memory, page 256</a>
ScramNet+	CWC SC150/150e	SCRAMNet+ network		<a href="#">↗ ScramNet+, page 257</a>
Toshiba ADMAP JAMI1	PCI card Toshiba ADMAP JAMI1	ADMAP-5M bus		<a href="#">↗ Toshiba ADMAP JAMI1, page 262</a>
XPact	Software license	SMS Siemag X-Pact	works with ibaLink-VME (SM128V) and Reflective Memory	<a href="#">↗ X-Pact, page 263</a>

---

Name	Type	Connection to...	Comment	Link
XPact Request	Software license	SMS Siemag X-Pact	works with ibaLink-VME (SM128V) and Reflective Mem- ory	<a href="#">↗ X-Pact Re- quest, page 266</a>

## 9.1 DTBox-Request

### Description

This is an interface between *ibaPDA* and the DTBox system from Dualis/Nidec. It provides *ibaPDA* with symbolic access to all signals defined in the DTBox system.

The DTBox is connected to *ibaPDA* for measured value acquisition via UDP or Reflective Memory. 4 time classes for the acquisition cycles are supported, which can be configured in *ibaPDA*.

Versions for 128, 1024 and an unlimited number of signals are available to choose from when it comes to the license.

An additional Ethernet TCP/IP connection between *ibaPDA* and DTBox is required for configuration. The *DTBox-Request* interface requires address books of the corresponding DTBox projects. The address books can be created by *ibaPDA*. A maximum of up to 1024 modules are supported per interface.

### Interface configuration (Configuration tab)

#### Active

Check this option if you want to use the interface.

#### <Manage address books>

Unless already done, you should create address books first. Clicking on this button opens the address book dialog with the “DTBox” tab. Clicking there on <Create address book> and enter the required information in the dialog “Import ISaGRAF Solution.” Then close the dialog by clicking <OK>.

#### Start acquisition even if a request agent is not accessible

If this option is enabled, the acquisition will start even if one or more request agents is/are not accessible. A warning is issued in the validation dialog. If the system has been started without a connection to request agents, then *ibaPDA* will try to connect to the missing request agents at regular intervals. As soon as the connection to one or more request agents can be established, the acquisition will automatically restart.

#### Allow inaccessible symbols

If this option is enabled and *ibaPDA* receives an error code for one or more variables from a DTBox request agent, then these variables will automatically be disabled and the acquisition will restart.

#### Resource change detection

At the start of the acquisition, *ibaPDA* checks the checksum of the DTBox that is transmitted by the DTBox with the watchdog telegram. The checksum is compared to the checksum saved in the address book. With the setting for the resource change detection, you determine how *ibaPDA* should respond if the checksums are no longer identical:

- Log alert
- Cancel validation and restart acquisition
- Reload address book during validation and restart acquisition.

**Available range of DTBox agent ID**

A DTBox resource offers 4 request agents with the IDs 0 to 3. It is possible to access a DTBox with several *ibaPDA* systems, but each system must use a different request agent, as otherwise there would be conflicts.

With the setting for the ID range, you determine which agents may be used by the *ibaPDA* system. The setting then applies to all resources that *ibaPDA* accesses.

Example: If there are 2 systems A and B, system A could use the IDs 0 and 1 and system B could use the IDs 2 and 3.

**Diagnostics tab**

Here you will find detailed information on the diagnostics of the connected DTBox systems.

**Available modules**

- DTBox Request

**Product name**

ibaPDA-Request-DTBox-128 (Art. no. 31.001380)

ibaPDA-Request-DTBox-1024 (Art. no. 31.001381)

ibaPDA-Request-DTBox-unlimited (Art. no. 31001382)

---

**Other documentation**

For a detailed description of this interface and its configuration, refer to the corresponding manual of the product *ibaPDA-Request-DTBox*.

---

## 9.2 DGM200E

**Description**

The data interface DGM200E is based on proprietary communication hardware from GE Energy Power Conversion (ex. Convertteam GmbH). The connection is only available for the systems HPC ("Logidyn D") and HPCi (P80i) when using the fast system bus DGM200. The DGM200E interface is a prerequisite for using the modules HPCi Lite and HPCi Request.

The following hardware components are required:

- at least one DGM200V module in a HPC/HPCi VME rack or one DGM200P board or a DGM 200-E adapter in/on an HPCi PC (RXi-042, RXi-142, APC 620, APC 810)
- in case of multiple HPC/HPCi racks or HPCi PCs, a DGM200C data concentrator should be installed
- at least one DGM 200-E adapter at the network interface of the *ibaPDA* computer
- the network interface must support 1000 Mbps and Jumbo Packets/Jumbo Frames.

Mixed operation of DGM200P and DGM 200-E in the *ibaPDA* computer is possible with some restrictions/limitations.

The DGM200 components have to be obtained exclusively from GE Energy Power Conversion (<http://www.gepowerconversion.com>).

### Available modules

When the DGM200E interface license is released in the dongle, you may configure the following modules in *ibaPDA*:

- HPCi Lite
  - Access on all data configured on the DGM200 bus by the CCM.
  - Access to data of all time classes of the DGM200 bus
  - max. 1000 analog values + 1000 digital values per module
  - Comfortable data selection via signal browser in the *ibaPDA* I/O Manager (`toc.ini` required, created by P80i address book generator)

---

### Note



One *ibaPDA* computer supports up to four DGM 200-E channels, provided that there are enough network adapters in the *ibaPDA* computer. One DGM 200-E channel corresponds to one DGM 200 network. The DGM 200-E device offers two separate channels.

- 
- DGM200E
    - Access to data from the DGM200-E adapter via physical addressing (offset / data type)
    - max. 1000 analog values + 1000 digital values per module
    - Applies to special cases
  - DGM200E dig512
    - Access to data from the DGM200-E adapter via physical addressing (offset / data type)
    - max. 32 x 16 = 512 bits per module, no analog values
    - Applies to special cases

A maximum of up to 1024 modules are supported per data interface.

Total number of signals to be measured is only limited by the *ibaPDA* license and performance of the involved systems.

### Product name

ibaPDA-Interface-HPCI-DGM200E (Art. no. 31.001009)

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-HPCI-DGM200E*.

### 9.3 DGM200P

The data interface DGM200P is based on a proprietary communication hardware from GE Energy Power Conversion (ex. Converteam GmbH). The connection is only available for the systems HPC ("Logidyn D") and HPCi (P80i) when using the quick system bus DGM200. The DGM200P interface is a prerequisite for using the modules HPCi Lite and HPCi Request.

The following hardware components are required:

- at least one DGM200V board in an HPC/HPCi rack or one DGM200P board in an HPCi PC
- in case of multiple HPC/HPCi racks or HPCi PCs, a DGM200C data concentrator should be installed
- at least one DGM200P PCI board in the *ibaPDA* PC

The DGM200 components have to be obtained exclusively from GE Energy Power Conversion (<http://www.gepowerconversion.com>).

#### Available modules

When the DGM200P interface license is released in the dongle, you may configure the following modules in *ibaPDA*:

- HPCi Lite
  - Access on all data configured on the DGM200 bus by the CCM.
  - Access on data of all time classes of the DGM200 bus
  - max. 1000 analog values + 1000 digital values per module
  - Comfortable data selection via signal browser in the *ibaPDA* I/O Manager (`toc.ini` required, created by P80i address book generator)

---

#### Note



Up to two DGM200P cards are supported in an *ibaPDA* PC. With HPCi Lite, you can use different address books (one per DGM200P card). Thus, access on separate DGM200 networks is provided.

---

- DGM200P
  - Access on data of the DGM200P memory by physical addressing (offset / data type)
  - max. 1000 analog values + 1000 digital values per module
  - Applies to special cases
- DGM200P dig512
  - Access on data of the DGM200P memory by physical addressing (offset / data type)
  - max. 32 x 16 = 512 bits per module, no analog values
  - Applies to special cases

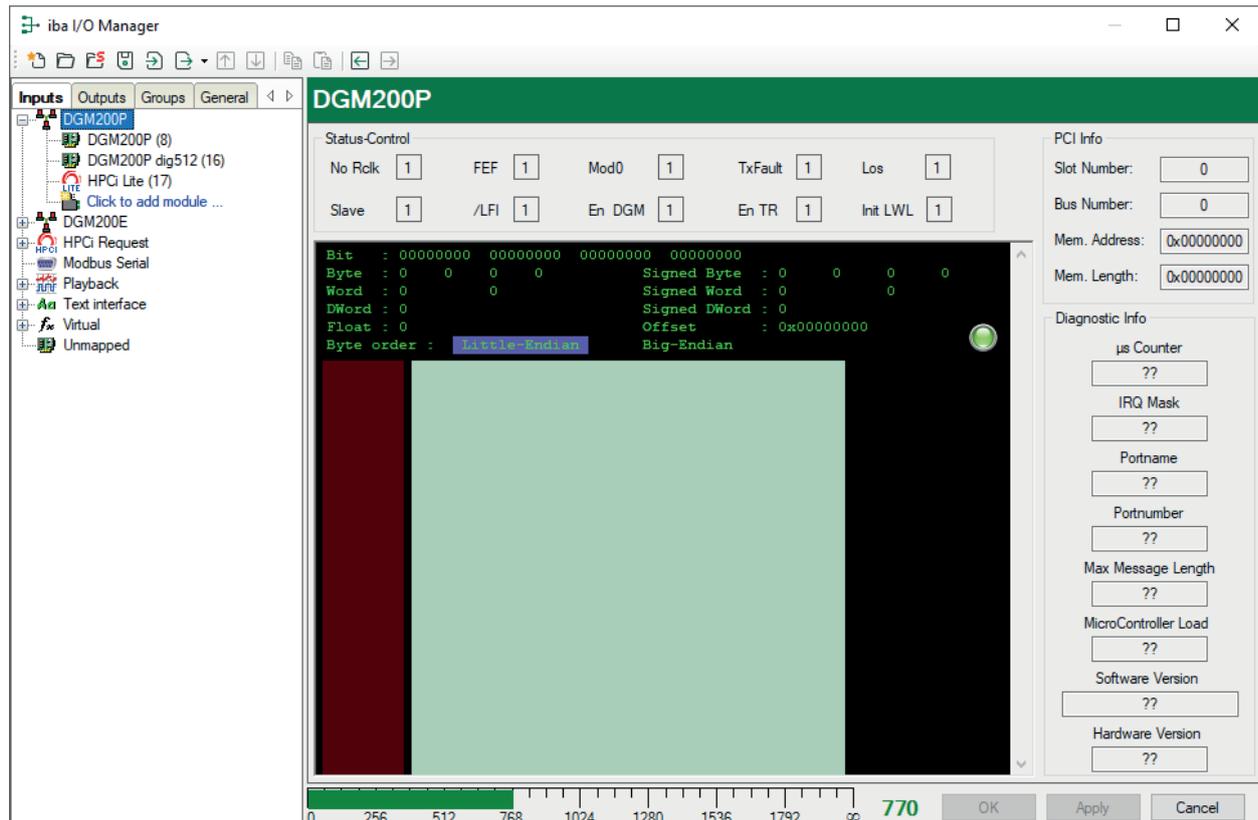
A maximum of up to 1024 modules are supported per interface.

Total number of signals to be measured is only limited by the *ibaPDA* license and performance of the involved systems.

For further information on module configuration, see:

- ↗ *HPCi Lite module*, page 240
- ↗ *DGM200P module*, page 236
- ↗ *DGM200P dig512 module*, page 239

### 9.3.1 DGM200P – Board information



#### Status control

The status of ten tabs of the DGM200P board is displayed here.

- NO\_RCLK (r)  
1 = no receiving clock is detected on the fiber optic input port.
- SLAVE (r)  
1 = the card is configured as "slave." The jumper "master" is removed.
- FEF (r)  
1 = all data has been copied from the input buffer to the DPR.

- /LFI (r)  
0 = link fault indication, one of the following conditions applies:
  - receive frequency out of range
  - receive amplitude too small
  - no flow change in input difference signal for at least 60 cycles
  - receiver is blocked
- MOD0 (r)  
0 = fiber-optical transceiver block is available.
- TXFAULT (r)  
1 = transmitting laser diode is deactivated.
- LOS (r)  
1 = loss of signal, no light at receiver
- INIT\_LWL (rw)  
A rising edge starts the initialization of the fiber optic link.  
**NOTE: The normal data exchange will be interrupted!**  
Force to zero in normal operation.
- EN\_DGM (r)  
1 = m-controller on the card has released transmission/receiving logical control.
- EN\_TR (rw)  
1 = release of transmitter

### PCI Info

In the PCI Info area of the dialog, you find the following information:

- Slot Number  
Number of the PCI slot where the card is plugged in.
- Bus Number  
PCI bus connected with this slot
- Mem. Address  
Start address of the memory region (hex)
- Memory Length  
Size of the memory region (hex), free / reserved

### Diagnosis Info

In the diagnosis info area, you will find information helpful for service and support.

- ms counter  
a running microseconds counter
- IRQ mask  
Interrupt mask / interrupt status (for further information, refer to the card documentation of the manufacturer)

- Port name  
Includes the station name. It is only valid after the connection to the data concentrator has been established.
- Port number  
Includes the number of the port at the concentrator to which the station is connected via fiber optic cable. It is only valid after the connection to the data concentrator has been established.
- Max. telegram length  
After the configuration has been attuned to the concentrator, you will find here the max. length of the telegrams sent by this port. The length is limited to 4096 (incl. overhead). If the microcontroller computes a greater max. length, no message will be send. The message length is only valid after the connection to the data concentrator has been established.
- CPU load  
Time in microseconds between falling edge of the SYNC pulse and termination of all tasks.
- Software version  
Contains software version as ASCII text
- Hardware version  
Contains hardware version (four BCD characters)

### Memory view

This view provides more detailed information about the message traffic for our support personnel.

## 9.3.2 DGM200P module

The DGM200P module type is used for the acquisition of up to 1000 analog and 1000 digital signals over a DGM200 connection.

The module size, i.e. the number of signals, can be specified. Default setting is 32 analog and 32 digital signals. If more signals are required, they can either be added to the module or another module can be added.

Similar to the Reflective Memory connection, this module type uses physical memory addresses (offsets), which have to be configured by the user. The data to be measured should be configured correspondingly on side of the HPCi -/ DGM200.

The use of module type DGM200P is therefore rather exceptional, e.g. when communicating over DGM200 without the HPCi control system.

### 9.3.2.1 DGM200P – General tab

#### Basic settings

For a description of the basic settings see [↗ Common and general module settings, page 20.](#)

## Advanced

### Swap mode

Set the swap mode according to the signal source.  
You can choose between the following 4 options:

Mode	16 bit	32 bit
No swap	AB	ABCD
Depending on data type	BA	DCBA
Swap 16 bit	AB	CDAB
Swap 8 bit	BA	BADC

The swap mode to be selected depends on the swap mode of the signal source.

### Asynchronous mode

In asynchronous mode, the data is copied from the card's memory into the memory of *ibaPDA* outside of the interrupt service routine (ISR). This mode can be used to measure large data volumes on a slower time base.

Enabling asynchronous mode is recommended if the duration of the ISR is longer than 2000  $\mu$ s. For checking the ISR duration, go to *Interrupt time* on the *Interrupt info* node of the *General* tab in the I/O Manager.

If you want to activate the asynchronous mode, set this option on TRUE.

## Module layout

### Number of analog and digital signals

Here, you can increase or decrease the number of signals in the module. By default, 32 signals are preset. You may enter any value between 0 and 1000. The signal tables will be adjusted accordingly.

## 9.3.2.2 DGM200P – Analog tab

### General columns in the signal table

For a description of the general signal table columns see [Columns in tables with analog and digital signals](#), page 22.

### Address

In this column, you should specify the offset of the first byte of the value within the raw data stream. The offset can be entered as hexadecimal or decimal values by selecting the appropriate option from the context menu. For a default allocation of the addresses within the column, just click on the column header. Based on the address offset and starting with the value in the first row or in the field the cursor is currently placed in, the address values are filled in automatically in accordance with the selected data types.

### Data type

In the fields of this column, you can select the relevant data type used for each signal. Click in the corresponding field and select the data type from the drop-down list. The address space depends on the data type. Therefore, an adjustment of address entries might be necessary after changing the data types.

Available data types:

Data type	Description	Value range:
BYTE	8 bit without positive or negative sign	0 ... 255
INT	16 bit with positive or negative sign	-32768 ... 32767
WORD	16 bit without positive or negative sign	0 ... 65535
DINT	32 bit with positive or negative sign	-2147483647 ... 2147483647
DWORD	32 bit without positive or negative sign	0 ... 4294967295
FLOAT	IEEE754; single precision; 32 bit floating point	$1.175 \cdot 10^{-38} \dots 3.403 \cdot 10^{38}$
DOUBLE	IEEE754; double precision; 64 bit floating point;	2.225E-308 ... 1.798E+308
FP_REAL	Fixed point real; Q15.16; 15 integer bits and 16 fractional bits;	-32768 ... 32767.9999

#### Note



It is recommended to configure the data to be transmitted in consecutive memory ranges, i.e. the signals should have consecutive addresses. Otherwise, the performance might decrease considerably.

### 9.3.2.3 DGM200P – Digital tab

#### General columns in the signal table

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals](#), page 22.

As for digital signals, it is possible to read 16 single bits out of an INTEGER (WORD) or 32 single bits out of a DINT (DWORD).

#### Address

In this column, you should specify the offset of the first byte of the data-carrying binary signal within the raw data stream. The offset can be entered as hexadecimal or decimal values by selecting the appropriate option from the context menu. For a default allocation of the addresses within the column, just click on the column header. Based on the address offset and starting with the value in the first row or in the field the cursor is currently placed in, the address values are filled in automatically according to the selected data types.

#### Bit no.

This number (0...15 or 0...31) specifies the position of the desired digital signal in a 16-bit or 32-bit block in the data stream with regard to the related offset address. Increase of bit no. by 1 up to 15 (31), then increase of address by 2 (4).

#### Note



It is recommended to configure the data to be transmitted in consecutive memory ranges, i.e. the signals should have consecutive addresses. Otherwise, the performance might decrease considerably.

### 9.3.3 DGM200P dig512 module

The DGM200P dig512 module type is used for the acquisition of up to 512 digital signals over a DGM200 connection, with the digital signals being packed in 32 16-bit integer signals.

This module type uses physical memory addresses (offsets), which have to be configured by the user. The data to be measured should be configured correspondingly on side of the HPCi -/ DGM200.

The use of module type DGM200P dig512 is therefore rather exceptional, e.g. when communicating over DGM200 without the HPCi control system.

#### 9.3.3.1 DGM200P dig512 – General tab

##### Basic settings

For a description of the basic settings see ↗ *Common and general module settings*, page 20.

#### 9.3.3.2 DGM200P dig512 – Digital tab

The signal tables for modules with dig512 format consist of two levels.

The first level shows the so-called connectors and activation attributes.

If you click on the small plus symbols in the table rows, the second level of the signal table opens and you can see the actual signals (16 per connector).

##### Connector level

##### Connector

Following the principle of the former ibaDig512 device, the separate data packages are referred to as connectors. One connector corresponds to an integer data item with 16 bits.

You can extend or hide the signal tables for each connector by click on the small + (resp. -) symbol at the connector name. You may also enter a name for the connector in the "Connector" column. This name is used for engineering assignment. Under each connector, 16 digital signals are grouped on the second level of the signal table.

##### Address

In the address column (connector line), the byte offset in the range of every single connector (= integer package) may be specified by the user. The offset can be entered as hexadecimal or decimal values by selecting the appropriate option from the context menu. Usually, the default values need to be adjusted. Consecutive addresses count in steps of 2 according to the size of the 16-bit integer package.

After you have entered the address in the first row and clicked on the column header, all following addresses are updated automatically.

##### Active

Activating the connectors

A click on the "Active" column header activates (checkmark) or deactivated (no checkmark) all the connectors for acquisition at the same time. Individual connectors can be activated in the corresponding selection box. No acquisition takes place for deactivated connectors, so that such

connectors are neither available for display nor storage. When enabling/disabling a connector's activation attribute in the parent table, all the channels it contains are enabled/disabled.

If you want to activate/deactivate the signals individually, go to the second level. If enabling of the signals of a connector is not uniform, the "Activation" selection box of the connector is grayed out.

Furthermore, disabled signals will not be taken into account in the signal statistics ("signal-o-meter").

### 9.3.4 HPCi Lite module

The HPCi Lite module type is used, in particular, for the acquisition of up to 1000 analog and 1000 digital signals from a GE HPCi system over DGM 200.

This module type can also be used in combination with an HPC ("Logidyn D2").

The HPCi Lite module type is included in the license for the DGM200P and DGM200E interface and supports a convenient signal selection by browser in the *ibaPDA* I/O Manager.

The selection of signals is limited to the signals configured in the HPC/HPCi for communication over the DGM 200 network ("CC100 signals"). A request function with access on all variables in the HPCi system is therefore not possible.

Preconditions for using this module type are a connection to the HPC/HPCi system over DGM 200-V, DGM 200-C, DGM 200-P or DGM 200-E as well as a data configuration for the CC100 bus with the CCM32 (Coordination Channel Manager) by GE Energy Power Conversion.

#### 9.3.4.1 HPCi Lite – General tab

##### Basic settings

For a description of the basic settings see [➤ Common and general module settings](#), page 20.

##### Advanced

##### Asynchronous mode

In asynchronous mode, the data is copied from the card's memory into the memory of *ibaPDA* outside of the interrupt service routine (ISR). This mode can be used to measure large data volumes on a slower time base.

Enabling asynchronous mode is recommended if the duration of the ISR is longer than 2000  $\mu$ s. For checking the ISR duration, go to *Interrupt time* on the *Interrupt info* node of the *General* tab in the I/O Manager.

If you want to activate the asynchronous mode, set this option on TRUE.

##### HPCi

##### Time class

Here, select the time class (from 1 to 4) which applies to the refresh rate of the signal to be measured on the DGM200. A module can only be assigned to one time class. So, only signals configured in the same time class on the DGM200 can be measured in one module.

## Module layout

### Number of analog and digital signals

Here, you can increase or decrease the number of signals in the module. By default, 32 signals are preset. You may enter any value between 0 and 1000. The signal tables will be adjusted accordingly.

### Hyperlink

#### Select HPCi symbols (Link)

A click on this link will open the browser for selecting the signals to be measured from the DGM200 address book. You have access to all symbols of all HPCi stations.

The selected signals will be entered automatically in the appropriate signal table of the module (next available free row).

So you can select several signals one after the other without the browser automatically closing. The browser will stay open until you press OK.

You also may open the browser directly from the signal tables ("HPCi symbol" column).

#### Select digital HPCi symbols (link)

Same function as with analog symbols.

## 9.3.4.2 HPCi Lite – Analog tab

### General columns in the signal table

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals](#), page 22.

### HPCi symbol

This column has the symbolic name of the signal, as it was configured in the HPCi system.

You may enter the symbol name manually. However, it is recommended using the signal browser, being easier and safer.

In compliance with the naming rules in the HPCi system, the full symbol path is indicated.

Two different methods are available:

a) via the link in the *General* tab of the module. With this method, the signals selected from the signal browser will be put into the next free row of the signal table. This is helpful if you fill the signal table for the first time or if you want to fill up a partially filled signal table.

b) via the small browser button  in the HPCi symbol field of the desired signal. With this method, you determine exactly the position where a symbol is to be entered in the table.

**Tip**

When using the modules HPCi Lite (also HPCi Request), even the comments are taken from the HPCi system, as long as they were configured there.

When selecting a signal from the signal browser, the HPCi symbol name is entered into the "Name" column. You may change the name manually afterwards.

Moreover, the *Update signal names with comments from address books* command in the context menu allows you to use "Comment 1" as signal name. This command will swap the positions of the current signal name and Comment 1.

You may also use the context menu command *Update signal names with symbols from address books*. This will enter the HPCi symbol in the name field and the HPCi signal comment 1 in the "Comment 1" field.

**9.3.4.3 HPCi Lite– Digital tab****Note**

In the DGM200 system there is no direct support for digital signals. Normally 32 digital signals are packed into one DINT that is put on the DGM 200. So when you want to select digital signals in the browser you have to select DINT values. *ibaPDA* adds 32 digital signals for each DINT value you select by using the link in the *General* tab of the module.

**General columns in the signal table**

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals](#), page 22.

**HPCi symbol**

This column has the symbolic name of the signal, as it was configured in the HPCi system.

You may enter the symbol name manually. However, it is recommended using the signal browser, being easier and safer.

In compliance with the naming rules in the HPCi system, the full symbol path is indicated.

Two different methods are available:

a) via the link in the *General* tab of the module. With this method, the signals selected from the signal browser will be put into the next free row of the signal table. This is useful if you fill the signal table for the first time or if you want to fill up a partially filled signal table.

b) Via the small browser button  in the HPCi symbol field of the desired signal. With this method, you determine exactly the position where a symbol is entered in the table.

**Bit no.**

The default allocation is 0, because "real" binary signals are initially assumed (e.g. flags).

In the case of packed bits, the number (bit index 0...31) can be entered here within the HPCi symbol variable. Applies to data formats INT, WORD, DINT, DWORD.

You can enter the value directly or set it using the arrow keys in the field.

### Tip



When using the modules HPCi Lite (also HPCi Request), even the comments are taken from the HPCi system, as long as they were configured there.

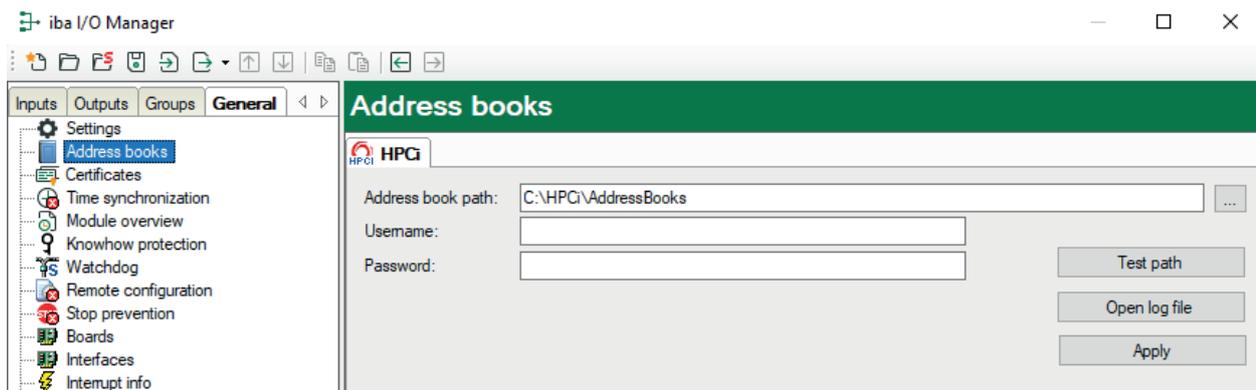
When selecting a signal from the signal browser, the HPCi symbol name is entered into the "Name" column. You may change the name manually afterwards.

Moreover, the *Update signal names with comments from address books* command in the context menu allows you to use "Comment 1" as signal name. This command will swap the positions of the current signal name and Comment 1.

You may also use the context menu command *Update signal names with symbols from address books*. This will enter the HPCi symbol in the name field and the HPCi signal comment 1 in the "Comment 1" field.

## 9.3.5 HPCi Lite

If the DGM200P or DGM200E option (each without HPCi request) is unlocked in the dongle, the *HPCi Lite* tab in the *Address book* node in the *General* register is displayed. If you are using an *HPCi Request* license, this tab is not visible.



Important information for the HPCi Lite function should be entered here.

### Addressbook path

Enter the complete path name where the `toc.ini` configuration file is stored. You can get a sample file of `toc.ini` from iba or from GE Energy Power Conversion (ex. Convertteam). The program CCM32 (Coordination Channel Manager), version 2.17a or higher, reads and supplements this file.

The actual address book will be generated by the CCM in the same directory.

### User name and password

If the `toc.ini` file is not stored on the local drive of the *ibaPDA* computer but on a network drive, then here you have to enter the credentials (user name and password) for the remote computer. This user must be registered on the remote computer with appropriate rights for reading data.

**<Test path>**

Click on this button in order to check the accessibility of the specified path.

**<Open log file>**

Mouse click this button in order to open a system log file which shows all relevant activities and events regarding the connection to the HPCi systems.

**<Apply>**

If changes were made in this dialog, they must be applied via this button. The changes will be applied and the driver restarted.

## 9.4 Hitachi MicroSigma

**Description**

The *Hitachi MicroSigma* interface is required to connect *ibaPDA* to an *IKS-LM-SN1G* or *IKS-LM-SN100* device, which in turn is a participant in a  $\mu\Sigma$ NETWORK-1000 network. These devices are distributed exclusively by iba Korea.

**Interface configuration**

In the Hitachi MicroSigma interface dialog, you can configure the connection to an *IKS-LN-SNx*-type device and select the address book for the signals to be acquired.

Usable modules are automatically created under the interface.

**Available modules**

- $\mu\Sigma$  UDP data modules
- $\mu\Sigma$  Diagnostic module

**Product name**

ibaPDA-Interface-Hitachi-MicroSigma (Art. No. 31.001100)

---

**Other documentation**

A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-Hitachi-MicroSigma*.

---

## 9.5 HPCi Request

The data interface HPCi Request is based on the licensed DGM200 communication hardware from GE Energy Power Conversion (ex. Converteam GmbH). HPCi Request provides comfortable access on all data in all HPCi stations connected to the DGM200 bus for measurement. This interface connection is available only for HPCi (P80i) systems for communication with the fast DGM200 bus.

Activation of DGM200P or DGM200E and HPCi Request licenses in the dongle as well as a DGM200P interface card or a DGM200E adapter installed in the *ibaPDA* computer are required for using the HPCi Request interface.

The hardware requirements are the same as for DGM200P or DGM200E (HPCi Lite); additionally a TCP/IP connection to all involved HPCi CPUs is required.

If the HPCi Request interface license is activated in the dongle, you may configure the following modules in *ibaPDA*:

- HPCi Request
  - Access to all data of all connected HPCi stations on request (incl. KK'S)
  - Access to data of all time classes of the DGM200 bus
  - max. 1000 analog values + 1000 digital values per module
  - Comfortable data selection via signal browser in the *ibaPDA* I/O Manager ([toc.ini](#) required, created by P80i address book generator)
  - Comfortable data selection via drag & drop from P80i to *ibaPDA* (with the *ibaPDA* client on the same PC as P80i).

A maximum of up to 1024 modules are supported per interface.

Total number of signals to be measured is only limited by the *ibaPDA* license and performance of the involved systems.

HPCi request is also supported for different physical interfaces:

- ibaLink-SM-128V-i-2o / ibaLink-VME (data channel via ibaFOB cards)
- Reflective Memory (data channel via PCI-boards VMIPCI 5565, 5576, 5579, 5587)

### Functional principle

A special tool in the HPCi programming software, the address book generator, creates a reference file, which consists of system configuration and the available signals on the DGM200 and in the HPCi stations. After selection of the signals to be measured in the I/O Manager of *ibaPDA* (via browser or drag & drop), request messages are sent to the corresponding HPCi stations over TCP/IP. In the HPCi stations, special service routines, so called agents, respond to the requests by assigning the requested signals to the DGM200 bus for measurement.

---

### Other documentation



Additional information about this interface and the configuration of the HPCi system can be found in the manual of the product *ibaPDA-Request-HPCi*.

---

## 9.5.1 HPCi Request – Overview

On the one hand, this tab provides status information, on the other hand, the user can enter configuration data.

### Active

Check this option if you want to use the interface.

### Multicast IP address and Port no.

Display of the multicast IP address and port number as configured in the configuration file (`toc.ini`). The address and port no. should be assigned to the *ibaPDA* system in the HPCi / DGM200 system configuration.

*ibaPDA* sends its request messages to the HPCi stations by multicast via this IP address and port number.

### Address book path

Enter the complete path name where the `toc.ini` configuration file is stored.

The file `toc.ini` is created by the P80i address book generator or edited manually.

### User name and password

If the file `toc.ini` is not stored on the local hard disk of the *ibaPDA*-PC but on a network drive, it might be necessary to enter here the user name and password of a user account, which can be used by the *ibaPDA*-PC for login on the remote computer. This user account should be sufficiently authorized for data access on the remote computer.

### <Test path>

Press this button in order to check the accessibility of the specified path.

### Disable signals of non-responding CPUs

At start of the measurement, all CPUs (in the HPCi stations) are polled. If a CPU is not responsive, the related signals will be disabled and the acquisition will start without these signals, provided this option is enabled.

The use of this option is recommended during commissioning or maintenance works, if some HPCi stations are unavailable.

If this option is not enabled, the measurement will not start until all CPUs have replied to the polling at start of measurement.

**HPCi CPU response timeout [s]**

This setting specifies the waiting time of idle communication until a CPU is considered as "non-responding". You may adjust this value to your requirements.

**Enable CPU reconnect detection**

If this option is enabled, missing CPUs will be checked also during the measurement. If a missing CPU is available again or restarts, the measurement will be stopped, a new request will be sent and the measurement will be restarted.

**Connect to CPUs without waiting for a multicast message**

When this option is enabled, *ibaPDA* will establish a unicast connection to each CPU, based on the IP address in the `toc.ini` file, as soon as possible. The option is disabled by default.

Generally, when starting the acquisition, *ibaPDA* waits for multicast messages sent by the CPUs before establishing the connection. In some cases the multicast messages can be delayed significantly due to network problems, particularly in networks with routers or switches using IGMP Snooping.

**Use asynchronous mode for timeclasses**

The asynchronous mode setting for the time classes determines when the driver of *ibaPDA* will copy data from the boards. If asynchronous mode is off then the data is copied during the interrupt service routine. If asynchronous mode is on then the data is copied on a separate thread outside of the interrupt service routine. Normally asynchronous mode should be off. Asynchronous mode is only needed when the interrupt service routine takes more than 1000  $\mu$ s to copy all the data from the boards. You can check this by going to the *General* tab and the *Interrupt info* node.

**<Open log file>**

Mouse click this button in order to open a system log file which shows all relevant activities and events regarding the connection to the HPCi systems.

**<Apply and restart>**

If changes were made in this dialog, confirm them with this button. The changes will be applied and the driver restarted.

All detected HPCi stations, i. e. those registered in the address book file, will be represented by colored blocks in the lower part of the dialog box.

The colors refer to the following states:

- green = control path (TCP/IP connection) and data path (DGM200) OK.
- yellow = control path (TCP/IP connection) OK, data path (DGM200) not OK.
- red = not connected with station, e. g. when station is switched off

## 9.5.2 HPCi Request – Diagnostics

The *Diagnostics* tab provides additional information about the HPCi system.

In the left part of the dialog box, you find a tree structure with HPCi stations and CPUs.

Click on an HPCi station in the tree to see a graphical representation of the rack configuration in the right part of the dialog. Type and slot position of the configured cards are shown here.

Select a CPU symbol in the tree to see some status information about the CPU, the TCP/IP connection and data connections.

### CPU Info

A proper function is indicated by the term "loaded" on green background in the *Symbols* field.

### TCP/IP Info

A proper function is indicated by the term "connected" on green background in the *Status* field.

### Data interfaces

This table shows the available data channels and the related memory offsets. In terms of DGM200, these are for example the time classes (TC) 1 to 4.

## 9.6 Modbus serial

### Description

This interface should be used when *ibaPDA* is connected to a Modbus network via a serial interface (COM port).

*ibaPDA* supports the Modbus modes master and slave as well as RTU and ASCII.

### Interface configuration

First, you should add a Modbus COM port (serial interface) . Select the *Modbus Serial* node in the interface tree and click on the <Add Modbus COM port> button.



Then, set up the COM port properties and click on the <Apply> button.

 A screenshot of a software window titled "Modbus Serial COM1 (Slave)" with a green header. The window is divided into two main sections. The top section, "COM port properties", includes:
 

- A checked "Active" checkbox.
- A "COM port status" field showing "Connected" in a green box.
- COM port: "COM1" (dropdown)
- Baudrate: "19200" (dropdown)
- Parity: "Even" (dropdown)
- Data bits: "8" (dropdown)
- Stop bits: "1" (dropdown)
- RTU framing: "Auto" (radio button selected), "Manual" (radio button), and a "1" ms spinner.
- Modbus mode: "RTU" (radio button selected), "Master" (radio button), "ASCII" (radio button), and "Slave" (radio button).
- An "Apply" button.

 The bottom section, "Modbus slaves", contains four counters:
 

- Received messages: 0
- Sent messages: 0
- Incomplete messages: 0
- CRC errors: 1

 Below these counters is a grid of 50 numbered buttons, arranged in two rows of 25, representing slave addresses from 1 to 50.

- Active  
Check this option if you want to use the interface.
- COM port  
Select a COM port from the drop-down list.

- Baud rate, parity, data bits and stop bits  
Set up these COM port parameters in compliance with the Modbus network.
- Modbus mode  
Choose whether *ibaPDA* should work in RTU or ASCII mode or as master or slave.
- RTU framing  
Choose whether RTU framing should be done automatically (recommended) or manually. If you choose manually, specify an appropriate framing time.

### Status display for Modbus slaves

Once the <Apply> button in the properties area has been clicked, *ibaPDA* starts working on the COM port.

- When running in slave mode, *ibaPDA* monitors the COM port for messages.
- When running in master mode, *ibaPDA* sends requests to the configured Modbus slaves at regular intervals.

If active Modbus slaves are connected, the status of the possible 247 Modbus devices is indicated by different colors.

Status	Master mode	Slave mode
Connected (green)	The slave responds to the periodic requests from <i>ibaPDA</i> .	The master is sending periodic requests to <i>ibaPDA</i> .
Disconnected (red)	The slave does not respond to periodic requests from <i>ibaPDA</i> .	The master does not send any requests to this slave.
Disabled (gray)	This slave is not configured.	This slave is not configured.

### Available modules

- Modbus: slave

### Product name

ibaPDA-Interface-Modbus-Serial (Art. no. 31.001021)

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-Modbus-Serial*.

## 9.7 PC Link

**PC Link**

Configuration
 Drops
 Memory view

PC Link configuration

In use  
 Firmware:   
 Passive mode  
 Active mode    Drop number:     Drop depth:

Card ID:   
 Firmware status: Not loaded  
 Board mode:

Drop status bits

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55																

Network diagnostics

Messages received:	<input type="text" value="?"/>	Messages transmitted:	<input type="text" value="?"/>
Receive timeouts:	<input type="text" value="?"/>	Overrun errors:	<input type="text" value="?"/>
CRC errors:	<input type="text" value="?"/>	Abort errors:	<input type="text" value="?"/>

### Description

*ibaPDA-Interface-PC Link* is a software interface for data acquisition on a DCS network as used by Reliance Automax PC Link.

Provided there is a DCS network card inserted in the *ibaPDA* PC, the *ibaPDA* software enables the measuring of data sent over the DCS network.

The main properties of the software interface are:

- Automatic detection of the DCS network interface card installed
- Up to 4 cards are supported per PC.
- Up to 1024 PC Link modules can be configured.
- 3 different module types for data acquisition
- The "Asynchronous" mode is supported on the PC Link module.
- Both the "Active" and "Passive" mode are supported.
- Signal browser

A Reliance Automax DCS Network consists of up to 56 drops or nodes. Each drop has 64K 16bit registers. Drop 0 functions as master and contains information on the network, such as which drops are active, broadcast data, etc. PC Link provides a connection to a DCS Network for an IBM AT compatible standard computer.

## Interface configuration (Configuration tab)

### "In use"

Check this option if you wish this interface to be used.

### Firmware

Firmware file to be selected and loaded. The PC Link interface card can run as a master or a slave (drop), depending on the firmware file loaded. The firmware files should be located in the program path of the *ibaPDA* server. If this is not the case then it should be installed using the CD ROM from the card's manufacturer (SST). The file can then be found under [SST\...\Reliance\Modules](#).

- For slave mode (drop), select [renet.ss3](#) from the list.
- For master mode, select [renetm.ss3](#).

### Passive mode

This mode is available if the PC Link interface card runs as slave (drop). If the Passive mode is enabled, *ibaPDA* is not "visible" on the bus to other connections or masters. The bus is only monitored.

### Active mode

This mode is available if the PC Link interface card runs as master or slave (drop). If Active mode is enabled, multiple drops can be mapped on the PC Link board. These drops are then "visible" on the bus for other participants and can be addressed for message transmission.

- Drop number and drop depth  
Enter the number of the first drop to which a PC Link card is allocated here. Enter the total number of drops according to the number of the following drops, starting with the previously entered Drop number.

### <Identify board>

By click on this button, you can identify and initialize the card.

### <Load firmware>

This button allows you to load the selected firmware file on the card.

### <Load address book>

Clicking this button, allows you to load the address book of the DCS network. This will open a file browser where you can select the correct database file. By default, this file is called [\\$NET.dbf](#). The address book is required for PC Link symbolic modules when using the DCS symbol browser.

### Drop status bits

There is a status indicator in this area for each drop in a DCS network.

When clicking on a drop's status field, the display will switch to the *Drops* tab, showing the 64 values of the corresponding drop.

### Drop diagnostics (Drops tab)

On this tab you'll find the contents of the 64 values of each drop. Enter the drop number in the input field in the upper left corner and choose between "Decimal" and "Binary" display mode.

A status indicator in the upper right corner shows the drop status.

### Available modules

- PC Link; 0 to 1000 analog signals and 0 to 1000 digital signals (default 32/32)
- PC Link Dig512; up to 512 digital signals, in 32 groups of 16 signals each
- PC Link Symbolic; 0 to 1000 analog signals and 0 to 1000 digital signals (default 32/32), featuring a DCS symbol browser for easier signal selection, based on the DCS address book.

For further information on module configuration see:

- [↗ PC Link module, page 252](#)
- [↗ PC Link Dig512 module type, page 253](#)
- [↗ PC Link Symbolic module, page 254](#)

### Product name

ibaPDA-Interface-PCLink-Automax

---

#### Other documentation



For a detailed description of this interface and its configuration, please refer to the corresponding manual of the product *ibaPDA-Interface-PCLink-Automax*.

---

## 9.7.1 PC Link module

The PC Link module type should be used if no address book of the DCS network is available.

### 9.7.1.1 PC Link – General tab

#### Basic settings

For a description of the basic settings see [↗ Common and general module settings, page 20](#).

#### Advanced

##### Asynchronous mode

In asynchronous mode, the data is copied from the card's memory into the memory of *ibaPDA* outside of the interrupt service routine (ISR). This mode can be used to measure large data volumes on a slower time base.

Enabling asynchronous mode is recommended if the duration of the ISR is longer than 2000  $\mu$ s. For checking the ISR duration, go to *Interrupt time* on the *Interrupt info* node of the *General* register in the I/O Manager.

If you want to activate the asynchronous mode, set this option on TRUE.

## Module layout

### Number of analog and digital signals

Here, you can increase or decrease the number of signals in the module. By default, 32 signals are preset. You may enter any value between 0 and 1000. The signal tables will be adjusted accordingly.

### 9.7.1.2 PC Link – Analog and Digital tab

#### General columns in the signal table

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals](#), page 22.

#### Drop number

Enter or dial the drop number.

#### Register

Enter the register number here. A register equals one 16-bit integer.

#### On digital tab only:

#### Bit number

Enter or dial the bit number corresponding to the digital signal within a register. Value range reaches from 0 to 15 per register.

## 9.7.2 PC Link Dig512 module type

The PC Link Dig512 module is a special module for measurement of 512 digital signals arranged in 32 groups (connectors) of 16 signals each.

### 9.7.2.1 PC Link Dig512 – General tab

#### Basic settings

For a description of the basic settings see [↗ Common and general module settings](#), page 20.

### 9.7.2.2 PC Link – Digital tab

The signal tables for modules with dig512 format consist of two levels.

The first level shows the so-called connectors and activation attributes.

If you click on the small plus symbols in the table rows, the second level of the signal table opens and you can see the actual signals (16 per connector).

#### General columns in the signal table

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals](#), page 22.

## Connector level

### Connector

Following the principle of the former ibaDig512 device, the separate data packages are referred to as connectors. One connector corresponds to an integer data item with 16 bits.

With a mouse click on the small + (or -) symbol at the connector name you can extend or hide the signal tables for each connector. You may also enter a name for the connector in the "Connector" column. This name is used for engineering assignment. Under each connector, 16 digital signals are grouped on the second level of the signal table.

### Drop number and register

Enter or dial the drop number and the register.

If you enter the drop number in the first row and then click on the column header, all following rows will be adjusted automatically.

Proceed analogously in the "Register" column, then the rows beneath will be filled with increments of 1.

The single bits will be addressed according to the 16 rows of each connector on the second level.

## 9.7.3 PC Link Symbolic module

The PC Link Symbolic module is very similar to the PC Link module. The difference lies in the "Browse symbols" hyperlink on the bottom of the "General" tab.

Some systems provide a database (address book) with all the signals available in the DCS network. ibaPDA can load this address book (see [PC Link](#), page 250). If there is an existing address book, it is recommended using PC Link Symbolic modules, for a more convenient selection of signals to be measured.

### 9.7.3.1 PC Link Symbolic – General tab

#### Basic settings

For a description of the basic settings see [Common and general module settings](#), page 20.

#### Advanced

##### Asynchronous mode

In asynchronous mode, the data is copied from the card's memory into the memory of ibaPDA outside of the interrupt service routine (ISR). This mode can be used to measure large data volumes on a slower time base.

Enabling asynchronous mode is recommended if the duration of the ISR is longer than 2000  $\mu$ s. For checking the ISR duration, go to *Interrupt time* on the *Interrupt info* node of the *General* tab in the I/O Manager.

If you want to activate the asynchronous mode, set this option on TRUE.

## Module layout

### Number of analog and digital signals

Here, you can increase or decrease the number of signals in the module. By default, 32 signals are preset. You may enter any value between 0 and 1000. The signal tables will be adjusted accordingly.

### Select symbols (link)

Click on this hyperlink to open the DCS symbol browser for selecting signals to be measured.

## 9.7.3.2 PC Link Symbolic – Analog and Digital tab

### General columns in the signal table

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals](#), page 22.

### Symbol

In this column, you will find the symbolic name of the signal as configured in the DCS system.

You may enter the symbol name manually, however, it is recommended using the DCS symbol browser, as it is easier and safer.

Two different methods are available:

a) via the link in the *General* tab of the module. With this method, the signals selected from the signal browser will be put into the next free row of the signal table. This is useful if you fill the signal table for the first time or if you want to fill up a partially filled signal table.

b) Via the small browser button  in the "Symbol" column of the desired signal. With this method, you determine exactly the position where a symbol must be entered in the table.

## 9.8 Reflective memory

### Description

The reflective memory (RM) interface is based on special hardware by General Electric (formerly GE Fanuc and VMIC). RM interface boards are available for a variety of systems, such as PCI Express, PCI and VME. The drivers for *ibaPDA* support the PC components, VMIPCI-5565, 5576, 5579, 5587, 5588 and the current components, PCI-5565PIORC and PCIE-5565PIORC.

The "Direct Memory Access" (DMA) mode is supported for the components, VMIPCI 5565, PCI-5565PIORC and PCIE-5565PIORC.

A maximum of up to 1024 modules are supported per interface.

The number of signals to be used is only limited by the *ibaPDA* license and the performance of the systems.

---

### Note



Please note that not all of the above components can be used in the current versions of Windows with 64 bit. Details on this can be found in *ibaPDA*'s compatibility overview in the [versions\\_pda.htm](#) file or in the *ibaPDA-Interface-Reflective-Memory* product manual.

---

### Interface configuration

#### 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. The setting is disabled for the newer cards, such as PCI 5565PIORC. You can select the swap mode in the settings of the data module in the *General* tab.

#### Memory length limit

This parameter describes the size of the mapped memory space. You should adjust the memory size according to your needs. Reduce the disk space if you do not need that much memory. 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 boundaries

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.

When addressing data, otherwise than on 4-byte boundaries, 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.

The option is enabled by default if the 5565PIORC component is used.

**Available modules**

- Reflective Memory with up to 1000 analog and 1000 digital signals per module, supports asynchronous mode and DMA
- Reflective Memory dig512, with up to 32 × 16 digital signals per module, supports asynchronous mode and DMA
- X-Pact Lite, with up to 1000 analog and 1000 digital signals per module, supports asynchronous mode and DMA (only with a license for X-Pact v1 or v2)
- HiPAC Request (only with HiPAC interface license)

**Product name**

ibaPDA-Interface-Reflective-Memory (Art. no. 31.001220)

**Other documentation**

A detailed description of this interface and its configuration can be found in the corresponding manual for the product, *ibaPDA-Interface-Reflective-Memory*.

## 9.9 ScramNet+

ScramNet+ is an acronym for Shared Common RAM. It is a communication system based on shared memory, which has been optimized for high speed and deterministic data transmission. Therefore, it is suits ideally for real-time applications and fast simulation.

The ScramNet+ interface bases on the specific hardware components from CWC Embedded Computing. ScramNet+ interface components are available for a variety of automation systems, such as PCI, VME6U, Compact PCI or PMC. The drivers for *ibaPDA* support the PC board SC150.

The data interface ScramNet+ allows the use of the following two modules:

- ScramNet, with up to 1000 analog signals and 1000 digital signals per module
- ScramNet dig512, with up to 32 x 16 digital signals per module
- A maximum of up to 1024 modules are supported per interface.

The number of signals to be used is only limited by the *ibaPDA* license and the performance of the systems.

For further information on module configuration, refer to ...

- ↗ *ScramNet module type*, page 258
- ↗ *ScramNet dig512 module type*, page 261

## 9.9.1 Interface configuration

### Swap mode

The only parameter to be adjusted in this dialog is the swap mode. The drop-down list offers several options of high- and lowbyte swapping (Endian control). Which mode is the right one depends on the connected 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>.

### Error indicators

Information about errors, which are detected by the card driver.

### Number of nodes

Information about number of connected nodes in the network. Automatically detected by the interface board.

### PCI Info

In the PCI Info area of the dialog, you find the following information:

- Slot Number  
Number of the PCI slot where the card is plugged in.
- Bus Number  
PCI bus connected with this slot
- IO Address  
Start address of the I/O address range of the card (hex)
- Mem. Address  
Start address of the memory region (hex)
- IO Length  
Size of the I/O address range (hex), free / reserved
- Memory Length  
Size of the memory region (hex), free / reserved
- Manufacturer  
Name of the board manufacturer
- Board ID:  
PCI board ID (hex); also listed in the PCI table when system is booting.

## 9.9.2 ScramNet module type

The ScramNet module type is used for the acquisition of up to 1000 analog and 1000 digital signals over a ScramNet+ coupling.

As for analog values, you may choose between six different data types: BYTE, INT, DINT, WORD, DWORD, FLOAT.

The module size, i.e. the number of signals, can be specified. Default setting is 32 analog and 32 digital signals. If more signals are required, they can either be added to the module or additional modules can be added.

### 9.9.2.1 ScramNet – General tab

#### Basic settings

For a description of the basic settings see [↗ Common and general module settings](#), page 20.

#### Advanced

##### Swap mode

Set the swap mode according to the signal source.

You can choose between the following 4 options:

Mode	16 bit	32 bit
No swap	AB	ABCD
Depending on data type	BA	DCBA
Swap 16 bit	AB	CDAB
Swap 8 bit	BA	BADC

The swap mode to be selected depends on the swap mode of the signal source.

##### Asynchronous mode

In asynchronous mode, the data is copied from the card's memory into the memory of *ibaPDA* outside of the interrupt service routine (ISR). This mode can be used to measure large data volumes on a slower time base.

Enabling asynchronous mode is recommended if the duration of the ISR is longer than 2000  $\mu$ s. For checking the ISR duration, go to *Interrupt time* on the *Interrupt info* node of the *General* tab in the I/O Manager tree.

If you want to activate the asynchronous mode, set this option on TRUE.

#### Module layout

##### Number of analog and digital signals

Here, you can increase or decrease the number of signals in the module. By default, 32 signals are preset. You may enter any value between 0 and 1000. The signal tables will be adjusted accordingly.

### 9.9.2.2 ScramNet– Analog tab

#### General columns in the signal table

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals](#), page 22.

##### Address

In this column, you should specify the offset of the first byte of the value within the raw data stream. The offset can be entered as hexadecimal or decimal values by selecting the appropriate option from the context menu. For a default allocation of the addresses within the column, just click on the column header. Based on the address offset and starting with the value in the first row or in the field the cursor is currently placed in, the address values are filled in automatically according to the previously selected data types.

**Data type**

In the fields of this column, you can select the relevant data type used for each signal. Click in the corresponding field and select the data type from the drop-down list. The address space depends on the data type. Therefore, an adjustment of address entries might be necessary after changing the data types.

Available data types:

Data type	Description	Values Range
BYTE	8 bit without positive or negative sign	0 ... 255
INT	16 bit with positive or negative sign	-32768 ... 32767
WORD	16 bit without positive or negative sign	0 ... 65535
DINT	32 bit with positive or negative sign	-2147483647 ... 2147483647
DWORD	32 bit without positive or negative sign	0 ... 4294967295
FLOAT	IEEE754; single precision; 32 bit floating point	$\pm 3.402823 \cdot E+38$ ... $\pm 1.175495 \cdot E-38$
DOUBLE	IEEE754; Double Precision; 64-bit floating point value	2.225E-308 ... 1.798E+308
FP_REAL	Fixed point real; Q15.16; 15 integer bits and 16 fractional bits;	-32768 ... 32767.9999

**Note**

It is recommended to configure the data to be transmitted in consecutive memory regions, i.e. the signals should have consecutive addresses. Otherwise, the performance might decrease considerably.

**9.9.2.3 ScramNet– Digital tab****General columns in the signal table**

For a description of the general signal table columns see [Columns in tables with analog and digital signals](#), page 22.

As for digital signals, it is possible to read 16 single bits out of an INTEGER (WORD) or 32 single bits out of a DINT (DWORD).

**Address**

In this column, you should specify the offset of the first byte of the data-carrying binary signal within the raw data stream. The offset can be entered as hexadecimal or decimal values by selecting the appropriate option from the context menu. For a default allocation of the addresses within the column, just click on the column header. Based on the address offset and starting with the value in the first row or in the field the cursor is currently placed in, the address values are filled in automatically according to the previously selected data types.

**Bit no.**

This number (0...15 or 0...31) specifies the position of the desired digital signal in a 16-bit block or 32-bit block in the data stream with regard to the related offset address. Increase of bit no. by 1 up to 15 (31), then increase of address by 2 (4).

**Note**

It is recommended to configure the data to be transmitted in consecutive memory regions, i.e. the signals should have consecutive addresses. Otherwise, the performance might decrease considerably.

### 9.9.3 ScramNet dig512 module type

The ScramNet dig512 module type is used for the acquisition of up to 512 digital signals over a ScramNet+ coupling, with the digital signals being packed in 32 16-bit integer signals.

#### 9.9.3.1 ScramNet dig512 – General tab

##### Basic settings

For a description of the basic settings see [↗ Common and general module settings](#), page 20.

#### 9.9.3.2 ScramNet dig512 – Digital tab

The signal tables for modules with dig512 format consist of two levels. The first level shows the so-called connectors and activation attributes. If you click on the small plus symbols in the table rows, the second level of the signal table opens and you can see the actual signals (16 per connector).

##### General columns in the signal table

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals](#), page 22.

##### Connector level

###### Connector

Following the principle of the former ibaDig512 device, the separate data packages are referred to as connectors. One connector corresponds to an integer data item with 16 bits. You can extend or hide the signal tables for each connector by click on the small + (or –) symbol at the connector name. You may also enter a name for the connector in the “Connector” column. This name is used for engineering assignment. Under each connector, 16 digital signals are grouped on the second level of the signal table.

###### Address

In the address column (connector line), the byte offset in the range of every single connector (= integer package) may be specified by the user. The offset can be entered as hexadecimal or decimal values by selecting the appropriate option from the context menu. Usually, the default values need to be adjusted. Consecutive addresses count in steps of 2 according to the size of

the 16-bit integer package. After you have entered the address in the first row and clicked on the column header, all following addresses are updated automatically.

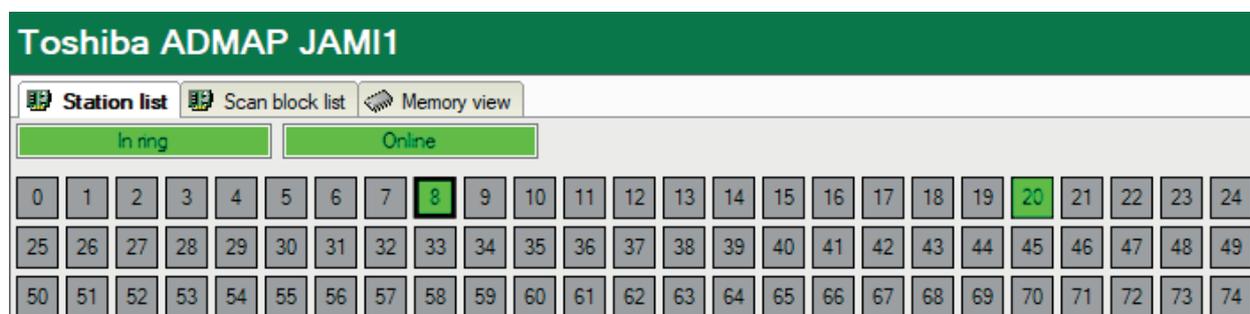
### Active

Enabling the connectors.

A click on the "Active" column header activates (checkmark) or deactivates (no checkmark) all the connectors for acquisition at the same time. Individual connectors can be activated in the corresponding selection box. No acquisition takes place for deactivated connectors, so that such connectors are neither available for display nor storage. When enabling/disabling a connector's activation attribute in the parent table, all the channels it contains are enabled/disabled. If you want to activate/deactivate the signals individually, go to the second level. If enabling of the signals of a connector is not uniform, the "Activation" selection box of the connector is grayed out.

Furthermore, disabled signals will not be taken into account in the signal statistics ("signal-o-meter").

## 9.10 Toshiba ADMAP JAMI1



### Description

The control network LAN ADMAP is a network of the CIE integrated control system from Toshiba, enabling a free communication with connected TOSDIC-CIE. The hardware interface to the ADMAP network is the JAMI1 board from Toshiba. It provides the computer access to the ADMAP legacy network (ADMAP-5M bus), which is a redundant network. *ibaPDA* only supports the scan transmission protocol and the ADMAP-5M bus.

### Interface configuration

The interface Toshiba ADMAP JAMI1 only shows status and diagnostic information. No settings required at this stage.

#### "Station list" and "Scan block list" tabs

#### "In ring" and "Online" indications

The first two indications give information on whether the card itself is connected to the ring or online. If the card is online, all data written to the card will be transferred to other stations in the ring.

The card is then automatically online when booting the computer.

#### "Station" field

Below, you will find a field with 256 boxes. Each box stands for a station in the ADMAP network and gives information on its individual status:

- Gray: station is not connected
- Red: station is connected and in standby mode
- Green: station is connected and in online mode
- Thick border: station corresponds to JAMI1 module of *ibaPDA* computer (like no. 8 in the fig. above)

#### Scan block list

The "Scan block list" tab shows the scan blocks. 1024 boxes are displayed. Each of them corresponds to a scan block and shows the current status of the block.

- Gray: scan block is damaged
- Red: scan block is damaged while *ibaPDA* is reading this block
- Green: Green: scan block is intact  
A scan block is intact when there is a talker for this block, i.e. when station is writing to that block.
- Thick border: *ibaPDA* is currently reading this block

#### Available modules

- Common memory; 0 to 1000 analog signals and 0 to 1000 digital signals (default 32/32)
- Scan block; 0 to 1000 analog signals and 0 to 1000 digital signals (default 32/32)

#### Product name

ibaPDA-Interface-Toshiba-ADMAP JAMI1

---

#### Other documentation



For a detailed description of this interface and its configuration, refer to the corresponding manual of the product *ibaPDA-Interface-Toshiba-ADMAP*.

---

## 9.11 X-Pact

### Description

This interface refers to the first implementation of X-Pact communication in *ibaPDA*, version 6.12.0. Data acquisition is made possible via different physical connections:

- Several physical links make data acquisition possible:
- ibaLink-SM-128V-i-2o / ibaLink-VME board, installed at the X-Pact rack Reflective Memory board, installed at the X-Pact rack or X-Pact computer

### Interface configuration

#### Hardware database

Enter here the full file path of the X-Pact hardware database. Store the file preferably in the server directory of *ibaPDA*.

**Project database**

Enter here the full file path of the X-Pact project database. Store the file preferably in the server directory of *ibaPDA*.

**User name and password**

If required, enter user name and password.

**<Load project>**

Click on this button after you have specified the database files in order to load the project data as well as to create address books and modules.

**Retrieve symbol comments from additional language**

If possible, select a language for the symbol comments to be retrieved.

**Ignore communication errors...**

If this option is activated, data acquisition starts even if communication errors occur on the DAQ server side.

**Available modules**

At the start, no modules can be added. Only after loading the project by clicking the <Load project> button, the appropriate modules can be configured in *ibaPDA*. The communication interfaces configured for X-Pact – SM128, ibaLink-VME or Reflective Memory – appear in the interface tree.

Now, it is possible to add an “X-Pact” module to each link.

For further information on module configuration, refer to [➤ X-Pact lite module \(Reflective Memory\)](#), page 264.

**Product name**

ibaPDA-Interface-Request-X-Pact (art. no. 31.001340)

---

**Other documentation**

On request.

---

**9.11.1 X-Pact lite module (Reflective Memory)**

This module type is only available if an X-Pact license is enabled in the dongle.

With the X-Pact lite module, you can measure the global variables that are already available in reflective memory. X-Pact does not require any agents. Only an address book is required containing all the variables in the reflective memory. This address book is generated by the X-Pact address book generator.

The paths of the X-Pact hardware database and project database should be entered under the “X-Pact” data interface in the tree of the I/O Manager. In this dialog, address books are generated with a click on the <Load project> button.

### 9.11.1.1 X-Pact lite – General tab

#### Basic settings

For a description of the basic settings see [↗ Common and general module settings](#), page 20.

#### Advanced

##### Swap mode

Set the swap mode according to the signal source.

You can choose between the following 4 options:

Mode	16 bit	32 bit
No swap	AB	ABCD
Depending on data type	BA	DCBA
Swap 16 bit	AB	CDAB
Swap 8 bit	BA	BADC

The swap mode to be selected depends on the swap mode of the signal source.

##### Asynchronous mode

In asynchronous mode, the data is copied from the card's memory into the memory of *ibaPDA* outside of the interrupt service routine (ISR). This mode can be used to measure large data volumes on a slower time base.

Enabling asynchronous mode is recommended if the duration of the ISR is longer than 2000  $\mu$ s. For checking the ISR duration, go to *Interrupt time* on the *Interrupt info* node in the *General* tab of the I/O Manager.

If you want to activate the asynchronous mode, set this option on TRUE.

#### Module layout

##### Number of analog and digital signals

Here, you can increase or decrease the number of signals in the module. By default, 32 signals are preset. You may enter any value between 0 and 1000. The signal tables are adjusted accordingly.

### 9.11.1.2 X-Pact lite – Analog and Digital tab

#### General columns in the signal table

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals](#), page 22.

##### Symbol

In this column, you will find the symbolic name of the signal as configured in the X-Pact system.

You may enter the symbol name also manually. However, it is recommended using the signal browser, as it is easier and saver.

In compliance with the naming rules in the X-Pact system the full symbol path is indicated.

Two different methods are available:

a) over the link in the "General" tab of the module. With this method, the signals selected from the signal browser will be put into the next free row of the signal table. This is helpful if you fill the signal table for the first time or if you want to fill up a signal table.

b) Via the small browser button  in the "Symbol" column of the desired signal. With this method, you determine exactly the position where a symbol must be entered in the table.

## 9.12 X-Pact Request

### Description

This is an interface between *ibaPDA* and the X-Pact system of SMS Siemag. It provides *ibaPDA* with symbolic access to all signals defined in the X-Pact system.

Currently, up to 2 interfaces are supported for data acquisition:

- ibaLink-SM-128V-i-2o / ibaLink-VME cards
- Reflective Memory cards VMIC 5576 and 5565.

An additional Ethernet TCP/IP connection between *ibaPDA* and X-Pact is required for configuration. The X-Pact Request interface requires address books of the corresponding X-Pact projects. They can be created with the address book generator or imported once they are available. A maximum of up to 1024 modules are supported per interface.

### Interface configuration (Overview tab)

#### Active

Check this option if you want to use the interface.

#### <Create address books>

Unless already done, you should create address books first. Clicking on this button, will open the address book. Enter here the full path of the X-Pact project.

Then load one or several projects by clicking on <Load projects> and check the resources in the tree structure, for which the address book is to be created. Finally click on <Create address books> and exit the dialog. You can create address books for up to 2 projects.

#### <Import address books>

Clicking on this button, will load an address book that was already created or previously extracted from a ZIP file into your file system.

#### <Export address books>

Click on this button in order to load address books that were currently loaded as zip-files in your system.

#### Multicast IP address and Port no.

By default, these values are set to 239.23.07.78 (Multicast IP address) and 17477 (Port no.); usually, these values can be kept. You can configure them according to your needs.

#### X-Pact CPU response timeout [s]

This value defines the maximum waiting time before an X-Pact CPU is considered as not available.

**Disable signals from non-responding CPUs**

At the start of the acquisition, all CPUs (in the X-Pact Controller) are requested. If a CPU is not responsive, the related signals will be disabled and the acquisition will start without these signals, provided this option is enabled.

**Enable CPU reconnect detection**

If this option is enabled, missing CPUs will be checked also during the measurement. In case the CPUs are available again, the measurement stops, a request will be executed and the measurement restarts.

**Enable DMA for request modules on reflective memory**

Enable this option if you use a reflective memory board, which supports DMA (Direct Memory Access), such as PCI-5565PIORC or PCIE-5565PIORC. Particularly, when processing large amounts of data and/or acquiring the data very fast via the reflective memory board, the CPU load of the *ibaPDA* computer can thus be reduced significantly though increasing the processing speed.

**<Apply and restart>**

Click on this button after all settings are made.

**Controller status area**

As soon as you apply the configuration and restart the measurement, you will find a grid showing all involved controllers below the properties area. The color of a controller corresponds to the status of the connection with the controller or its first CPU. There are 4 possible color displays:

- Red: there is no TCP connection and no data connection to the controller.
- Yellow: there is a TCP connection but no data connection to the controller.
- Green with exclamation point: there is a TCP connection and at least one data connection to the controller was detected.
- Green without exclamation point: there is a TCP connection and all data connections to the controller were detected.

A flashing controller indicates an existing connection, which is not listed in the project and a missing address book. In this case, refresh the address books by reloading the project in the address book generator.

**Diagnostics tab**

On this tab, you will find detailed information on the diagnosis of the connected X-Pact system.

**Available modules**

- X-Pact request; 0 to 1000 analog signals and 0 to 1000 digital signals (default 32/32)

For further information on module configuration, refer to [➤ X-Pact Request module](#), page 268.

**Product name**

ibaPDA-Interface-Request-X-Pact (art. no. 31.001340)

---

**Other documentation**

For a detailed description of this interface and its configuration, refer to the corresponding manual of the product *ibaPDA-XPact-Request*.

---

### 9.12.1 X-Pact Request module

The X-Pact Request module type provides two ways for the acquisition of up to 1000 analog and 1000 digital signals from an SMS SIEMAG X-Pact system.

- ibaLink-SM-128V-i-2o boards via ibaNet or
- Reflective Memory cards VMIC 5576 and 5565.

An additional Ethernet TCP/IP connection between *ibaPDA* and X-Pact is required for configuration.

The X-Pact Request interface requires address books of the corresponding X-Pact projects. They can be created with the address book generator or imported once they are available.

This module type can be configured under the “X-Pact Request” data interface in the I/O Manager tree.

The X-Pact Request module provides access to all variables of a connected X-Pact station.

The signals to be measured can be selected conveniently by browser in the *ibaPDA* I/O Manager.

Preconditions for using this module type are a connection to the X-Pact system over ibaNet or Reflective Memory as well as a TCP/IP connection to the X-Pact computer and the X-Pact system.

The two connections are required for different purposes:

- Control path (TCP/IP)
  - Registering X-Pact CPUs over multicast
  - Exchange of watchdog and administrative messages
  - Transmission of data request telegrams
- Data path (Reflective Memory link or ibaNet)
  - Data transmission of the requested signals from X-Pact to *ibaPDA*.

Before the Request function is ready for use, the address books must have been created or imported.

#### 9.12.1.1 X-Pact Request – General tab

##### Basic settings

For a description of the basic settings see [↗ Common and general module settings, page 20](#).

## Module layout

### Number of analog and digital signals

Here, you can increase or decrease the number of signals in the module. By default, 32 signals are preset. You may enter any value between 0 and 1000. The signal tables will be adjusted accordingly.

### X-Pact

#### Update time

The update time is the rate at which the X-Pact system will send module data to *ibaPDA*. The time base is the rate at which *ibaPDA* will request the data from the agent. Usually, the values of refresh time and time base are set equal.

### Hyperlink

#### Select symbols (link)

A click on this link will open the browser for selecting the signals to be measured from the address book. You have access to all symbols of all X-Pact stations configured in the loaded project.

The selected signals will be entered automatically in the appropriate signal table of the module (next available free row).

So you can select several signals one after the other without the browser automatically closing. The browser will stay open until you press OK.

You may also open the browser directly from the signal tables ("Symbol" column).

## 9.12.1.2 X-Pact Request – Analog and Digital tab

### General columns in the signal table

For a description of the general signal table columns see [↗ Columns in tables with analog and digital signals](#), page 22.

### Symbol

In this column, you will find the symbolic name of the signal as configured in the X-Pact system. You may enter the symbol name also manually. However, it is recommended using the signal browser, as it is easier and safer.

In compliance with the naming rules in the X-Pact system the full symbol path is indicated.

Two different methods are available:

a) via the link in the *General* tab of the module. With this method, the signals selected from the signal browser will be put into the next free row of the signal table. This is useful if you fill the signal table for the first time or if you intend to fill up a partially filled signal table.

b) Via the small browser button  in the "Symbol" column of the desired signal. With this method, you determine the exact position where a symbol has to be entered in the table.

**Note**

The "Actual value" column is not available in the signal tables of the Request modules. You find the actual values in the data modules automatically generated in the sub-branch of the interface.

---

## 10 Cloud, database and message broker interfaces

This section describes various interfaces that are used to exchange data with SQL databases, cloud systems and message broker services.

In some cases, these interfaces can not only be used for data acquisition (reading) but also for data output (writing) to the external systems.

### 10.1 SQL database interface

#### Description

The SQL interface of *ibaPDA* forms the basis for a series of database interfaces via which *ibaPDA* can read data both from databases as well as write data in databases.

The connections to one or more databases are configured and managed with the SQL interface. One or more connections can be configured per database. All DB connections can be used simultaneously. The databases can be of the same or different types.

Database systems of different vendors are supported:

- Maria DB
- MySQL
- Oracle
- PostgreSQL
- SAP Hana
- SQL Server

The data exchange with the databases occurs via corresponding SQL modules, which are configured according to the specific database type. The data is accessed with SQL statements.

- Input direction (read)
  - For read access, custom SQL queries (e.g., SELECT) are used and their results are mapped to input signals in *ibaPDA*.
  - Optional parameters assigned to signals can be used in the queries in order to thus change the queries dynamically during the ongoing acquisition.
- Output direction (write)
  - SQL commands (e.g., UPDATE, INSERT) are used for write access in order to write data from *ibaPDA* into the database.
  - In the commands, optional parameters assigned to signals can be used in order to therefore write signal actual values.

The execution of queries and commands can be cyclical or triggered.

**Caution**

**The network and DB can be overloaded by highly cyclical DB accesses and/or very large data volumes. In the SQL statements, any SQL commands can be used and therefore cause damage (e.g. “delete table,” “drop database,” etc.).**

Therefore...

- Only have qualified users create/change SQL statements (-> user management, security administration).
  - Involve your DB administrator for questions about configuring the SQL statements. This way, for example, the DB user used in *ibaPDA* for the DB connection only has the rights that are actually required.
  - Set an appropriate update time (as short as necessary, as long as possible).
  - Assess and test the results of your SQL statement.
- 

**Available modules**

SQL query (read from a database using SQL statement)

SQL command (write to a database using SQL command)

**Product name**

- ibaPDA-Interface-MySQL (Art. no. 31.001116)
  - ibaPDA-Interface-Oracle (Art. no. 31.001113)
  - ibaPDA-Interface-PostgreSQL (Art. no. 31.001115)
  - ibaPDA-Interface-SAP-HANA (Art. no. 31.001117)
  - ibaPDA-Interface-SQL-Server (Art. no. 31.001114)
- 

**Other documentation**

A detailed description of these interfaces and their configuration can be found in the corresponding product manuals for *ibaPDA-Interface-MySQL*, *ibaPDA-Interface-Oracle*, *ibaPDA-Interface-PostgreSQL*, *ibaPDA-Interface-SAP-HANA* oder *ibaPDA-Interface-SQL-Server*.

---

## 10.2 MQTT interface

### Description

MQTT (Message Queue Telemetry Transport) is a communication protocol that is essentially based on an event-driven publication/subscription architecture.

The core is a central server (broker) to which both the transmitter as well as the receiver connect. The data is sent (published) and received (subscribed) via so-called topics. Topics are basically communication channels in which the transmitters, e.g. sensors, write their data.

The broker checks which receivers have opened (“subscribed”) a channel (topic) for this data and then passes on the data to these receivers. *ibaPDA* with the *ibaPDA-Interface-MQTT* acts like the receiver (client).

As a client, *ibaPDA* subscribes to all topics with the measured values to be acquired and made available via the MQTT broker. This could include, for example, data from higher-level systems (levels 2 and 3) or from sensors that publish their measured values via MQTT.

MQTT v3.1.1 is supported in order to receive data from an MQTT-Broker.

---

### Note



The MQTT broker is not part of *ibaPDA*. This must be purchased, installed and configured separately. The transmission behavior is decisively influenced by the configuration and the performance of the broker.

---

### Interface configuration

At the interface level, no special settings need to be made in the I/O Manager. The connection is configured at the module level.

### Available modules

- MQTT

### Product name

*ibaPDA-Interface-MQTT* (Art. no. 31.001112)

---

### Other documentation.



A detailed description of this interface and its configuration can be found in the corresponding manual for the product, *ibaPDA-Interface-MQTT*.

---

## 10.3 HTTP(S) interface

### Description

*ibaPDA* provides the interface *ibaPDA-Interface-HTTP(S)* for sending HTTP(S) requests. By using this very common HTTP(S) protocol you get access to web services of any kind.

All standard methods for requests are supported: Get, Post, Put, Delete, Patch, Options, Trace and Head.

Thus, you cannot only read data (“Get”) but also write data (“Put”).

In terms of security, *ibaPDA* supports the use of SSL certificates and the authentication methods Basis Authentication, Json Web Token or OAuth2.0.

Under this interface you may configure any number of modules where each module can execute exactly one request. In the modules you configure all required parameters for the request, e.g. connection, query method, security settings etc.

The requests can be executed periodically or on trigger.

### Interface configuration

At the interface level, no special settings need to be made in the I/O Manager. The connection is configured at the module level.

### Available modules

- HTTP(S)

### Product name

ibaPDA-Interface-HTTP(S) (Art. no. 31.001018)

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product, *ibaPDA-Interface-HTTP(S)*.

---

## 11 Other interfaces and ibaPDA add-ons

Some add-ons like *ibaInSpectra* or *ibaInCycle* are visible in the signal tree as interfaces, if the corresponding license is available. The modules can then be created under these interfaces.

Name	Type	Connection to...	Comment	Link
Audio	Software license	Windows sound source (local computer)	<i>Inputs</i> tab	<a href="#">↗ Audio, page 276</a>
ibaInCycle	Software license	internal, InCycle functions	<i>Analytics</i> tab	<a href="#">↗ ibaInCycle, page 277</a>
ibaInSpectra	Software license	internal, InSpectra functions	<i>Analytics</i> tab	<a href="#">↗ ibaInSpectra, page 278</a>
Snapshots	Software license	internal, Computation module, InSpectra Expert	<i>Analytics</i> tab	<a href="#">↗ Snapshots, page 279</a>

## 11.1 Audio

### Description

The audio interface in *ibaPDA* is used to acquire audio data from a Windows audio source. For this purpose, a suitable component must be installed in the *ibaPDA*-server computer, e.g., a sound card or a USB device that can work as audio source. This allows microphone signals, e.g., from a headset, to be acquired and recorded.

However, other audio sources connected via a "line-in" input can also be acquired. Ultimately, any source that can be configured as an audio source in Windows can be used.

Possible applications include signal-synchronous acquisition and recording...

- of voice traffic via industrial intercom systems
- of radio traffic at the plant
- of loudspeaker announcements
- of announcements via automated audio information systems (e.g., text-to-speech)
- of acoustic recordings on a machine for troubleshooting purposes

*ibaPDA* can process up 4 audio input modules (mono or stereo) with one interface license. You can extend the number of audio modules by 4 with each step-up license to a total of 20 modules.

The audio interface is not designed for high-quality sound recordings (HiFi).

### Interface configuration

At the interface level, no settings need to be configured. However, *Exclusive Mode* should be enabled in the Windows sound settings so that the audio source can be used by *ibaPDA*. If several audio components are installed in the computer, you can select the desired component in the module settings.

### Available modules

Audio

### Product name

ibaPDA-Interface-Audio (Art. no. 31.001101)

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-Audio*.

---

## 11.2 ibaInCycle

### Description

*ibaInCycle* is a technology module that is integrated as an add-on in *ibaPDA*.

*ibaInCycle* monitors all types of cyclically repeating processes, including recurring process sequences and rotating components, such as rollers and gears.

Process signals from cyclical processes ideally exhibit similar behavior within a cycle. *ibaInCycle* compares the “learned” or defined good process with the actual process signal and indicates deviations immediately, for example via alarm message or e-mail.

In addition, feedback into the plant control system is also possible in order to automatically adjust corresponding parameters.

Since *ibaInCycle* is seamlessly integrated into *ibaPDA*, the full range of *ibaPDA* connectivity is available to acquire all possible process signals in a system and to use them to define the respective states.

### Available modules

- *InCycle Expert module*: This offers diverse, individual configuration options for analysis of the cycles
- *InCycle Auto-Adapting module*: This module automatically learns the behavior of the cycles in different process conditions and uses this as a reference to automatically identify deviations.

### Product name

ibaInCycle (Art. no. 30.681215)

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaInCycle*.

---

## 11.3 ibaInSpectra

### Description

With ibaInSpectra, any vibrations are permanently monitored in real time in order to detect possible sources of error at an early stage.

Since ibaInSpectra is integrated in ibaPDA, possible correlations between vibration effects and process behavior can be immediately identified in addition to the pure vibration analysis.

Spectral analyses can be used to monitor vibrations online and correlate them with other process parameters. If vibrations reach critical states an indication is generated immediately, for example via an alarm message or e-mail. In addition, feedback into the plant control system is also possible in order to automatically adjust corresponding parameters.

### Available modules

- *Expert module*: This offers the most versatile parameterization options for frequency band analysis and is the preferred tool for vibration experts. The frequency bands to be monitored can be defined as fixed or dependent on process variables, and can be checked for adherence to the limit values.
- *Auto-Adapting module*: This module automatically learns spectra under different process conditions and uses them as a reference to detect changes in the spectrum over time.
- *Orbit module*: This is used to monitor and analyze shaft motion, for example of plain bearings.
- *Universal module*: This module is easy to configure and calculates the most common characteristic values for vibration monitoring within the time domain.
- *Fan module*: This is used to monitor fans and specifically calculates status indicators for fans.

### Product name

ibaInSpectra (Art. no. 30.681223, limited to 4 modules)

---

### Other documentation



A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaInSpectra*.

---

## 11.4 Snapshots

### Description

The *snapshots* feature facilitates the buffering of signal data over a limited period of time. The name “Snapshot” refers to this limited time period (given in seconds). One or more snapshots can be made repeatedly as often as needed in a configurable interval (given in minutes).

If you have a corresponding license, the feature is available in the I/O Manager in the *Analytics* tab.

Below the main node *Snapshots*, you can configure any number of snapshots. For each snapshot you have to create a snapshot module, which determines the properties of the snapshot like duration and interval by profiles.

With each snapshot the buffered data is available for subsequent computations performed by suitable modules. After the finalization of the computations, the results and - if requested - the buffered data will be written to a data file.

The computations and writing of the data file are executed asynchronously, meaning that the recording of other signal data is not hindered by possibly long computation times of snapshot modules.

### Available modules

The following modules can be used together with the snapshot feature:

- Computation module: Like for the standard computation module you can configure computations in profiles which are then applied to the buffered data in the snapshot. Profiles, which already have been configured for standard computation modules may be used for snapshot computation modules too and vice versa.
- InSpectra Expert module: When using this module you have all the ibaInSpectra functions at hand in order to apply vibration analyses to the buffered data in the snapshot.

### Product name

ibaPDA-Snapshot (Art. no. 30.770026)

---

### Other documentation



A detailed description of this feature and its configuration can be found in the corresponding manual for the product *ibaPDA-Snapshot*.

---

## 12 Diagnostic modules

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

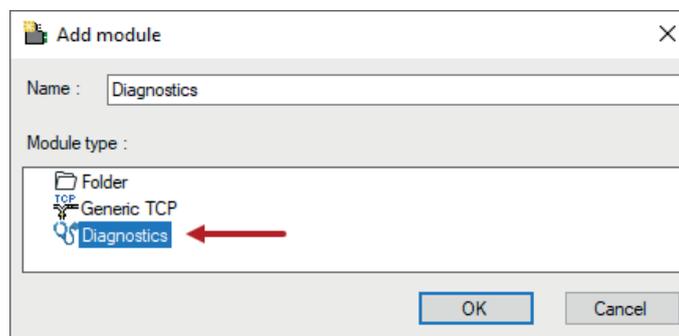
A diagnostic module is always assigned to a data acquisition module of the same interface and supplies its connection information. By using a diagnostic module you can record and analyze the diagnostic information continuously in the *ibaPDA* system.

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

Example for the use of diagnostic modules:

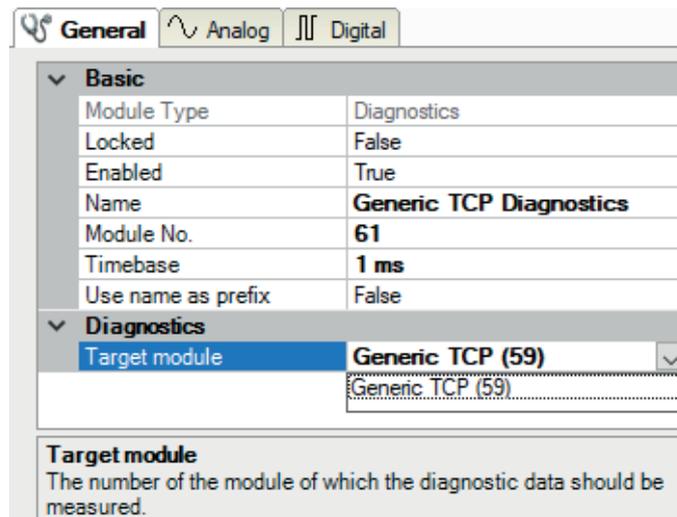
- A notification can be generated, whenever the error counter of a communication connection exceeds a certain value or the connection gets lost.
- In case of a disturbance, the current response times in the telegram traffic may be documented in an incident report.
- The connection status can be visualized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

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



### Module settings diagnostic module

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



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.

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

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

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

## Diagnostic signals

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

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

Signal name	Description
Incorrect message type	Number of received messages with wrong message type
Input data length	Length of data messages with input signals in bytes ( <i>ibaPDA</i> receives)
Invalid packet	Invalid data packet detected
IP address (part 1-4)	4 octets of the IP address of the target system
Keepalive counter	Number of KeepAlive messages received by the OPC UA Server
Lost images	Number of lost images (in) that were not received even after a retransmission
Lost Profiles	Number of incomplete/incorrect profiles
Message counter	Number of messages received
Messages per cycle	Number of messages in the cycle of the update time
Messages received since configuration	Number of received data telegrams (in) since start of acquisition
Messages received since connection start	Number of received data telegrams (in) since the start of the last connection setup. Reset with each connection loss.
Messages sent since configuration	Number of sent data telegrams (out) since start of acquisition
Messages sent since connection start	Number of sent data telegrams (out) since the start of the last connection setup. Reset with each connection loss.
Multicast join error	Number of multicast login errors
Number of request commands	Counter for request messages from <i>ibaPDA</i> to the PLC/CPU
Output data length	Length of the data messages with output signals in bytes ( <i>ibaPDA</i> sends)
Packet size (actual)	Size of the currently received message
Packet size (max)	Size of the largest received message
Ping time (actual)	Response time for a ping telegram
Port	Port number for communication
Producer ID (part 1-4)	Producer ID as 4 byte unsigned integer
Profile Count	Number of completely recorded profiles
Read counter	Number of read accesses/data requests
Receive counter	Number of messages received
Response time (actual/average/max/min)	Response time is the time between measured value request from <i>ibaPDA</i> 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 <i>ibaPDA</i> )
Source IP address (part 1-4) T->O	4 octets of the IP address of the target system Input data (from <i>ibaPDA</i> to target system)
Statements processed	Number of executed statements since last start of acquisition
Synchronization	Device is synchronized for isochronous acquisition
Time between data (actual/ max/min)	Time between two correctly received messages  Actual: between the last two messages  Max/min: statistical values since start of acquisition or reset of counters
Time offset (actual)	Measured time difference of synchronicity between <i>ibaPDA</i> and the <i>ibaNet-E</i> device
Topics Defined	Number of defined topics
Topics Updated	Number of updated topics
Unknown sensor	Number of unknown sensors
Update time (actual/average/ configured/max/min)	Specifies the update time in which the data is to be retrieved from the PLC, the CPU or from the server (configured). Default is equal to the parameter "Timebase". During the measurement the real actual update time (actual) 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

## 13 Support and contact

### Support

Phone: +49 911 97282-14

Email: [support@iba-ag.com](mailto: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](mailto: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](http://www.iba-ag.com)**