ibaPDA

Introduction and Installation

Manual Part 1

Issue 8.14

Measurement Systems for Industry and Energy

www.iba-ag.com

Manufacturer

iba AG

Gebhardtstrasse 10-20

90762 Fuerth

Germany

Contacts

Main office	+49 911 97282-0
Support	+49 911 97282-14
Engineering	+49 911 97282-13
E-mail	iba@iba-ag.com
Web	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 2026, 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 and can be found in the iba help center docs.iba-ag.com.

Version	Date	Revision	Author	Version SW
8.14	06-2026	License ibaPDA-Interface-Sigmatek-Xplorer removed; license ibaPDA-Request-S7 modified	RM	8.14.0

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

1 About this documentation 6

1.1 Target group and previous knowledge 6

1.2 Notations 6

1.3 Used symbols 7

1.4 Documentation structure 8

2 Product information 9

2.1 Introduction 9

2.2 Safety aspect 10

2.3 License for base software 10

2.4 Licenses for interfaces 12

2.5 Licenses add-ons 17

2.6 Licenses for special data storage 18

2.7 Upgrade policy 19

2.8 Update policy 19

2.9 Demo mode 21

2.10 Demo licenses 22

3 Installation and program start 23

3.1 System requirements 23

3.2 Installation 25

3.2.1 Standard installation 25

3.2.2 Installing by command line 28

3.2.2.1 /HELP command line switch 28

3.2.2.2 /SILENT command line switch 28

3.2.2.3 /VERYSILENT command line switch 28

3.2.2.4 /SUPPRESSMSGBOXES command line switch 28

3.2.2.5 /Dir= command line switch 29

3.2.2.6 /COMPONENTS= command line switch 29

3.2.2.7 /FORCECLIENTCLOSE command line switch 29

3.2.2.8 /LOG, /LOG='filename' command line switch 30

3.2.2.9 /TASKS= command line switch 30

3.3 Version alignment of ibaPDA-server and ibaPDA-Client 31

3.4 Start program 32
3.4.1 Startup of the server 32
3.4.2 Starting the client 32
3.4.3 Starting the ibaPDA client via command line 33
3.4.3.1 General 33
3.4.3.2 /config: command line switch 33
3.4.3.3 /reuse command line switch 35
3.4.3.4 /displaystyle: command line switch 35
3.4.3.5 /lang: command line switch 36
3.4.4 Starting multiple clients on the same computer 37
3.4.5 Disconnect another client on client startup 39
4 User interface 40
4.1 The screen 40
4.2 The menu bar 41
4.2.1 The File menu 42
4.2.2 The Configuration menu 42
4.2.3 The View menu 46
4.2.4 The Help menu 51
4.3 Toolbars 52
4.3.1 Configuration toolbar part 52
4.3.2 Layout toolbar part 52
4.3.3 Data storage toolbar part 55
4.4 Status bar 56
4.5 Operation via keyboard shortcuts 57
5 Configuring the system 59
5.1 Project management 59
5.2 Remote configuration 64
5.2.1 Principle of operation 64
5.2.2 Setting up the remote configuration 65

6 ibaPDA-server 67

6.1 Select an ibaPDA server 68

6.2 Server status display 72

6.2.1 Server Status, General 72

6.2.2 Server status I/O Manager 75

6.2.3 Server Status, Data storage 77

6.2.4 Server status, Event log 78

6.3 Server – more information 79

6.4 Server access control 81

7 User management 85

7.1 Local user management in ibaPDA 87

7.1.1 ibaPDA server rights 92

7.1.2 ibaPDA client rights 93

7.2 User management in a domain with Active Directory 96

7.2.1 Advantages of the Active Directory method 96

7.2.2 Establish a connection to a domain 96

7.2.3 Add domain user 98

7.2.4 Manage user rights 98

7.2.5 Effective rights 99

7.2.6 Logging in as a domain user 101

7.2.7 Indication of logon name and display name 102

7.3 Rights when simultaneously logged in to ibaPDA and ibaHD-Server 103

7.3.1 ibaPDA-client rights for PDA and HD clients 104

7.3.2 ibaPDA server rights for PDA and HD clients 106

8 Printing 107

8.1 Printer setup 107

8.2 Settings for printing views 108

9 Support and contact 110

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 they are capable of assessing safety and recognizing possible consequences and risks on the basis of their specialist training, knowledge and experience and knowledge of the standard regulations.

1.2 Notations

In this manual, the following notations are used:

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

1.3 Used symbols

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

Danger!

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

Observe the specified measures.

Warning!

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

Observe the specified measures.

Caution!

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

Observe the specified measures.

Note

A note specifies special requirements or actions to be observed.

Tip

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

Other documentation

Reference to additional documentation or further reading.

1.4 Documentation structure

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.

Part 1	Introduction and installation	General notes, license policy and add-ons Installation and program start User interface, system architecture, client server User management, printing
Part 2	I/O Manager	Basic information about the I/O Manager, general settings Groups and vector signals, text signals, outputs, configuration files
Part 3	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.
Part 4	Expression builder	All functions for calculating virtual signals
Part 5	Data storage	Types of data recording, recording profiles, signal selection
Part 6	Data visualization	All display modes for live data, their operation and settings
Part 7	Appendix	Various additions, error listings, etc.

2 Product information

In this section you find information about the overall system as well as security aspects, the license policy, add-ons and demo mode.

2.1 Introduction

The *ibaPDA* (Process Data Acquisition) system is a PC-based system for the acquisition, recording and analysis of measured signals.

It consists of coordinated hardware and software components for the acquisition, recording, evaluation and processing of measurement data.

Thanks to the modular design and easy configuration, the system can be comfortably adapted to the most diverse tasks and is scalable in size at any time. As a result, a wide range of applications can be covered: from small systems for mobile commissioning with 64 measuring signals to large, stationary systems with several thousand signals. If a system is not sufficient on its own, multiple *ibaPDA* computers can be synchronized in the multistation mode. Similar to a flight data recorder, the *ibaPDA* system records a wide range of measurement data and stores it for long-term availability.

The recorded measurement data is saved in a measurement file continuously (24/7) or only with the occurrence of defined events. The measurement files can be up to 64 GB in size and support UTF-8 encoding (UNICODE), so that texts and signal names can also consist of Asian characters, for example. For protection of data and know-how, the measurement files can be protected with a password. Alternatively or additionally, measurement data can also be continuously recorded by means of an *ibaHD-Server*.

An essential feature of the *ibaPDA* system is the strong connectivity to automation systems and field bus technologies. An iba data acquisition system can be connected to almost any common automation system via vendor-specific special assemblies, field bus connections or Ethernet protocols, regardless of manufacturer or device generation.

There is also a wide range of high-quality analog/digital converters for discrete and time-critical signals available, which offer acquisition rates of up to 100 kHz and are connected to the *ibaPDA* computer via the proprietary ibaNet bus system.

Besides the acquisition of measurement data, *ibaPDA* also offers the possibility to output signals and calculated values. For output signals, different interfaces are available. Discrete analog and digital signals can be used for controlling or alarming via ibaNet and suitable hardware components. All configured signals are available as tags for SNMP, OPC and OPC-UA servers integrated in *ibaPDA*. Further interfaces to databases, MQTT brokers and cloud systems complete the range of functionality.

The analysis of the data recorded by *ibaPDA* is carried out with the license-free software, *ibaAnalyzer*.

The benefits of the system can be considerably extended with special add-ons, such as *ibaCapture* for signal-synchronous video recording or *ibaQPanel* for HMI-like visualization of measured values.

With *ibaPDA*, metrological solutions can be implemented regardless of sector or technology and for a wide range of applications:

- Troubleshooting
- Process data analysis
- Quality documentation
- Condition monitoring
- Fault recorder (TFR) and energy quality
- Test bench automation
- Retrofit

2.2 Safety aspect

ibaPDA is a software package that is not designed for process or plant control. Special warnings regarding risks for people and machines are therefore not required.

The signal output option of *ibaPDA* is only used for information purposes in the sense of signaling and alarms or the transmission of recorded signals without security-related effects. If output signals from *ibaPDA* are to be incorporated into a security plan, then this must be done with appropriate systems in process or plant automation.

Safety and hazard warnings for using the hardware components can be found in the corresponding documentation.

2.3 License for base software

Order no.	Product name	Description
30.770064	ibaPDA-64	64 signals, 1 client [1], 2 data storages, standard interfaces
30.770128	ibaPDA-128	128 signals, 1 client [1], 2 data storages, standard interfaces
30.770256	ibaPDA-256	256 signals, 1 client [1], 2 data storages, standard interfaces
30.770512	ibaPDA-512	512 signals, 1 client [1], 2 data storages, standard interfaces
30.771024	ibaPDA-1024	1024 signals, 1 client [1], 2 data storages, standard interfaces
30.772048	ibaPDA-2048	2048 signals, 1 client [1], 2 data storages, standard interfaces
30.774096	ibaPDA-4096	4096 signals, 1 client [1], 2 data storages, standard interfaces
30.778192	ibaPDA-8192	8192 signals, 1 client [1], 2 data storages, standard interfaces

Order no.	Product name	Description
30.779999	ibaPDA-unlimited	>8192 signals, 1 client [1], 2 data storages, standard interfaces
30.770021	ibaPDA-One-Data-Store	1 additional data store
30.770022	ibaPDA-Two-Data-Stores	2 additional data stores
30.770023	ibaPDA-Ultra-Data-Store	Max. 255 data storages with up to 20 signals each; more signals per data storage are possible but with every additional multiple of 20 or part of it the number of usable data storages decreases by 1. 1…20 signals require 1 data storage 21…40 signals require 2 data storages 41…60 signals require 3 data storages
30.770024	ibaPDA-Client	1 additional client
30.770025	ibaPDA-Multi Client	5 additional clients
30.770001	Upgrade ibaPDA-64 to 128	Increase number of signals
30.770002	Upgrade ibaPDA-128 to 256	Increase number of signals
30.770003	Upgrade ibaPDA-256 to 512	Increase number of signals
30.770004	Upgrade ibaPDA-512 to 1024	Increase number of signals
30.770005	Upgrade ibaPDA-1024 to 2048	Increase number of signals
30.770006	Upgrade ibaPDA-2048 to 4096	Increase number of signals
30.770007	Upgrade ibaPDA-4096 to 8192	Increase number of signals
30.770008	Upgrade ibaPDA-8192 to unlimited	Increase number of signals
30.770009	Upgrade PLC-Xplorer to PDA-64	Upgrade to a full-fledged ibaPDA license
30.001930	ibaPDA-Multistation-License	License for the synchronous operation of multiple ibaPDA computers

[1] One client for use on other computers included. There is another client available, which can only be used on the server computer.

2.4 Licenses for interfaces

Order no.	Product name	Description
31.000001	ibaPDA-Interface-S7-Xplorer	Single license for 16 connections, expandable by up to 14 licenses for max. 240 connections
31.000002	ibaPDA-Interface-Codesys-Xplorer	Single license for 16 connections, expandable by up to 14 licenses for max. 240 connections
31.000003	ibaPDA-Interface-AB-Xplorer	Single license for 16 connections, expandable by up to 14 licenses for max. 240 connections
31.000005	ibaPDA-Interface-TwinCAT-Xplorer	Single license for 16 connections, expandable by up to 14 licenses for max. 240 connections
31.000006	ibaPDA-Interface-B&R-Xplorer	Single license for 16 connections, expandable by up to 14 licenses for max. 240 connections
31.000007	ibaPDA-Interface-Logix-Xplorer	Single license for 16 connections, expandable by up to 14 licenses for max. 240 connections
31.000008	ibaPDA-Interface-MELSEC-Xplorer	Single license for 16 connections, expandable by up to 14 licenses for max. 240 connections
31.000009	ibaPDA-Interface-ABB-Xplorer	Single license for 16 connections, expandable by up to 14 licenses for max. 240 connections
31.000030	ibaPDA-Interface-SINAMICS-Xplorer	Access to SINAMICS Control Units Single license for 32 connections, expandable by up to 15 licenses for max. 512 connections
31.000031	ibaPDA-Interface-SIMOTION-Xplorer	Access to SIMOTION motion controllers Single license for 32 connections, expandable by up to 15 licenses for max. 512 connections
31.000033	ibaPDA-Interface-SINUMERIK-Xplorer	Access to SINUMERIK controllers (NCUs) Single license for 16 connections, expandable by up to 15 licenses for max. 256 SINUMERIK NCUs
31.000034	ibaPDA-Interface-Bachmann-Xplorer	Single license for 16 connections, expandable by up to 14 licenses for max. 240 connections

Order no.	Product name	Description
31.000035	ibaPDA-Interface-OMRON-Xplorer	Single license for 16 connections, expandable by up to 14 licenses for max. 240 connections With WIBU CodeMeter license only!
31.001005	ibaPDA-Interface-Ethernet-IP	Single license for 64 connections, expandable by up to 3 licenses for max. 256 connections
31.001006	ibaPDA-Interface-ibaNet-E	Single license for up to 16 ibaNet-E connections to foreign devices
31.001007	ibaPDA-Interface-eLumina	Single license for up to 16 connections to eLumina CCUs via ibaN-2E; expandable by up to 15 licenses for max. 256 connections
31.001009	ibaPDA-Interface-HPCi-DGM200E	Use of a DGM200E adapter device from GE
31.001010	ibaPDA-Interface-HPCi-DGM200P	Use of a DGM200P card from GE
31.001011	ibaPDA-Interface-LANDSCAN	Single license for 2 connections, expandable by up to 7 licenses for max. 16 connections
31.001012	ibaPDA-Interface-LMI-Gocator	Single license for 2 sensors, expandable by up to 7 licenses for max. 16 connections
31.001013	ibaPDA-Interface-Raytek	Single license for 2 scanners, expandable by up to 7 licenses for max. 16 connections
31.001014	ibaPDA-Interface-LAND-SPOT+	License for unlimited connections to LAND SPOT+ pyrometers
31.001015	ibaPDA-Interface-ibaLogic-TCP	Single license for 64 modules, each with 32 analog values (REAL) and 32 digital values.
31.001016	ibaPDA-Interface-Micro-Epsilon	Single license for 2 scanners, expandable by up to 7 licenses for max. 16 connections
31.001017	ibaPDA-Interface-FBGS	Single license for 4 channels, expandable by up to 63 licenses for max. 256 channels
31.001018	ibaPDA-Interface-HTTP(S)	License for unlimited connections/modules for sending HTTP(s) requests
31.001019	ibaPDA-Interface-AMEPA	License for one connection to one AMEPA sensor (TSD); expandable for up to 256 connections
31.001020	ibaPDA-Interface-Modbus-TCP-Server	Single license for 64 connections, expandable by up to 3 licenses for max. 256 connections

Order no.	Product name	Description
31.001021	ibaPDA-Interface-Modbus-Serial	Driver license to use the Modbus protocol via serial interface (Master/Slave, RTU/ASCII)
31.001022	ibaPDA-Interface-Modbus-TCP-Client	Single license for 64 connections, expandable by up to 3 licenses for max. 256 connections
31.001024	ibaPDA-Interface-Modbus-TCP-Client-16	Single license for 16 connections, expandable by up to 15 licenses for max. 256 connections
31.001025	ibaPDA-Interface-PCLink-Automax	License to use a PCLink connection via Automax DCS Network to Rockwell (Reliance).
31.001026	ibaPDA-Interface-AN-X-DCSNet	License for communication via Automax DCS Network to Rockwell (Reliance) automation systems.
31.001027	ibaPDA-Interface-CAN	License for two connections to CAN bus devices via a CAN-to-Ethernet gateway
31.001030	ibaPDA-Interface-RAW-Ethernet	License to receive IEEE 802.3 Multicast Frames on max. 4 network interfaces.
31.001040	ibaPDA-Interface-S7-TCP/UDP	Single license for 64 connections, expandable by up to 3 licenses for max. 256 connections
31.001042	ibaPDA-Interface-PLC-Xplorer	License package for all currently available PLC-Xplorer-interfaces
31.001044	ibaPDA-Interface-Drive-Xplorer	License package for all currently available Drive-Xplorer-interfaces
31.001045	ibaPDA-Interface-SCRAM-Net	License for direct access to the Reflective Memory area of the SCRAM-Net SC150 interface card
31.001046	ibaPDA-Interface-Toshiba-ADMAP JAMI1	License for communication via an ADMAP network card; access to the variables of the connected stations; Scan Transmission, ADMAP-5M
31.001047	ibaPDA-Interface-TC-net	License for communication via a Tc-net 100 PCI, TC-net 100 PCIe or TC-net 1G network card. Max. 2 cards of the same type may be used.
31.001050	ibaPDA-Interface-SIMOLINK	License for data acquisition on the SIMOLINK drive bus via bus module ibaBM-SLM

Order no.	Product name	Description
31.001051	ibaPDA-Interface-GH180-Xplorer	Access to Innomotics Perfect Harmony GH180 medium voltage drives. Single license for one (1) connection, expandable by up to 254 licenses for max. 255 connections
31.001055	ibaPDA-Interface-Sisteam-TCP	License for communication with Ingelectric automation systems
31.001056	ibaPDA-Interface-TDC-TCP/UDP	Single license for 64 connections, expandable by up to 3 licenses for max. 256 connections
31.001065	ibaPDA-Interface-VIP-TCP/UDP	Single license for 64 connections, expandable by up to 3 licenses for max. 256 connections
31.001070	ibaPDA-Interface-EGD	Single license for 64 connections, expandable by up to 3 licenses for max. 256 connections
31.001075	ibaPDA-Interface-Generic-UDP	Single license for 64 connections, expandable by up to 3 licenses for max. 256 connections
31.001076	ibaPDA-Interface-Generic-TCP	Single license for 64 connections, expandable by up to 3 licenses for max. 256 connections
31.001080	ibaPDA-Interface-GCOM	Connection to the ABB Stressometer rolls
31.001090	ibaPDA-Interface-IEC61850-Client	Single license for 64 modules, expandable by up to 3 licenses for max. 256 modules
31.001100	ibaPDA-Interface-Hitachi-MicroSigma	Single license for connection to one IKS-LM-SN1G/-SN100 device in a µΣNETWORK-1000 network; including 32 modules with up to 4096 bytes each
31.001101	ibaPDA-Interface-Audio	Single module license for acquisition and recording of audio signals from the Windows system.
31.001111	ibaPDA-Interface-OPC-UA-Client	Single license for 16 connections, expandable by up to 15 licenses for max. 256 connections; number of signals limited to 1000 analog + 1000 digital signals per license.
31.001112	ibaPDA-Interface-MQTT	License for connecting one or more MQTT brokers in order to receive data
31.001113	ibaPDA-Interface-Oracle	Single license for 8 modules with interface to an Oracle DB, expandable by up to 3 licenses for max. 32 modules

Order no.	Product name	Description
31.001114	ibaPDA interface SQL Server	Single license for 8 modules with interface to a SQL Server DB, expandable by up to 3 licenses for max. 32 modules
31.001115	ibaPDA-Interface-PostgreSQL	Single license for 8 modules with interface to a PostgreSQL DB, expandable by up to 3 licenses for max. 32 modules
31.001116	ibaPDA-Interface-MySQL	Single license for 8 modules with interface to a MySQL DB, expandable by up to 3 licenses for max. 32 modules
31.001117	ibaPDA-Interface-SAP-HANA	Single license for 8 modules with interface to a SAP-HANA DB, expandable by up to 3 licenses for max. 32 modules
31.001220	ibaPDA-Interface-Reflective-Memory	Reflective Memory connection from GE Fanuc (formerly VMIC)
31.001350	ibaPDA-Interface-Profinet-CP	License to use the Siemens cards CP1616 or CP1626
31.001300	ibaPDA-Request-HPCi	HPCi system from GE Energy Power Conversion (formerly Converteam)
31.001302	ibaPDA-Request-HiPAC	HiPAC system from Danieli
31.001303	ibaPDA-Request-TwinCAT	Beckhoff controllers
31.001310	ibaPDA-Request-S7	SIMATIC S7 from Siemens via PROFIBUS/ PROFINET/ibNet-E/UDP
31.001320	ibaPDA-Request-SD	Simadyn D from Siemens
31.001330	ibaPDA-Request-TDC	SIMATIC TDC from Siemens
31.001340	ibaPDA-Request-X-Pact	X-Pact from SMS Siemag
31.001360	ibaPDA-Request-FM458/TDC	SIMATIC TDC from Siemens with FM458 board
31.001380	ibaPDA-Request-DTBox-128	DT-Box request interface for 128 signals
31.001381	ibaPDA-Request-DTBox-1024	DT-Box request interface for 1024 signals
31.001382	ibaPDA-Request-DTBox-unlimited	DT-Box Request interface for more than 1024 signals
31.001400	ibaPDA-Interface-IEC61850-9-2	Single license for 2 streams via the IEC61850-9-2 interface, expandable by up to 3 licenses for 2 streams each

2.5 Licenses add-ons

Order no.	Product name	Description
30.670040	ibaQPanel-Add-on	License for using ibaQPanel on an ibaPDA and/or ibaHD client
30.681223	ibaInSpectra	Complete package for ibaInSpectra functions (acquisition, recording and analysis of measured vibration signals, frequency band analysis, orbit analysis); 4 modules
30.681215	ibaInCycle	Functions for monitoring and analysis of cyclical processes and rotating machines, visualization by cycle view and circle view; 4 modules
30.770026	ibaPDA-Snapshot	Feature for buffering of signal data with subsequent computation; unlimited number of modules

2.6 Licenses for special data storage

Order no.	Product name	Description
30.670141/2/3	ibaPDA-Data-Store- SAP-HA-NA-64/256/1024	Data storage in an SAP HANA DB for 64/256/1024 signals
30.670160/1/2/3	ibaPDA-Data-Store-Kafka-16/64/256/1024	Data storage in an Apache Kafka cluster for 16/64/256/1024 signals
30.670180/1/2/3	ibaPDA-Data-Store-MindSphere-16/64/256/1024	Data storage in a Siemens MindSphere Cloud for 16/64/256/1024 signals
30.671000/1/2/3	ibaPDA-Data-Store-MQTT-16/64/256/1024	Data storage to an MQTT broker for 16/64/256/1024 signals
30.671020/1/2	ibaPDA-Data-Store-Oracle-64/256/1024	Data storage in an Oracle DB for 64/256/1024 signals
30.671030/1/2	ibaPDA-Data-Store-SQL-Server-64/256/1024	Data storage in an SQL-Server DB for 64/256/1024 signals
30.671040/1/2	ibaPDA-Data-Store-PostgreSQL-64/256/1024	Data storage in a PostgreSQL DB for 64/256/1024 signals
30.671050/1/2	ibaPDA-Data-Store-MySQL-64/256/1024	Data storage in a MySQL DB for 64/256/1024 signals
30.671060/1/2	ibaPDA-Data-Store-InfluxDB-64/256/1024	Data storage in a Influx-DB for 64/256/1024 signals

2.7 Upgrade policy

Upgrades can be purchased to increase the number of signals, clients, data storages or interfaces according to the table shown above.

Starting with an *ibaPDA-PLC-Xplorer* license, the purchase of an upgrade to a full *ibaPDA* license with at least 64 signals is required if you want to use Add-ons or other interfaces such as *ibaFOB*-cards etc.

2.8 Update policy

For *ibaPDA* our conditions for software maintenance and support apply.

In the first two years after purchasing the software, you can use the software maintenance and support services free of charge. After that, a maintenance contract with an annual fee is required.

Once a licensed software has been purchased it can be used thoroughly and independently from the software maintenance and support services!

Just the update of the used software towards a newer version is unacceptable without maintenance service fee.

If and to what extend the license extensions of your system affect the maintenance service fee can be learned usually from our offers.

Alternatively, you can contact our service and support team.

You can check the validity period for free updates and support services of your license wherever the license information for your dongle is displayed:

- *ibaPDA* I/O Manager: *General* tab - *Settings* node
- *ibaPDA* service status
- *ibaDongle viewer* (Marx Dongles only)

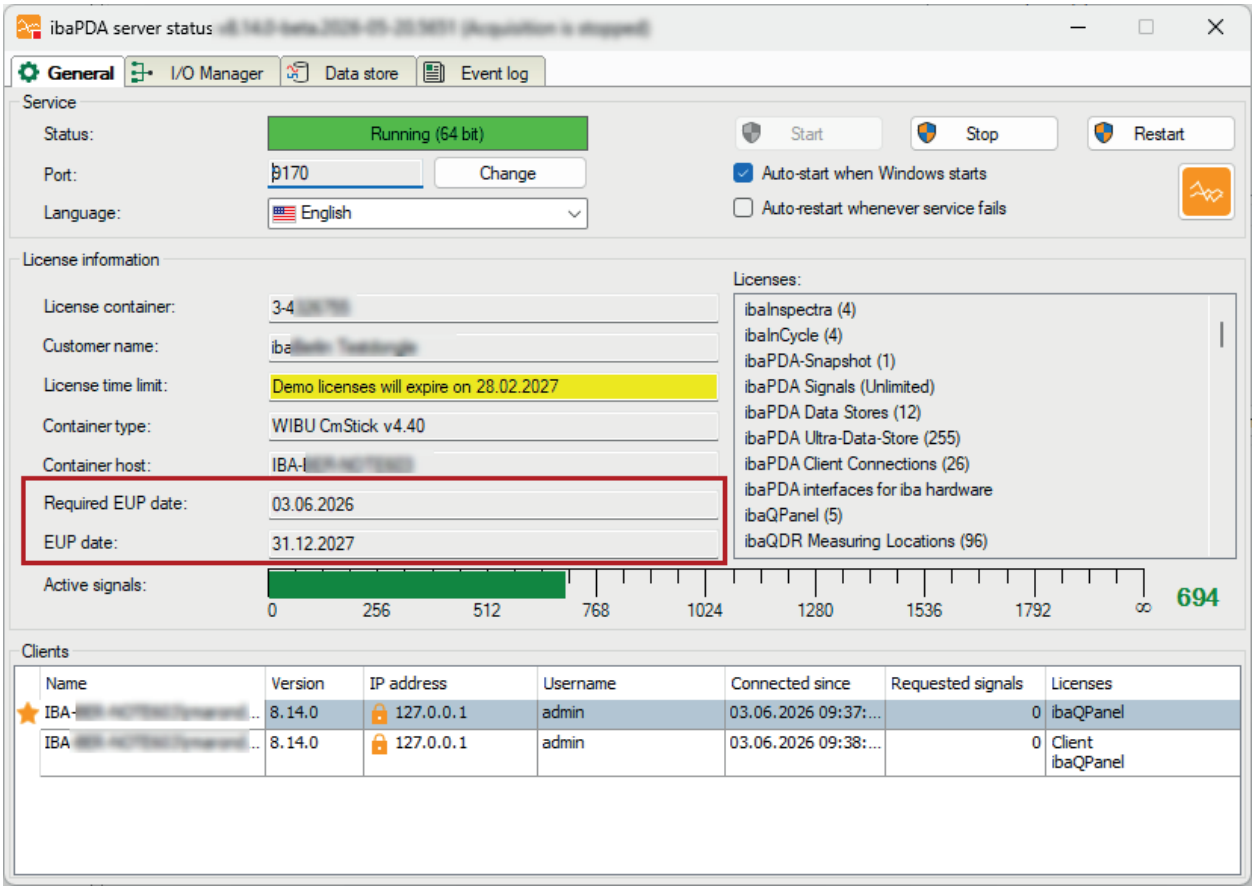
EUP date
In these programs you'll find the "EUP date". This date is set with purchase of an applicable software license (date of purchase + 2 years) and will be adapted with each extension of the update period (maintenance contract). It determines the end of the period for free updates. All major versions of a software, which had been and still will be released prior to this EUP date can be installed, including the associated bugfix versions. Relevant for this is not the date of installation but the release date of the software version.

Hence, you can still install updates after the EUP date, as long as versions are involved which had been released before the EUP date.

Required EUP date

In order to control the installation procedure there is the “required EUP date” which is hard coded in the software. Practically, this date corresponds to the production or release date of a major version of the software. The “required EUP date” is only visible in the dialogs of the corresponding software (*ibaPDA*, *ibaHD-Server*, *ibaCapture*), not in *ibaDongle-Viewer*.

During the installation, the “required EUP date” and the “EUP date” set in the dongle or in the soft license are compared in order to check if the installation is still permissable. You can find the EUP date of your software product e.g. in the status app.



If the “Required EUP date” is before the “EUP date,” then you still have update protection and can use the support.

If the “Required EUP date” is later than the “EUP date”, then the software version in question had been released outside of your update period and must not be installed.

You will be advised to this fact by the installation wizard. In this case, abort the installation and contact the iba support desk.

In the course of installation of *ibaPDA* you have the option in the installation wizard (step "EUP information") to search for available license updates. This facilitates the task in case you already have a DongleUpgrade available.

For further information, please see ↗ *Standard installation, page 25*.

2.9 Demo mode

ibaPDA supports a demo mode.

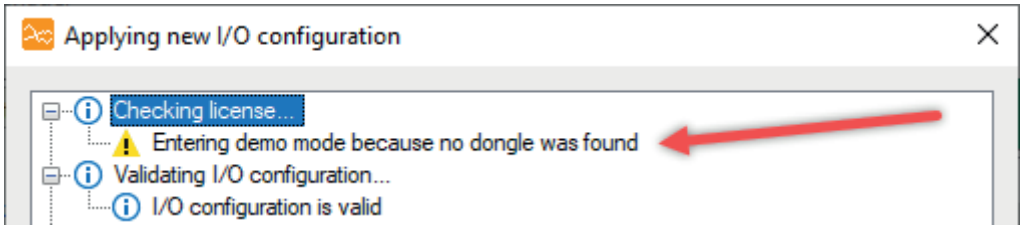
If no dongle is inserted, *ibaPDA* automatically switches to demo mode.

Only 2 types of interfaces are allowed in demo mode:

- Playback, for more information see Part 3, *Playback*
- Virtual, for more information see Part 3, *Virtual*

Up to 1024 signals are allowed. Data storage is disabled.

The switch to demo mode is displayed when validating the configuration.



This mode is suitable, in principle, to gain an impression of the system or to become familiar with the basic functions.

Note

If you have used *ibaPDA* based on a demo license the switch-over to demo mode does not work after the demo license expired.

If you want to continue using the software without another demo license then close the program, unplug the license container (dongle) and restart the program.

If you use a soft license, the license needs to be re-hosted in the *CodeMeter License Central WebDepot*.

You'll find detailed information about re-hosting licenses in the "WIBU CodeMeter for iba users" manual, in chapters *Re-hosting licenses* (with internet connection) or *Re-hosting licenses file-based* (without internet connection). Alternatively, you can find this information in the iba Help center:

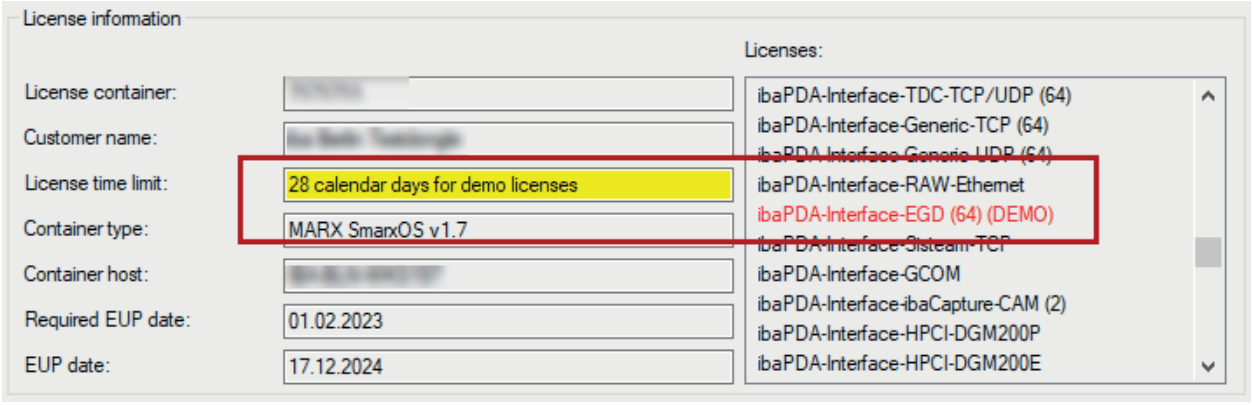
Re-hosting licenses or Re-hosting licenses file-based

2.10 Demo licenses

For testing license-dependent functions or interfaces that are not activated on the dongle or soft license on an existing system, an option is available to provide individual functions for demonstration purposes with a limited useful life.

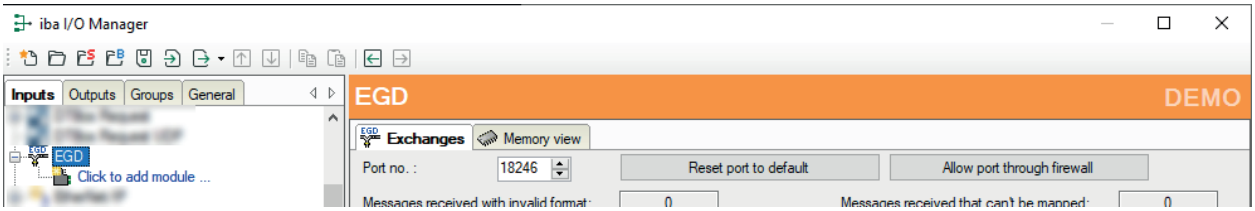
Users can therefore equip their existing system with new features and test them. Once the test time has elapsed, the function in question is disabled but the rest of the system continues to operate according to the original license time limit.

The functions that are enabled with a demo license are colored and identified with the addition of “(DEMO)” in the license options in the I/O Manager, *General* tab, *Settings* node. The remaining time for (all) demo licenses is displayed in the license time limit field. The following figure shows an example of a demo license for the *ibaPDA-Interface-EGD*.



The same information is displayed as in the server status program.

When you open the module of a demo license in the I/O Manager, the demo status is also indicated here by a yellow header bar and the word, "DEMO".



3 Installation and program start

This chapter provides information about the system requirements, the installation and about starting the program.

3.1 System requirements

Software

- Operating system: Windows 10 (x86/x64) version 1607 or higher, Windows 11 (x64), Windows Server 2016 (x64), Windows Server 2019 (x64), Windows Server 2022 (x64) or Windows Server 2025 (x64)
- .NET Framework v4.8 or higher

Note

Since v8.0.0 of *ibaPDA* the operating systems Windows 7 and Windows Server 2008 R are not supported anymore. Since v8.9.0 the installation of *ibaPDA* with these operating systems is not possible anymore.

Further pre-requisites and compatibility conditions can be found in the file `versions_pda.htm` which is part of the data storage medium "iba Software & Manuals" or of the download ZIP file. The following figure shows an example of this file.

Open this file in your Web browser and click on the header *Version compatibility.* Besides information about the current program version you can find a history, too.

Some older PC-cards made by iba as well as by other manufacturers are not supported by the x64 version anymore. You can find a listing of those cards in this file as well.

ibaPDA

Version history	Version compatibility	
Current version		
Version	Operating system	Requirements
	Windows 8.1 (x86/x64)	.NET Framework 4.8
	Windows 10 (x86/x64)	Version 1607 or later .NET Framework 4.8 (integrated in OS in version 1903 and later)
	Windows 11 (x64)	.NET Framework 4.8 (integrated in OS)
	Windows Server 2012 (x64)	.NET Framework 4.8
	Windows Server 2012 R2 (x64)	.NET Framework 4.8
	Windows Server 2016 (x64)	.NET Framework 4.8
	Windows Server 2019 (x64)	.NET Framework 4.8
	Windows Server 2022 (x64)	.NET Framework 4.8 (integrated in OS)
	Supported hardware	Remarks
	License	WIBU CmStick or CmActLicense MARX dongle
	ibaFOB-D, ibaFOB-Dexp, ibaFOB-io-ExpressCard, ibaFOB-io-USB, ibaFOB-SD, ibaFOB-SDe, ibaFOB-TDC, ibaFOB-TDCe, ibaCom-L2B, VMIC-5565, VMIC-5576, DGM200P, CP1616, CP1626, PC Link SST 5136-RE2-PCI	Supported on both x86 and x64 OS

Tip

You can also get this overview in the *ibaPDA* program under menu *Help – Version History*.

Hardware

- PC, Multicore CPU 2 GHz, 8 GB RAM, 512 GB HDD (program needs approx. 3.5 GB)

Virtual machines

The use of iba software in virtualized Windows systems is basically possible if no special PCI or PCIe cards are needed and slight response delays can be tolerated. Thus, the operation of *ibaPDA* using Ethernet-based interfaces (TCP/IP, UDP, etc.) in a virtual environment is possible.

It is not permitted to operate iba cards (*ibaFOB-D*, *-Dexp*, *-SD*, *-TDC*, etc.) in virtualized environments.

The provision of the licenses stored on an USB-dongle can be granted by use of a so called dongle server which can be purchased from iba AG. Alternatively, licensing can also be done via a soft license. If necessary, speak with your respective iba representative or iba Support.

The selection and adaption of the virtualization layer (e.g. VMWare, Microsoft HyperV, Citrix, XEN, Oracle etc.) and/or the use of hardware interfaces via I/O pass through are not in the responsibility of iba AG.

Note

Further specific system pre-requisites, e.g. for special interfaces, can be found in the respective chapters and manuals.

3.2 Installation

You can install the software either by using the installation wizard or by using the command line.

3.2.1 Standard installation

If an older version of *ibaPDA* is already installed, you can simply install the new one. The older version of the program is automatically removed after a prompt and confirmation. The settings and configurations chosen with the older version are preserved.

If you have a ZIP file of the new *ibaPDA* version (e.g., after a download), unzip it into a (temporary) folder of your choice.

You will find the file in the "...\01_iba_Software\ibaPDA" directory of the "iba Software & Manuals" data medium.

Execute the `ibaPDASetup_v8.exe` file, e.g., by double-clicking on the filename in Windows Explorer. Follow the instructions of the installation wizard.

1. Accept the license agreement.
2. Select the installation folder. (on first installation only)
3. Select the components to be installed:

- ibaPDA Client
  The client component must always be installed unless there is another computer in the network on which a client is installed and which has access to the local computer. Without a client, the server cannot be configured and signals cannot be viewed online. A local installation of the client on the server PC is recommended for being independent from the network.
- ibaPDA ActiveX Control
  You only need to select this component if, on the computer, *ibaPDA* is to be embedded as an ActiveX control in another application, like in an HMI application. See part 7, chapter *Setting up the ibaPDA client as an ActiveX control*.
- ibaPDA Server
  The server component must be installed if the measurement data is physically acquired and processed with the relevant computer locally.
  - ibaFOB-D network card
    This option is enabled for the server installation by default. The *ibaFOB-D* network drivers are required if the 32Mbit Flex protocol is to be used to connect to iba peripheral devices. Communication to the devices is carried out with an Ethernet protocol via the ibaNet fiber-optic connection. If you do not use *ibaFOB-D* cards or the 32 Mbit-Flex protocol, you can disable the installation of this driver. The drivers can be installed at any time. Note Hardware such as *ibaFOB*-cards is not supported by an *ibaPDA-PLC-Xplorer* license. The installation of the card drivers is not required but won't harm either.
  - Driver for iba-external hardware
    This option is enabled for the server installation by default. *ibaPDA* provides the driver for a number of external PC cards, such as reflective memory or DGM200P. If you are not using external cards, then you can disable the installation of this driver. The drivers can

be installed at any time. Note: Hardware such as ibaFOB-cards is not supported by an *ibaPDA-PLC-Xplorer* license. The installation of the card drivers is not required but won't harm either.

- ibaPDA-S7-Xplorer proxy
  You only need to select this component if you want to access a SIMATIC CPU with the S7-Xplorer interface by PC/CP without STEP 7 or SIMATIC Net being installed on the *ibaPDA* computer. For more information about this, Part 7, *ibaPDA-S7-Xplorer Proxy for PC/CP interface*.
- ibaDongleViewer
  This is not a component of *ibaPDA*, but rather a separate tool you can use to easily query the data of the license dongle (Marx Dongle only, no WIBU Dongle). It is offered for installation for all iba programs, is optional and only needs to be installed on a computer once.
- ibaManagementStudio Agent
  If there is an *ibaManagementStudio* server available in the network you can manage this *ibaPDA* installation by *ibaManagementStudio*. Therefore, you have to enable this option in order to install the *ibaManagementStudio* agent. If this program is already installed on your computer you can disable this option.

4. Check the EUP information. The EUP date should be in the future (green). In case you are still using an USB-dongle of "Marx CryptoBox Smarx" type and if a DongleUpgrade for this license is available you may enable the option "Search for license updates". If you enable this option and click on <Next>, the dialog of the ibaLicenseUpdater opens. Enter or browse the path where the DongleUpgrade files are stored. If a DongleUpgrade for this license is available in the path, first the DongleUpgrade process gets started before the installation continues. This setting (checkmark and path) will be stored and applied to upcoming installations.

5. If you are using the software "WIBU Codemeter Runtime" for licensing, the installer verifies the version of this software. If its version is outdated the installer offers to update this software. Please note, that other iba applications may use this software for licensing as well and be subject to stop and restart in the course of updating. If you you don't want to update the software at this point, just uncheck this option.

6. In the next step select further actions if needed.

7. Select the user account (on first installation only).
Use the user account to determine the user rights for the service process. To function properly, *ibaPDA* requires administrator rights, regardless of which user you choose. The default account is the local system account.

- Local system account (default)
  The local system account has administrator rights for the local computer. This is sufficient if the server only requires local access.
- This user account
  Select this option if you want to run the *ibaPDA* service (server) under a specific user account. Possible reasons for this are e.g. IT security aspects or the use of the OPC DA server.

8. Select the mode the program should be executed in (on first installation only).
ibaPDA (server) can run in 64-bit-mode or 32-bit-mode. A few functions are not available in

64-bit-mode. Consider the notes in the dialog in order to select the correct mode. This option is offered only for the first or a new installation. In case of updates the same mode like for the previous installation will be used.

9. Click <Next>. The installation process starts and the drivers are installed.

10. Then click <Finish> in order to finish the installation.

Note on installation of CodeMeter Runtime
iba software products use the WIBU licensing system to license certain features. The WIBU license containers are managed by the CodeMeter Runtime software which must be installed on the respective computer.

When installing the iba software the installer checks for existing CodeMeter Runtime installation and timeliness of version.

Along with an interactive installation (standard installation) a dialog opens which you can use to prevent or postpone the installation of CodeMeter Runtime. In case an update is offered, you have the option to prevent stopping other iba programs. This is relevant if other iba programs already use CodeMeter Runtime.

Along with a "silent" installation started by commandline, CodeMeter Runtime is only installed, if no previous installation exists. In such case, no running program on this computer uses this software. Updates of CodeMeter Runtime will not be initiated at all in "silent" mode.

3.2.2 Installing by command line

The installation of *ibaPDA* can also be started via a command line. This is help for the central software administration or when using deployment systems.

The installation process can also occur in so-called ‘silent mode’ so that the interaction with the user (clicking the <Next> button in the installation wizard) is omitted.

You can control how the installation is to go and which components are to be installed with the command line switches, which are set after calling up the installation program.

Notation (case-insensitive):

```
ibaPdaSetup_v8.x.y.exe [/HELP] [/SILENT | /VERYSILENT | /COMPO-
NENTS=xx,yy,...]
```

3.2.2.1 /HELP command line switch

This switch shows a message box containing all supported command line parameters.

3.2.2.2 /SILENT command line switch

This switch installs *ibaPDA* in "silent mode", i.e., interaction is not required during installation. All installer pages are skipped except the *Installing* page showing the installation progress.

The installer runs without specifying additional parameters, as if <Next> were always pressed in the dialog. In the case of a fresh installation, the program components are installed from the default settings (client + ActiveX + server including FOB-D drivers and third-party drivers + DongleViewer). The S7-Xplorer proxy is not installed. In the case of an installation update, the selected program components remain the same as the previous installation.

The selection of the execution mode (see standard installation step 8) in a silent installation is handled as follows. When installing for the first time, the 64-bit-mode is selected on an x64-system and the 32-bit-mode on an x86-system by default. When installing updates the same mode as of the previous installation is selected.

Combine this switch with /SUPPRESSMSGBOXES, in order to prevent message boxes popping up.

3.2.2.3 /VERYSILENT command line switch

Works like /SILENT, but nothing is shown.

Combination with /SUPPRESSMSGBOXES is recommended.

3.2.2.4 /SUPPRESSMSGBOXES command line switch

This switch prevents message boxes popping up. Always use in combination with /SILENT or /VERYSILENT.

3.2.2.5 /Dir= command line switch

This switch sets the installation directory.

Example:

```
ibaPdaSetup_v8.4.0.exe /Dir=c:\programfiles\iba\ibaPDA
```

3.2.2.6 /COMPONENTS= command line switch

Add the names of the components to be installed right after the equal sign in a comma-separated list. Only the components specified in the list will be installed.

Following components are permitted:

- ibaPDAClient
- ibaPDAActiveXControl
- ibaPDAServer
- ibaPDAServer\ibaFOBNetwork
- ibaPDAServer\NonIbaHW
- ibaPDAS7XplorerProxy
- ibaDongleViewer
- ibaManagementStudioAgent

Example:

```
ibaPdaSetup_v8.4.0.exe /COMPONENTS=ibaPDAClient,ibaPDAServer
```

This will install *ibaPDA* client and *ibaPDA* server.

3.2.2.7 /FORCECLIENTCLOSE command line switch

When updating an *ibaPDA* installation, the previous version is uninstalled first. When uninstalling the old version, the installer checks for open *ibaPDA* clients at this time. In the course of a standard installation, a dialog pops up in this case requesting the manual termination of the client. In the course of a “Silent mode” installation, the installation will be interrupted without notice.

This switch allows to terminate an eventually open client automatically along with the installation. This may cause the loss of configuration changes of the client, which have not yet been saved. You may use this switch in scripts facilitating the automatic installation of new *ibaPDA* versions.

The switch can be combined with other switches, e.g.:

```
ibaPDASetup_v8.5.0.exe /SILENT /FORCECLIENTCLOSE
```

3.2.2.8 /LOG, /LOG='filename' command line switch

This switch activates a logging function for the installation.

Alle steps and events in the course of installation will be logged in a file, which is stored in the user's TEMP folder, if you only use `/LOG`.

You can determine the storage location of the log file by specifying full path and filename with the switch `/LOG='filename'`, where 'filename' should be replaced by path and filename.

Example:

```
ibaPdaSetup_v8.6.0.exe /LOG=C:\iba\logs\MyInstallerLog.txt
```

3.2.2.9 /TASKS= command line switch

Use this switch to determine whether a desktop icon should be created after installation or not.

Example:

With desktop icon: `ibaPdaSetup_v8.4.0.exe /TASKS=desktopicon`

Without desktop icon: `ibaPdaSetup_v8.4.0.exe /TASKS=nodesktopicon`

3.3 Version alignment of ibaPDA-server and ibaPDA-Client

An *ibaPDA*-Client can establish a connection to an *ibaPDA*-Server of a different version without difficulty.

If the client version does not support features of a more recent server version, the use of these features is not possible or restricted. For instance, the I/O configuration or the data storage configuration may be opened in read-only mode only.

Correspondingly, the same applies if you use a client version for connecting to an older server version.

In case of such a version conflict you can use the command *Install client version from server* in the *Help* menu at any time in order to install the matching client version.

An alignment in the inverse direction, i.e. adaption of the server version according to the client version is not possible.

Note

If an *ibaPDA*-Server is installed on the client computer too, this method of adapting the client version is not possible. The client version should always match the server version on the same computer.

3.4 Start program

After *ibaPDA* had been installed as described in chapter *Installation*, the application can be started in different ways.

3.4.1 Startup of the server

The *ibaPDA* server is a service under Windows. It is configured by installation to start automatically when installation is finished and with each system start (boot).

After installation (incl. server) you can find the *ibaPDA* server service icon in the Windows task bar.

	ibaPDA server service and data acquisition are running.
	ibaPDA server service is running, data acquisition is stopped.
	ibaPDA server service is not running. A double-click on this icon opens the *ibaPDA* server status application to start the service.

Table 1: Taskbar icons

You can operate the basic server functions conveniently with the *Server Status* tool, which was automatically installed with the server.

You can start the *ibaPDA Server Status* program as follows:

- Double-click on the *ibaPDA* Server icon in the info tray of the task bar
- Using the context menu of the *ibaPDA* Server icon (right mouse click – *Status*)
- Using the Windows Start menu *All programs – iba – ibaPDA – Server Status*

3.4.2 Starting the client

After the installation, you will find the *ibaPDA* client icon on your Windows desktop.

The client process is started by a double-click on the icon.

Alternatively, you may start the client via the Windows program menu (Start button):

Start – All Programs – iba – ibaPDA – ibaPDA client

3.4.3 Starting the ibaPDA client via command line

3.4.3.1 General

The *ibaPDA* client can be started in the command line. This makes it possible to use batch files or scripts to start *ibaPDA* client(s). The syntax for such a command line is:

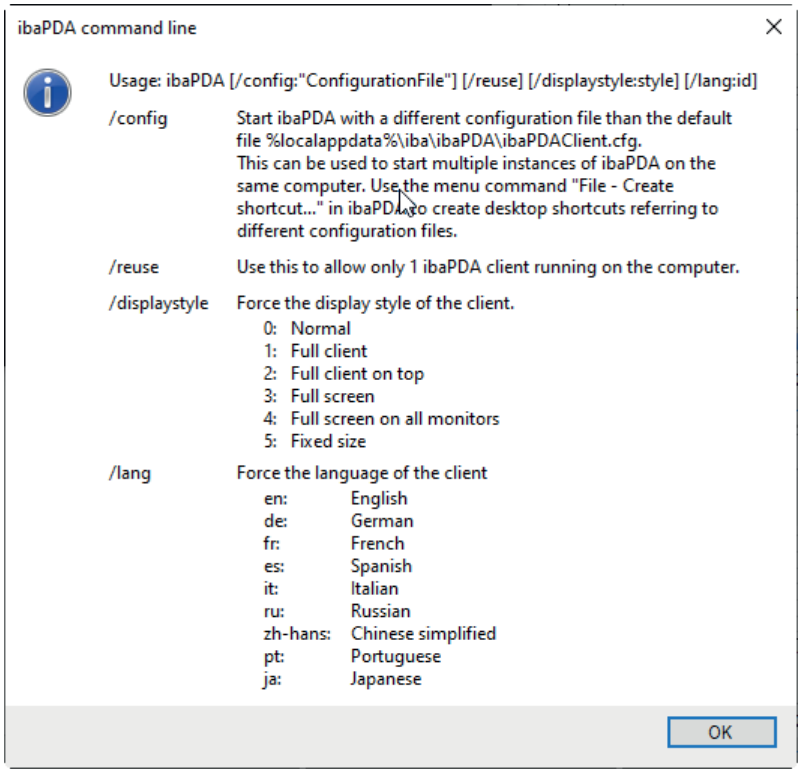
`ibaPDA.exe` */switch*

The "/switch" optional switch serves as a placeholder for different parameters, which can be used to start the *ibaPDA* client.

Full syntax description:

```
ibaPDA.exe [/config:"ConfigurationFile"][/reuse][/displaystyle:-
style][/lang:id]
```

If you enter the `ibaPDA /?` command in the command line, you receive complete assistance:



3.4.3.2 /config: command line switch

You can start the *ibaPDA* client with the /config: command line switch.

Using this function you can load an individual configuration. It is therefore possible to operate several clients on one computer. The individual clients can be connected to different servers, each of them with a different layout. The /config: switch requires a configuration file. The configuration file contains the server that the client connects to and the initial position and size of the client window. The directory containing the configuration file is also used as the base directory for the client files, e.g., the layout file, log files, address books, etc.

Example

```
ibaPDA.exe /config:"c:\ibaPda\Configuration1\Configuration 1.cfg"
```

The *ibaPDA* client will use the specified configuration file. The configuration file is an XML file with the following format.

XML configuration file

```
1   <?xml version="1.0"?>
2   <ClientConfiguration>
3   <Servers>
4   <Server>
5   <Address>localhost</Address>
6   <PortNr>9170</PortNr>
7   </Server>
8   </ Servers>
9   <AutoReconnect>false</AutoReconnect>
10  <Monitor>0</Monitor>
11  <FormPlacement>2;3;-1;-1;-1;-1;161;1152;189;867</FormPlacement>
12  </ClientConfiguration>
```

Lines 1 to 3
This is the general XML header.

Lines 4 to 9
This section contains the address and port number of the *ibaPDA* server to which the client will attempt to connect. By setting "AutoReconnect" to "true" or "false", you can specify whether or not the client should automatically attempt to establish a connection to the server. In this case, enter "true" for <AutoReconnect>.

Line 10
The "Monitor" setting determines on which monitor the client will appear. If "Monitor" is set to 0, the primary display will be used. If "Monitor" is set to 1, display number 1 will be used. If "Monitor" is set to 2, display number 2 will be used and so on.

If you are unsure about the numbering of your displays, use the *Identify* function in the Windows display settings of the relevant PC.

If another monitor is set in the "FormPlacement" setting, the "Monitor" setting is overridden.

Line 11
The "FormPlacement" setting makes the "Monitor" setting obsolete. *ibaPDA* saves the position, size and state of the main window in "FormPlacement" when it closes. If "FormPlacement" is available, *ibaPDA* will restore the position, size and status of the main window when *ibaPDA* starts. It is not advisable to manually change the values in "FormPlacement".

Tip

You can simplify the configuration by starting *ibaPDA* with a configuration file and then set up the display as desired. Afterwards, close the client again. When closing the client, the settings (including the selected monitor) are updated and saved in the configuration file. The next time you start the client with the the same configuration file, the client window opens at the selected position.

3.4.3.3 /reuse command line switch

Example

```
ibaPDA.exe /reuse
```

With this switch you can limit the number of clients on a computer to 1.

When the client is started with the /reuse switch, it will first search whether there is already another client running with this switch. If another client is found, the new client will activate the running client instead of starting itself.

Tip

If you do not want to start *ibaPDA* client using the command line, you can also do so by enabling the *Allow only one instance of ibaPDA client* option in the *ibaPDA* preferences, *General* Register.

3.4.3.4 /displaystyle: command line switch

```
ibaPDA.exe /displaystyle:0
```

Use this switch to start the *ibaPDA* client with a specific display style.

For more information, see ↗ *The View menu, page 46*.

The following switches are available:

/displaystyle:0	Normal
/displaystyle:1	Full client
/displaystyle:2	Full client on top
/displaystyle:3	Full screen
/displaystyle:4	Full screen on all monitors
/displaystyle:5	Resized to a predefined window size

3.4.3.5 /lang: command line switch

```
ibaPDA.exe /lang:de
```

Use this switch to start the *ibaPDA* client in a specific language.

The following switches are available:

/lang:de	German
/lang:en	English
/lang:fr	French
/lang:it	Italian
/lang:es	Spanish
/lang:ru	Russian
/lang:zh-hans	Chinese
/lang:ja	Japanese
/lang:pt	Portuguese

Note

With the simple online language switching in the *ibaPDA* client, it is possible that some objects or dialogs, such as the *ibaQPanel* toolbox, continue to be displayed in the original language in which the *ibaPDA* client was started.

Starting the *ibaPDA* client with the language switch has the advantage that all instanced objects of the application are started in the specified language.

Note

Note that some languages require licenses. If the desired language is not available, please contact the iba support or your regional iba representative.

3.4.4 Starting multiple clients on the same computer

By means of the *Create shortcut* command in the *File* menu, you can have multiple clients running on the same computer. This can then be very helpful especially when administrating different *ibaPDA* systems in large scale installations from a central workstation. The individual clients can be connected to different servers, each of them with a different layout.

The configuration file, which is stored with the creation of every shortcut, contains:

- the server that the client connects to and
- the initial position and size of the client window

The directory to which the configuration file is stored serves as the base directory for client files (layout file, log files, address books, etc.)

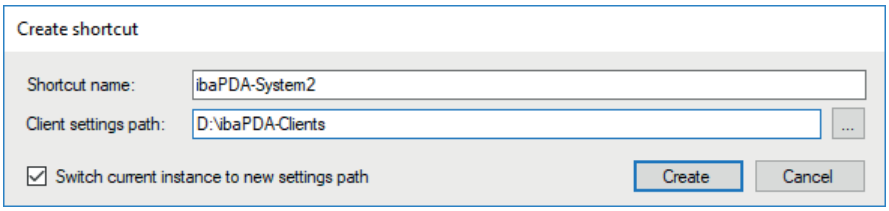
Without setting up different desktop shortcuts, you can start multiple clients with individual configuration files on a computer by using the command line call and the "/config" switch.

For more information, see ↗ */config: command line switch, page 33*.

In order to create a shortcut, follow these steps:

1. Go to the menu item *File – Create shortcut…*.

→ The *Create shortcut* dialog opens.



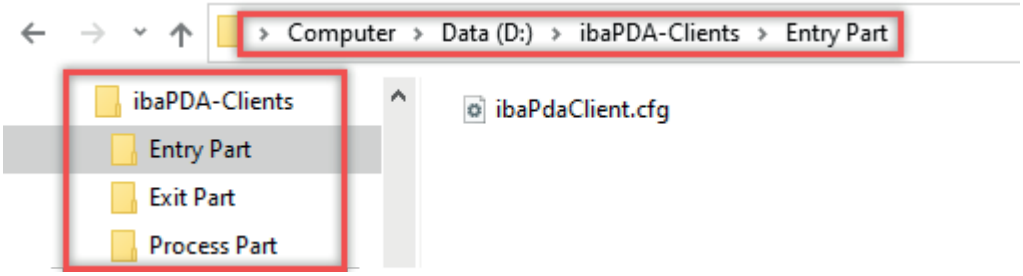
2. Enter a plain-text name for the shortcut, indicating the server it links to. By default, the name is formed from "ibaPDA" followed by the name of the currently connected server.
3. The path for the client settings is the base directory for the client files corresponding with the shortcut. Enter an appropriate path according to your needs if required. With the settings shown in the screenshot above, *ibaPDA* will create the `D:\ibaPDA-Clients\ibaPDA-System2` directory.
   In this directory, *ibaPDA* will then create a configuration file as well as the current layout file. A shortcut with the name, "ibaPDA-System2", is subsequently created on the desktop that links to `ibaPDA.exe /config:"D:\ibaPDA-Clients\ibaPDA-System2\ibaPDaClient.cfg"`.
4. Check the *Switch current instance to new settings path* option if you want *ibaPDA* to switch its current directory to the new shortcut directory. This will save all subsequent changes made to the layout or the server connection in the directory of the shortcut.
   If this selection box is activated, *ibaPDA* will act as if it was started from the shortcut. If the selection box is deactivated, *ibaPDA* will not change its current directory and no changes will be made.
5. Click the <Create> button.

If you want to create more shortcuts to other *ibaPDA* servers, iba AG recommends the following procedure.

1. Close the current instance of *ibaPDA* client and start a new one (not by the previous shortcut).
2. Connect the desired *ibaPDA* server.
3. Open the client preferences, select the *Connections* node and choose "Last connected server" for connection on startup. You may choose "Default server", but in this case make sure that the correct server is specified in the field *Default server*.
4. Then create a new shortcut as described above.

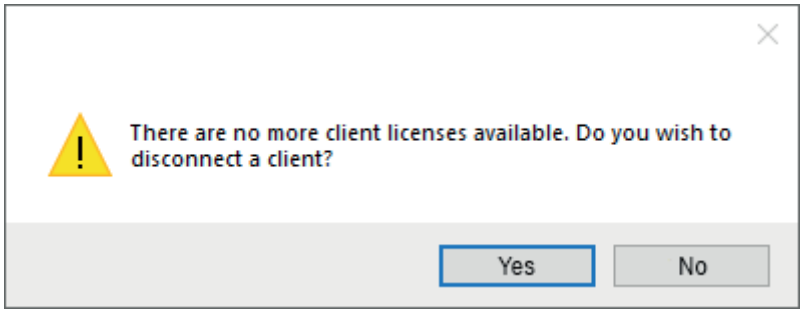
Example
In the case of a large plant with multiple *ibaPDA* systems, e.g., for the plant's lead-in area, processing area and lead-out area, you can create three shortcuts for the *ibaPDA* clients on your desktop.

In the file system, this appears as follows:

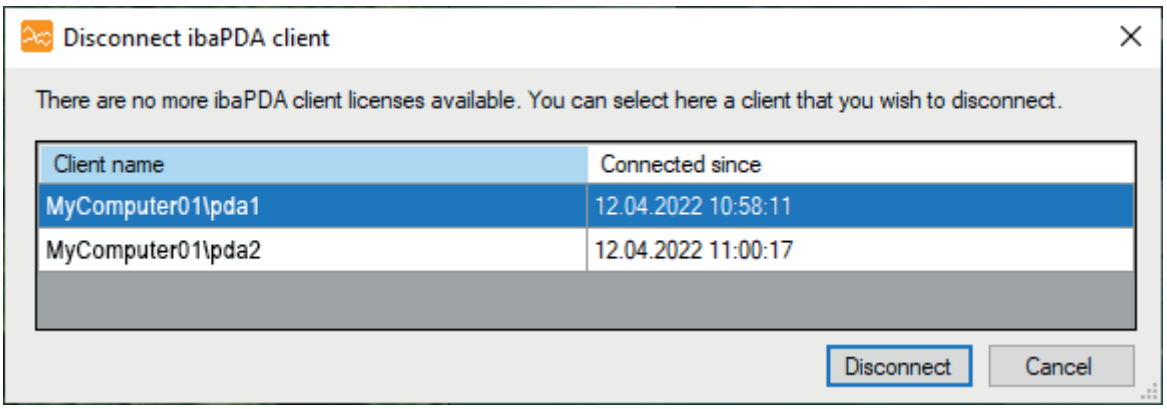


3.4.5 Disconnect another client on client startup

When a client tries to connect to an *ibaPDA* server and the server reports that there are no more client licenses available, the client displays the following message box.



If you click <Yes>, then a dialog opens:



The table shows all clients currently connected. Select one of them and click the <Disconnect> button in order to disconnect the client. If required, you will be asked to enter the password. If the other client is disconnected, your client will connect to the server and replace the other client.

Note

If the user management is active, this function can be disabled depending on the logged in user (*Server rights – Disconnect other user*).

4 User interface

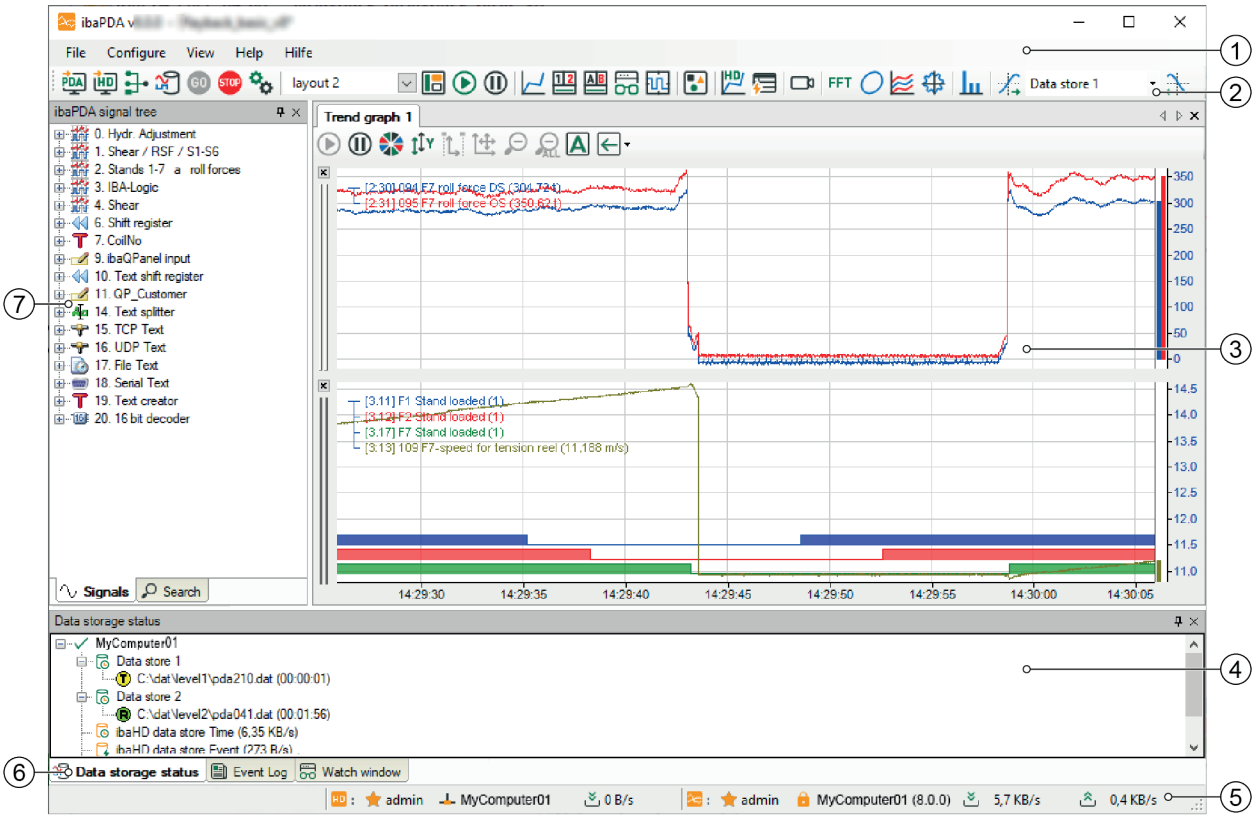
This chapter provides information about the main screen, menus, tool bars and hot keys.

4.1 The screen

The different areas of the *ibaPDA* screen are presented as an example in the following image.

However, the arrangement of the areas is not fixed. Signal views, signal tree, event log, data storage status and watch window are dockable windows which you can move or hide. You can adapt the user interface to your needs. The arrangement is always saved in a layout file.

In the following you'll find an overview of the default *ibaPDA* main screen.



Legend

1	Menu bar
2	Toolbar
3	Display area
4	Windows for the data storage status, watch window, event log
5	Status bar
6	Tab to switch windows
7	Signal tree and signal search

4.2 The menu bar

The main menu includes:

- File
- Configuration
- View
- Help

If you click on the *ibaPDA* symbol in the upper left corner (window header), a menu appears where you can minimize, move, resize or close the current window of the *ibaPDA* client. The *Close* command or double-clicking the icon closes the current window, i.e. the *ibaPDA* client. *ibaPDA* server, data acquisition and recording remain unchanged.

4.2.1 The File menu

The *File* menu offers some basic operations such as opening and saving projects, activation of password protection and exiting the program.

New project / Open project / Save project
These functions refer to the *ibaPDA* project management. You can save different projects, i.e. I/O configurations, data stores and layouts.

For more information, see ↗ *Configuring the system, page 59*.

Create shortcut
With this command, users can create shortcuts on the desktop for starting the *ibaPDA* client. By using this function, multiple clients can be run simultaneously on the same computer, as a configuration file is saved for every shortcut created. The individual clients can be connected to different servers, each of them with a different layout.

For more information, see ↗ *Starting multiple clients on the same computer*, page 37.

Print setup / Print
The print function includes only those parts, shown in the signal monitor area, i.e. trend graphs, FFT views, camera views, scope views, etc.
Log, status and signal tree windows will NOT be printed. By clicking on the *Print* command, the printing process of the entire signal monitor area starts immediately.

If you want to print only certain views, right-click on the caption of the subject view and select "*Print* from the context menu. The caption is standardized and contains date and time, layout name and the version number of *ibaPDA*.

Click on *Print setup* to open the dialog for the printer setup. In addition to the common print settings such as paper size, orientation, etc., you can decide whether or not to add a caption and/or a border around the image.

To change the color settings for printing, see ↗ *Settings for printing views, page 108*.

Log out from ibaPDA server/ibaHD server
This command drops the connection to the *ibaPDA* server or *ibaHD-Server*.

Exit
Exit the *ibaPDA* client program. The server and hence the data acquisition and data storage continue.

4.2.2 The Configuration menu

The *configuration* menu offers all important functions for system configuration, data acquisition and data storage.

Select ibaPDA server...
This menu function opens the *Select ibaPDA server* dialog (see ↗ *ibaPDA-server, page 67*) for establishing a connection between the *ibaPDA* client and an *ibaPDA* server.

You can open this dialog also by this button in the toolbar:

In the "Address" field, enter the name of the server PC in the network. In the "Address" field, enter the name of the server PC in the network. If client and server run on the same computer, just enter "localhost".

The system automatically detects and lists all active servers available in the network. You only need to select the desired server from the list and press <OK> in order to establish the connection. The port number will be given automatically.

If the desired server is not in the list, you can use the search function. Click on the <Search> button. The bar indicates the search progress.

You can establish a connection to multiple servers one after another, e.g. for configuration purposes.

Select ibaHD server (only available if the license for the *ibaHD-Server* is available)
This menu function opens the dialog for selecting an *ibaHD-Server* for historical data recording. This function requires a license for the *ibaHD-Server* and *ibaHD-Server* client.

You can open this dialog also by this button in the toolbar:

For more information, see Part 6, *Connecting to HD server*.

I/O Manager
This menu function opens the dialog of the *iba I/O Manager,* hereafter referred to as *I/O Manager.* This is the main dialog for the configuration of the signals to be measured, groups and outputs.

You can open this dialog also by this button in the toolbar:

With the I/O Manager, you define among others

- the integrated servers, such as *OPC- , OPC UA- , SNMP- and IEC 61850-Server*
- the *Address books* of the PLC interfaces
- *Time synchronization*, *Knowhow protection* and *Certificates*
- the *data sources* (interfaces) you want to use
- the *modules* (devices) to be connected
- the *analog* and *digital signals* as well as the *text signals*, you want to acquire
- the *virtual signals* you want to use
- the *signal groups* containing selected signals
- the *outputs*, i. e. signals which should be sent from *ibaPDA* to other systems over appropriate interfaces or by e-mail.

In addition, real time diagnostics provides information on the status of the connected

- modules in this dialog
- actual values of the signals

A bar chart always informs you about the actual number of signals used with regard to your license limit.

Data Store Manager
This menu item opens the configuration dialog for the *data storage*.

You can open this dialog also by this button in the toolbar:

In this dialog you can define and configure

- the *archive profiles*, you want to use for the data recordings
- the different *data stores* for creating standard data files, but also for *ibaQDR*- and *ibaHD-Server*-data stores as well as for sending data to external systems like databases, MQTT-broker etc.

Furthermore, for each data store

- the trigger settings and the recording mode
- the selection of the signals to be stored
- file name and path of the data files
- Usage of text signals
- Automatic cleanup for the hard disk
- Buffer (for HD and database/cloud related data stores only)

User Management
This menu item opens the configuration dialog for the user management.

The user management provides service for

- adding or removing users
- assigning user rights for server and client functions
- creating and changing passwords

Layout Manager
This menu function opens the dialog for layout management.

You can open this dialog also by this button in the toolbar:

With the Layout Manager you can make general settings for the layout, create layouts and assign layouts to users. You can save, load, export and import the created layouts here.

For more information, see Part 6, *Managing layouts*.

Server Access Manager
By using this menu function, you open a dialog for reserving client-licenses for selected clients. Furthermore, you can filter by clients (IP addresses) in order to limit the access to the *ibaPDA* server.

For more information, please see ↗ *Server access control, page 81*.

Start
By using this menu function, you start the data acquisition (measurement) on the connected server.

You can open this function also by this button in the toolbar:

This function can be locked with a *password.*

At start of the measurement, a pop-up window will inform you about the start sequence.

Stop
With this menu function, you stop the data acquisition (measurement) on the connected server.

You can open this function also by this button in the toolbar:

This function can be locked with a *password.*

After the Stop command is issued, a warning will appear which must be confirmed before the measurement is stopped.

Log out from ibaPDA server/Log out from ibaHD server
With these menu functions you can selectively log out the actual client from the corresponding server, e.g. in order to unblock client licenses. The function is only available if the user management is enabled.

Client preferences
This menu item opens the configuration dialog for the signal monitor and screen layout preferences.

You can open this dialog also by this button in the toolbar:

In this dialog, you can define the following:

- General settings for the client (behavior, buffer, signal names etc.)
- Display properties of all views for print and display
- Scaling settings for *X and Y axis* of the graphs

Note

These settings will be stored in the local registry of the client computer. They apply exclusively to the *ibaPDA* client instances running on the same computer but not to clients on other computers.

For more information, see Part 6, *Data visualization*.

4.2.3 The View menu

This menu offers all the features for the design of the screen and the management of different panes and layouts. The term "layout" means the individual arrangement of signal monitors, signal selection and status information windows. At program startup, *ibaPDA client* uses the `CurrentLayout.lay` configuration file.

Add view
By clicking on this menu item, a submenu opens providing a selection of different display types.

Symbol	Function	Description
	Add Trend graph	Adds a new trend graph view. All views on the screen reorganize automatically. Every signal monitor has its own scrolling control, zoom function and X axis. For more information, see Part 6, *Trend graphs*.
	Add Numeric digital display	Adds a new digital numeric display to the digital display window. Afterwards, you can drag the digital display out of the window and drop it at a different position on the screen. For more information, see Part 6, *Numeric digital display*.
	Add Text digital display	Adds a new digital text display to the digital display window. Afterwards, you can drag the digital display out of the window and drop it at a different position on the screen. For more information, see Part 6, *Text digital display*.
	Add Watch view	Adds a new watch view to the signal monitoring area, in order to display signal values in tabular manner. For more information see Part 6, *Watch view*.
	Add Scope view	Adds a new oscilloscope view to the signal monitor area. For more information, see Part 6, *Scope view*.
	Add ibaQPanel	Adds a new *ibaQPanel* view to the signal monitoring area. The QPanel view requires the *ibaQPanel* add-on. For more information, see Part 6, *ibaQPanel*.
	Add HD trend graph	Adds a new trend graph of historical data to the signal monitor area. A connection to an *ibaHD server* as well as a time-based storage are required to display data. For more information, see Part 6, *Historical data trend graph (HD trend graph)*.
	Add HD event table	Adds an event table to the signal monitoring area. A connection to an *ibaHD server* as well as an event-based HD storage are required to display events. For more information, see Part 6, *HD event table.*

Symbol	Function	Description
	Add HD time period table	Adds a time period table to the signal monitoring area. A connection to an *ibaHD server* as well as a time-based HD storage are required to display time periods. For more information, see Part 6, *HD time period table.*
	Add camera view	Adds a new camera view to the signal monitor area. A connection to an *ibaCapture* server is required to display video images. For more information, see Part 6, *ibaCapture camera view*.
	Add FFT view	Adds a new FFT view to the signal monitor area. To use the FFT view in its full functional extend, the add-on *ibaInSpectra* is required. For more information, see Part 6, *FFT view*.
	Add Orbit view	Adds a new orbit view to the signal monitor area. The Orbit view requires the *ibaInSpectra* add-on. For a full description of the view, please read the corresponding manual for the software product *ibaInSpectra.*
	Add Cycle view	Adds a new cycle view to the signal monitoring area, in order to display values of an *ibaInCycle* module. For a full description of the view, please read the corresponding manual for the software product *ibaInCycle.*
	Add Circle view	Adds a new circle view to the signal monitoring area, in order to display values of an *ibaInCycle* module. For a full description of the view, please read the corresponding manual for the software product *ibaInCycle.*
	Add PQU spectrum	Adds a new spectrum bar chart view to the signal monitor area. The view is designed with respect to the device for power quality analysis (*ibaPQU-S*) and shows the harmonics. For more information, see Part 6, *PQU spectrum*.
	Add Phasor view	Adds a new vector diagram to the signal monitor area. The view is designed with respect to the device for power quality analysis (*ibaPQU-S*) and shows voltage and current vectors. For more information, see Part 6, *Phasor view*.

Table 2: Toolbar icons for views

Hide tabs

With this menu function, you can hide the tabs showing the name of the respective signal monitor. Thereby, you can save some space when using grouped (stacked) views. Note that you only

see the current topmost monitor and that there is no way to switch to the other monitors behind unless tabs are shown.

Hide tab borders
This menu function allows you to hide the frames of tabs. This is useful for "Full screen" or "Full client" mode.

ibaPDA signal tree
This menu function allows you to show and hide the signal tree window of the ibaPDA server.

ibaHD signal tree
This menu function allows you to show and hide the signal tree window of a historical data server.

Data storage status
This menu function allows you to show and hide the data storage status window.

ibaPDA event log and ibaHD event log
These menu functions allow you to show and hide the event log windows of the corresponding servers.

Watch window
This menu function allows you to show and hide the watch window for the tabular display of signals and their actual values. Operation and function is the same as for *Watch view*.

Digital displays
This menu function allows you to show and hide the window for the digital numeric and text displays. The window is dockable and is shown when a new digital display is added via the menu or the toolbar button. If the digital display window is opened, you can add digital displays by dragging more signals or text signals into the window.

Reset window layout
This menu function allows you to reset all free floating windows (e.g. signal tree, watch window, data storage status etc.) in the currently active layout to their default position, size and visibility state.

There are similar commands available in the Layout Manager in order to apply this function to all layouts and not just the currently active layout.

For more information see *ibaPDA* manual part 6, chapter Layout pool.

Display style
This menu function provides a submenu offering 6 different display styles. You can activate and deactivate the display styles here.

The keyboard shortcuts allow you to switch (toggle) between the "Normal" display style and the selected style.

Tip

It makes sense to know the keyboard shortcuts by heart or to have them at hand in case operation with the mouse is not possible (since toolbars or menus are no longer shown).

You can press the <F10> key several times to set and exit all of the aforementioned styles. Use the <ESC> key to exit any style and go back to the normal display style.

Style	Shortcut key	Description
Normal		Standard Windows frame with signal monitor area, toolbars, menus, signal tree Data storage status event log, data storage status, etc.
Full client	<Ctrl>+<Shift>+<C>	The signal monitor area (with the layouts) is expanded over the full area of the *ibaPDA* client window including the frame; no toolbars, menus or other windows are visible. Other Windows programs can be positioned above the *ibaPDA* client. When you restore the *ibaPDA* client window, you can still see the desktop background.
Full client on top	<Ctrl>+<Shift>+<T>	Equivalent to *Full client*, but the *ibaPDA* client is always in the foreground.
Full screen	<Ctrl>+<Shift>+<F>	Equivalent to *Full client on top*, but the signal monitor area covers the whole screen. The Windows task bar is also covered.
Full screen on all monitors	<Ctrl>+<Shift>+<A>	Equivalent to *Full screen* when using several monitors.
Fixed size	<Ctrl>+<Shift>+<S>	Shows the Client window in size and with options as set in dialog *Configure fixed size …* (in the same menu). You can adjust the size of the client pixel-precisely and enable/disable some options regarding tabs and scroll bars. The settings of *Hide tabs* and *Hide tab borders* overwrite the corresponding settings made in view menu above. If a layout contains a QPanel you can use the *Fixed size* display style to place and move the QPanel view freely on the screen. Therefore, an empty area in the QPanel without any element is required in order to grab and drag the view with the mouse.

Table 3: Client display modes

You can use these display modes to maximize the available display area for signal monitors or to avoid an unauthorized use of the *ibaPDA* client. However, we recommend user management to effectively restrict operational access.

Language
The *Language* submenu makes it possible to switch between the GUI languages. The menu offers all the languages that are available in *ibaPDA*. note that some languages are only available with a corresponding license.

Free languages:	Licensed languages:
■ English ■ German ■ French	■ Spanish ■ Portuguese ■ Italian ■ Russian ■ Chinese ■ Japanese

ibaQPanel language dictionary
With this menu item, you open the dialog for editing the dictionary for the add-on *ibaQPanel*. In this dictionary you can enter all the texts which have been configured in the *ibaQPanel* in all available languages. Depending on the language setting in the *Language* menu of the display, the texts in the *ibaQPanel* user interface will switch their language accordingly.

Unit system
Under this menu item you can switch to a different unit system for the proper display of numerical values given in physical or other units. As a pre-requisite you should have configured one or more unit systems in the unit table and have enabled this function in the I/O Manager, section *General - Unit table*. The default setting is "Metric" (for the metric unit system).

4.2.4 The Help menu

The help menu provides information about the *ibaPDA* software version (*Version history…*), changes (*Version History…*) and iba contact information (*Support…*). In addition, it offers a user-friendly function in case you need help.

Help
This menu item opens the *ibaPDA*-online help.

Contact iba Support
Here, you will find the iba Customer Support and contact information for all branch offices worldwide.

Save information for iba Support
With this command, you can create a ZIP file containing the following information:

- the current project with the current layout
- a complete log directory of the server
- system information about the server

This command allows the user to gather all information relevant to the support with only one click. In general, we recommend submitting an updated support file to our support desk.

You can also create a support file via command line, e.g. by executing a script. The program `ibaPdaGenerateSupportFile.exe` is available for that purpose.

You will find more detailed information about how to generate a support file by command line in the Appendix of the ibaPDA manual (part 7).

Version history
This menu function starts the Internet file in an Web browser showing the history of the different software versions. We recommend having a look at this file after every *ibaPDA* update. Here, you will find interesting information about new functions, improvements and bugfixes.

Install client version from server
This command allows you to download the *ibaPDA* client version from the *ibaPDA*-Server you are currently connected to and install it on the client computer. This way, you install the appropriate client version which matches the server's version in order to gain the best functional compliance. We recommend doing so particularly if the client version is lower than the server version.

About
Here, you'll find relevant information about the current version of the *ibaPDA* software (client, server) as well as about the license number and the registered customer name.

4.3 Toolbars

The toolbars offer the most important functions you can also find in the menus.

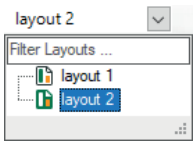
ibaPDA comes with a default toolbar and menu configuration. You can customize the toolbar and menus according to your needs, save them as a profile and assign them to *ibaPDA* users.

For more information, see chapter *Manage layouts* in part 6 of the *ibaPDA* manual.

4.3.1 Configuration toolbar part

Symbol	Name	Description
	Select ibaPDA server...	Opens the dialog for the connection to an *ibaPDA* server. For more information, please see ↗ *Select an ibaPDA server, page 68*.
	Select ibaHD server...	Opens the dialog for the connection to an *ibaHD-Server*. For more information, see Part 6, *Connecting to HD server*.
	I/O Manager	Opens the I/O Manager. For more information, see Part 2 *I/O-Manager*.
	Data Store Manager	Opens the dialog for the data storage configuration For more information, see Part 5 *Data storage*.
	User Management	Opens the dialog for the ibaPDA and ibaHD-Server user management. For more information , see chapter ↗ *User management, page 85*.
	Start	Starts the acquisition on the server
	Stop	Stops the acquisition on the server
	Client preferences	Opens the dialog for the preferences of signal monitors and views, printing and preferred ibaPDA/ibaHD server connections. For more information, see Part 6, *Display preferences and view properties*.

4.3.2 Layout toolbar part

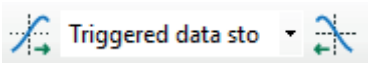
Symbol	Name	Description
	Selection list Layout	Pick-list for selecting the desired and saved layout (overview). You can also toggle between layouts with the <F11> function key.
	Layout Manager	Opens the dialog for managing the layouts. In this dialog you can make general settings for the layout, create layouts and assign layouts to users.
 	Start views / Stop views	Starts or stops the feeding of trend graphs. This function applies to all visible trend graphs of any view.

Symbol	Function	Description
	Add Trend graph	Adds a new trend graph view. All views on the screen reorganize automatically. Every signal monitor has its own scrolling control, zoom function and X axis. For more information, see Part 6, *Trend graphs*.
	Add Numeric digital display	Adds a new digital numeric display to the digital display window. Afterwards, you can drag the digital display out of the window and drop it at a different position on the screen. For more information, see Part 6, *Numeric digital display*.
	Add Text digital display	Adds a new digital text display to the digital display window. Afterwards, you can drag the digital display out of the window and drop it at a different position on the screen. For more information, see Part 6, *Text digital display*.
	Add Watch view	Adds a new watch view to the signal monitoring area, in order to display signal values in tabular manner. For more information see Part 6, *Watch view*.
	Add Scope view	Adds a new oscilloscope view to the signal monitor area. For more information, see Part 6, *Scope view*.
	Add ibaQPanel	Adds a new *ibaQPanel* view to the signal monitoring area. The QPanel view requires the *ibaQPanel* add-on. For more information, see Part 6, *ibaQPanel*.
	Add HD trend graph	Adds a new trend graph of historical data to the signal monitor area. A connection to an *ibaHD server* as well as a time-based storage are required to display data. For more information, see Part 6, *Historical data trend graph (HD trend graph)*.

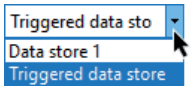
Symbol	Function	Description
	Add HD event table	Adds an event table to the signal monitoring area. A connection to an *ibaHD server* as well as an event-based HD storage are required to display events. For more information, see Part 6, *HD event table.*
	Add HD time period table	Adds a time period table to the signal monitoring area. A connection to an *ibaHD server* as well as a time-based HD storage are required to display time periods. For more information, see Part 6, *HD time period table.*
	Add camera view	Adds a new camera view to the signal monitor area. A connection to an *ibaCapture* server is required to display video images. For more information, see Part 6, *ibaCapture camera view*.
FFT	Add FFT view	Adds a new FFT view to the signal monitor area. To use the FFT view in its full functional extend, the add-on *ibaInSpectra* is required. For more information, see Part 6, *FFT view*.
	Add Orbit view	Adds a new orbit view to the signal monitor area. The Orbit view requires the *ibaInSpectra* add-on. For a full description of the view, please read the corresponding manual for the software product *ibaInSpectra.*
	Add Cycle view	Adds a new cycle view to the signal monitoring area, in order to display values of an *ibaInCycle* module. For a full description of the view, please read the corresponding manual for the software product *ibaInCycle.*
	Add Circle view	Adds a new circle view to the signal monitoring area, in order to display values of an *ibaInCycle* module. For a full description of the view, please read the corresponding manual for the software product *ibaInCycle.*
	Add PQU spectrum	Adds a new spectrum bar chart view to the signal monitor area. The view is designed with respect to the device for power quality analysis (*ibaPQU-S*) and shows the harmonics. For more information, see Part 6, *PQU spectrum*.
	Add Phasor view	Adds a new vector diagram to the signal monitor area. The view is designed with respect to the device for power quality analysis (*ibaPQU-S*) and shows voltage and current vectors. For more information, see Part 6, *Phasor view*.

Table 4: Toolbar icons for views

4.3.3 Data storage toolbar part



The buttons in this part of the toolbar allow you to manually control the data storage. In order to avoid erroneous operation, this toolbar is not shown by default. It must be activated manually.

Symbol	Name	Description
	Start	Manual triggering of the start trigger; starts the recording, which is selected in the field on the right next to the button. This button is only enabled if a start trigger has been defined for the recording. Very useful for testing and commissioning
	Stop	Manual firing of the stop trigger; stops the recording, which is selected in the field on the left next to the button. The button is enabled for both triggered and continuous recordings. Thus, also continuous recordings can be stopped manually. Very useful for testing and commissioning
	Select data store	Pick-list for selecting the desired data storage to be triggered manually.

4.4 Status bar

The status bar at the bottom of the *ibaPDA* program window contains some information about the connection between client and server:

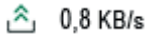
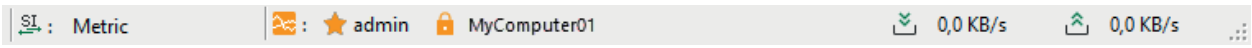
Symbol	Description
	A connection has been established between client and server. On the right side of the symbol the name of the connected server and the version of the server software (in brackets) are displayed.
	Connection established between *ibaPDA* client and *ibaPDA* server with status of main connection (MC) and data connection (DC): MC+DC encrypted / only MC encrypted / MC+DC not encrypted For details, see ↗ *Server access control, page 81*
	The connection between client and server is broken; no server connected
	Data rate (downstream) from server to client Refers to the load of the server due to the data that it has to send to the client.
	Data rate (upstream) from client to server
	Server type: PDA server, HD server
	The administrator or user currently logged on
	The currently selected unit system

Table 5: Status bar symbols

Example



ibaPDA client is connected to the *ibaPDA* server on the MyComputer01 computer and logged in as the administrator. Actual data upload rate amounts to 0.0 kBps and download rate to 0.0 kBps. The currently selected unit ystem is metric.

4.5 Operation via keyboard shortcuts

General and display

Keyboard	Function
<Ctrl>+<U>	Open user management
<Ctrl>+<Shift>+<C>	*Full client* display style on/off
<Ctrl>+<Shift>+<T>	*Full client on top* display style on/off
<Ctrl>+<Shift>+<F>	*Full screen* display style on/off
<Ctrl>+<Shift>+<A>	*Full screen on all monitors* display style on/off
<Ctrl>+<Shift>+<U>	Menu to customize toolbars and menus
<F11> / <Shift>+<F11>	Next/previous layout
<Ctrl>+<D>	QPanel display: Designer mode on/off
<Ctrl>+<I>	Open I/O Manager
<Ctrl>+<O>	Open Data Store Manager
<Ctrl>+<P>	Print all visible views
<Ctrl>+<L>	Open Layout Manager

Signal tree and signals

Keyboard	Mouse	Function
<Ctrl>	Double-click	a signal in the signal tree: Adds the signal to an existing and marked signal strip (trend graph) and attaches it to the Y axis of the lowest signal.
<Shift>	Double-click	a signal in the signal tree: Inserts a signal in a marked trend graph display with its own Y-Axis.
<Ctrl>	Drag & Drop	a signal in the signal tree: Inserts a signal in the trend graph display and attaches it to the Y-axis of the lowest signal when it is dropped in the free area of the display.
<Shift>	Drag & Drop	a signal in the signal tree: Inserts a signal in trend graph display with its own Y-axis when it is dropped in the free area of the trend view.

Marker in trend graph

Keyboard	Mouse	Function
<Ctrl>	Move marker	Marker locks in place precisely at the samples.
<Shift>	Move marker	Markers X1 and X2 move in the same direction at a constant distance
<Ctrl>+<Shift>	Move marker	Markers X1 and X2 lock into place precisely at the samples and move in the same direction at a constant distance

I/O Manager

Keyboard	Mouse	Function
	Double-click	Toggling between binary settings, such as true/false; Double-click on the field where "True" or "False" appears, e.g., "Locked" in the general module settings.
<Ctrl>+<E>		Enable/disable the selected module

5 Configuring the system

ibaPDA is configured in the *ibaPDA* client program. So, the first step is the establishment of a connection between the client and the server. If only one server is available, usually *ibaPDA* automatically establishes a client server connection. If, however, a network includes several *ibaPDA* servers, you ensure your client to be connected to the right server before starting the configuration.

For more information, see ↗ *Select an ibaPDA server, page 68*.

Generally, the system configuration consists of 3 parts:

1. The configuration of the data sources of the signals to be measured
   The data sources and signals to be measured are generally configured in the I/O Manager.

   Also see Part 2, *I/O Manager.*
2. The configuration of the data storage
   In the dialog of the data storage, you configure the type of data storage in data files, databases or *ibaHD-Server*.

   Also see Part 5, *Configuring data storage*.
3. The configuration of the display layout
   The layout of *ibaPDA* client is configured on the screen itself by arranging signal displays, views and windows.

   The layout configuration is mainly described in Part 6, *Data visualization*.

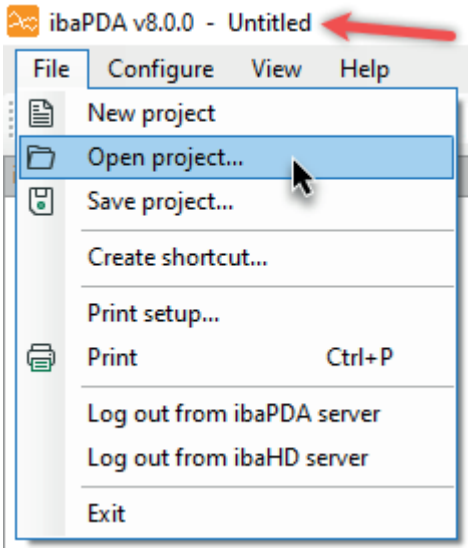
You can store each of the different configurations individually in configuration files as described in the above-mentioned chapters. For a more convenient, secure and flexible management of different configurations there are two more features:

- Project management
- External configuration

5.1 Project management

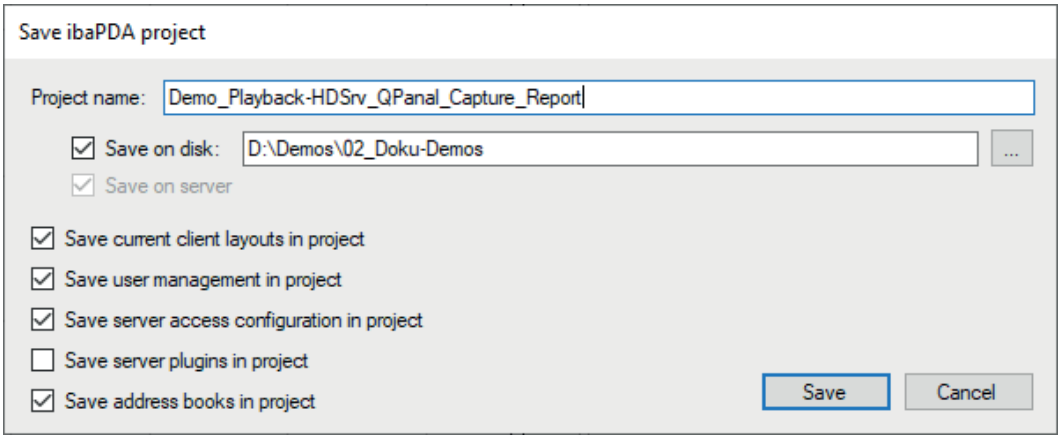
The project management in *ibaPDA* was invented for a more convenient and saver workflow with different or changing configurations. This may apply to different users of the same system with changing configurations, for instance in testing facilities, or to service engineers traveling from plant to plant always facing different systems.

Instead of saving and reloading the files for the configuration of the I/O Manager as well as Data storage and views, only one project needs to be loaded. An *ibaPDA* project consists of an I/O configuration and a data storage configuration. It can also contain one or more views (layouts), user management and server access data, as well as server plugins and address books. The project is saved with its respective individual name on the server. It can also be saved as ZIP file. The name of the currently loaded project is displayed in the caption bar of the client.



By default the project is called "Untitled" as you can see in the picture above. You can open or save the project via the file menu. As soon as you have finished an I/O configuration or data storage configuration (as described later in this manual) you can save it in an *ibaPDA* project.

When you click *Save project...*, the following dialog appears:



In this dialog, you can enter the project name. The project includes at least the current I/O configuration and the configuration of the data storage By default, it contains also the current layout. If you do not want to save the layout, uncheck the checkbox *Save current layout in project*. The project is always saved in the “Projects” subdirectory of the server's program data path (default: `C:\ProgramData\iba\ibaPDA\Projects`). If you also want to save the project to a file on the hard disk or another storage medium, then check the checkbox *Save on disk* and enter a full path. Saving a project on a hard disk or a network share can be reasonable for security reasons or for exchange with other users.

In addition, there are the following options:

Save user management in project
If you enable this option, then the configured user management, i.e. all defined users together with their granted client and server credentials as well as all passwords is stored in the project. Please note, that at least one administrator password (for user **admin**) must be known in order to open such a project.

For more information, see ↗ *Local user management in ibaPDA, page 87*.

Save server access configuration in project
If you enable this option, then the server access configuration (permitted access to the server by *ibaPDA* clients) – if available – is stored in the project on *ibaPDA* server.

For more information, see ↗ *Server access control, page 81*.

Save server plugins in project
If you enable this option, then any existing function files (.dll) that you have created with the *ibaPDA*-Plugin as well as other files which are required by the plugin (.ini, .xml etc.) are saved in the project.

If you want to open and load a project in which plugins are stored, then first a dialog opens for selecting the plugins. The system shows you a comparison of the status of the Plugins in the project file and on the *ibaPDA* server. Carefully check which plugins you want to load and select them specifically.

For more information, see Part 4, *Plugins*.

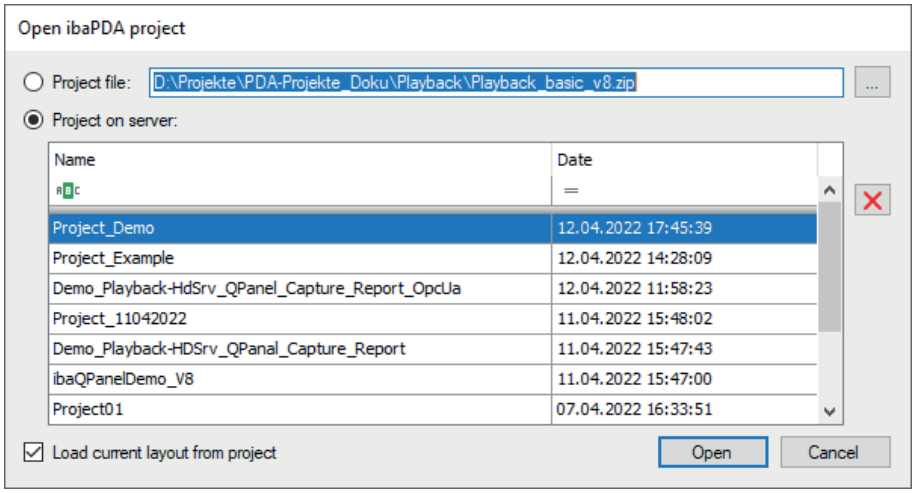
Save address books in project
If you enable this option, then all address books configured in the I/O Manager under *General – Address books* are saved in the project.

Depending on the amount and size of the address books, this can make the project file very large!

If you want to open and load a project in which address books are stored, then first a dialog opens for selecting the address books. The system shows you a comparison of the status of the address books in the project file and on the *ibaPDA* server. Carefully check which address books you want to load and select them specifically.

As soon as you have saved the project, the *ibaPDA* server will inform all the connected clients about the project name change and the clients will refresh their caption bar. If you change the I/O configuration or the data storage configuration, then an asterix (*) is added to the end of the project name, indicating that the project has been modified.

To open a project, click *Open project…* in the file menu. The following dialog appears:



You can select a project either from a file or from the server. Check the radio button *Project file* and browse the file system for a ZIP file if you want to load the project from a file.

The table shows the projects available on the server. If you want to open a project from the server, select the project in the table and click on “Open”.

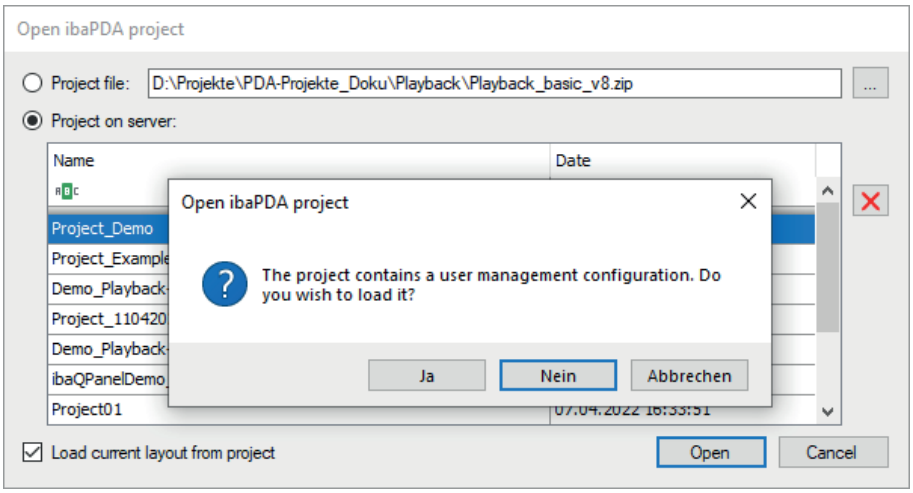
If the list of projects is very long, you can use the filter row in the table header to find the desired project more easily.

For detailed information about the filter functions, please refer to part 7 of the manual, chapter *Advanced filters in tables*.

If you wish to remove a project from the server, select it and click on ✕.

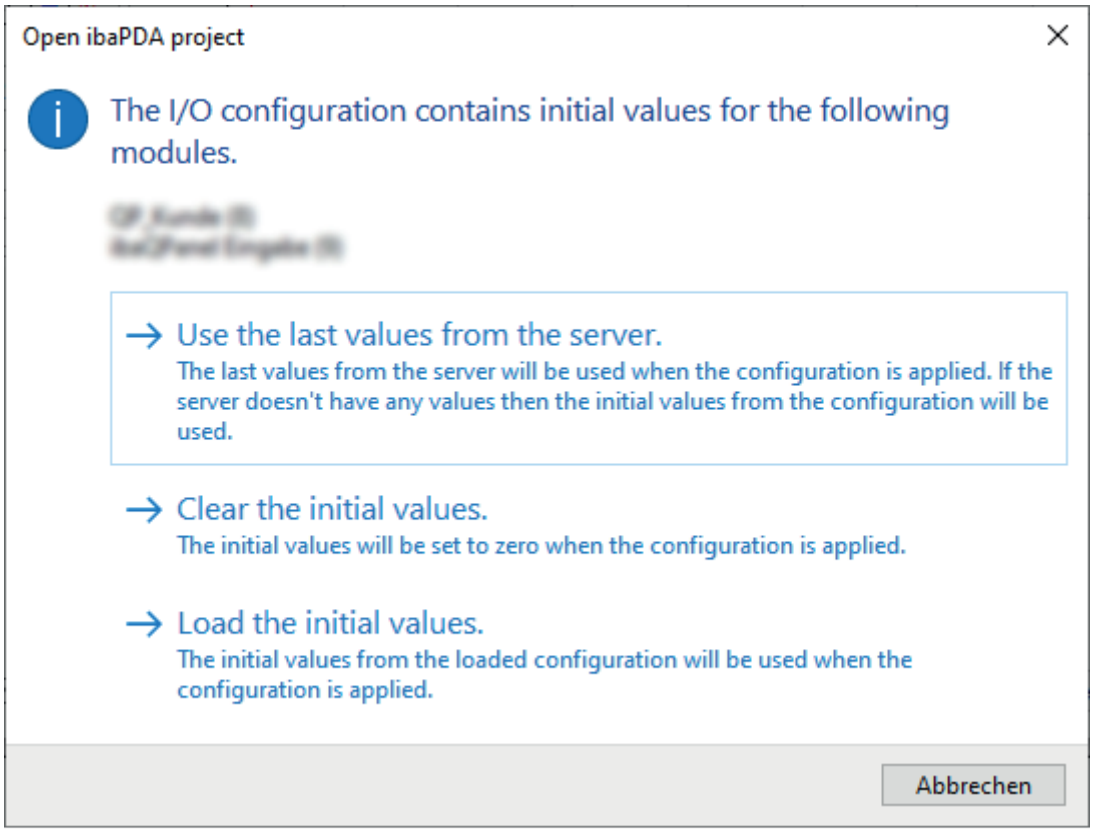
When you open a project, then *ibaPDA* stops the acquisition if it was running. *ibaPDA* then loads the I/O configuration and data storage configuration from the project.

If the *Load current layout from project* checkbox is activated and if the project contains a layout, then the layout is transferred from the server to the client and the client loads the layout. If a user management configuration had been stored in the project, you will be asked when opening whether to load the user management configuration or not.



Only confirm the query with <Yes> if you know the administrator password!

If the I/O configuration of the project to be opened contains retentive modules or modules with initial values (e.g. ibaQPanel input, ibaQPanel text input, shift register, Virtual rententive etc.), you are asked whether to clear or to load the initial values or to use the last values from the server.



If you choose "Clear the initial values", the initial values are replaced by zero when applying the configuration.

If you choose "Load the initial values", the initial values as saved in the configuration are applied.

If you choose "Use last values from server", the last valid values of the currently running configuration will be used in the new configuration. If there is no corresponding module available in the loaded configuration, the last values from the loaded project will be used.

After the loading process, the data acquisition must be restarted manually.

Note

If you open an I/O configuration in the I/O Manager you will get the same dialog.

5.2 Remote configuration

The remote configuration feature enables the user to control the configuration of one or more *ibaPDA* systems from a remote location via a configuration file.

5.2.1 Principle of operation

The remote configuration is an automatic function with immediate effect on the running data acquisition, resulting in a stop and restart of the acquisition.

When activated, the remote configuration feature periodically checks if the configured file is available. If the file is available, it is executed and afterwards deleted.

The processing depends on the file extension.

- ZIP file (.zip):
  ibaPDA tries to open this file as a project file. At success, the I/O configuration and the data store configuration will be loaded and the data acquisition starts. Layout configurations will not be loaded even if the ZIP file contains those files.
- IO file (.io):
  ibaPDA tries to open this file as a new I/O configuration. At success, the new I/O configuration is applied and the data acquisition starts.
- DS file(.ds).:
  ibaPDA tries to load this file as a new configuration of the Data storage. At success, the new data storage configuration is applied.
- other files:
  ibaPDA interprets this file as a text file. It reads out the first line. This line can contain one of the following commands.
  - STOP
    ibaPDA stops the data acquisition.
  - RESUME
    ibaPDA continues the data acquisition.
  - EXPORT
    ibaPDA exports the current I/O configuration to the file specified in the rest of the first line.

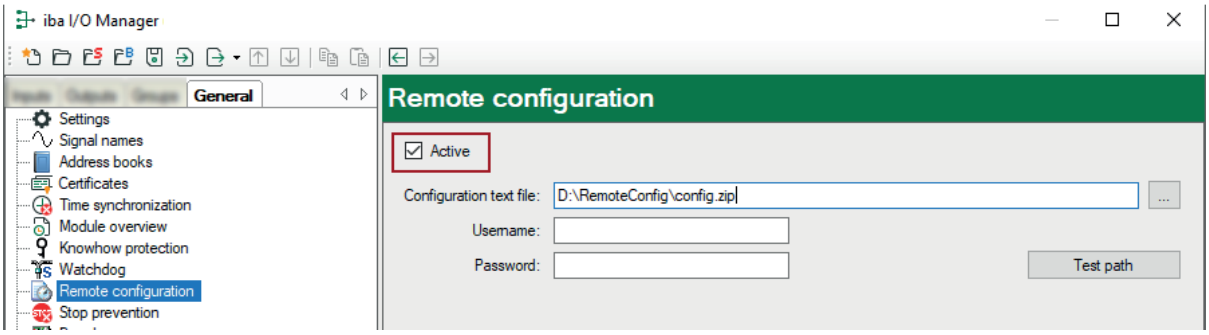
Example: `EXPORT c:\export\MyPdaIoComfog.io`

If none of these commands are in the first line, then *ibaPDA* tries to import this file as a new I/O configuration. At success, the new I/O configuration is applied and the data acquisition starts.

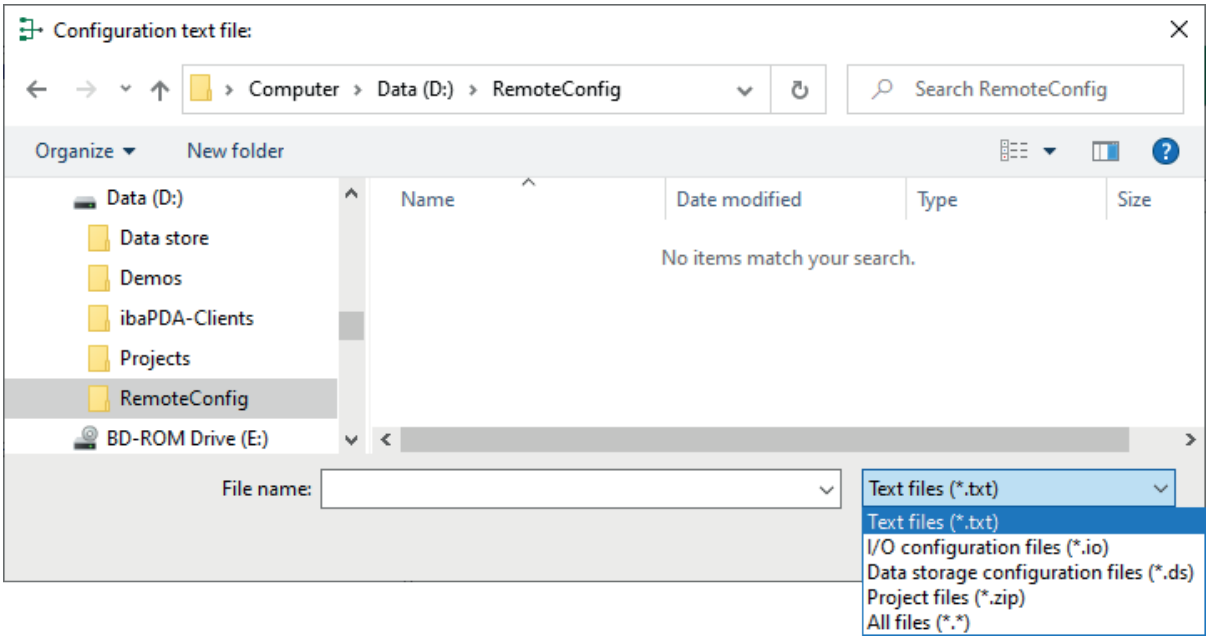
5.2.2 Setting up the remote configuration

The remote configuration needs to be configured and activated.

1. Open the I/O Manager and select the *Remote configuration* node under *General* register.



2. Enable the function by checking the *Active* selection box.
3. Enter the full path and name of the configuration file you wish to use. The file name specified here including the file extension must exactly match the file name you will copy later into this directory.
   It may be useful to choose a file name which will be reused every time you want to change the configuration, such as "config".
   In that case you are not urged to change the filename in the I/O manager every time. On the other hand, the configuration name in the caption of the *ibaPDA* client window would always be the same.
4. You can type in the information manually or you can use a browser which makes it easier if the path is very long or on a network share.



5. Via the button <Test path> you can check the accessibility of the directory, particularly if it is a network share.
6. Click on <OK> to close the I/O Manager. If the real configuration file is already stored in the specified directory, *ibaPDA* will immediately take the file, process and delete it.

Tip

Setting up the remote configuration is part of the I/O configuration. To make sure that the remote configuration works again after it was executed, the remote configuration feature must be enabled and properly setup in every configuration file which is loaded via remote configuration.

Attention!

As the system reacts immediately when a proper configuration file was copied to the specified directory, you should take care that all settings or changes made are correct.

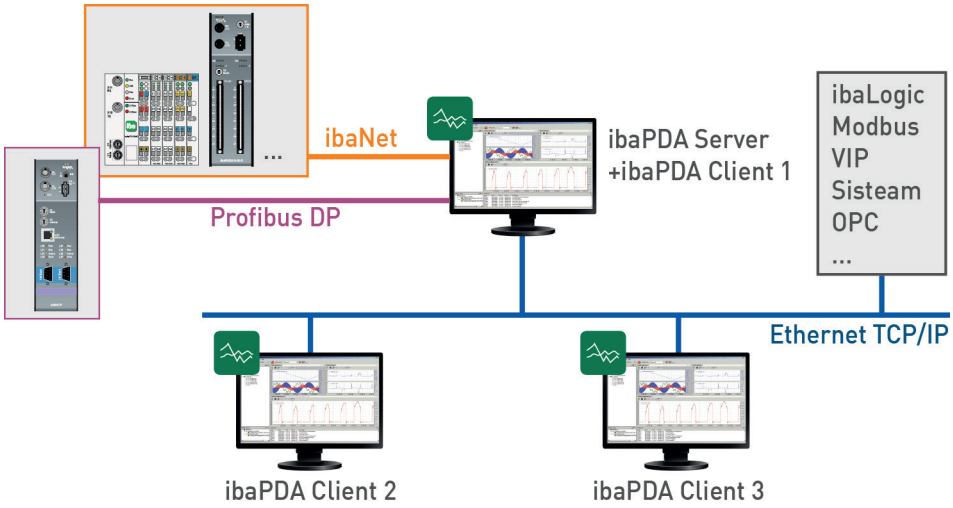
Otherwise, the system might have a wrong configuration which leads to incorrect data files, displays or even prevent the system from restarting.

6 ibaPDA-server

"Server" describes a basic process in *ibaPDA* which is relevant for the data acquisition and storage in *ibaPDA*. Data is displayed online not via the server but the client. In addition, the client is the user interface of *ibaPDA* and serves not only for display, but also for server configuration.

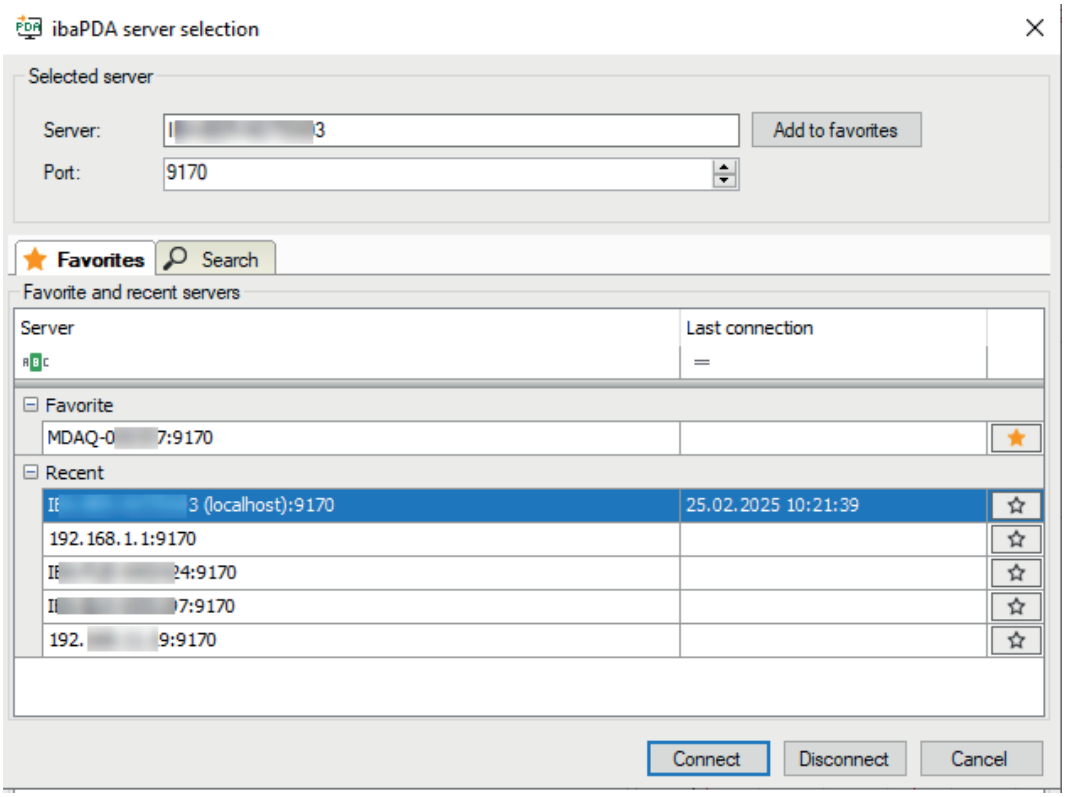
The server can run independently and without a client.

You can see an example for a topology of a Single Server Multiple Client system in the following figure.



6.1 Select an ibaPDA server

In order to open the dialog for server configuration, click on the following toolbar button .



In this dialog, you can select the server to be connected with the client. All changes made in the I/O configuration and data storage configuration will also concern the selected server. You can always connect and disconnect a server in this dialog.

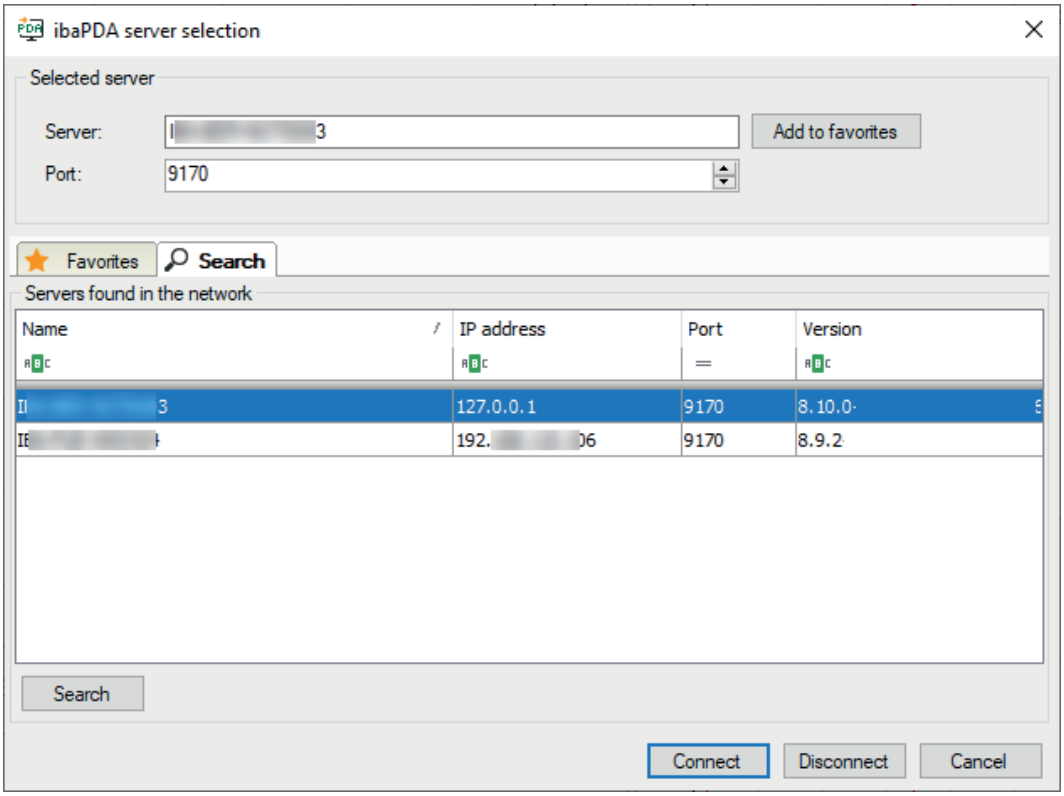
Use the client preferences to determine the behaviour of the client at startup in terms of server connection. For more information see ↗ *Configure server connections for client startup, page 70*.

If the client is already connected to a server, this server is indicated on top of the dialog in the area *Selected server*.

As soon as the dialog opens, *ibaPDA* automatically searches the local computer and the network for active *ibaPDA* servers for approximately 3 s. The found servers will be listed in the *Search* tab of the dialog, in the area *Servers found in the network*..

Search tab

Select this tab if you want to search for active servers locally or in the network.



Click on the <Search> button to start a search for available servers.

The found servers will be listed in a table, which offers filter and sort functions in its header. If the right server is listed you can connect the client with this server by doubleclick on the corresponding row in the table or by selecting the server, followed by click on <Connect>.

Enter the server manually

You can also specify the server manually in the *Server* field in the upper part of the dialog. Enter the host name or the IP address of the server computer. If client and server run on the same computer, you may enter "localhost". If the server runs on a remote device, you may enter the host name or the IP address.

Tip

Some cases may occur where the name resolution works not properly in a network and hence the server cannot be found under its host name. Enter then the IP address of the server computer instead.

Port

The port number will be given automatically. (Default=9170)

Click on <Connect> to establish the connection. If the corresponding server is not accessible you will get an error message.

Favorites tab

In the *Favorites* tab you find an overview of preferred and also previous server connections. You may use this list to determine preferred servers (favorites). When you double-click on a server the client starts a connection attempt to this server.

This dialog can be helpful, especially in case of frequently changing sites (service), in testing environments or when configuring multiple distributed servers at a central desk.

The *ibaPDA* server can be selected either by double-click on the server in the list or by clicking on the server and pressing <OK>.

Favorite and recent servers

In the *Favorites* tab you find an overview of preferred and also previous server connections. You may use this list to determine preferred servers (favorites). When you double-click on a server the client starts a connection attempt to this server.

This dialog can be helpful, especially in case of frequently changing sites (service), in testing environments or when configuring multiple distributed servers at a central desk.

The table as well, offers filter and sort functions in its header.

Besides the column with host name or IP address there is a column *Last connection* showing date and time when the connection to this server was established for the last time.

Recent

The group *Recent* contains servers, the client was connected to in the past.

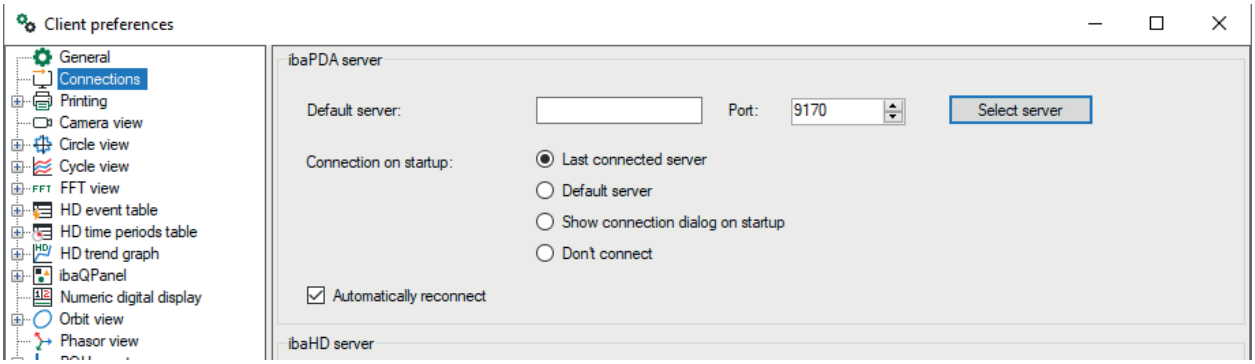
Favorite

The *Favorite* group is only available if at least one server out of the *Recent* group has been declared as favorite. Click on the star button in order to enable or disable the favorite status of a server.

If you want to add the currently connected server to the favorites, click on the <Add to favorites> button in the upper part of the dialog.

Configure server connections for client startup

In the client preferences you can define the server the client should connect to on startup. Therefore, select the node *Connections* in the client preferences.



Default server
If there is a server the client should always connect to by default, then enter it here. Enter the computer name or the IP address of the server computer. Alternatively, you can click on the <Select server> button in order to open the *Server selection* dialog and select an available server of your choice.

Connection on startup
Choose one of the options for startup of the client.

- Last connected server: The client connects to the same server it was connected to when it was last closed. This is not necessarily the default server.
- Default server: The client connects to the server which is set as default server above.
- Show connection dialog on startup: The dialog *ibaPDA or ibaHD server selection* opens. Here you can select the server manually.
- Don’t connect: The client starts without connecting to any server.

Automatically reconnect:
If you enable this option, the client tries to reconnect to the server automatically, in case the connection is interrupted.

6.2 Server status display

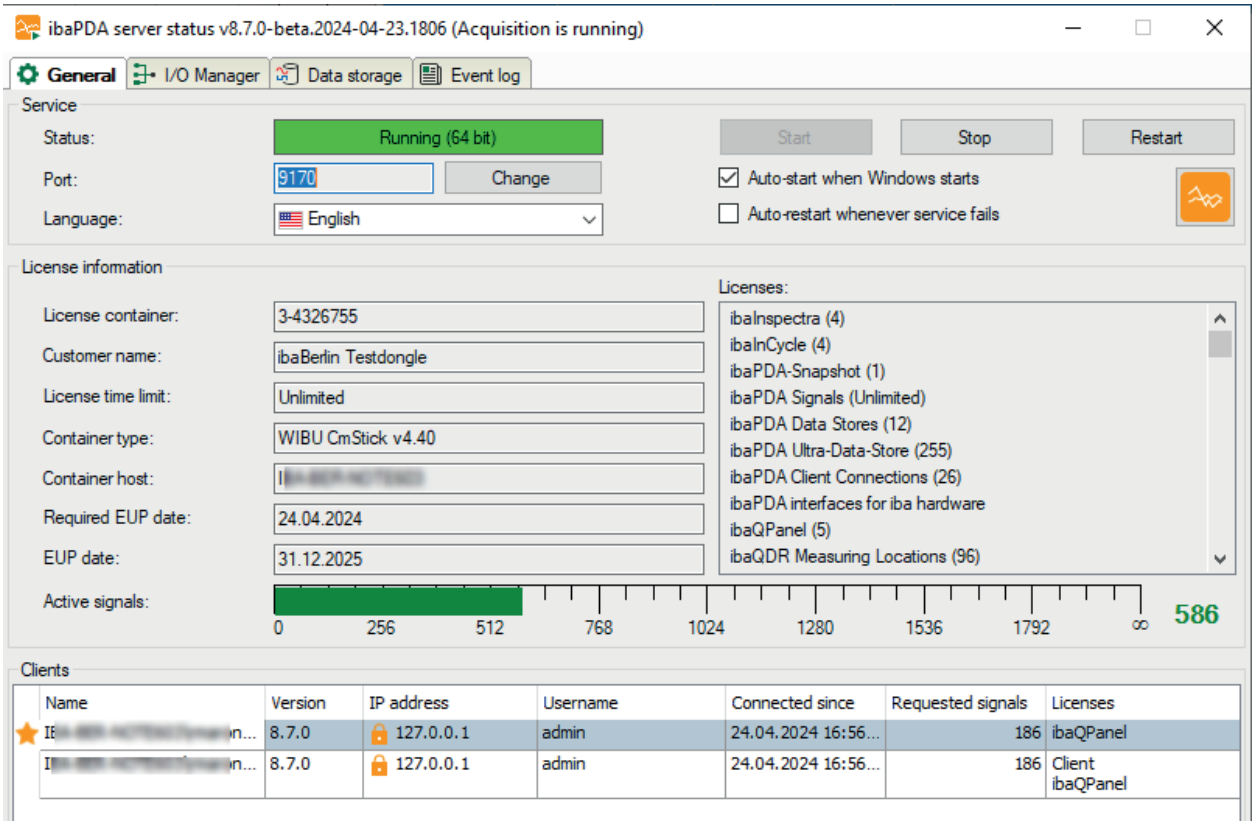
The *ibaPDA* software package includes the separate tool *Server Status* for control and diagnostic analysis of the *ibaPDA* server.

This software is auto-installed with the server.

You can execute it as follows:

- the Windows Start menu *Start – Programs – iba – ibaPDA – Server Status*
- by double-click on the server icon in the Windows system tray.

6.2.1 Server Status, General



Service

Status
Indication whether service is running or not.

Port
The port number is assigned automatically, but can be changed here manually if necessary (<Change> button).

<Start>, <Stop> and <Restart> buttons
Use these buttons to start, stop or restart the *ibaPDA* service

Auto-Start when Windows starts
This option must be enabled if an automatic restart of the system is planned after a computer restart.

Auto-restart whenever service fails
Enable this option if you want the system to restart after the *ibaPDA* service failed.

Language
Here you can select one of the languages supported by *ibaPDA*, which will be used for both the dialogs of this program *ibaPDA Server Status* and all messages which are generated by the *ibaPDA*-Server. By the setting “System” the program will adopt the Windows language setting. A change of language will come into effect after a restart of the service.

Open ibaPDA client button
Click on the button with the *ibaPDA* icon in order to open *ibaPDA* client. If *ibaPDA* client is already open, then another instance of the program will be opened.

License information

In the *License information* area you'll find important information about your current software license.

License container
The license container number is important for all service queries and upgrades. Please provide the license container number to our support team. It is linked to your license container, i.e. either MARX dongle, WIBU dongle or WIBU soft license and stored in our database.

Customer name
In this field, you will find the name of the customer for which this license has been approved. For projects, which were initially completed via system integrators, registration of the end user should have been requested from iba either from the outset or, at the latest, after the warranty period has expired. This makes it easier to assign the license in the case of later extensions or support cases.

License time limit
This field shows the validity time of the licenses. Depending on the container type there is either a remaining validity time given in days or hours or there is an expiration date.

Container type
This field indicates whether a MARX dongle, a WIBU dongle (WIBU CmStick) or a WIBU soft license (WIBU CmActLicense) is used.

Container host
This field shows the name of the computer where the license container is attached, which supplies the licenses obtained by the application.

Required EUP date and EUP date
The EUP date specifies the date until which the period of free software updates is valid. After the update period has expired you still can continue running the program as usual but only with the features which were already available until then.

For more information, see Part 1, *Update policy*.

Licenses
This window displays all the available basic and additional licenses, such as clients, data storages, interfaces, plug-ins etc.

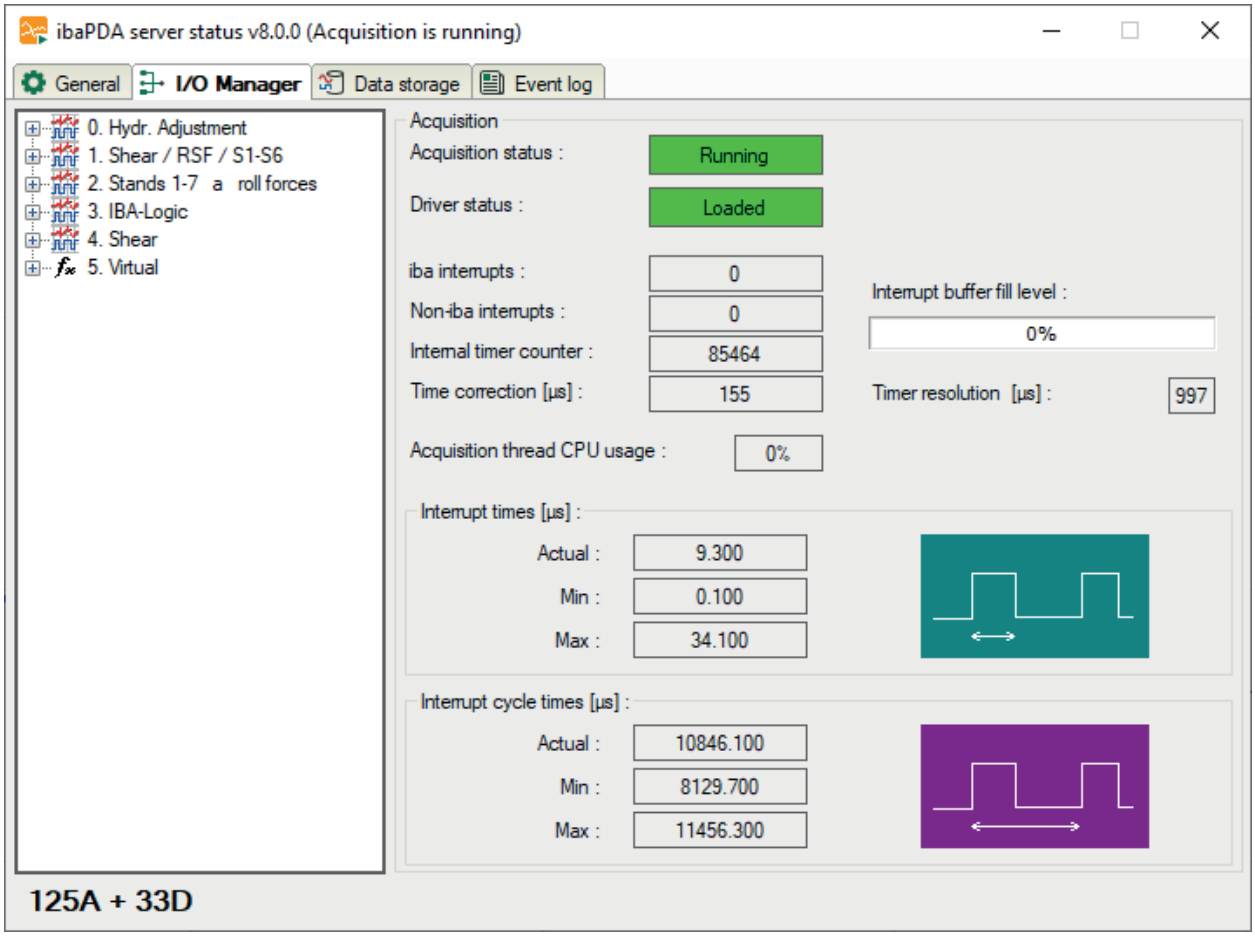
If multiple licenses are applicable, e.g. clients, Xplorer interfaces, data storages etc., the number of respective licenses is given in brackets.

Active signals
A bar on the scale and the value behind the scale show the number of currently active signals. The end value of the scale indicates the maximum permissible number of signals according to the license.

Clients
This table shows all the clients currently connected to the server. It provides information about

- the host name of the client and the logged on user,
- the *ibaPDA* software version of the client,
- the IP address of the client computer,
- date and time of the establishment of the current connection to the server,
- number of currently requested signals (only with running acquisition), i.e. of the signals which are currently used in all layouts of the client and
- special licenses, which are claimed by the client

6.2.2 Server status I/O Manager



Acquisition

This dialog offers relevant information about data acquisition.

Beside the signal tree, which contains all defined modules and signals, you can see whether or not the acquisition (=measurement) is running. Below the signal tree, you will find the number of currently used signals (analog and binary).

The display for the driver status (*ibaPDA* driver) must show the "Loaded" status on green background for proper operation. In case of a failure, the field turns red and an error message appears below the start button. If, for example, another process is already running with *ibaPDA*-drivers, the message would indicate that an instance of the program is already running.

In addition, you will find a number of displays that reflect the interrupt behavior. The interrupts are good indicators of the proper function of the system.

iba interrupts

This is an interrupt counter for interrupts, which are generated by iba PC cards. In proper operation, it counts up with a rate of 1000 / sec corresponding to an interrupt cycle of 1 ms.

Non-iba interrupts

This counter will only change if other cards in the computer generate interrupts. This could occur in case of “shared interrupts”. If the iba card generating the interrupt is assigned an exclusive interrupt, the counter must not change.

Internal timer counter
The internal timer counter is active when no interrupt source (PC card) is available. In this case the Windows timer is used.

The counter usually also count upwards with 1000 / sec.

Time correction
The value for time correction will only be displayed during acquisition and the internal timer counter serves as interrupt source.
Generally, the internal timer counter has not exactly the same resolution like the basic sample time. Hence, each counter step has a deviation of a few microseconds. These microseconds are added in the time correction counter. As soon as the sum matches the timer resolution, a double counter pulse will be triggered. This method enables the internal timer to match the basic sample time in average.

Acquisition thread CPU usage
This is the CPU time the server needs to read data from the driver and process them, e.g. evaluation of virtual signals, writing to the data file, buffering for transmission to clients etc.

Interrupt buffer (fill level)
This display shows how much data is in the interrupt buffer of the driver.
The *ibaPDA* server periodically reads from this buffer. If the interrupt service routine (ISR) of the driver takes too long, the *ibaPDA* server has not enough time to read all data out of the buffer and the buffer overflows. This would cause a stop of the acquisition. In that case, either a reduction of the number of signals or an increase of the basic sample time is required in order to ensure proper operation. This would reduce the read-out time for the ISR.

In the *I/O-Manager – General* tab – *Interrupt Info* node you may increase the size of the interrupt buffer up to 1024 MB, in order to prevent fast buffer overflow. Default size of the buffer is 50 MB.

Timer resolution
This time value given in ms should approximately correspond to the measurement time base. The timer is used for the internal generation of interrupts.

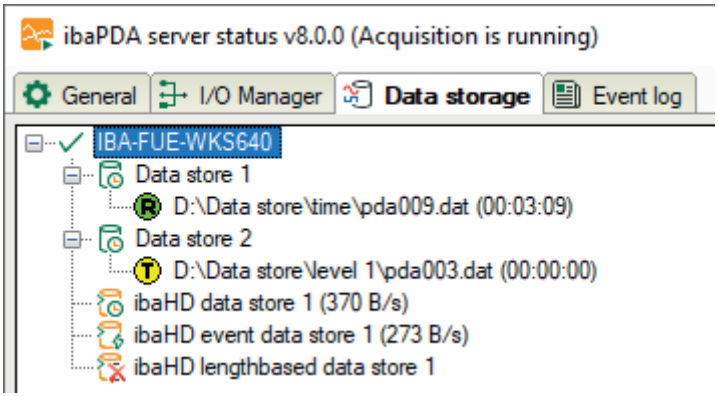
Interrupt times
These values are only displayed when the acquisition is running.

The system measures the actual processing time of the ISR (=interrupt time) and stores the shortest (Min) and longest (Max) time. The ISR reads the requested data from the different PC cards. The ratio of interrupt time to interrupt cycle time shows the percentage of CPU time required to read the data from the cards.

The actual interrupt time should never exceed double the interrupt cycle time. If so, interrupts will get lost and the measurement is not accurate.

Interrupt cycle times
These values are only displayed when the acquisition is running.
The system measures the actual cycle time of the interrupt and stores the shortest (Min) and longest (Max) time.

6.2.3 Server Status, Data storage



You will find the same information in the *ibaPDA* client program in the menu *View – Data storage status*.

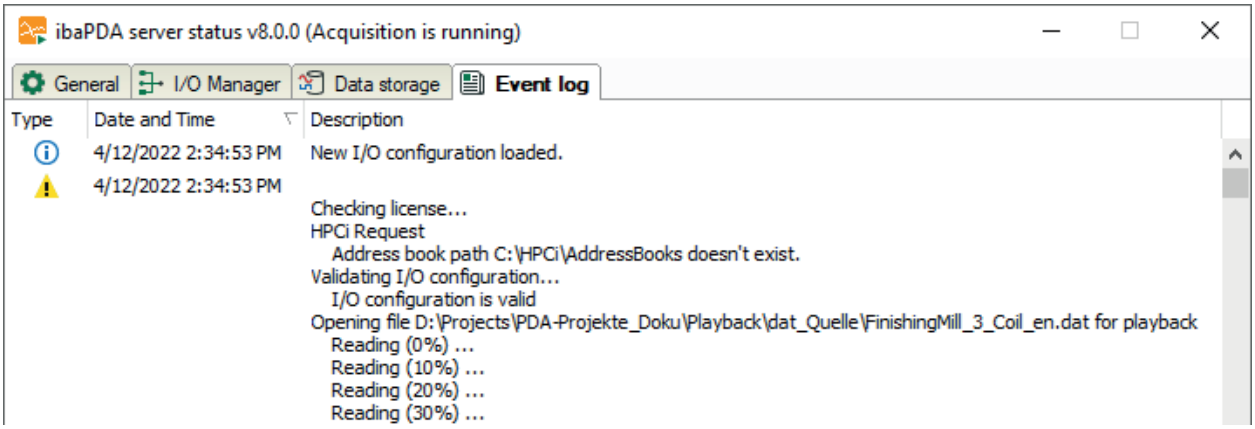
Here, all configured data stores are displayed with their status, sorted by server name, data storage name and data file.

The colored symbols in the tree indicate the current status of the recording or the data file respectively.

Icon	followed by ...	Display in ...	Description
	Path and file name	Data storage status	Data recording is running
	Path and file name	Data storage status	Waiting for (start) trigger; appears only if trigger is configured for data storage. Measured values are buffered in the RAM memory for the defined pre-trigger time and saved in the data file when the start trigger occurs.
	Path and file name	Data storage status	Post-trigger phase; stop trigger occurred but acquisition continues until the post-trigger time is over. The symbol only appears if a stop trigger is configured for the data storage.
	Server name	Data storage status signal tree	Acquisition stopped, no recording
	Server name	Data storage status signal tree	Server not available (no connection), thus no signals available and no recording
	behind a signal name	Signal monitor / legend	Signal not available, e.g. when acquisition stopped or server not available.

Table 6: Status icons

6.2.4 Server status, Event log



You will find the same information in the *Event log* window in the *ibaPDA* client program (*View – Event log*).

All relevant system events, such as configuration changes, are logged and listed in a table.

6.3 Server – more information

The *ibaPDA* server is a service under Windows. A Windows service always runs with reference to a user, i.e. it is assigned to a user account.
The *ibaPDA* server runs only if the user has administrator rights. The default setting is the local system account. This system-defined account has administrator rights on the local machine.

Note

If *ibaPDA* server only needs to access the local machine, i.e. also the data storage uses only the local hard disk, then the local system account is sufficient.

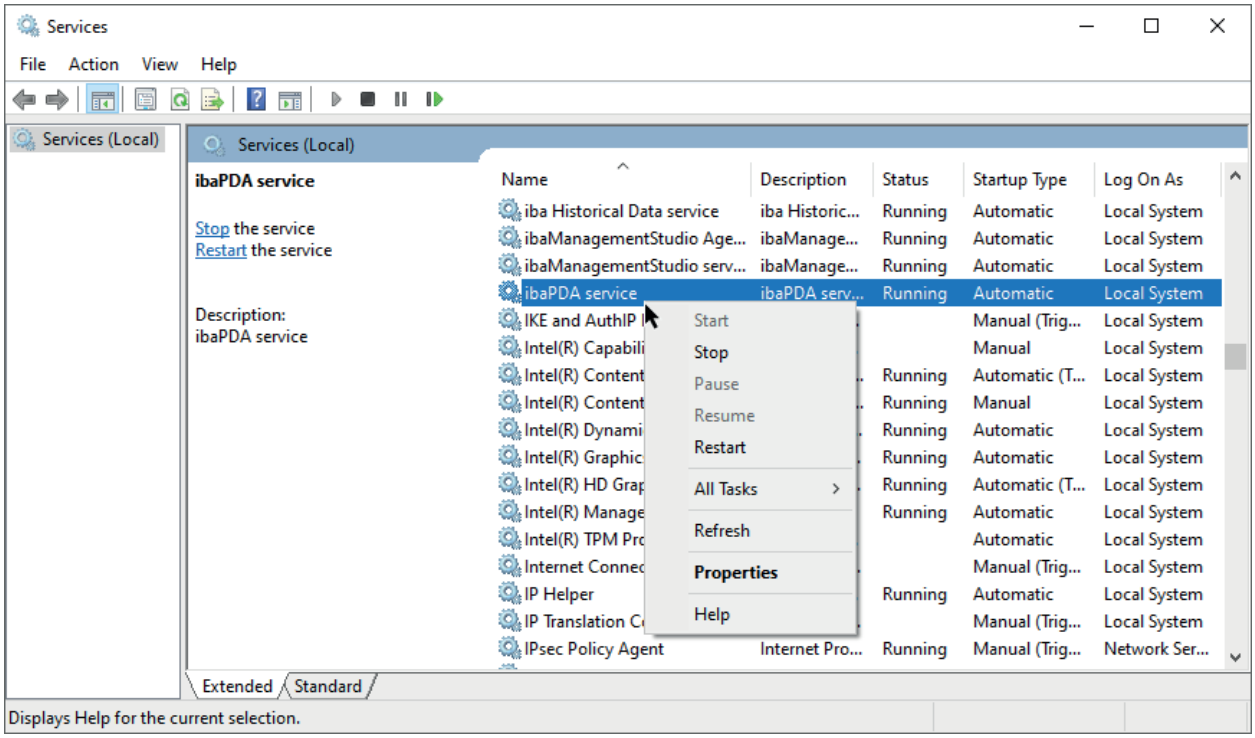
If *ibaPDA* server requires access to remote computers in the network which requires special rights on the remote machine, e.g. writing data files on a remote file server, you should create an appropriate user account.

This user account must be available on all computers involved.

Furthermore, this user account should be either set during installation of ibaPDA server (see ↗ *Installation*, page 25) or configured later in the Windows service administration (*Control Panel – Administrative Tools – Services – ibaPDA service / Properties – Log On*).

In order to view or change the current user account set in the *ibaPDA* server, please go to the Windows service management. Enter "Services" in the Windows search bar. Open the *Services* app and go to the service *ibaPDA service*.

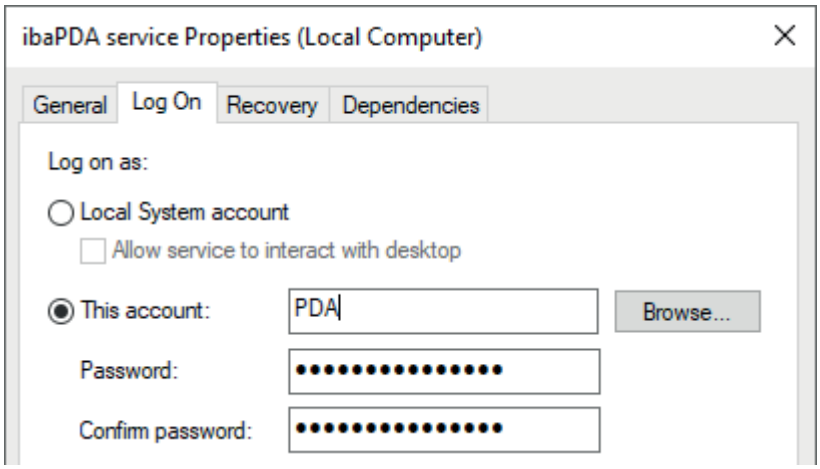
In order to see the settings of the service, right-click on *ibaPDA service* and select *Properties* in the context menu.



In the properties, select the *Log On* tab.

Here, you can verify if the service is currently running under the local system account or if another user account is logged on.

If you want to change the user account, please click on *This Account*, fill in the registration information and exit the dialog by clicking <OK>.

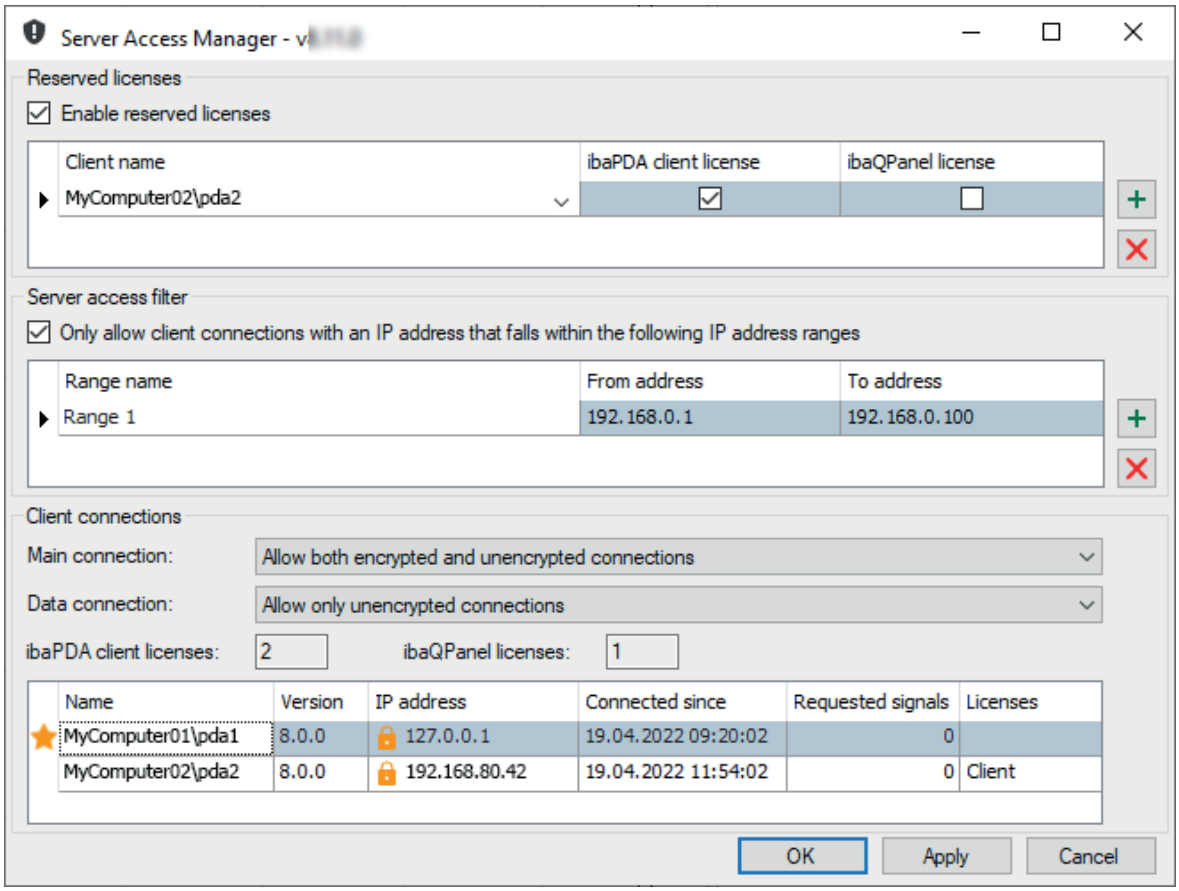


6.4 Server access control

By setting up the server access control, you have the option to reserve licenses for selected users and to limit the access to the *ibaPDA* server.

For the configuration of the server access control, in the *ibaPDA* client, select the *Configuration – Server Access Manager* menu…

The following dialog is shown.



Reserved licenses

In the *Reserved licenses* section, you can reserve *ibaPDA* client and *ibaQPanel* licenses for certain computers on the network.

Because the *ibaPDA* license model works with free (floating) licenses, any client that wants to access the server can take advantage of a free license. However, this entails the risks that clients that need to access the server no longer find a free license if other clients were "faster" and have taken all the available licenses.

By reserving, you can now guarantee that the clients that absolutely need to gain server access, e.g., a computer on a control desk with *ibaPDA* client and *ibaQPanel*, always receive a license.

You must identify the computer with its name. You can reserve multiple licenses for a computer if multiple instances of the *ibaPDA* client are to run on the same computer. Reserved licenses can only be used by computers on the list.

Clients that are not on the list can only use generally available licenses.

In the *Current client connections* section at the bottom of the dialog window, the available *ibaPDA* client and *ibaQPanel* licenses are displayed.

For reserved licenses for clients that are not currently connected to the server, the row has a gray background.

Enable reserved licenses
If you select this option, the allocation of licenses is regulated according to the reservations. If you disable this option, all licenses of the server become available again.

Reserve license

1. Click the button + to add a new line.
   A computer name is then immediately displayed in the *Client name* column if there are active clients in the network. The name of the server computer is usually in the first row.
2. Select the desired client in the drop-down list with available clients.
3. Then put a check mark in the selection fields of the desired license(s).

Cancel reservation

1. Click in the table row of the client for which you no longer need a reservation.
2. Click the button ×.

Server access filter

In the *Server access filter* area, you can determine to which clients a connection to the server is allowed. You do this by specifying the IP address ranges for the allowed clients.

Only allow client connections with an IP address...
If you select this option then connections are only allowed from clients in one of the IP address ranges in the table below. When you disable this option, access is not filtered.

Note

If this option is enabled without an IP address range in the list, the access to the server is only granted for local clients which are running on the server computer.

Add address range for filter

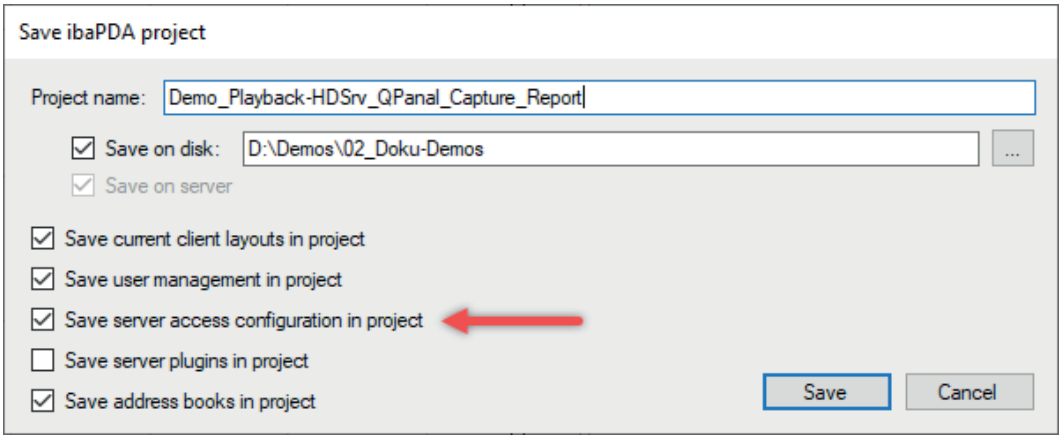
1. Click the button + to add a new line.
   A default name is provided in the *Range name* column, which you can change if you want.
2. Click in the field in the *From address* column and enter the start address of the range allowed.
3. Click in the field in the *To address* column and enter the start address of the range allowed.

Delete filter range

1. Click in the table row of the range you no longer want to filter.
2. Click the button ×.

Save configuration

The server access configuration can be stored in an *ibaPDA* project.



If a project containing a server access configuration is to be loaded, the user is asked whether to also load the server access configuration.

Client connections

The communication between client and server can run encrypted or unencrypted. The following connection types can be distinguished:

- Main connection for the transmission of configuration data and diagnostic data
- Data connection for the transmission of signal data when acquisition is running

In this area you can setup if the communication should run either encrypted or unencrypted or both.

Main connection

There are two settings for the main connection:

1. Allow both encrypted and unencrypted connections
2. Allow only encrypted connections

Purely encrypted connections are only possible with version v7.3.0 or higher of the *ibaPDA* client and the *ibaPDA* server. Clients running on version <v7.3.0 will always connect unencrypted. If you want to make sure that even older clients can connect to the current server, choose option 1. The other way round, clients running on a version ≥v7.3.0 connect unencrypted to a server with an older version.

Data connection

There are two settings for the data connection:

1. Allow both encrypted and unencrypted connections
2. Allow only encrypted connections
3. Allow only unencrypted connections

Since an encrypted data connection sometimes requires a lot of computing power depending on the number of signals to be transmitted, option 3 is the default setting here.

A data connection can only be encrypted if the main connection is also encrypted. Option 2 for the data connection therefore also requires option 2 for the main connection.

If you select option 1 for the data connection, the data connection is encrypted, provided that the main connection is also encrypted. If the main connection is unencrypted, then an unencrypted data connection is also used.

ibaPDA client licenses and ibaQPanel licenses
These fields show the number of the respective licenses.

Connection table
The table below shows one row for each currently connected client. In it you will find the

- name/user, client version and IP address of the connected computer
- date and time the connection was established
- the number of signals transmitted to the respective client
- licenses used by the client.

The first client running locally on the *ibaPDA* server does not need a license. It is marked with a star icon in the table.

There is a lock symbol in front of the IP address which indicates the encryption status of the respective client connection.

	Both the main connection and the data connection are encrypted
	The main connection is encrypted, the data connection is unencrypted
	Both the main connection and the data connection are unencrypted

Table 7: Encryption status of the client connection

The same symbols are shown in the status bar of the *ibaPDA* client.

7 User management

Common user management for ibaPDA and ibaHD-Server

The user management system of *ibaPDA* and *ibaHD-Server* allows different user accounts to be created for use of each program.

A distinction is made between local users and domain users.

Both user types can be managed via the *user management*.

As the program *ibaPDA* client is used for both *ibaPDA* and *ibaHD-Server*, the user management for both programs is possible in the *User management* dialog of *ibaPDA* client. Some differences exist with regard to the program-specific permissions.

In *ibaPDA* client, you can access the User management dialog either

- by choosing *Configure - User management…* menu or
- via the shortcut <Ctrl>+< U>.

For *ibaHD-Server*, there is the additional user management option in the *ibaHD Manager* program.

ibaPDA-specific user management

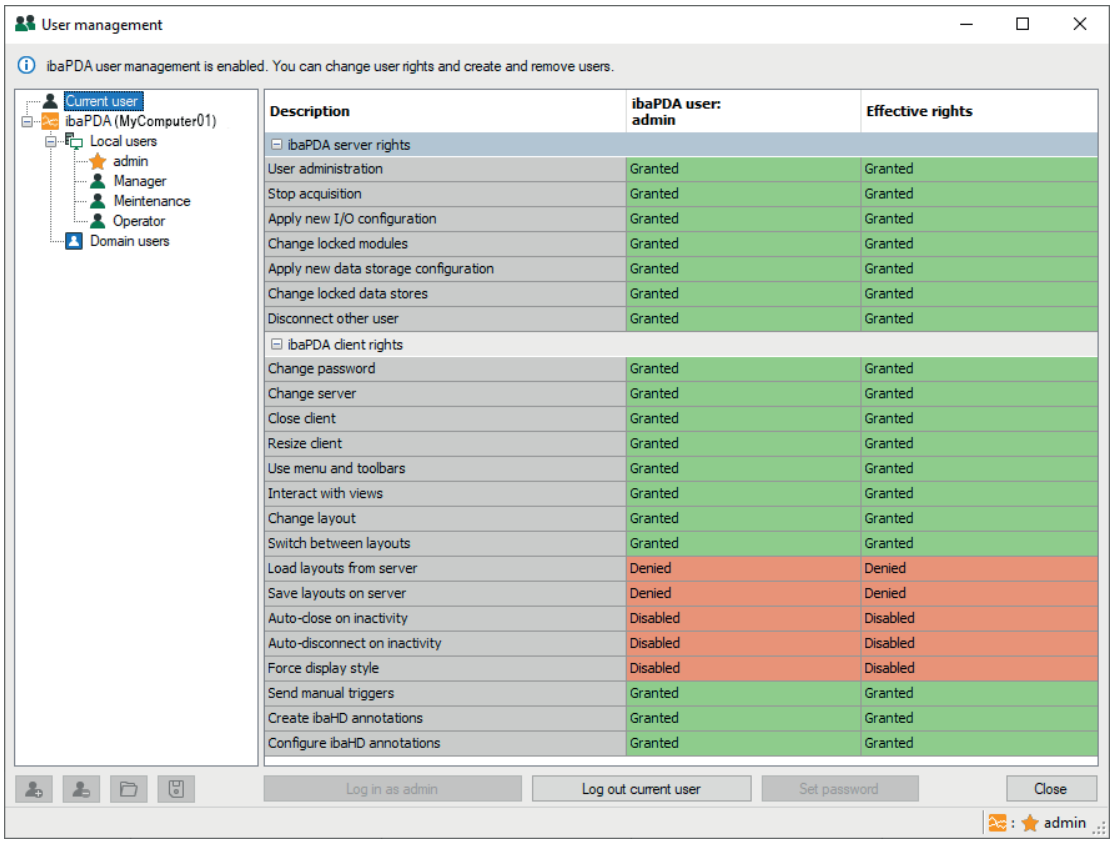
For *ibaPDA*, rights for the *ibaPDA* client and *ibaPDA* server can be assigned to users via the user management dialog.

Other documentation

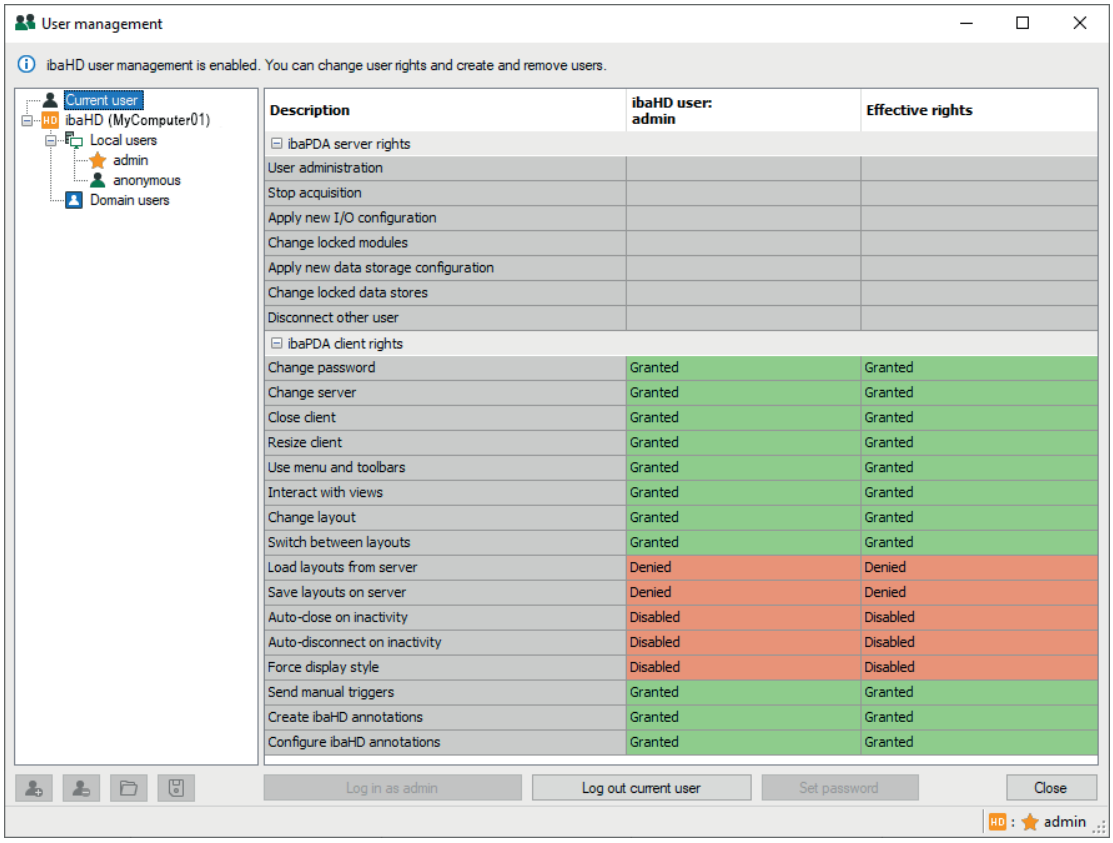
The program-specific rights for *ibaHD-Server* are described in the *ibaHD-Server* product manual.

After opening the dialog, the *Current user* is always initially selected in the user tree in the left-hand window. This is the user who is currently logged in to the server. The identity of this user is shown in the column header of the table to the right. Depending on whether the *ibaPDA* client is connected only to an *ibaPDA* server, only to an *ibaHD-Server* or to both, different user management screens will appear.

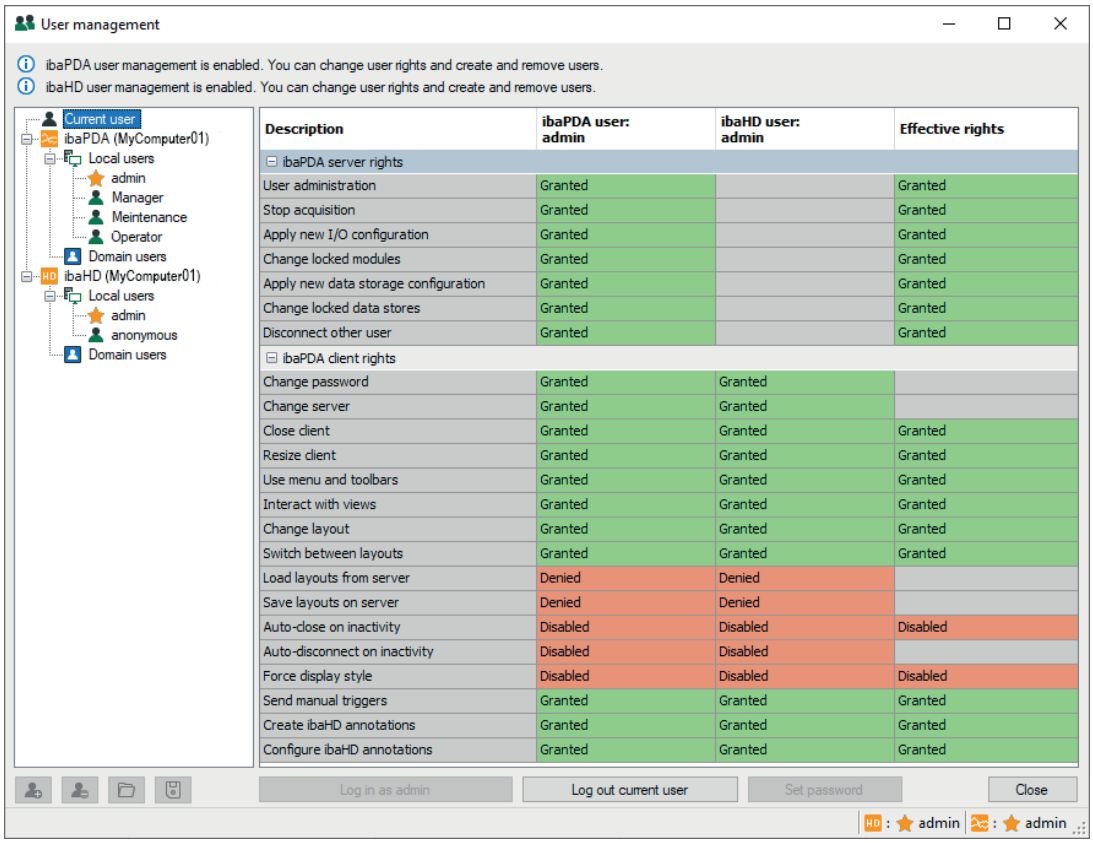
Currently logged in user, *ibaPDA* only:



Currently logged in user, *ibaHD-Server* only:



Currently logged in user for *ibaPDA* and *ibaHD-Server*:



If the *ibaPDA* client is connected to an *ibaPDA server* and an *ibaHD-Server*, the rights for both servers are displayed. Different users can be logged in via the *ibaPDA* client.

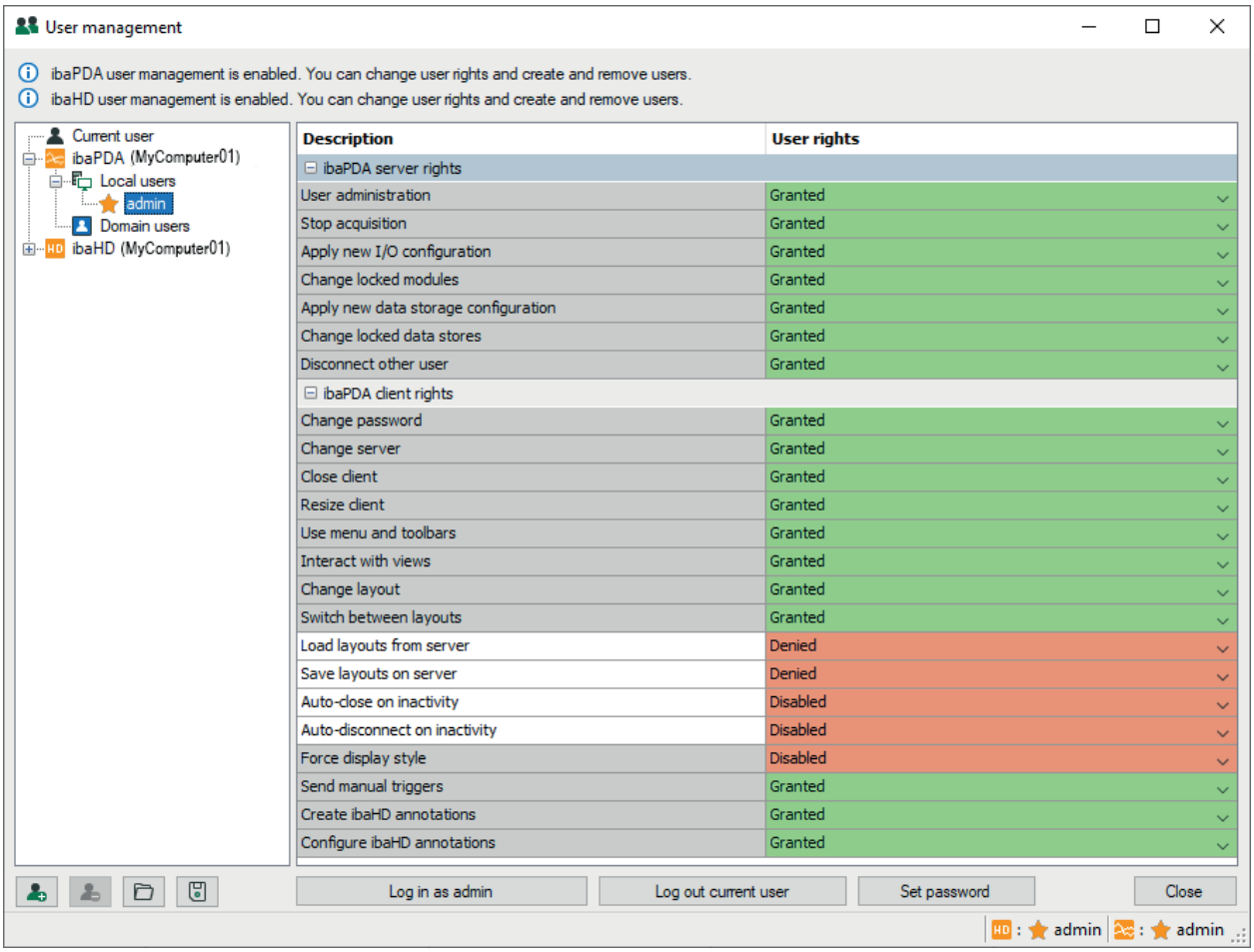
Since the *ibaPDA*-client permissions are relevant for both *ibaPDA* and *ibaHD-Server* users, the settings are linked and the result is displayed in the "Effective rights" column.

For more information about this, see ↗ *Rights when simultaneously logged in to ibaPDA and ibaHD-Server, page 103*.

7.1 Local user management in ibaPDA

Local users are created only on application level for *ibaPDA* (or *ibaHD-Server* respectively), provided with application-specific permissions. These users are independent from the Windows-user. They only exist on the *ibaPDA* server or *ibaHD-Server*.

Each user account has several server rights and client rights. There is one user, called **admin**, who generally has all rights and whose rights cannot be altered. At first, only the user **admin** is able to create other users. Other users can only do this if they have been granted the "user administration" right. The following picture shows the user administration from the **admin** user's point of view.



By default, there is only the **admin** user. The user management is disabled as long as the **admin** user does not have a password. If the user management is disabled, every *ibaPDA* client automatically connects to the *ibaPDA* server as **admin**. This ensures that every client has full rights.

To enable the user management system, you have to set a password for the **admin** user. For this purpose, click the <Set password> button.

Note

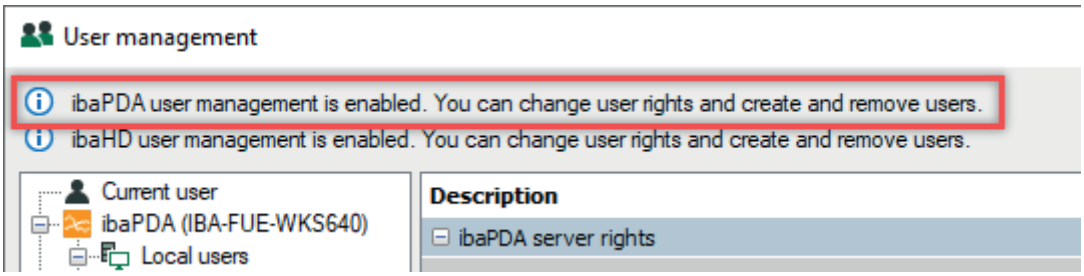
Please note, that there is a difference between Marx and WIBU license containers in terms of password length and characters.

Marx dongle: max. 15 ASCII characters

WIBU dongle or soft license: no limitation in terms of length and character set

Enter a password, repeat it and close the dialog via <OK>.

An active user management is indicated by a text line on top of the user management dialog.



If the user management is enabled and you are logged in as **admin**, you can add or delete users and change user permissions. You can also assign passwords for each user account.

To add and delete users, the following buttons are available at the bottom (green icons for local users):

	Add local user/domain user (in *ibaPDA* client, a distinction is made between *ibaPDA* users and *ibaHD-Server* users).
	Delete selected local user/domain user
	Load users from an existing user file (*.pda_users)
	Save the configured (*ibaPDA*) users in a user file (*.pda_users), incl. all permissions and passwords

Table 8: User management buttons

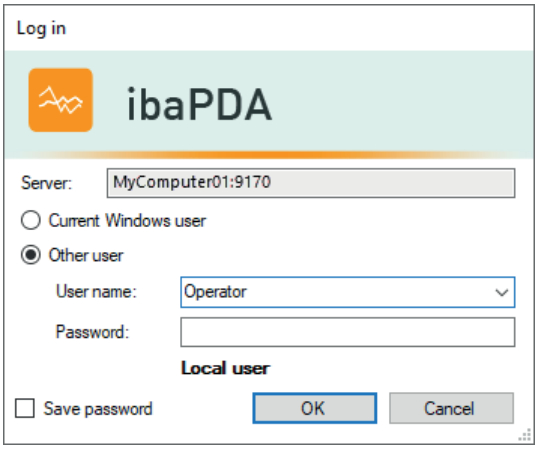
User groups can only be created for domain users. For additional information on adding domain users, see ↗ *Add domain user, page 98*.

You can change the permissions by selecting the relevant user in the tree and choosing the desired option in the table to the right using the fields' drop-down menu.

Tip

Using the context menu of the table (right mouse click on the table) you can copy the settings of one user and transfer them to another user.

If the user management is enabled, a login dialog will appear when you connect to the *ibaPDA* server.

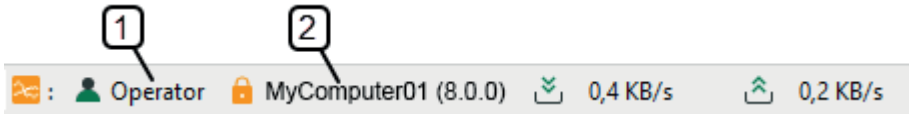


The *Current Windows user* selection works only for domain users.

Available users can be selected from a drop-down *User name* list.

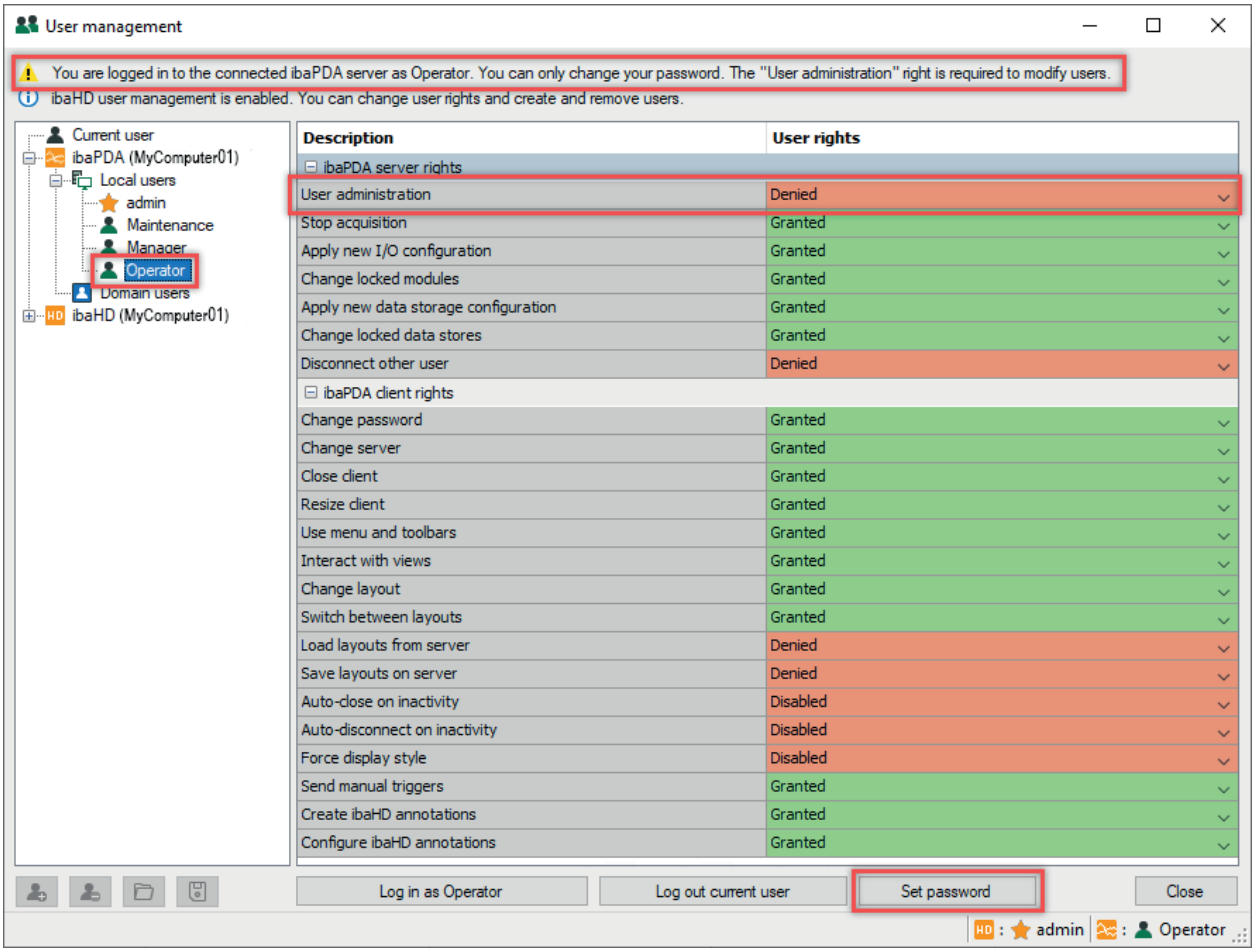
If you enable the "Save password" option, the user name and password will be saved and re-used during the next connection attempt to the *ibaPDA* server to enable automatic login. It is not possible to save the password for the **admin** user for security reasons.

In the status bar of the client, you can see your current user name and the server you are connected to.



1	Logged in user	2	Connected server

If you are not logged in as **admin**, or do not have the user administration permission, you can only change your own password in the user management dialog. You cannot add or delete users or change their permissions. To log in as **admin**, first highlight the **admin** user in the user tree and then click the <Log in as admin> button.



Note

If your *ibaPDA* client is connected to an *ibaHD-Server* (too), there is a separate user management available for the *ibaHD-Server* which can also be configured in this dialog. A simultaneous connection to both servers will partially result in effective permissions.

For more information, please see ↗ *Rights when simultaneously logged in to ibaPDA and ibaHD-Server, page 103*.

7.1.1 ibaPDA server rights

Option	Description
User administration	The right to add/delete users and to change the assigned rights
Stop acquisition	The right to stop the measurement
Apply new I/O configuration	The right to apply a new I/O configuration
Change locked modules	The right to change the settings of a locked module. A locked module is a module, whose configuration cannot be changed. All the module properties are locked. All the signal properties like name, scaling, unit etc. are locked.
Apply new data store configuration	The right to apply a new data store configuration
Change locked data stores	The right to change the "Locked" property of a data store. A locked data store is a data store whose configuration cannot be changed.
Disconnect other user	The right to disconnect another client in case the server has no more free client licenses left at the time the current user tries to connect to the server with another client. If no more client licenses are available, a dialog will appear showing a list of the currently connected clients and offering the possibility to select and disconnect a client.

Table 9: User management, ibaPDA server rights

Note

The majority of the client rights for the user **admin** is fix and cannot be altered by this user.

Tip

One way to prevent acquisition from being unintentionally terminated by a user with permission to do so is to use the "Stop prevention" function in the I/O Manager. More information about this can be found in part 2 of the *ibaPDA* manual, chapter 4.11 "Stop prevention".

7.1.2 ibaPDA client rights

Option	Description
Changing the password	The right for a user to change his own password
Change server	The right to connect to a (different) *ibaPDA* server
Close client	The right to close the *ibaPDA* client
Resize client	The right to change the size of the program window or the display style, either using the mouse or using a key combination, e.g. to enable/disable full screen mode.
Use menu and toolbars	The right to use menu commands and buttons of the toolbar. Without this right, the menus and toolbars are disabled (grayed out), except for the toolbar of the views. The user can add/remove signals to/from the current views.
Interact with views	The right to interact with views, e.g. zoom in/out, show/hide signals or change the scaling in trend graphs. If a user is not authorized to interact with views, the right “Change layout” must also be disabled. This right does not apply to FFT and oscilloscope views. Interaction with those views is always possible.
Change layout	The right to change the current layout. If you do not have this right, then you cannot move, reduce or enlarge the *ibaPDA* client window. You cannot change views. You cannot change docking windows.
Switch between layouts	The right to switch layouts. Users with this right can switch between different layouts by using the layout toolbar or function key <F11>.
Load layouts from server	The right to load layouts from the server Users with this right can manually load layouts from the server via the Layout Manager. When logging in to the server, a query appears asking if layouts should be loaded from the server. Users who do not have this right cannot manually load layouts from the server, nor is there a query when they log in. Exception user admin: ■ Denied: Layouts can be loaded manually from the server. There is no query when logging in. ■ Granted: Layouts can be loaded manually from the server. There is an additional query when logging in.

Option	Description
Save layouts on server	The right to save current or modified layouts on the server. Users with this right can manually save layouts on the server via the Layout Manager. If layouts have been changed, an additional query appears asking whether layouts should be saved on the server when the user logs out or exits *ibaPDA*. Users who do not have this right cannot manually save layouts on the server, nor is there a query when they log out. Exception user admin: ■ Denied: Layouts can be saved manually on the server. There is no query when logging out. ■ Granted: Layouts can be saved manually on the server. There is an additional query when logging out if layouts have been changed. When saving on the server, the layouts are saved in the layout file (*.layouts) in the path for the server configuration files, e.g. `C:\ProgramData\iba\ibaPDA\Layouts`
Auto-close on inactivity	Clients that are logged in under a user with this right are automatically exited after the set time if no use occurred in this time. Times from 5 min to 24 h can be selected.
Auto-disconnect on inactivity	Clients, which are logged in by a user with this right will be disconnected from the *ibaPDA* server automatically if the set time has elapsed without any operation during this time. The last layout remains loaded and the log-in dialog appears on the screen. Times from 5 min to 24 h can be selected.
Force display style	Clients that are logged in under a user with this right show the set display style during starting.
Send manual triggers	The right to send start and stop triggers using the buttons in the toolbar of the data store or the context menu on the data store status tree.
Create ibaHD annotations	The right to enter new annotations based on a pre-configured annotation type (selection from a drop-down list).
Configure ibaHD annotations	The right to configure new annotation types

Note

The majority of the client rights for the user “admin” is fix and cannot be altered by this user. Exception: Auto- close / disconnect on inactivity.

Tip

If the menus and tool bars are disabled and you need to return to the user management dialog, press <Ctrl>+<U> for a new login to the server.

7.2 User management in a domain with Active Directory

The iba software products *ibaPDA*, *ibaHD-Server*, *ibaCapture* and *ibaDaVIS* support user management via Active Directory. Thus, not only are local users of the programs accepted, but also domain users or groups defined by the IT admins.

Note

ibaPDA only supports Active Directory security groups for user management. Distribution groups are not supported.

Further information on security groups and distribution groups can be found in the Microsoft documentation.

7.2.1 Advantages of the Active Directory method

For the IT administration, Active Directory offers the advantage of central user management for applications such as *ibaPDA*, *ibaHD-Server*, *ibaCapture* or *ibaDaVIS* in one domain. This simplifies the maintenance of user accounts and their authorizations within the applications.

Active Directory users, or *domain users* as they are called in the softwares, are users or groups defined in an Active Directory.

The individual users inherit the permissions from the group to which they belong. Therefore, it is not necessary to create all users individually in the software's user management system. If the group is created as a domain user, any user belonging to this group can log in with their domain credentials and use the software with the permissions assigned to the group. This means that adjustments can be made very easily by the IT administration, e.g., in the case of staff changes or new employees.

Individual users can still be created locally in the software's user management system if required, e.g., to obtain rights that differ from those of the group.

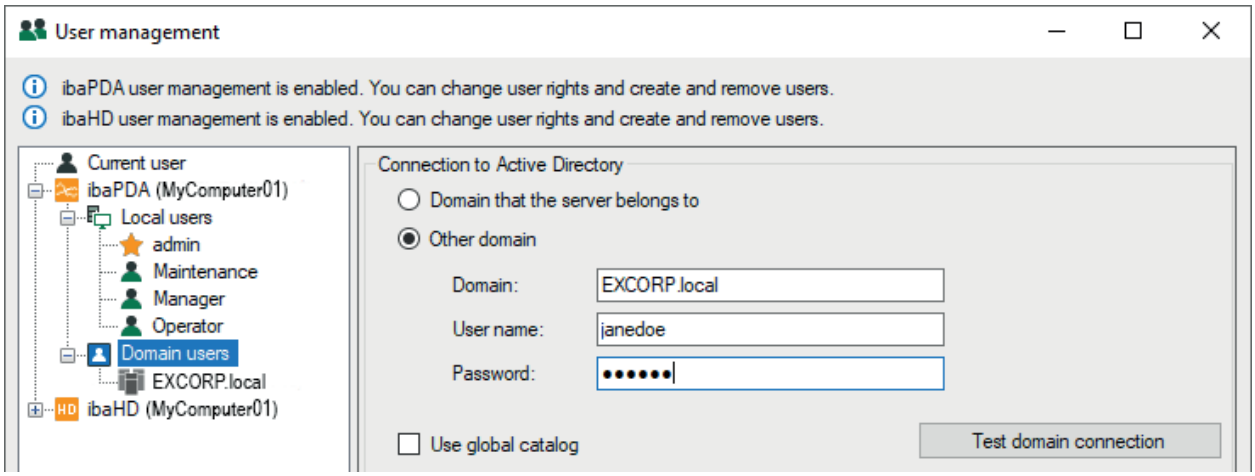
The software-specific rights are granted or denied to the individual users via the software's user management interfaces (*ibaPDA* or *ibaHD-Server*). This can be done by all users who have the *User administration* right.

7.2.2 Establish a connection to a domain

To add a domain user, you must first establish a connection to a domain. Under the *domain user* node in the user management dialog, you can set which Active Directory domain *ibaPDA* or *ibaHD-Server* you should connect to.

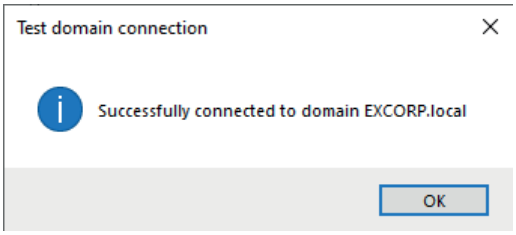
If the *ibaPDA*/*ibaHD-Server* computer is a member of a domain, you can decide to use this domain. It is also possible to use another domain by entering the domain name and the credentials of a user from that domain. Thus, it is possible to use the users of an Active Directory on an *ibaPDA* computer that is not integrated in a domain (e.g., only part of a workgroup). The prerequisite is, of course, that the domain is accessible so that *ibaPDA* can query the corresponding information.

ibaPDA or *ibaHD-Server* will then search for domain users and groups in the set domain and its sub-domains.



Proceed as follows:

1. If you have not already done so, log on as a user with the "User administration" right.
2. In the user tree, select the *Domain users* node. The *Connection to Active Directory* section appears in the right-hand dialog box.
3. If the computer is already in a domain and you want to access the users of this domain, select *Domain that the server belongs to*.
4. If you want to access the users or groups of another domain, select *Other domain* and specify the credentials of a valid domain user. It is recommended to use the full domain name (FQDN: Fully Qualified Domain Name) with all name components (e.g., including the suffix *.local*).
5. Click on <Test domain connection> to check that the information is correct. If the connection is successful, you will receive a message.



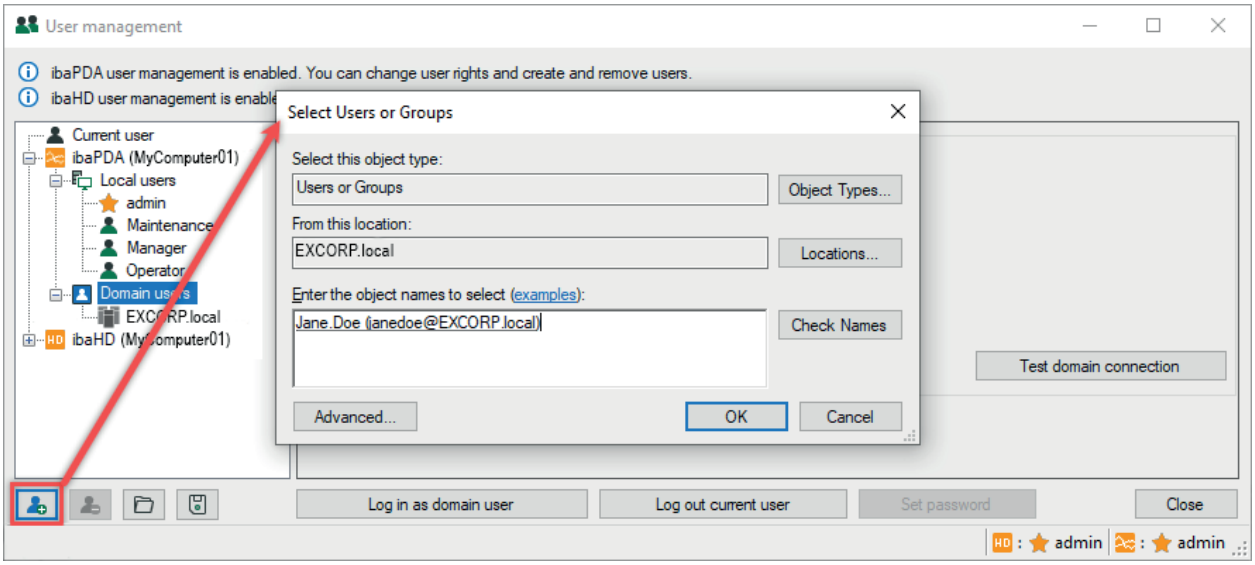
6. If you enable the *Use global catalog* option, then parent domains and trusted domains are also included when searching for users and groups.

At the bottom of the user tree there are four buttons (blue icons for domain users) with the following functions:

	Add local user/domain user (in *ibaPDA* client, a distinction is made between *ibaPDA* users and *ibaHD-Server* users).
	Delete selected local user/domain user
	Load users from an existing user file (*.pda_users)
	Save the configured (*ibaPDA*) users in a user file (*.pda_users), incl. all permissions and passwords

Table 10: User management buttons

7.2.3 Add domain user



Proceed as follows:

1. Make sure that any node in the *Domain users* section of the user tree is selected.
2. Click the "Add user" button at the bottom of the tree. The tooltip for the button shows the program and computer name for which the user is created, e.g., "ibaPDA (*computer name*)". The "Select users or groups" dialog opens (Windows standard dialog). The default domain is set as configured in the domain settings (see figure in 7.2.2, page 96).
3. Enter a user or group name in the empty field if you know it. If not, you can use the search function via <Advanced...>. Then select the desired user or group from the search results. If necessary, you can select several users or groups here, separated by semicolons.
4. Close the dialog with <OK>. The new user or group now appears in the user tree under a domain node.
5. If you want to add additional users, make sure that the *Domain users* node or the child domain node is selected.

7.2.4 Manage user rights

If a user or group is selected in the user tree, the corresponding permissions are displayed to the right of it. Select the top node *Current user* if you want to see the granted permissions for the currently logged-in user.

You can assign or revoke rights to users or groups if you are logged in to the *ibaPDA* server or *ibaHD-Server* as a user with the "User administration" right (e.g., admin).

Rights that are not available or cannot be changed are grayed out. For example, the "Change password" right is not available for domain users or groups.

By right-clicking in the table you can copy the user rights and transfer them to another user.

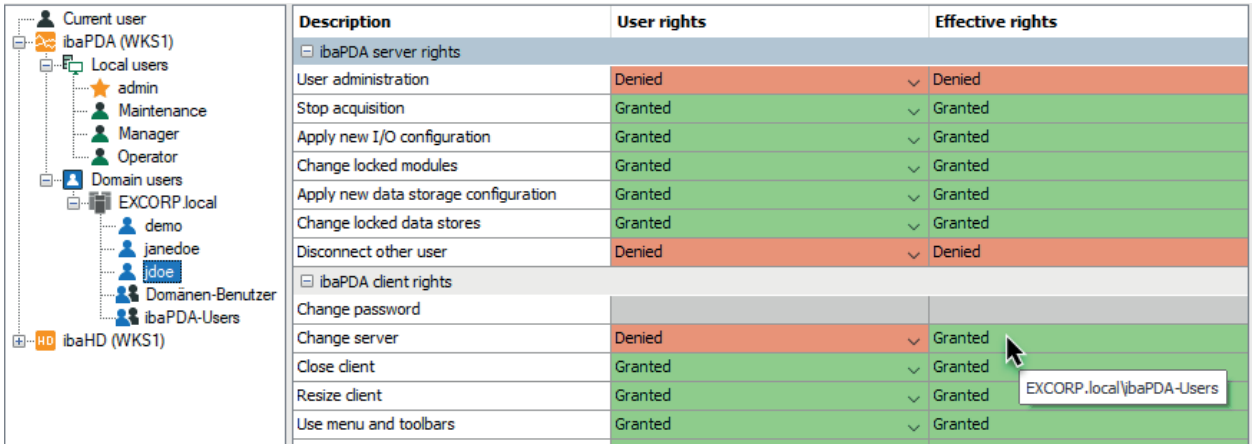
7.2.5 Effective rights

One domain user can belong to several groups. Permissions can be assigned both directly to the user or inherited from the groups to which they belong.

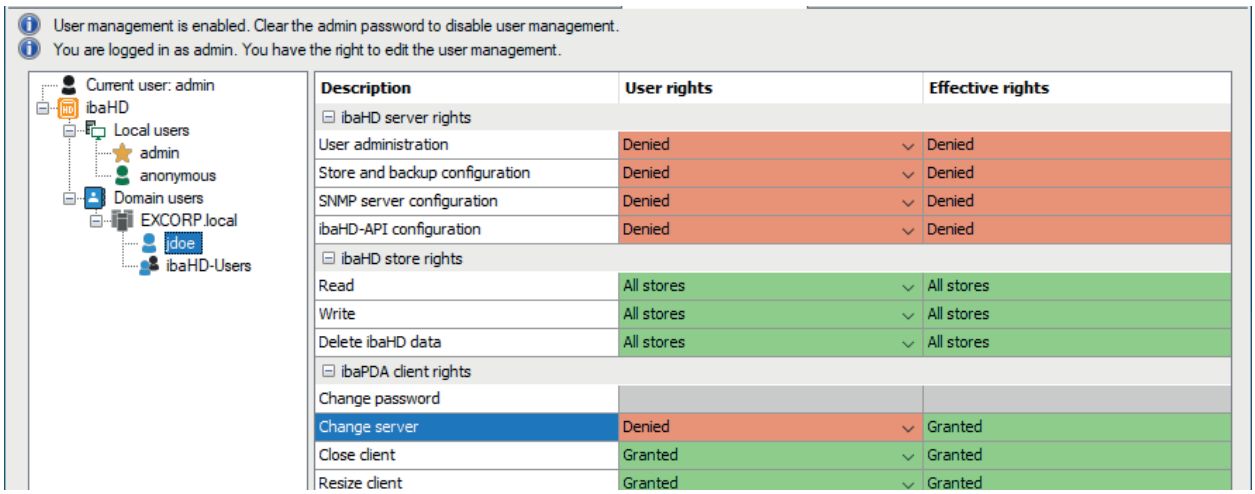
The "Effective rights" column shows the result of this mechanism and is only displayed for domain users, not for groups or local users.

If both a domain user and the user group to which it belongs are entered in the user management, there may be different settings for group and user.

In such a case, the **granted** rights always prevail over the denied ones. Example *ibaPDA:*



Example *ibaHD-Server*:



The figures show that the user “jdoe” has been denied the “Switch server” right. However, this user still gets the right granted because it is also a member of the “ibaPDA-users” or “ibaHD-users” group, which has this permission.

If a right has been granted, the tooltip on the mouse cursor shows the source(s) that inherited the right.

In this context, similar cases are also possible:

The configured domain user is a member of a group

User	Group	Effective
✓	✓	✓
✓	x	✓
x	✓	✓
x	x	x

✓ = Granted, x = Denied

The configured domain user is a member of more than one group

User	Group 1	Group 2	Effective
✓	✓	✓	✓
✓	✓	x	✓
✓	x	✓	✓
✓	x	x	✓
x	✓	✓	✓
x	✓	x	✓
x	x	✓	✓
x	x	x	x

✓ = Granted, x = Denied

In the case of users who are members of several groups, *ibaPDA* successively verifies the group based rights in their listed order in the list of SIDs received from the Active Directory controller (AD controller). This list is requested by ibaPDA from the AD controller whenever a domain user logs on. The group rights are combined with one another according to fixed rules. If the domain user is entered individually in the *ibaPDA* user management system, its rights are used as the basis for the additional combinations. If the user is not entered in the user management system, only the group rights are taken into account.

The order of the groups in the SID list is important for certain rights.

1. They are determined as granted/denied via a Boolean "OR" function, whereby "Granted" is dominant.
2. For the rights "Auto-close on inactivity" and "Auto-disconnect on inactivity", the lowest value is always applied. For example, if one group has a value of 10 minutes and the other group has a value of 1 minute, 1 minute is applied.
3. The "Force display style" right is applied from the user or the first group with a preselected display style.
4. For the rights "Load layouts from server" and "Save layouts on server", the layout directory is applied from the user or the first group for whom one of these rights is set.

The domain user is not entered in the user management system but is a member of multiple groups

Group 1	Group 2	Effective (current user)
✓	✓	✓
✓	x	✓
x	✓	✓
x	x	x

✓ = Granted, x = Denied

7.2.6 Logging in as a domain user

You can log in as a domain user in the user management dialog. To do this, click the button <Log in as domain user> at the bottom of the dialog. The button is only available if the *Domain user* node or one of its child nodes is selected in the user tree.

In the subsequent login dialog, you have various options for logging in.

Current Windows user

Logging in with the current Windows user is only possible for domain user accounts. Local Windows users are not supported.

The *ibaPDA server* or *ibaHD-Server* uses the SID (Security IDentifier) of the logged-in user to determine whether this user is present in the configured Active Directory (setting is configured via user management – *Domain users* node). If the search is successful, the user's group memberships in the Active Directory are determined. Based on these, the user's rights within *ibaPDA/ibaHD-Server* are configured via the *ibaPDA*/*ibaHD* user management dialog. It is not necessary to enter a password for the current Windows user, because their credentials have already been verified during Windows logon.

Other user

You can also log in under "Other user" with the full credentials of a user. The drop-down list in the *Users* field shows all configured local users and domain users that are available for selection.

To log in as a domain user, enter the user name in the user field using the syntax `Domain\User name`. It is recommended to use the full domain name (FQDN: Fully Qualified Domain Name) with all name components (e.g., including the suffix *.local*). Additionally, you must enter the password of the domain user.

The *ibaPDA*/*ibaHD-Server* logs on to the desired domain with the entered credentials, thereby verifies the user and queries the SID.

The rest of the procedure is identical to that for the "Current Windows user":

The *ibaPDA server* or *ibaHD-Server* uses the SID to determine whether the user exists in the configured Active Directory. If the search is successful, the user's group memberships in the Active Directory are determined. Based on these, the user's rights within *ibaPDA*/*ibaHD-Server* are configured via the *ibaPDA*/*ibaHD* user management dialog.

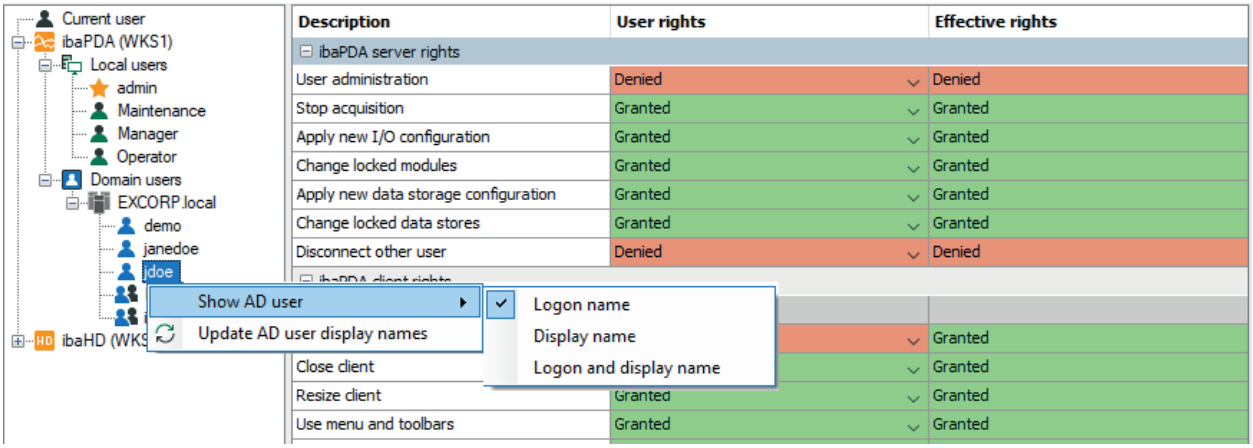
7.2.7 Indication of logon name and display name

Active Directory users have a logon name and a display name. The logon name is indicated in the user management by default.

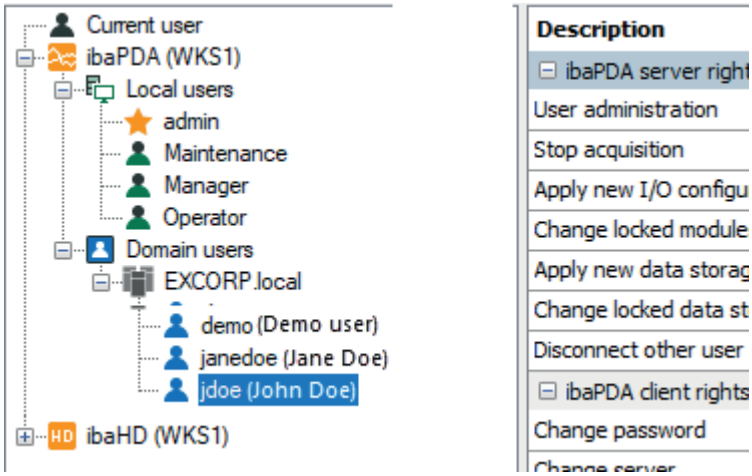
ibaPDA will retrieve the display name of new Active Directory users that are added. For existing Active Directory users, you can let *ibaPDA* retrieve the display name by executing the command *Update AD user display names*.

By using the context menu in the domain users area of the user tree you can decide which name is shown in the tree:

- Logon name: The logon name is shown in the tree and the display name is shown as tooltip.
- Display name: The display name is shown in the tree and the logon name is shown as tooltip.
- Logon and display name: The logon name is shown in the tree, followed by the display name in brackets.



Context menu on user tree, domain users area



Logon name and display name (in brackets)

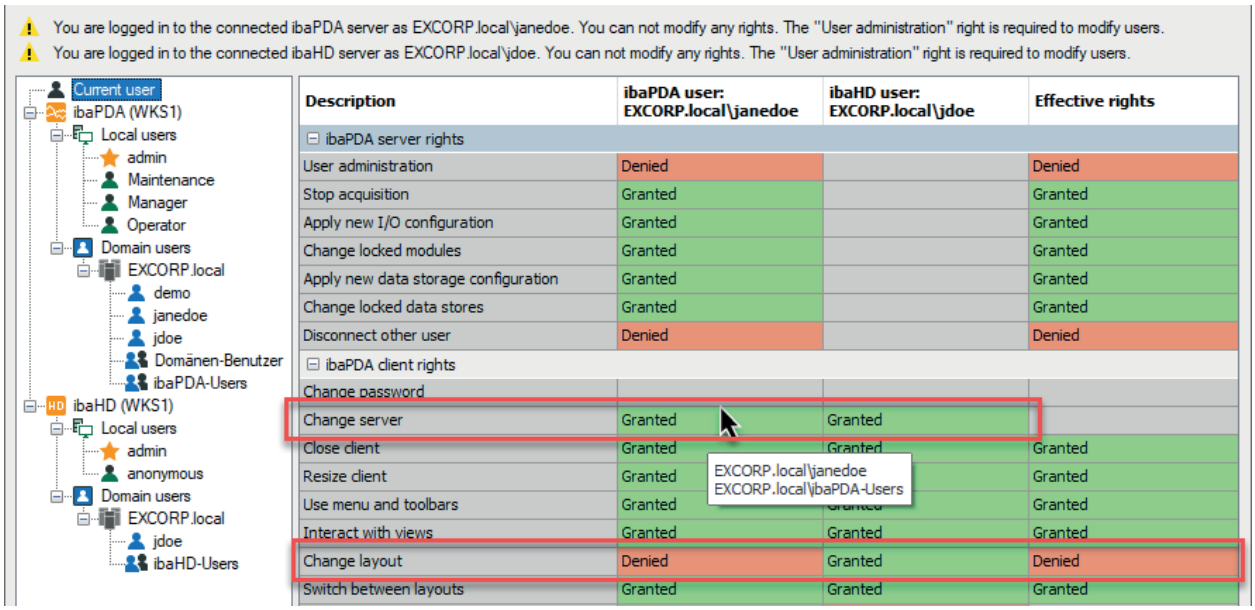
7.3 Rights when simultaneously logged in to ibaPDA and ibaHD-Server

The *ibaPDA* client program can connect to an *ibaPDA* server as well as to an *ibaHD-Server*. Users can be connected to both servers simultaneously and logged in as the same or different users. On the other hand, exclusive connections only to the *ibaPDA server* or *ibaHD-Server* are also possible. The *ibaPDA* client program can thus assume the role of an *ibaPDA* client and an *ibaHD* client (both are also possible at the same time).

Note

Simultaneous login to *ibaPDA server* and *ibaHD-Server* with different users is possible, but not necessarily useful. For example, in cases where both an *ibaPDA* user and an *ibaHD-Server* user have the same rights, the system will determine the combined effective rights.

If you select the top node *Current user* in the user tree, then the table shows the user accounts that were used to log in to *ibaPDA* and *ibaHD-Server* respectively.



The column "ibaPDA user" shows the rights that are granted when logging in to the *ibaPDA* server. If the user is a domain user, then the mouse cursor tooltip indicates the source(s) from which the rights were inherited. This user does not need to exist as a single user in the user tree if it belongs to a group that is set as a domain user.

The column "ibaHD user" shows the same for *ibaHD-Server* users.

The "Effective rights" column shows the result of combining the permissions of *ibaPDA* and *ibaHD-Server* users. For most rights, an AND operation is applied for this purpose. This means that both users must have been granted each the right in order to also receive the effective right.

ibaPDA server rights are only available to *ibaPDA* users. Some of the *ibaPDA* client rights are handled differently and are not combined. These rights will then be unavailable (grayed out) in the Effective rights column.

7.3.1 ibaPDA-client rights for PDA and HD clients

ibaPDA client right	ibaPDA client	Log. operation	ibaHD client	Effective	Comment
Change password	X		X	-	Change you rown password on the relevant server
Change server	X		X	-	Change the respective server
Close client	X	AND	X	X	"Denied" is dominant; this affects the shared *ibaPDA* client program.
Resize client	X	AND	X	X	"Denied" is dominant; this affects the shared *ibaPDA* client program.
Use menu and toolbars	X	AND	X	X	"Denied" is dominant; this affects the shared *ibaPDA* client program.
Interact with views	X	AND	X	X	"Denied" is dominant; this affects the shared *ibaPDA* client program.
Change layout	X	AND	X	X	"Denied" is dominant; this affects the shared *ibaPDA* client program.
Switch between layouts	X	AND	X	X	"Denied" is dominant; this affects the shared *ibaPDA* client program.
Load layouts from server	X		X	-	CurrentLayout.lay is loaded from the relevant server;if granted for both, the layouts overwrite each other.
Save layouts on server	X		X	-	Layout is saved on the relevant server; PDA user saves to PDA server, HD user saves to HD server.
Auto-close on inactivity	X		X	X	Lowest current value is applied.
Auto-disconnect on inactivity	X		X	-	The user concerned is logged out; *ibaPDA* client program remains open.

ibaPDA client right	ibaPDA client	Log. operation	ibaHD client	Effective	Comment
Force display style	X	AND/OR	X	(X)[1)]	May only be enabled on one server or settings must be the same
Send manual triggers	X	AND	X	X	"Denied" is dominant.
Create ibaHD annotations	X	AND	X	X	"Denied" is dominant.
Configure ibaHD annotations	X	AND	X	X	“Denied" is dominant.

Table 11: Combination of ibaPDA and ibaHD-Server client rights for ibaPDA clients

[1)] Disabled, if settings are not the same

7.3.2 ibaPDA server rights for PDA and HD clients

ibaPDA server right	ibaPDA client	Log. operation	ibaHD client	Effective	Comment
User administration	X			= PDA client	Right only exists on *ibaPDA* server
Stop acquisition	X			= PDA client	Right only exists on *ibaPDA* server
Apply new I/O configuration	X			= PDA client	Right only exists on *ibaPDA* server
Change locked modules	X			= PDA client	Right only exists on ibaPDA server.
Apply new data storage configuration	X			= PDA client	Right only exists on *ibaPDA* server
Change locked data stores	X			= PDA client	Right only exists on *ibaPDA* server
Disconnect other user	X			= PDA client	Right only exists on *ibaPDA* server

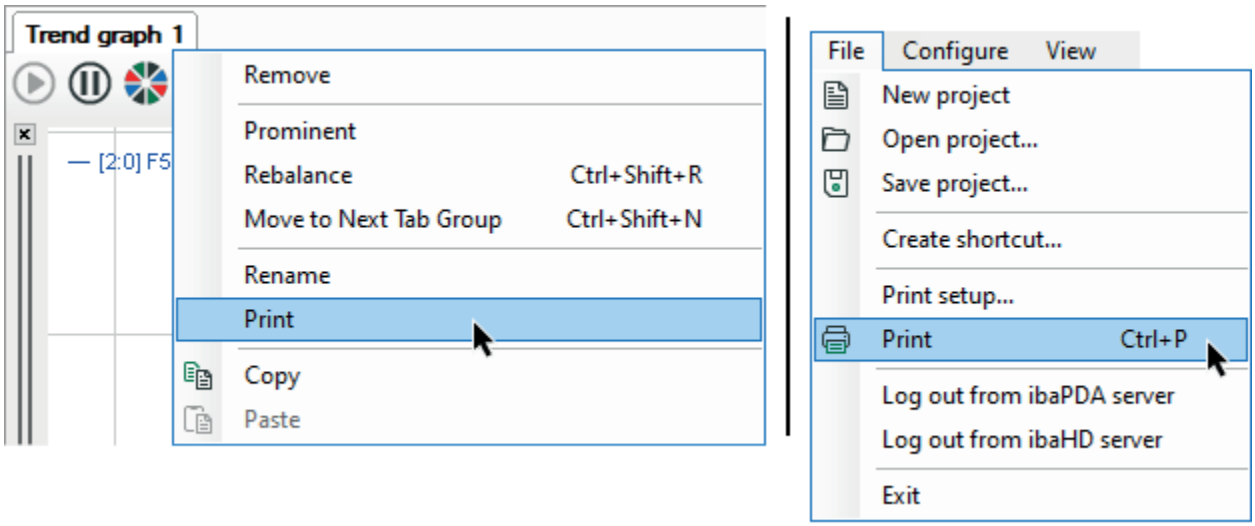
Table 12: Combination of ibaPDA and ibaHD-Server client rights for ibaPDA server

8 Printing

ibaPDA supports printing of each view separately, or all panes at once.

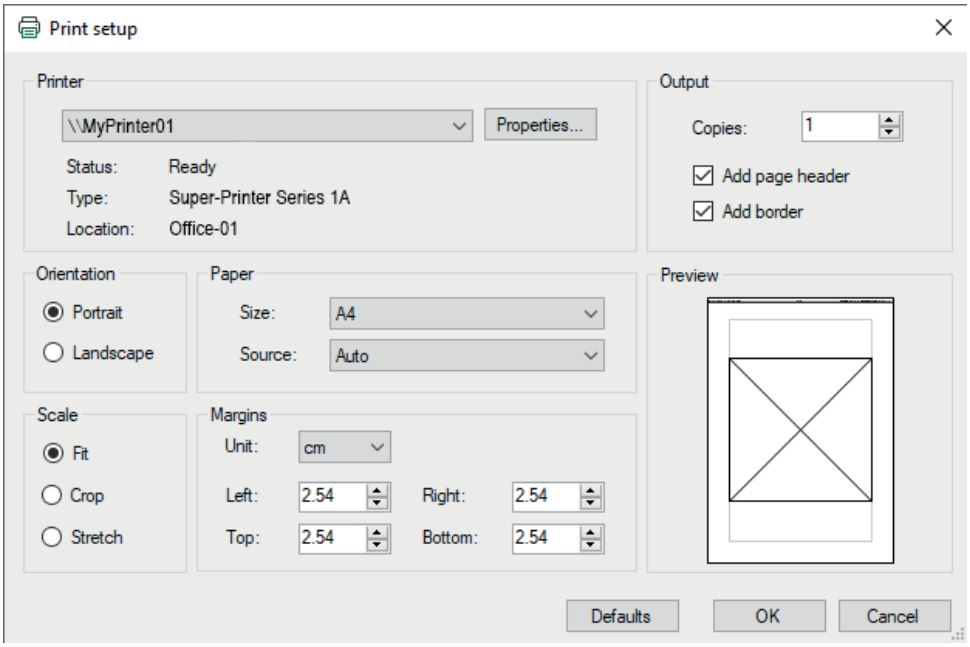
In order to print a single view, right-click on the tab of the relevant view and select *Print* from the context menu.

To print all views at once, select *Print* in the *File* menu or press <Ctrl>+<P>.



8.1 Printer setup

The *ibaPDA* print setup dialog can be accessed in the *File– Print settings…* menu. The dialog also opens when printing a single view via the context menu of the View tab.



This dialog provides the usual printing options:

- printer selection and the possibility to open the printer specific options dialog (<Properties...> button)

- Number of copies
- Orientation
- Page size and printer tray
- Page margins

Some printing options are more specific to *ibaPDA*:

- Output
  - A header containing time stamp, title of the printed view and the current *ibaPDA*-version will be inserted on the page.
  - A frame will be printed around the view.
- Scale
  - *Fit*: the views to be printed are scaled to fit the print range without changing their aspect ratio.
  - *Crop*: the views to be printed are not scaled. The views are printed in screen size. The images are cropped to the print range if larger.
  - *Stretch*: the views to be printed fit the entire print area. The aspect ratio is not observed.
- Margins
  - The minimal values depend on the non-printable page area of the selected printer.
- Preview
  - Shows a schematic print preview. If the *Fit* scaling mode is selected, the view's aspect ratio is considered square.

8.2 Settings for printing views

In addition to the aforementioned general print setup, there are more settings to be configured in the C*lient preferences* dialog or in the *Properties* dialog of the views to be printed.

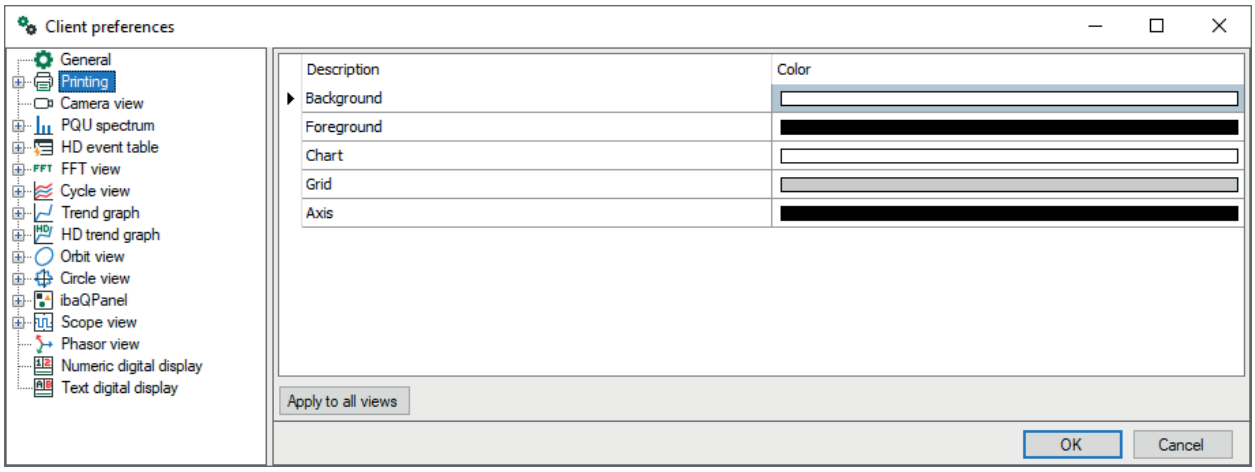
This dialog configures the color settings for printout of views. If, for instance, a view with a black background works well on a screen, it is a reasonable approach to use a white background in a printout in order to save toner.

General print settings are available on the "Printing" item, followed by subordinated branches for FFT view, trend graph/HD trend graph, camera, cycle, scope view etc.; *ibaQPanel* elements, like multi-state picture, label, bar chart, text input control and QPanel.

For all these element types, foreground and background colors can be usually set in general and individually. Some additional color settings are available depending on the element.

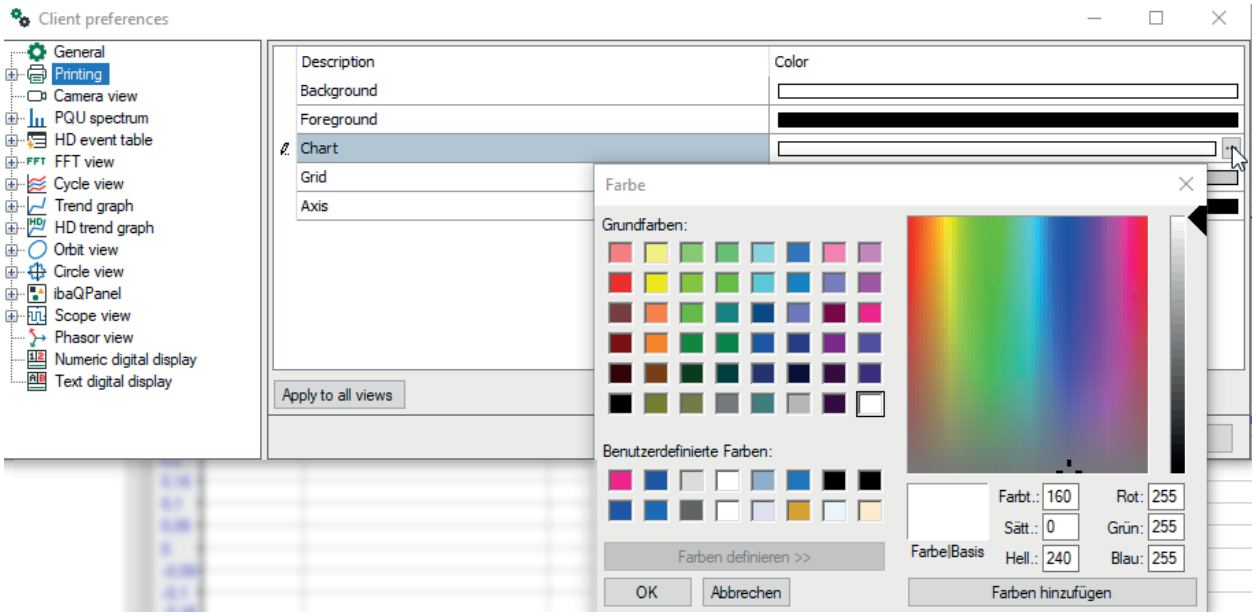
Note

If you want to set or modify the printing options of a specific view, use the "Properties" dialog of the view in question.



In order to make a general color setting, select the "Printing" main branch in the tree. Click in a cell in the color column and select the desired color. If you want the new settings to be applied to all types of views, click on the <Apply to all views> button.

Click <OK> to close the dialog.



In order to change the color settings for each element individually, select an element from the tree and click in a cell of the color column. Select the desired color and click <OK>.

Note

Changes of the color settings in the preferences apply to newly opened views only.

9 Support and contact

Support

Phone: +49 911 97282-14

Email: support@iba-ag.com

Note

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

Product Security Incident Response Team (PSIRT) at iba

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

Email: psirt@iba-ag.com

Contact

Headquarters

iba AG
Gebhardtstrasse 10–20
90762 Fuerth
Germany

Phone: +49 911 97282-0

Email: iba@iba-ag.com

Mailing address

iba AG
Postbox 1828
D-90708 Fuerth, Germany

Delivery address

iba AG
Gebhardtstrasse 10
90762 Fuerth, Germany

Regional and worldwide

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

www.iba-ag.com

ibaPDA

I/O Manager

Manual Part 2

Issue 8.14

Measurement Systems for Industry and Energy

www.iba-ag.com

Manufacturer

iba AG

Gebhardtstrasse 10-20

90762 Fuerth

Germany

Contacts

Main office	+49 911 97282-0
Support	+49 911 97282-14
Engineering	+49 911 97282-13
E-mail	iba@iba-ag.com
Web	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 2026, 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 and can be found in the iba help center docs.iba-ag.com.

Version	Date	Revision	Author	Version SW
8.14	06-2026	Module overview: new column Samples/s; OPC UA server: refresh rate of the tags	RM	8.14.0

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

1 About this documentation 9

1.1 Target group and previous knowledge 9

1.2 Notations 9

1.3 Used symbols 10

1.4 Documentation structure 11

2 Introduction 12

3 I/O Manager toolbar 13

4 General settings and information 16

4.1 Settings 16

4.2 System information 19

4.3 Signal names 21

4.4 Unit table 23

4.4.1 Unit systems 26

4.4.2 Units 27

4.4.3 Abbreviations 29

4.4.4 Using the unit system in the signal table 31

4.4.5 Changing the unit system 33

4.4.6 Indicating the active unit system 34

4.4.7 Unit table in the data file 35

4.5 Address books 36

4.6 Certificates 38

4.6.1 Certificates 38

4.6.2 Central certificate store 39

4.6.3 Manage certificates 41

4.6.3.1 Generate a new certificate 42

4.6.3.2 Add certificate 43

4.6.3.3 Export certificates 43

4.6.4 Use certificates 43

4.6.5 Save and protect certificates 44

4.6.6 Check information about a certificate 45

4.7 Time synchronization 45
4.7.1 DCF77 46
4.7.1.1 DCF77 Configuration 47
4.7.2 IEC 1131 49
4.7.3 PTPv2 50
4.7.4 ibaClock 53
4.7.5 NTP via Windows 54
4.8 Module overview 55
4.9 Knowhow protection 59
4.9.1 Application range of know-how protection 59
4.9.2 Creating a protection scheme 60
4.9.3 Application of a protection scheme 62
4.9.4 Removing the protection 62
4.9.5 Importing and exporting protected elements 63
4.9.6 Unlocking protected items 63
4.10 Watchdog 64
4.10.1 Information about the watchdog telegram structure, ASCII format 65
4.10.2 Information about the watchdog telegram structure, binary format 66
4.11 External configuration 67
4.12 Stop prevention 68
4.13 Boards 69
4.14 Interfaces 71
4.15 Interrupt Info 74
4.16 Multistation 77
4.17 OPC server 78
4.18 OPC UA Server 79
4.18.1 General information 79
4.19 SNMP server 80
4.20 IEC61850 server 81
4.21 Signal usage display 81

5 Groups and vector signals 82

5.1 Adding groups 82
5.2 Renaming a group 83
5.3 Deleting groups 83
5.4 Move groups 83
5.5 Assigning signals to a group 83
5.6 Deleting signals from a group 84
5.7 Remove inactive signals from a group 84
5.8 Creating vector signals 84
5.9 Export groups 85

6 Text signals and text processing 86

6.1 Using the text signals 86
6.2 Properties of the text signals 87
6.3 Using text signals 94
6.3.1 General 94
6.3.2 Configuration 95
6.3.3 Creating text signals 97
6.3.4 Delete the text signals 98
6.3.5 Defining and assigning text signals 98
6.4 Text interface 99
6.5 Text splitter module 100
6.5.1 Add text splitter module 101
6.5.2 Module settings 101
6.5.2.1 Fixed width split mode 103
6.5.2.2 Delimited split mode 104
6.5.2.3 Split mode JSON 106
6.5.3 Signal configuration 107
6.6 Text signals via TCP/IP and UDP 108
6.6.1 Add module 108
6.6.2 Module settings 109
6.6.3 Signal configuration 110

6.7 Text signals via serial (COM) interface ... 111
6.7.1 Add module ... 111
6.7.2 Module settings ... 111
6.7.3 Signal configuration ... 112
6.8 Text signals via text file (file text) ... 113
6.8.1 Add module ... 113
6.8.2 Module settings ... 113
6.8.3 Signal configuration ... 115
6.9 Custom text (text creator) ... 116
6.9.1 Add module ... 116
6.9.2 Module settings ... 116
6.9.3 Text creator (Plaintext) ... 118
6.9.4 Text creation XML/JSON ... 121
6.9.4.1 Data structure tree ... 122
6.9.4.2 Signal tree ... 126
6.9.4.3 Synchronize arrays ... 128
6.9.5 Signal configuration ... 129
6.10 Text via ibaQPanel text input ... 130
6.10.1 Add module ... 130
6.10.2 Module settings ... 130
6.10.3 Signal configuration ... 131
6.11 Text shift register ... 132
6.11.1 Add module ... 132
6.11.2 Module settings ... 132
6.11.3 Signal configuration ... 134
6.11.4 Application example for text shift register ... 135
6.12 Text signals via OPC / OPC UA ... 137
6.13 Text signals via other interfaces ... 137
7 Outputs ... 138
7.1 Create outputs ... 140
7.2 Adding output modules ... 140
7.3 Setting up output modules ... 141

7.3.1 General module settings ... 141
7.3.2 OPC output module ... 142
7.3.2.1 General tab ... 142
7.3.2.2 Analog and Digital tab ... 143
7.3.3 Email ... 143
7.3.3.1 Basic settings (accounts and fields) ... 144
7.3.3.2 Email module settings ... 145
7.3.4 ibaNet750-BM, -BM-D output modules ... 149
7.3.4.1 General tab ... 149
7.3.4.2 Analog and Digital tab ... 150
7.3.5 ibaL2B... output modules ... 151
7.3.5.1 General tab ... 151
7.3.5.2 Analog and Digital tab ... 151
7.3.6 Reflective Memory output module ... 152
7.3.6.1 General tab ... 152
7.3.6.2 Analog and Digital tab ... 152
7.3.7 EtherNet/IP I/O output modules ... 152
7.3.8 Generic TCP output module ... 153
7.3.8.1 General tab ... 153
7.3.8.2 Analog and Digital tab ... 154
7.3.9 Generic UDP output module ... 155
7.3.9.1 General tab ... 155
7.3.9.2 Analog and Digital tab ... 155
7.3.10 Modbus-TCP Client output module ... 156
7.3.11 ABB-Xplorer output module ... 157
7.3.12 OMRON-Xplorer output module ... 158
7.3.13 S7-Xplorer output module ... 159
7.3.14 TwinCAT-Xplorer output module ... 159
7.3.15 ibaCapture output module ... 161
7.3.15.1 General tab ... 161
7.3.15.2 Video server tab ... 161
7.3.15.3 Analog and Digital tab ... 162
7.3.16 ibaVision output module ... 163
7.3.17 FOB alarm output module ... 163

7.3.17.1 General tab 163
7.3.17.2 Analog and Digital tab 163
7.3.18 Outputs to the iba modular system (ibaPADU-S) 164
7.3.19 Outputs to iba bus modules (ibaBM-...) 165
7.3.20 Outputs to iba system interconnections (ibaLink-...) 165
7.3.21 OPC UA client output module 166
7.3.22 MQTT output module 168
7.3.23 Output module for SQL commands to SQL databases 172
7.3.24 ibaNet-E output modules 173
7.3.25 ibaM-DAQ outputs 174
8 The I/O configuration file 175
8.1 Saving an I/O configuration 175
8.2 Loading an existing I/O configuration 176
8.3 Loading a backup configuration file 177
8.4 Exporting an I/O configuration to a text file 177
8.5 Importing a configuration text file 180
9 Support and contact 181

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 they are capable of assessing safety and recognizing possible consequences and risks on the basis of their specialist training, knowledge and experience and knowledge of the standard regulations.

1.2 Notations

In this manual, the following notations are used:

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

1.3 Used symbols

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

Danger!

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

Observe the specified measures.

Warning!

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

Observe the specified measures.

Caution!

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

Observe the specified measures.

Note

A note specifies special requirements or actions to be observed.

Tip

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

Other documentation

Reference to additional documentation or further reading.

1.4 Documentation structure

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.

Part 1	Introduction and installation	General notes, license policy and add-ons Installation and program start User interface, system architecture, client server User management, printing
Part 2	I/O Manager	Basic information about the I/O Manager, general settings Groups and vector signals, text signals, outputs, configuration files
Part 3	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.
Part 4	Expression builder	All functions for calculating virtual signals
Part 5	Data storage	Types of data recording, recording profiles, signal selection
Part 6	Data visualization	All display modes for live data, their operation and settings
Part 7	Appendix	Various additions, error listings, etc.

2 Introduction

"I/O" stands for "Input/Output". The I/O Manager is the main dialog for all configuration settings related to signals and interfaces.

All data sources and signals must first be defined in the I/O Manager. How the signals are then eventually used (display, recording, etc.) is subsequently determined in the other dialogs.

All setting options are clearly displayed in the I/O Manager. When configuring the input and output modules, the user is actively supported through context-sensitive selection menus. This helps to avoid configuration errors. Each change in configurations is checked by the system before being applied.

Note

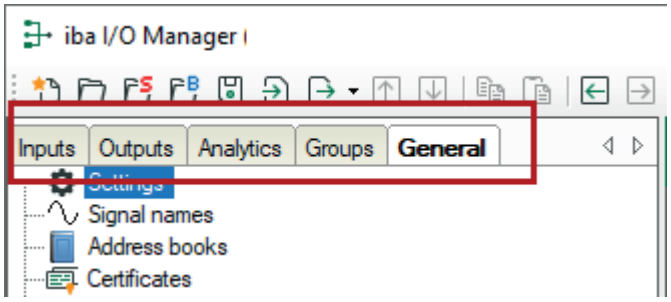
Changing the configuration settings in the I/O Manager always causes acquisition to automatically stop and restart!

The I/O Manager can be accessed as follows:

- Via the *Configure - I/O Manager* menu
- Via the button

You will find four main sections in the I/O Manager where adjustments can make other settings:

- Inputs
- Outputs
- Analytics
- Groups
- General



Due to the extent and scope, the description of the data interfaces, which are an integral part of the I/O Manager, is addressed separately in part 3 of the manual.

3 I/O Manager toolbar

Toolbar

Symbol	Description
	New configuration; then choose whether an empty configuration (reset to factory settings without auto-detect) or a new configuration with auto-detecting the connected modules should be created. If you click on this button while the acquisition is running, an additional option appears offering a new configuration with auto-detect after automatic stop of acquisition.
	Opens an existing I/O configuration file (XML-file .io). Afterwards, if applicable, the possibility to keep or delete initial values, if the configuration contains corresponding modules.
	Opens an existing I/O configuration file (.io or .zip) in simulation mode. The loaded configuration is not aligned with the currently existing hardware or licenses and cannot be applied. The simulation mode only provides for display of any I/O configuration.
	Opens access to the automatically saved backups of the I/O configuration in the path `%ProgramData%iba\ibaPDA\Backup` on the *ibaPDA*-Server computer. In a dialog window you can select and load the desired backup. Furthermore, you can use the dialog to delete a backup.
	Saves the current, full I/O configuration in an XML-file. You can select or enter the path and filename in the following dialog. The file extension .io is added automatically.
	Importing an I/O configuration, which is available in the form of a text file. The I/O configurations of an older *ibaPDA* system (< V6) as well as modules, groups or profiles that were exported from another *ibaPDA* system or configured using other tools, e.g., MS Excel, can also be imported. Signal addresses can be imported as decimal or hexadecimal (prefix “0x”) values. You cannot import a complete I/O configuration by using the import function.
	Button with drop-down list to export to a text file one out of the following: ■ the current I/O configuration, i.e. all modules, groups and profiles ■ all or selected modules ■ all groups ■ all *InSpectra*-profiles
	These buttons are disabled, unless you can move selected modules within the interface. Move the module by one position up or down in the signal tree of the interface.
	This Button is only enabled if a signal table is open. Pressing this Button will copy the open signal table to the Windows clipboard in order to paste it into another signal table or e.g. Microsoft Excel/Word.

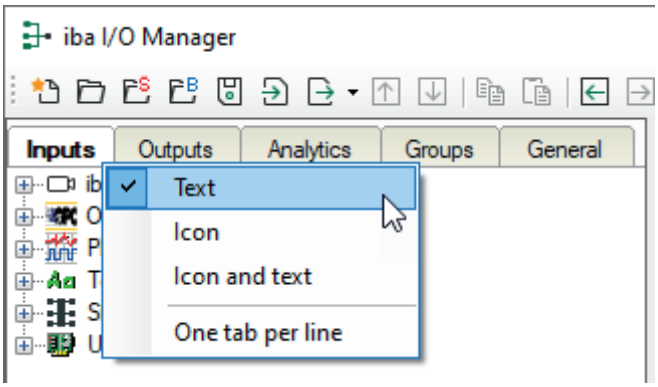
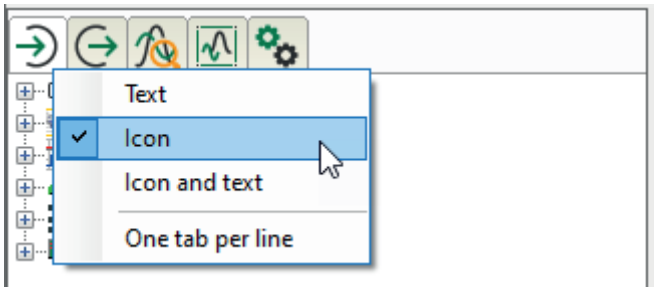
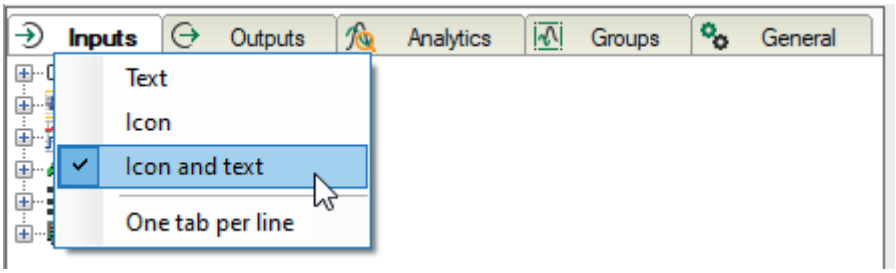
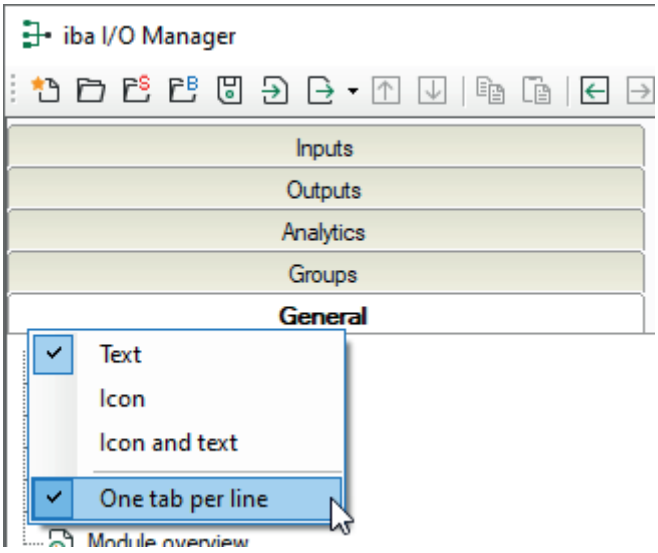
Symbol	Description
	This Button is only enabled if a signal table is open. Clicking this Button will paste the contents of the Windows clipboard, provided it is a compatible table, created in I/O Manager or e.g. Microsoft Excel/Word.
	Go forward/backward buttons: Navigate to the previous or following step in the I/O Manager.

Tabs

Tabs	Icon	Description
Inputs		Here, you can configure data sources (interfaces), modules and signals.
Outputs		Here, you can configure output signals, e-mails and SQL commands for interfaces and modules, which support output of data.
Analytics		Here, you'll find internal interfaces and modules which are rather used for analytic processing of the input and other signals. Included interfaces and add-ons are: ■ Virtual ■ InCycle ■ InSpectra ■ Snapshots Before introducing the *Analytics* tab, these interfaces were placed in the *Inputs* tab. When updating from older *ibaPDA* versions to v8.4.0 or higher these interfaces will be moved automatically to the *Analytics* tab.
Groups		Here, you can define signal groups, assign signals to groups and define multidimensional signals (vectors).
General		Here, you can carry out general settings for interfaces and modules, such as address books and certificate management, time synchronization set-up or visibility of interfaces.

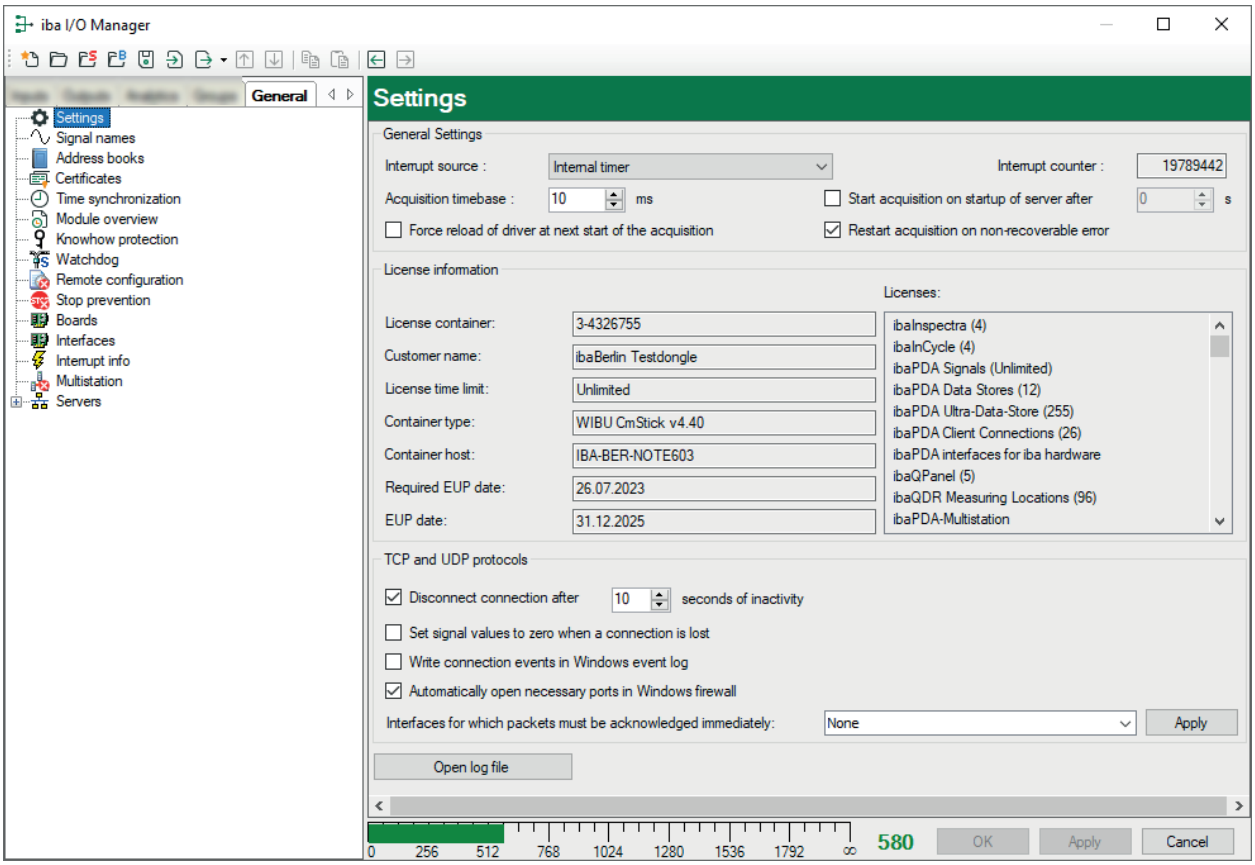
You can set up the representation and arrangement of the tabs according to your needs. The arrangement of the tabs is responsive to the available size of the window. The program tries to show the complete text in the tabs. If there is not enough space, the tabs will be arranged one above the other.

Concerning the contents of the tabs you may choose between text, icon or text + icon. Use the context menu on any tab to make your choice.

Text (default)	
Icons	
Icons and text	
One tab per line	

4 General settings and information

This section describes the information provided in the *General* tab.



4.1 Settings

General settings

Interrupt source

The interrupt source determines the entire timing of the data acquisition in *ibaPDA*. From this drop-down list, select one card which should act as the interrupt source on the PCI-/PCIe-bus.

The iba cards are fitted with a very precise and stable timer chip. If iba cards are installed in the computer they are listed in the drop-down list for the interrupt source. If cards of the *ibaFOB-D* family are used they are preferred as interrupt source compared to older cards.

If no iba cards are used, the less stable Windows timer will be used instead.

Interrupt counter

The interrupt counter is incrementing with a rate of 1000 / sec when the drivers and *ibaPDA* server are started and the application time base is set to 1 ms.

Acquisition timebase

Enter the timebase here.

The acquisition timebase is the superior clock for the entire system and the fastest period for sampling data in running acquisition. Slower sampling can can be configured for the individual modules by the module's timebase setting.

Excluded are data which are acquired over fiber optic connections with ibaNet 32Mbit Flex protocol, data from *ibaInSpectra* modules and *ibaInCycle* modules as well as virtual modules. In these cases timebases faster than the acquisition timebase can be configured.

Faster sampling induces higher load and data volume. Therefore, it is always a reasonable approach to set the sampling on an appropriate level , i.e. as low as required for the intended application .

The configured module timebases cannot be chosen arbitrarily and independently from each other in order to ensure the good performance of the system even with high amounts of data. An important quantity is the “driver period” which results from the module timebases. It is determined by the least common multiple of all module timebases.

The longest driver period should not exceed 1000 ms. The ratio of smallest module timebase and driver period must be limited. Bad or inadmissible configurations will be indicated during the validation on start of the acquisition by warning or error messages.

You will find an overview of all modules including their configured time bases in *Module overview* (see ↗ *Module overview, page 55*).

As the name implies, the acquisition timebase refers only to data acquisition (sampling) not recording. The timebase for the data recording which is used to write the measured values into the data file is configured in the data storage profiles.

Start acquisition on startup of server after ... s
If you check this option, the acquisition will be started automatically after a system restart of the server (equivalent of the GO button). You have to select this option if you need a full automatic startup sequence of the system, e.g. after a power failure. A prerequisite is, of course, that the *ibaPDA* server service starts automatically when the system is booting.

In the field next to it, you can set a delay time in seconds. A delay of >0 may be useful if other components, drivers or interfaces are involved, which are needed to correctly start the acquisition and that need some time to be available (e.g. remote OPC server).

Force reload of driver at next start of the acquisition
If you check this option, the drivers will be reloaded and restarted with the next change of the I/O configuration. The selection of this option will not be retained. The option will be disabled automatically after the driver has been loaded and started. You have to enable it again for the next repetition.

Selecting this option is recommended if *ibaPDA* does not experience any interrupts. The system always checks the interrupts when the acquisition starts. If no interrupts are detected, an error message is shown.

Restart acquisition on non-recoverable error
If you check this option, data acquisition will be stopped automatically in the event of an non-recoverable error.

The automatic restart is the attempt to fix the error. At the same time, faulty modules may eventually be disabled. For most of the applications it is vital, that the data acquisition with *ibaPDA* is running continuously. It is therefore more beneficial to have a running data acquisition, even with reduced signal count, than a stopped data acquisition.

License information

In the *License information* area you'll find important information about your current software license.

License container

The license container number is important for all service queries and upgrades. Please provide the license container number to our support team. It is linked to your license container, i.e. either MARX dongle, WIBU dongle or WIBU soft license and stored in our database.

Customer name

In this field, you will find the name of the customer for which this license has been approved. For projects, which were initially completed via system integrators, registration of the end user should have been requested from iba either from the outset or, at the latest, after the warranty period has expired. This makes it easier to assign the license in the case of later extensions or support cases.

License time limit

This field shows the validity time of the licenses. Depending on the container type there is either a remaining validity time given in days or hours or there is an expiration date.

Container type

This field indicates whether a MARX dongle, a WIBU dongle (WIBU CmStick) or a WIBU soft license (WIBU CmActLicense) is used.

Container host

This field shows the name of the computer where the license container is attached, which supplies the licenses obtained by the application.

Required EUP date and EUP date

The EUP date specifies the date until which the period of free software updates is valid. After the update period has expired you still can continue running the program as usual but only with the features which were already available until then.

For more information, see Part 1, *Update policy*.

Licenses

This window displays all the available basic and additional licenses, such as clients, data storages, interfaces, plug-ins etc.

If multiple licenses are applicable, e.g. clients, Xplorer interfaces, data storages etc., the number of respective licenses is given in brackets.

Active signals

A bar on the scale and the value behind the scale show the number of currently active signals. The end value of the scale indicates the maximum permissible number of signals according to the license.

TCP and UDP protocols

Disconnect connection after… seconds without activity

Enabling of this option will activate a timeout for TCP/IP connections. If no messages are received over the TCP/IP connections within the specified time (to be entered in the entry field next to the checkbox), the corresponding connection will be disconnected. Disconnecting unused connections saves resources.

Set signal values to zero when a connection is lost
Enabling this option sets all measured signals of a connection to zero if the connection is interrupted. Otherwise the signal values would show the value at the time of the connection interruption.

Write connection events in Windows event log
Use this option to enter selected events from communication field in the Windows event log so that they are visible in the Windows event display "System". This refers to Ethernet-based interfaces which are incapable to generate their own log, e.g. because they are implemented in the driver. Most of the Ethernet-based interfaces which are supported by *ibaPDA* generate their own log files and write them to a dedicated subfolder in the server program path on the hard disk. You'll find more information on this in the manuals of the interfaces.

Automatically open necessary ports in Windows firewall
The installer of *ibaPDA* only adds the application exceptions for *ibaPDA* client, server and S7-Xplorer proxy to the Windows firewall. Exceptions for network protocols which are handled by the *ibaPDA* driver can be added automatically or manually. If you enable this option (default) the required port exceptions are added automatically. If you disable this option then you can manually add the port exceptions to the firewall by clicking the button <Allow ports through firewall> on each interface that requires a listen port.

Interfaces for which packages must be confirmed immediately...
For TCP-based interfaces that are to receive data faster than 200 ms, a so-called "fast acknowledge" must occur. For this purpose, the parameter "TcpAckFrequency" must be changed in the Windows registry. So that the user does not need to edit the registry, this option can be used to select the network adapter(s) for which a quick response is required. The changes in the registry are then made automatically.

<Open log file> button
Via this button, you open the log file of the *ibaPDA* server in the standard text editor of the (Windows) system. All system-related events and processes from the perspective of the server acquire logged this log.

If you create a support file in the support case using the *Save information for iba support...* function in the *Help* menu then this log file is included.

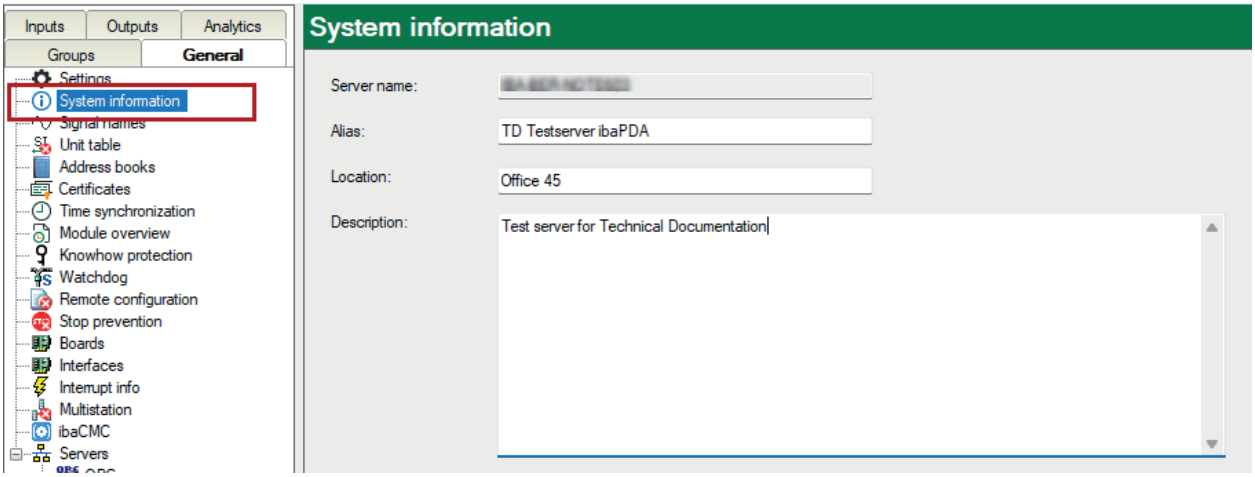
4.2 System information

The node *System information* offers some information about the *ibaPDA* server computer. The display of the server name derives from the Windows computer name.

You can add more information:

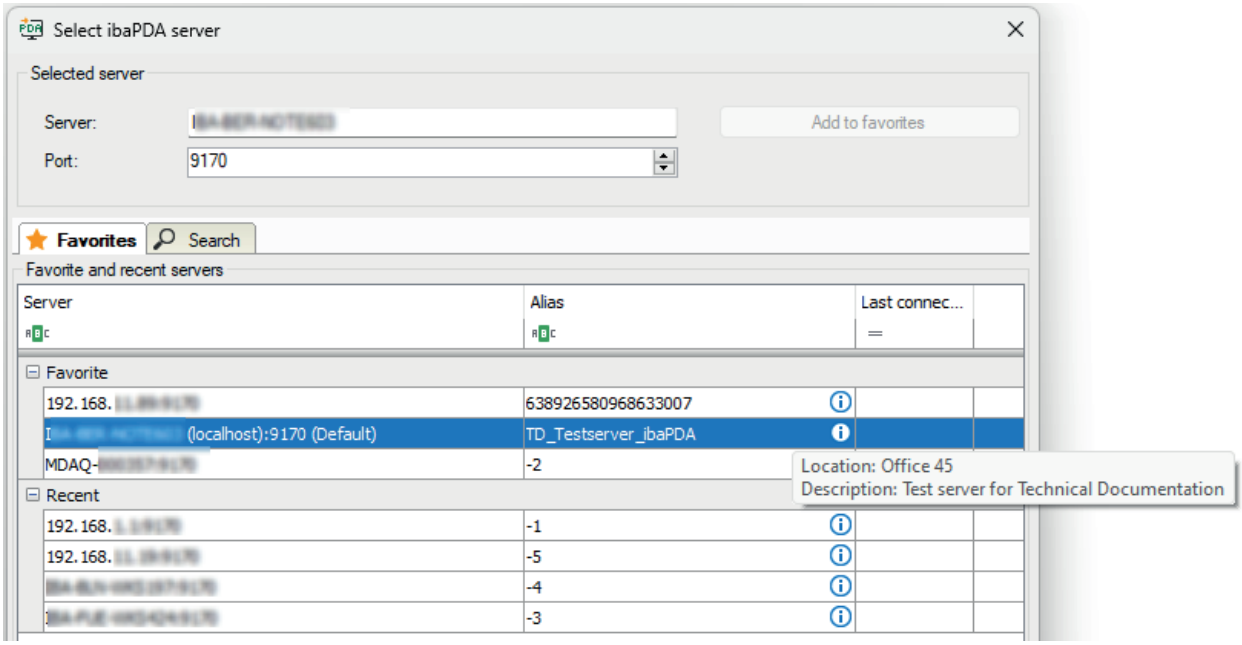
- Alias
- Location
- Description

The information is stored in the I/O configuration file.



Alias

If needed, you can assign an alias to the computer, which is, for example, more intelligible than the host name. This alias will also be available on client side in the *Select server* dialogs in the *Search* and *Favorites* tabs and hence, helps to identify the desired server.



The other information *Location* and *Description* is displayed in the tooltip.

Location

Enter a clear denomination for the location where the computer is installed, e.g. describing text or a standardized location designation (+ ...).

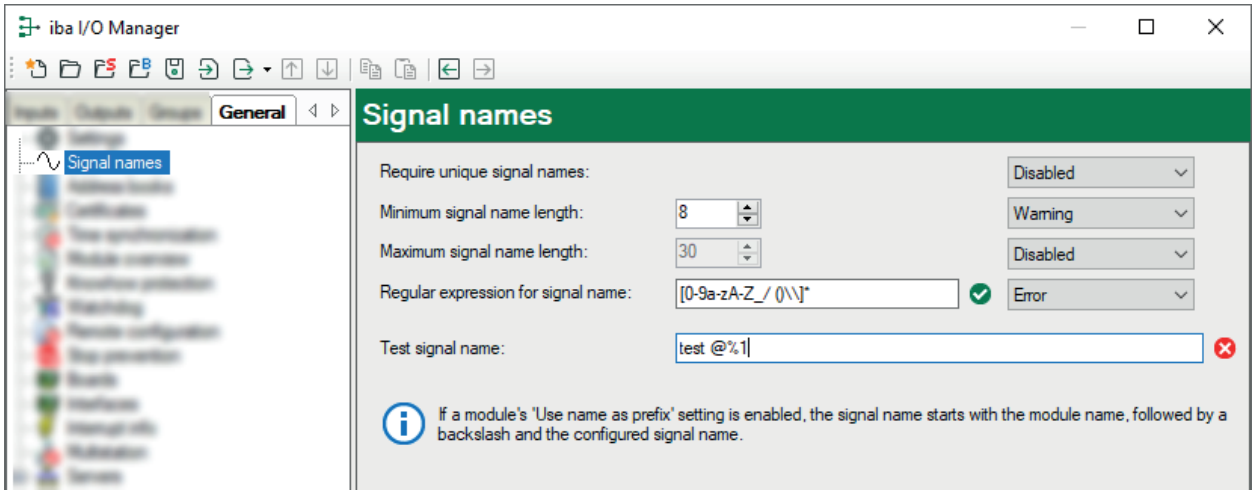
Description

Enter more describing information here, if needed.

Note

If you transfer an I/O configuration file to another computer in another location, make sure to update the system information in the configuration to avoid confusion.

4.3 Signal names



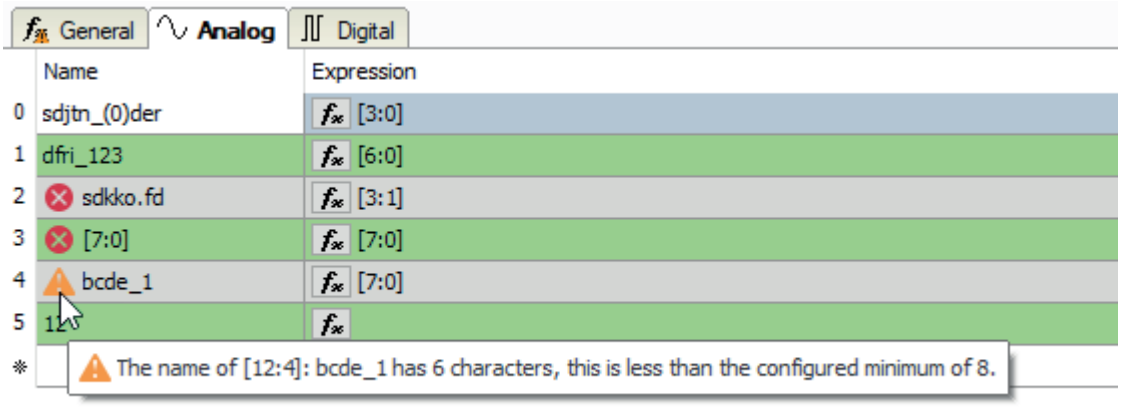
The creation of signal names can be checked against four rules, which can be enabled or disabled individually:

- The signal name must be unique through the entire I/O configuration
  A signal name must not be created twice in the I/O configuration.
- Guarantee a minimum signal name length
  The signal name must at least consist of the number of characters as specified.
- Limit the maximum signal name length
  The number of characters in the signal name must be equal or less than the specified number.
- The signal name must comply with a regular expression
  The signal name must only consist of characters, which are permitted by the specified regular expression. (Default: `[0-9a-zA-Z_/()\\]*`)

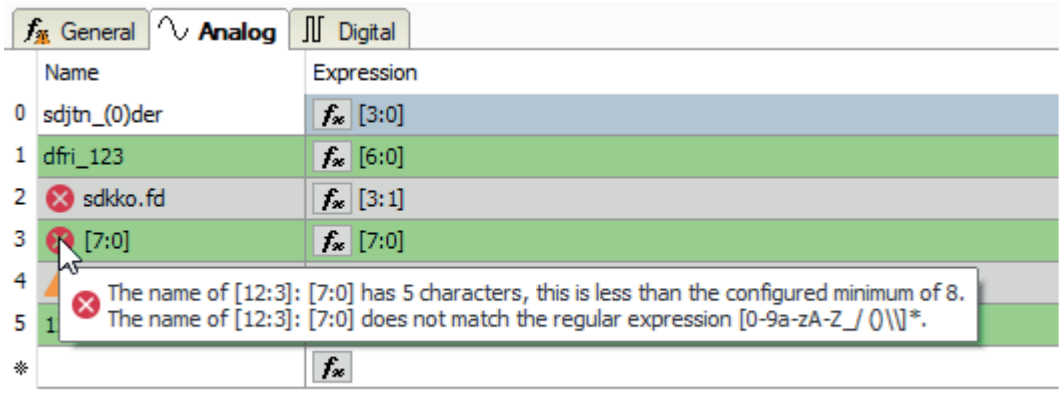
For each rule you can use the selection button on the right side to determine whether the rule is disabled or whether it should generate a warning or an error in case of violation.

You can enter any name into the field *Test signal name* in order to check if the rules work properly. As long as the test signal name does not follow the rules, a red circle with a white cross appears on the right end of the field.

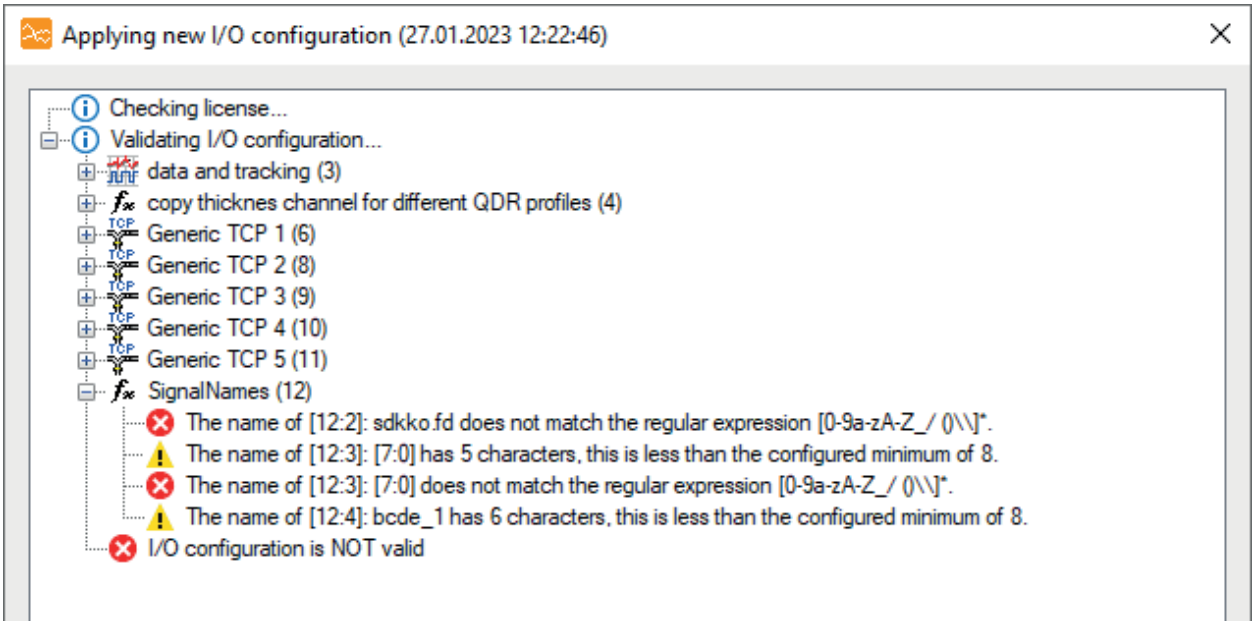
The compliance with the enabled rules is already checked while you work in the I/O Manager. As soon as a signal name does not comply with the rules a warning or error icon indicates this in the signal table. The following examples refer to the settings in the picture above.



In case more than one rule has been broken, all reasons are mentioned in the tooltip on mouse-over.



Furthermore, the compliance with the rules is checked during the validation of the I/O configuration, e. g. at start of acquisition. The corresponding warning or error messages will be displayed and in case of an error the I/O configuration will be invalid.



4.4 Unit table

The node Unit table offers the possibility to configure different unit systems in order to be able to switch the display of numerical values to a preferred unit system, if needed.

For example, a German manufacturer can configure the *ibaPDA* system in accordance with the metric SI unit system. Later, another user in the anglosaxon region can switch to the British or American unit system (Imperial units or U.S. Customary units) in the *ibaPDA* client and will find the values and units in his or her preferred system.

Note

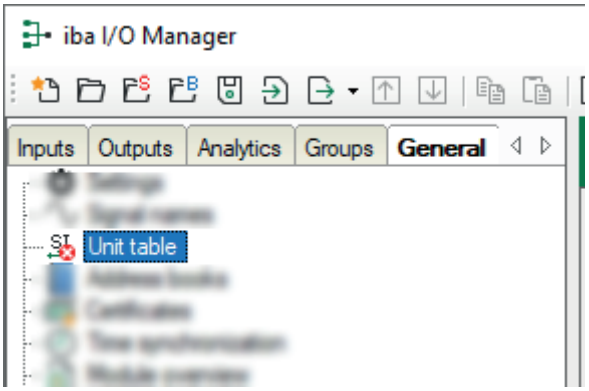
Numerical values in data files or in *ibaHD-Server* are always stored with the original unit according to the I/O configuration. Usually this is the primary unit system. The conversion into other units is exclusively done for visualization purposes.

Switching between unit systems has the following effects:

- The views of ibaPDA client will show the values of all *ibaPDA* and *ibaHD* signals converted to the new unit system.
- The signal units in the legend will be switched.
- All views in *ibaQPanel*, incl. the offline trendgraph and HD Time periods table, will show the values in the new unit system.

Configuring and enabling the unit table

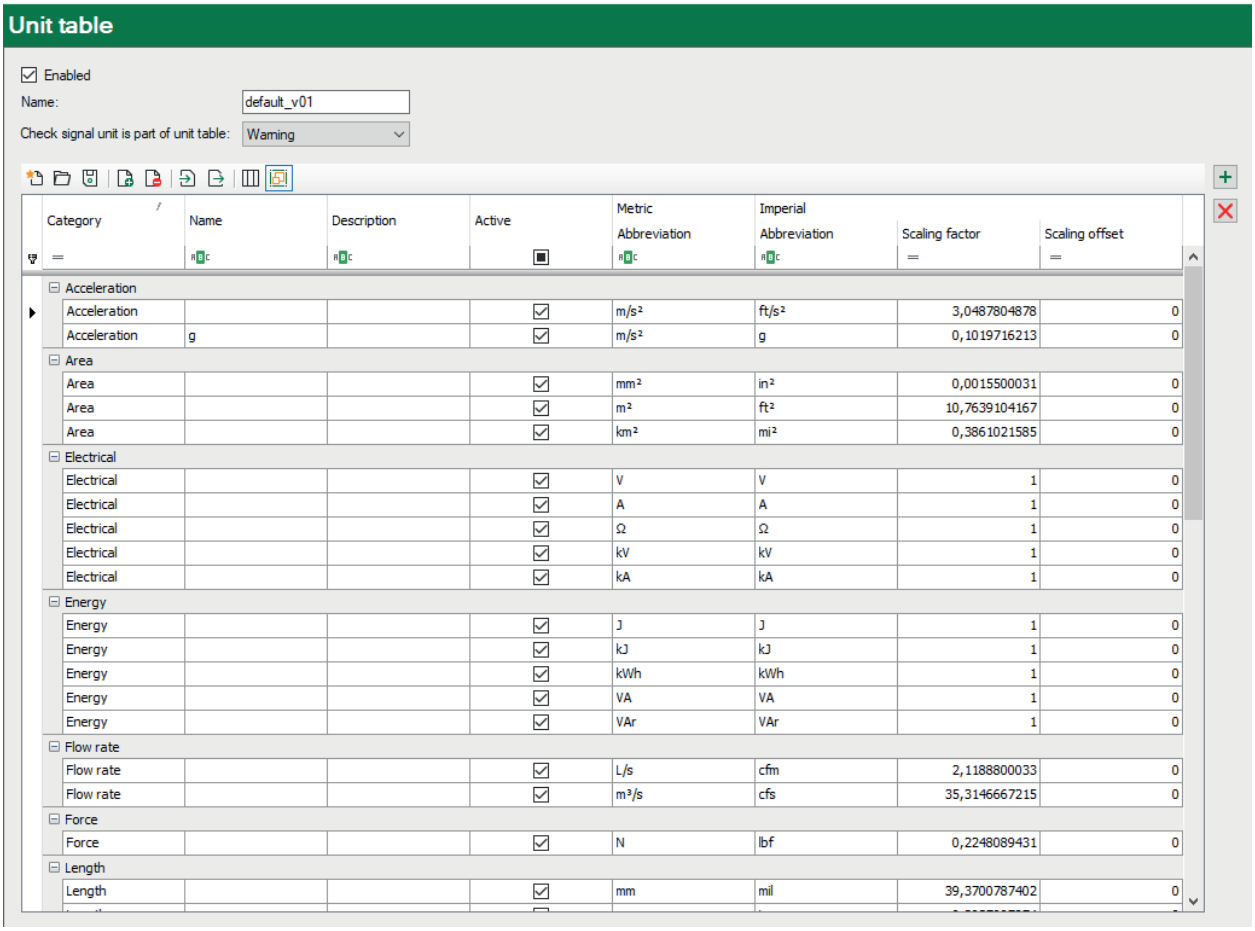
Select the node *Unit table* in the *General* tab of the I/O Manager.



If no unit table is available yet, you'll be asked whether you want to load the standard table. This is a table with the most common units in the metric and imperial systems. You can extend or change the table if needed.

If you reject loading the standard table, you'll get an empty table which you can fill by yourself.

Description of the unit table



Category	Name	Description	Active	Metric Abbreviation	Imperial Abbreviation	Scaling factor	Scaling offset
Acceleration							
Acceleration			☑	m/s²	ft/s²	3,0487804878	0
Acceleration	g		☑	m/s²	g	0,1019716213	0
Area							
Area			☑	mm²	in²	0,0015500031	0
Area			☑	m²	ft²	10,7639104167	0
Area			☑	km²	mi²	0,3861021585	0
Electrical							
Electrical			☑	V	V	1	0
Electrical			☑	A	A	1	0
Electrical			☑	Ω	Ω	1	0
Electrical			☑	kV	kV	1	0
Electrical			☑	kA	kA	1	0
Energy							
Energy			☑	J	J	1	0
Energy			☑	kJ	kJ	1	0
Energy			☑	kWh	kWh	1	0
Energy			☑	VA	VA	1	0
Energy			☑	VAr	VAr	1	0
Flow rate							
Flow rate			☑	L/s	cfm	2,1188800033	0
Flow rate			☑	m³/s	cfs	35,3146667215	0
Force							
Force			☑	N	lbf	0,2248089431	0
Length							
Length			☑	mm	mil	39,3700787402	0

The table consists of unit systems and units. One of the unit systems is the primary unit system (standard table: metric). For the other unit system(s) there are additional columns with scaling factor and scaling offset containing the values for the conversion. Optionally, you can open test columns to verify the conversion with sample values.

There is one row for each unit.

Each unit has a category, a name, a description, an abbreviation (unit symbol) and scaling information. For a clear view you can group the units by categories. See chapter ↗ *Units, page 27* for more information about units.

Above the table there are 3 settings.

Enabled
If you don’t enable the unit table, then you cannot switch the unit system in the views of *ibaPDA* client. Default setting: disabled.

Name
Enter a name for the unit table here. The name will be used when you export the table or save it as a file. The name will also be used when saving the table in the data file.

Check signal unit is part of unit table
Select what should happen if a used signal unit is not part of the unit table:

- Disabled: Signals and their units are not checked.
- Warning: A warning message will be issued in the signal table and at validation of the I/O configuration. However, the I/O configuration will be accepted and the acquisition starts.
- Error: An error message will be issued in the signal table and at validation of the I/O configuration. The I/O configuration is not valid and the acquisition does not start until the error is fixed.

Symbol buttons
Above and next to the table, there are some buttons with the following features:

	Create a new unit table
	Load a unit table from file (*.ut); replace current unit table
	Save unit table in file (*.ut)
	Add new unit system
	Delete unit system
	Import unit table (unit system); adds the imported unit system to the existing unit table.
	Export unit table (unit system)
	Show/hide test columns
	Group/ungroup units by category
	Add unit/row
	Delete selected unit(s)/row(s)

Create a new unit table
Click on the corresponding icon in the tool bar and choose whether you want to load the standard table or an empty table.

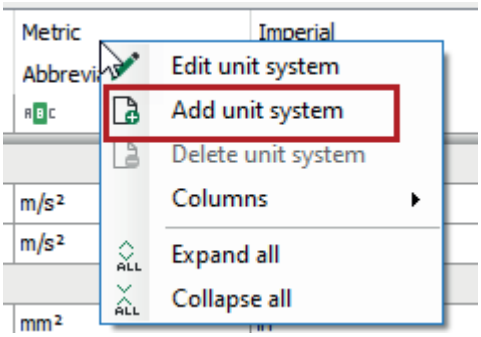
A table based on the metric unit system as primary system will be created automatically. Then add the destination unit system of your choice or enter the units first.

4.4.1 Unit systems

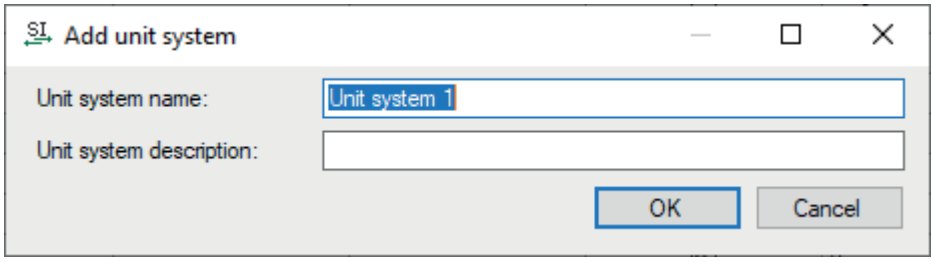
You can add, edit and delete unit systems.

Add unit system

Click on the corresponding icon in the toolbar or make a right mouse click on a unit system column header and choose *Add unit system*.



A dialog opens, where you can enter a name and a description. Enter at least a unique name. It is not allowed to have multiple unit systems with the same name in the unit table. If you enter a description, it will be shown as tooltip when hovering the column header of a unit system in the table with the mouse.



Edit unit system

Make a right mouse click on the unit system column header and choose *Edit unit system*. A dialog opens, where you can edit the name and the description.

Delete unit system

Make a right mouse click on the column header of the unit system you want to delete and choose *Delete unit system*. Alternatively, you can click on the corresponding icon in the toolbar and select the unit system(s) to be deleted in the dialog.

Import/export unit systems

Use the corresponding icons in the toolbar for these functions.

You can import one or more unit systems from a file (*.ut).

In the import dialog you can select the unit system(s) to be imported. The primary unit system is always selected because it is the reference for the conversion. Existing units will be overwritten with new values during import. New units will be added.

You also can export one or more unit systems and save them in a file (*.ut).

In the export dialog you can select the unit system(s) to be exported. The primary unit system is always selected.

4.4.2 Units

This chapter provides information about the configuration of units in the unit table.

Properties

Each unit (row) has the following properties (columns):

Category

The category can be used to group the units. Physical entities are used as categories, such as temperature, velocity, force. Click in the table cell to select the appropriate category. The list of categories is predefined.

Name

This is the name of the unit. This field can be left empty as long as no other unit has the same abbreviation. See also ↗ *Abbreviations, page 29*.

Description

Here, you can enter additional information or a description regarding the unit. This information is only visible in this table.

Active

This property determines whether the unit is taken into account or not. If you uncheck the checkbox the unit will be ignored.

Per unit system a unit has the following properties:

Abbreviations

This is the unit symbol which stands next to the signal value if the unit table is enabled. This field cannot be empty. See also ↗ *Abbreviations, page 29*.

Scaling factor (not available for primary unit system)

This is the factor that a signal value in the primary unit system should be multiplied with to get the correct value in the destination unit system.

Scaling offset (not available for primary unit system)

This is an additional offset which should be added to or substracted from the calculated value in the destination unit system.

See for example the temperatures in the standard unit table.

The scaling always refers to the primary unit system (metric in the standard table). Therefore, the columns *Scaling factor* and *Scaling offset* are not available for the primary unit system.

Testing the unit conversion

Click on the corresponding icon in the tool bar to show or hide additional test columns. You can enter a value in the test column of one unit system and *ibaPDA* will calculate immediately the values for all other unit systems (works in both directions).

Grouping units

Click on the corresponding icon in the tool bar to group the units by categories. The groups are built by category names in ascending alphabetical order.

Adding and deleting units

In order to add units to or delete units from the table use the symbol buttons on the right side of the table or the context menu inside the table. The delete command refers to the selected unit(s) in the table. Multiselection is possible by using <Ctrl> or <Shift> keys.

Showing and hiding columns

Open the context menu by a right mouse click on the header of the table. Select here which columns are to be displayed. *Reset* restores the default column selection and column arrangement. You can change the column arrangement in the table by drag & drop.

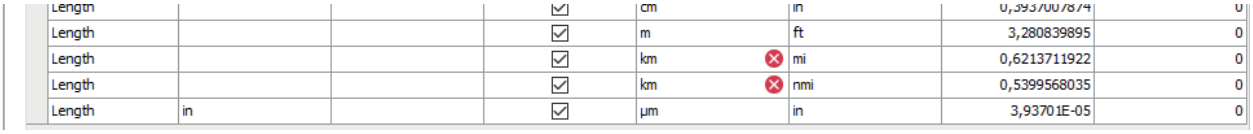
Furthermore, there is a filter row in the table header, which can be used for filtering the table by content.

You will find more detailed information about the filter functions in Part 7 of the *ibaPDA* manual, in chapter *Advanced filter in tables*.

4.4.3 Abbreviations

Basically, the abbreviation of a unit must be unique in order to identify the unit and convert the values correctly. Empty abbreviations are not allowed. An error symbol in the table indicates an invalid abbreviation. The tooltip provides more information when hovering the mouse about.

However, there are use cases where the same abbreviation is needed multiple times, e.g. when converting kilometers (km) in miles (mi) and also kilometers (km) in nautic miles (nmi).

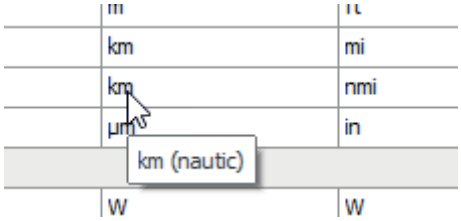


Length			☑	m	ft	3,280839895	0
Length			☑	km ⊗	mi	0,6213711922	0
Length			☑	km ⊗	nmi	0,5399568035	0
Length	in		☑	µm	in	3,93701E-05	0

In such cases just add a name to the unit. Later, this name will be shown together with the abbreviation, when you select the unit in the signal table.

Length			☑	m	ft	3,280839895	0
Length			☑	km	mi	0,6213711922	0
Length	nautic	nautical mile	☑	km	nmi	0,5399568035	0
Length	in		☑	µm	in	3,93701E-05	0

In this example, the name “nautic” is added to the abbreviation. A tooltip on the abbreviations indicates the display name of the unit.



For a better understanding you may enter additional information in the *Description* column.

Another use case of multiple usage of the same abbreviation is, that a unit has the same abbreviation in different unit systems.

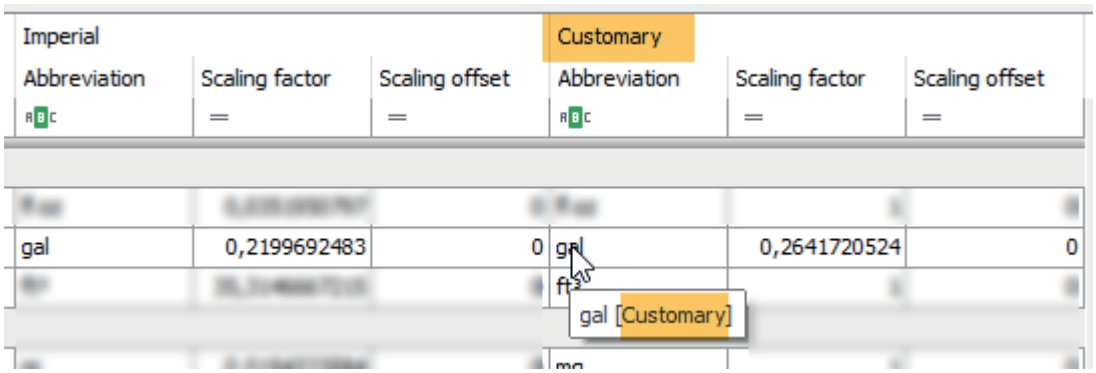
For example, you can use the abbreviation “s” for second in all unit systems.

If this unit is identical over all unit systems, i.e. the scaling factor is 1 and the scaling offset is 0, then the unit abbreviation is displayed whithout any supplement.

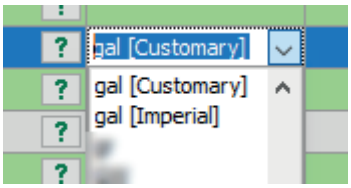
If the unit abbreviation is the same but the scaling factor is not 1, then the unit system is indicated after the abbreviation.

This is the case, for instance, when the unit systems Imperial Units and Customary Units are configured. In both unit systems there are identically named units such as gallon (gal) or fluid ounce (fl.oz.).

If you hover the mouse about the abbreviation, the tooltip will show the corresponding unit system.

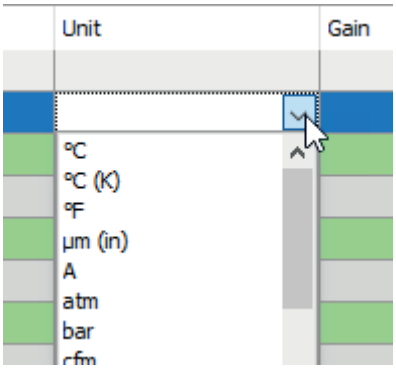


In the same way, the unit will be offered for selection in the signal tables.



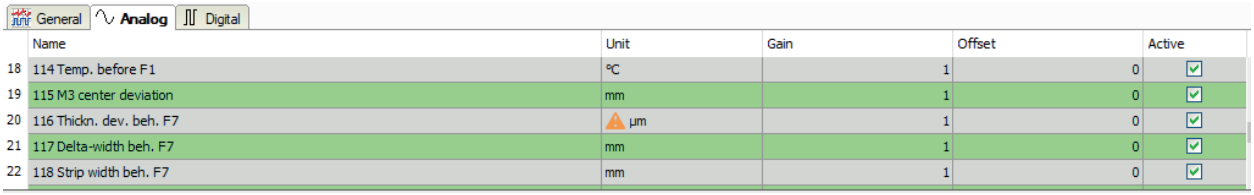
4.4.4 Using the unit system in the signal table

If you have enabled the unit table (see ↗ *Configuring and enabling the unit table, page 23*), the units are available for selection in the signal tables in the *Unit* column. You can either select a unit from the unit table or you can enter another unit into the field.



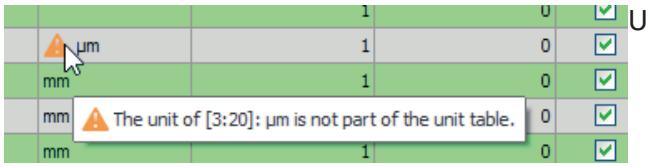
Units in the signal table, which are also available in the unit table are displayed without any markup.

Depending on the setting for checking the units, those units which are not part of the unit table will be marked with a warning or error symbol.

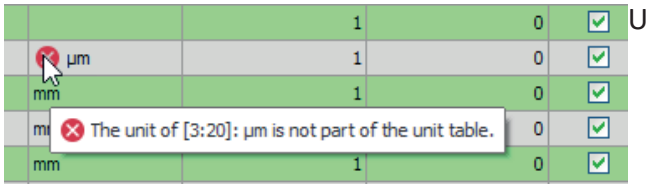


In this example, the unit µm is not part of the unit table.

The tooltip on the warning or error symbol provides more information.



Unit check with warning



Unit check with error

If the unit check is disabled there will be no indication.

You can use units even if they are not part of the unit table. However, those units will not be considered when changing the unit system and hence the values will not be converted.

Measured values in different units

When configuring the signals, you can mix units from different unit systems. Therefore, all units which are part of the unit table are available in the selection list and not only those units from the primary system.

If, for instance, two temperature gauges transmit their measured values like one in °C and the other in °F, you can enter the different units in the signal table. The conversion of the values into the unit of the active unit system for display is done automatically where needed.

Note

The signal tables of the I/O Manager, always show the signals with their configured units. This means that the actual values shown in the modules or diagnostic modules are always the received values but not the converted values.

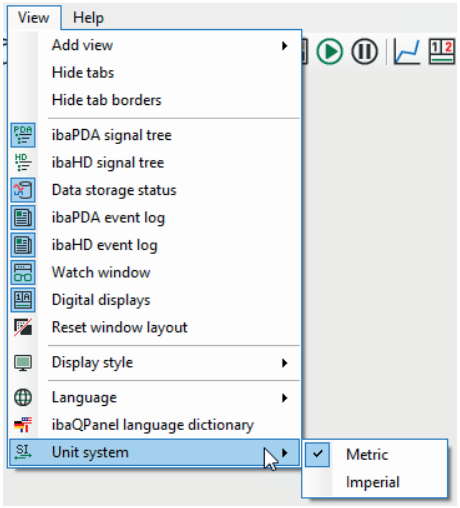
4.4.5 Changing the unit system

The metric unit system is selected, when the unit table is enabled for the first time. After changing the unit system, the new unit system will be stored as active unit system in the registry.

There are different methods to change the unit system.

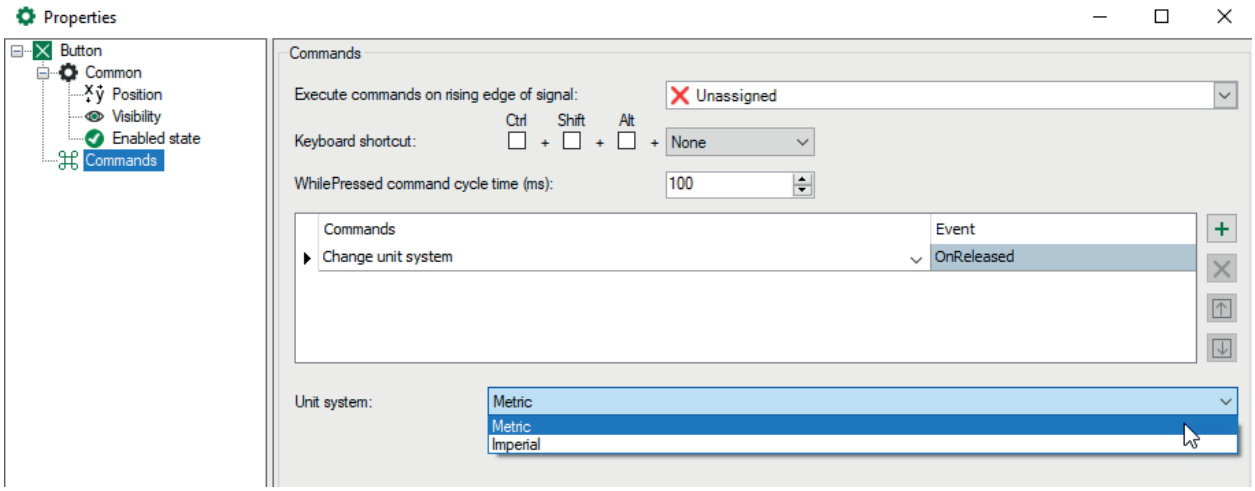
Switching in the View menu

► In *ibaPDA* client go to menu *View - Unit system* and select the desired system.



Switching by button in ibaQPanel

The button in *ibaQPanel* has a new command *Change unit system*.



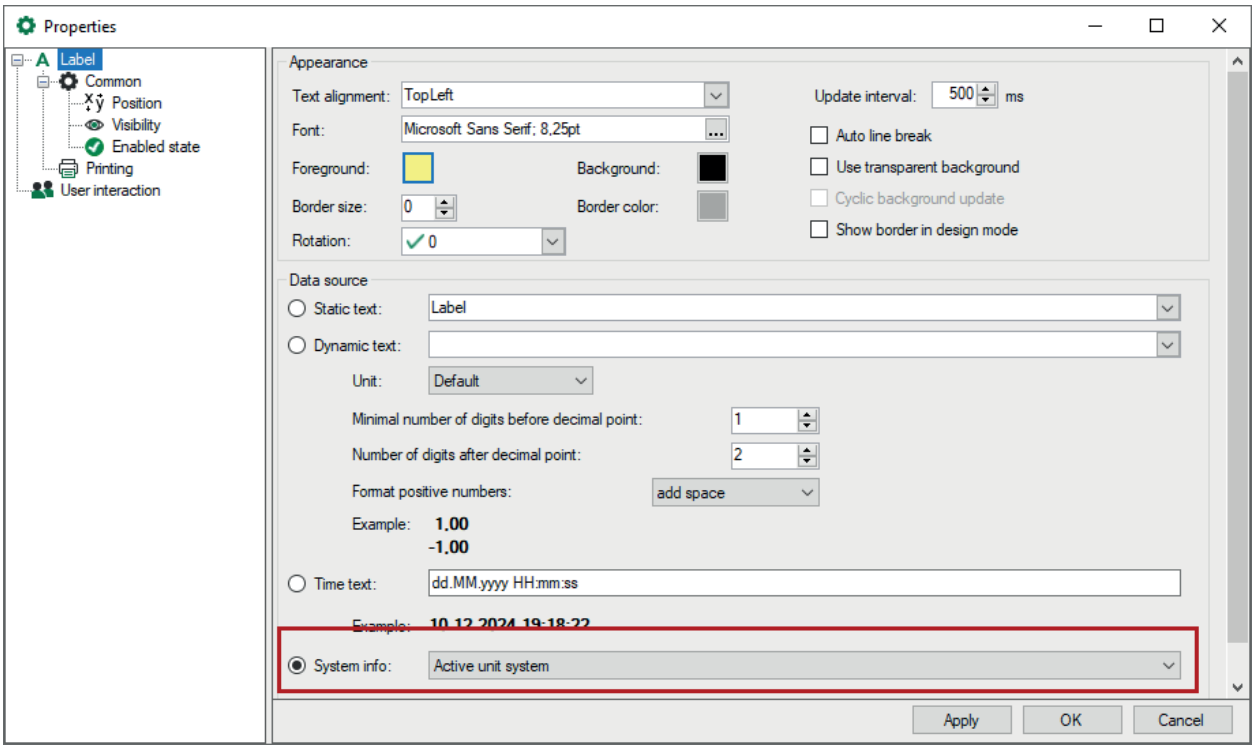
In the settings of the command you can define the unit system to be activated when clicking the button.

4.4.6 Indicating the active unit system

ibaPDA offers several features for users being informed about the currently active unit system.

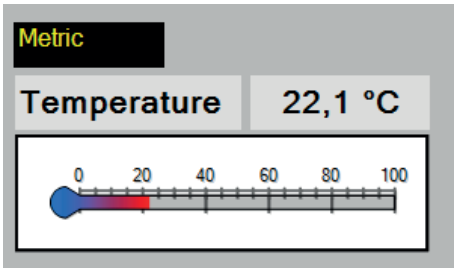
Indication in ibaQPanel

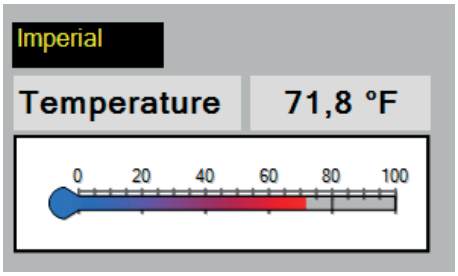
In the properties of the *Label* control you can select *Active unit system* for *System info* in the *Data source* area.



If you place this label at a highly visible position in the QPanel, then the actual unit system is always noticeable.

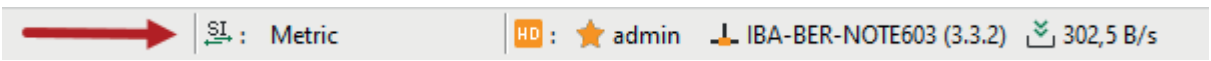
The following example shows such a label in the upper left corner.





Indication in the ibaPDA client status bar

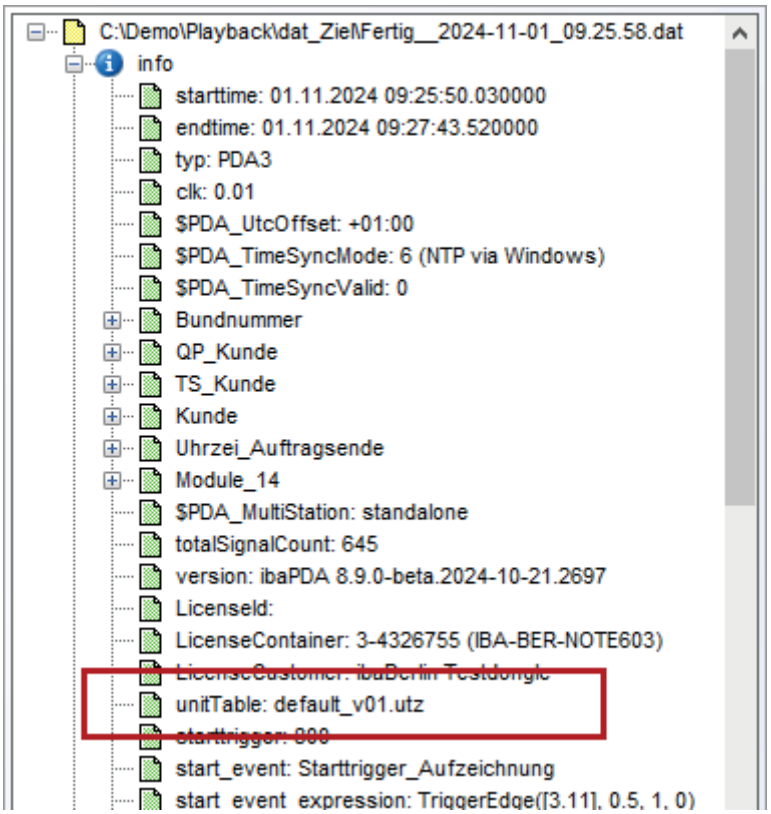
If the unit table is enabled, the actual unit system is indicated in the status bar of the *ibaPDA* client.



4.4.7 Unit table in the data file

If the unit table has been enabled, it will be stored automatically in the data file.

If a unit table is stored in the data file, then the info-field “unitTable” is available in the info-branch of the data file. The following picture shows the view in *ibaAnalyzer*.

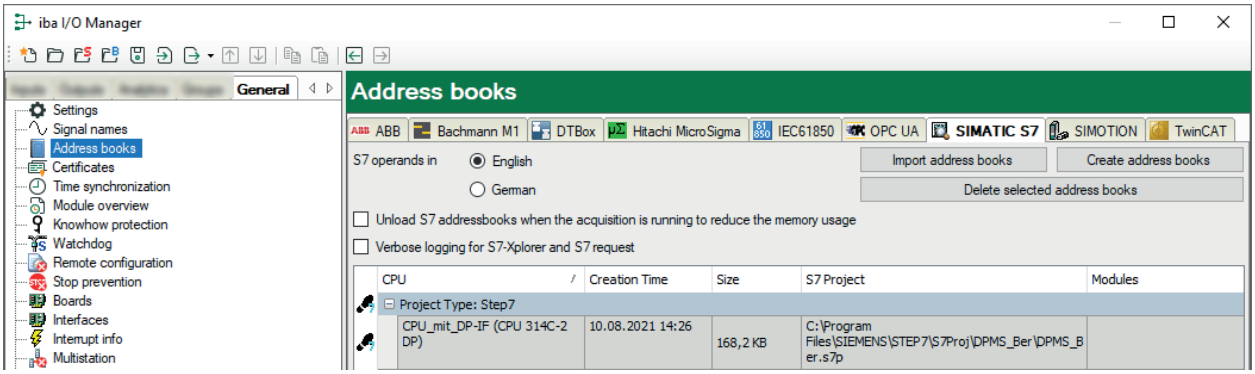


If no unit table is embedded in the data file, the info-field is not available.

Next to the name of the info-field you’ll find the name of the unit table as it was configured in the I/O Manager. The filename extension of the unit table is `*.utz`, because it is stored as a compressed file in the data file.

4.5 Address books

The *Address books* node is used to configure and manage address books, which are needed for address book-based connections to other systems, such as some Xplorer or SIMATIC interfaces.



The respective address book tabs are only visible if appropriate interfaces and/or licenses are active on the *ibaPDA* system.

The basic structure is the same for all tabs.

In the upper part there are some settings that depend on the connection type as well as the Button for importing, generating and deleting address books.

Below is a table listing all of the address books available on the system, with reference to the PLC, the creation date and – if available – the project. The last column lists the modules that have been configured for the specific CPU.

Address book tabs are available for the following systems:

Name	Tab visible ...
ABB	... with license *ibaPDA-Interface-ABB-Xplorer*
B&R	... with license *ibaPDA-Interface-C&Rodesys-Xplorer*
Bachmann M1	... with license *ibaPDA-Interface-Bachmann-Xplorer*, *ibaPDA-Request-Bachmann-M1*
Codesys	... with license *ibaPDA-Interface-Codesys-Xplorer*
DTBox	... with license *ibaPDA-Request-DTBox-xxxx*
Hitachi MicroSigma	... with license *ibaPDA-Interface-Hitachi-MicroSigma*
IEC61850	... with license *ibaPDA-Interface-IEC61850-Client*
Logix	... with license *ibaPDA-Interface-Logix-Xplorer*
OPC UA	... with license *ibaPDA-Interface-OPC-UA-Client*
SIMATIC S7	... with licenses *ibaPDA-Interface-S7-Xplorer*, *ibaPDA-Request-S7* or *ibaPDA-Request-S7-UDP* ... if an S7 module (all types) is included in the I/O configuration.
SIMOTION	... with license *ibaPDA-Interface-SIMOTION-Xplorer*
TwinCAT	... with license *ibaPDA-Interface-TwinCAT-Xplorer*, *ibaPDA-Request-TwinCAT*

Table 1: Systems with address book support

Simatic S7 example

S7 operands in English/German
Here you can choose the language in which the S7 operands will later be available when browsing through the signal tables.

<Create address books>
This button opens the "S7 address book generator" dialog. You can select the source directory of an S7 project for creating the S7 address book. This can be a local or network drive.

<Import address books>
Import address books which are already available as ZIP files.

<Delete selected address books>
Delete address books from the *ibaPDA* server's directory.

Unload S7 address book when the acquisition is running to reduce the memory usage
By enabling this option, the address book is outsourced to the hard disk during the acquisition in order to free up the main memory for the acquisition.

4.6 Certificates

The *Certificates* node gives you access to the central certificate store. This node is available in both the I/O manager and the data store index. From both locations, you can access the same pool of certificates and can manage the relevant certificates.

4.6.1 Certificates

For secure and encrypted TLS/SSL communication between a client and a server, so-called certificates are used because they enable secure authentication.

Certicates used by iba programs can be administered in a central certificate store.

Before a client can connect to a server, an application certificate must first be configured. Certificates can be provided from both the server and client side. Communication can only take place if each partner trusts the partner certificate.

Certificates can either be exchanged spontaneously when a connection is established or registered as trusted in advance. If a previously unknown certificate is offered when a connection is established, the user must manually accept or reject the certificate. Accepted certificates are automatically entered into the table in the *certificate store* and marked as trusted. If the certificate is rejected, then no communication will take place.

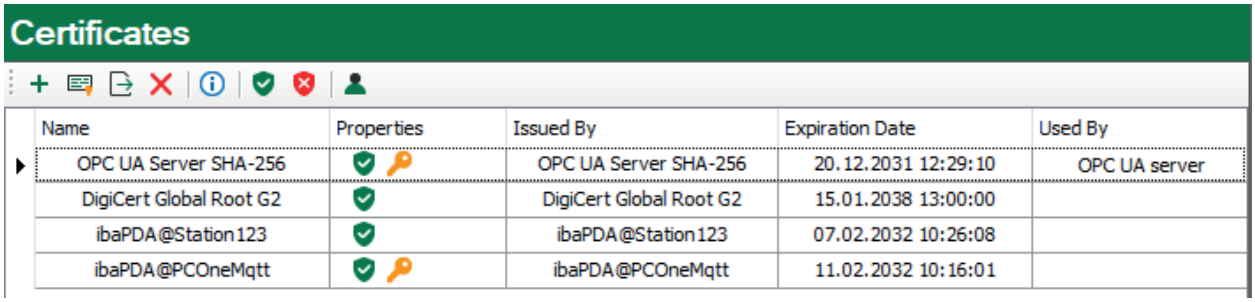
You can also register certificates and then mark them as "not trusted". Communication with partners with such certificates is then always denied. Once certificates have been registered, i.e., entered in the table in the certificate store, the user will no longer be notified or prompted when communication is established – regardless of whether the certificates are marked as "trusted" or "not trusted".

Note

Some interfaces in *ibaPDA*, such as the e-mail output, use Windows certificates. Other features, such as OPC UA server or MQTT data stores use certificates from the central certificate store of *ibaPDA*.

4.6.2 Central certificate store

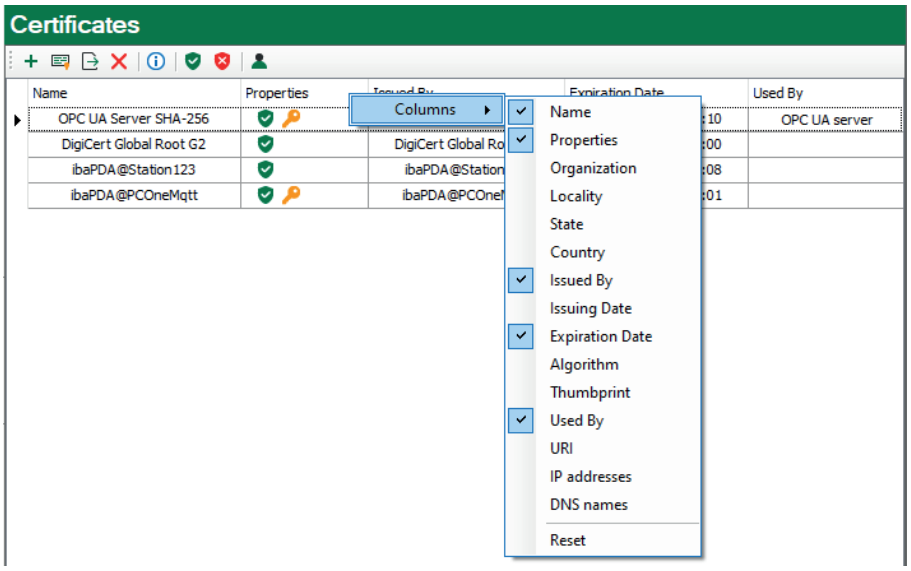
All registered certificates are listed in a table. The following figure shows the certificate store in the I/O Manager of *ibaPDA* as an example.



Each row refers to one certificate.

The columns *Name*, *Properties*, *Expiration Date* and *Used By* are displayed by default.

If needed, you may add or remove other columns via the context menu.



The *Name* column holds the name of a certificate. Different certificates may have the same name, thus it is not unique. Only the finger print of a certificate is unique.

The symbols in the *Properties* column have the following meaning:

Symbol	Meaning
	The certificate is trusted as long as it has not expired.
	This certificate is not trusted.
	A private key for this certificate is available.
	This certificate can also be used for user authentication.
	This certificate is invalid. If the certificate is invalid because it expired, the expiration date is highlighted in red color.

Table 2: Symbols for certificate properties

The *Used By* column shows by which application/function the certificate is used. In the example shown in the figure above, the certificate *OPC UA Server SHA-256* is used by the OPC UA server in *ibaPDA*. This means that this certificate was selected during configuration of the OPC UA server.

Note

One characteristic of *ibaPDA* is that certificates from different areas are managed in the certificate store: from the I/O Manager and from the data storage configuration.

For that matter the *Used By* column field has a link function. Via a double-click on a filled field you jump to the respective dialog.

If the entry belongs to the other manager, the link does not work. In the example above, a double-click on the “OPC UA-Server” entry would open the configuration dialog of the OPC UA server, provided it’s done in the I/O Manager. If you had opened the certificate store in the data storage configuration, the link to the OPC UA server does not work.

Vice versa, jumping to an “MQTT data store” would only work if the certificate store was opened in the data storage configuration.

The column *Expiration date* shows the date which marks the end of the validity of the certificate. Beyond this date the certificate cannot be used anymore. You have to renew the certificate or replace it by another, yet valid certificate. A red highlighted date indicates an expired certificate.

4.6.3 Manage certificates

The central certificate store is used to manage the certificates. Here you can add, create and delete certificates.

In the certificate store toolbar you will find a number of buttons with the following functions:

Button	Function
	This button opens a dialog that allows you to load an existing certificate file. Various file formats are supported (.der, .cer, .crt, .cert, .pem, .pfx, .p12). If you have a certificate with an unknown file extension, expand the file filter to "*.*" and try to open the file anyway. This works in most cases.
	This button opens a dialog that lets you create a new certificate file.
	This button lets you export a certificate to a file to register it for Windows or another application, e.g. on an OPC UA client. Multiple file formats are supported here as well.
	Use this button to remove the selected certificate from the table.
	Use this button to open a window with all detailed information about the selected certificate.
	Use this button to designate the selected certificate as "trusted".
	Use this button to designate the selected certificate as "not trusted". However, the certificate will still remain in the certificate store table. However, certificates that are not trusted are not available in the selection list for use in the corresponding configuration dialog.
	With this button you can define whether a certificate can also be used for user authentication for OPC UA.

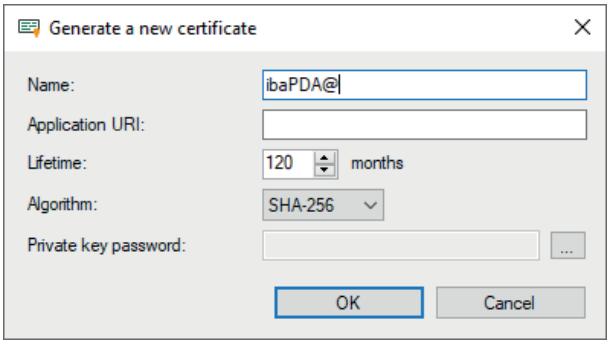
Table 3: Buttons in the toolbar for certificate management

The commands always refer to the certificate selected in the table, which is indicated by an arrow on the left at the start of the line.

4.6.3.1 Generate a new certificate

If no certificates are available to load, it is necessary to generate one.

1. Click the button and the following dialog box will open:



2. Enter a name of your choice for the certificate.
3. If required, enter an Application URI.
   The URI (Uniform Resource Identifier) is a global unique identifier for the application. If you do not fill in this field, a standard URI will be generated, provided that the OPC UA client verifies an Application URI. This standard URI consists of the machine name and the name of the application:
   `urn:machinename:applicationName.`
4. Define the desired validity period (lifetime) of the certificate.
5. Select the desired hash algorithm for the encryption.
   You have the choice between the algorithms SHA-256, SHA-384 and SHA-512.
   Make sure that the other communication partners support the selected algorithm too.
6. Define a password for the private key. If no password has been entered, the <OK> button remains inactive. To assign the password, click the <...> button and enter the password twice and confirm with <OK>. There are no special requirements for the password. Keep the password in a safe place so that the self-generated certificate can be exported and used for Windows or other applications.
7. Close the dialog with <OK>.

The new certificate is now entered into the list held by the certificate store and immediately assigned the properties "trusted" + private key.

You can now also export the certificate and register it with the communication partner, e.g., an OPC UA client. Afterwards, the client can then connect to *ibaPDA* (OPC UA-Server).

4.6.3.2 Add certificate

1. In the certificate store toolbar, click the button + .
   A dialog will open that lets you navigate to the desired certificate file and open it.
   Different file formats are supported (.der, .cer, .crt, .cert, .pem, .pfx, .p12).
   If you have a certificate with an unknown file extension, expand the file filter to "*.*" and try to open the file anyway. This works in most cases.
2. When the certificate is loaded, it appears in the certificate store list.
3. If you have not already done so, trust the certificate.

Certificates can sometimes be added without manual import.

Thus, during the first connection attempt by an OPC UA client to the OPC UA server (*ibaPDA*), the application certificate of the OPC UA client is automatically added to the certificate list and initially rejected.

Once you have selected the OPC UA client certificate in the list and confirmed it as trusted with the button the OPC UA client can subsequently connect automatically.

Use the button to reject a certificate at any time or to classify it as not trusted.

4.6.3.3 Export certificates

All certificates in the certificate store can be exported individually as a file and subsequently used for Windows or other applications. An exported certificate can also be re-imported into.

To export a certificate, first select the desired certificate in the table and then click the button in the toolbar for the certificate store.

If you wish to export a certificate without a private key, a dialog that lets you save the file opens immediately.

If the certificate to be exported has a private key, there are some options.

First, you will be asked if the existing private key should be exported as well. If you answer "no", the certificate will be saved immediately, just like a certificate without a key.

If you answer "yes", then you must enter the correct password afterwards. The correct password is the password used when importing or generating the certificate. If the password is correct, the certificate can be saved as a PFX-file. This file is password protected and contains the certificate and the private key.

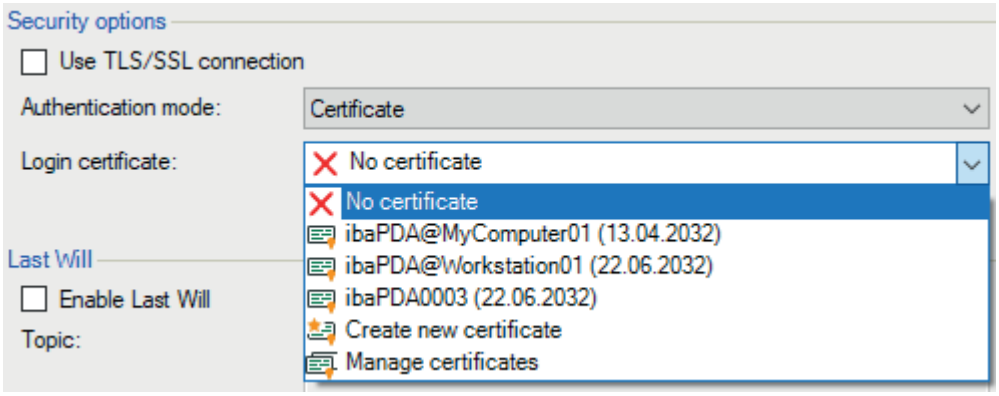
If the password is incorrect, the certificate will not be exported.

Under certain circumstances, a certificate with a private key may be stored, however the key is not password protected. In this case, the certificate can only be exported without a private key. You will then be notified accordingly.

4.6.4 Use certificates

At the points where certificates are applied, you will find a drop-down list offering the available certificates for selection.

There are the following options:



- No certificate: No certificate is used. As a rule, this leads to an invalid configuration.
- Available certificates: All certificates are displayed that are contained in the central certificate store, are valid, and are suitable for use at this point.
- Create a new certificate: The dialog for creation of a certificate opens. If the operation is successful, the new certificate is also selected immediately. If not, "No certificate" is selected.
- Manage certificates: Calling up the central certificate store.

Note

The selected certificate is saved in the registry file of the computer. In case of a new configuration, the same certificate will be selected unless another certificate is actively selected.

Other documentation

For more information on the use and functions of the certificates, please refer to the descriptions and manuals of the relevant applications, such as *ibaPDA*, div. *ibaPDA* interfaces, *ibaHD-Server*, *ibaDatCoodinator* etc.

4.6.5 Save and protect certificates

The certificates are stored in the `settings.xml` file, which is located in the folder `c:\ProgramData\iba\Name of application\Certificates`. This file is automatically encrypted.

There are a number of measures whereby certificates with private keys can be used to protect your identity or that of your organization. Specifically, these are measures that make their simple export and reuse in Windows or other applications more difficult.

- Certificates are always stored in encrypted form.
- For certificates with a private key, the input of a password is required...
  - when a new certificate is generated
  - when a certificate with a private key is exported
  - when a certificate with a private key is imported
- Certificates with a private key can only be exported if there is also a password for the key. If there is no password or the password is unknown, the certificate can no longer be exported. Therefore, keep the passwords in a safe place.
- The password for a private key cannot be changed.
- It is not necessary to enter a password to use a certificate. The `settings.xml` file can be copied from one installation to another to transfer the certificates there. Password entry is not required for this either.

Should the private key fall into the wrong hands, many types of misuse are possible. Therefore, make sure that the passwords are kept safe.

4.6.6 Check information about a certificate

If you want to get more detailed information about a certificate then proceed as follows:

1. Go to the certificate overview in the certificate store.
2. Select the certificate in question by clicking on the corresponding row.
3. In the certificate store toolbar above the table, click on the info-button (i).

→ A window opens showing all relevant information, which are available for this certificate.

4.7 Time synchronization

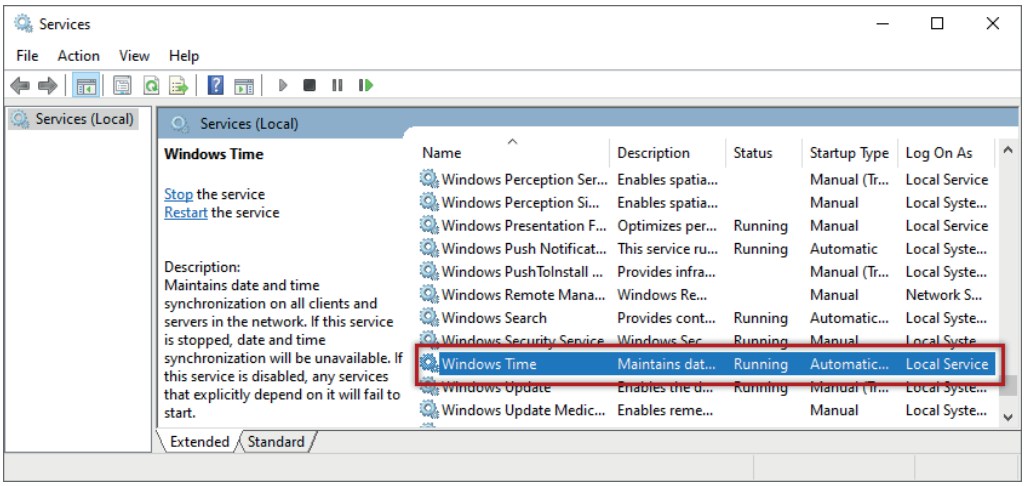
ibaPDA offers several methods of time synchronization. The purpose of time synchronization is to set the time of the *ibaPDA* system in accordance with a central time server. In this case, *ibaPDA* is a "*time slave*". In the case of PTP time synchronization, *ibaPDA* can itself be a time server for other systems and be configured as a "*time master*".

Note

You should use the “Use system time” option for data storage when time synchronization is active. Otherwise, the start time of the data file will not be accurate.

Note

To avoid conflicts between the Windows time management and the *ibaPDA* time synchronization, you should disable the local service *Windows time source* on the computer as soon as you enable one of the *ibaPDA* time synchronization modes.



There are the following ways to synchronize time:

- DCF77
- IEC 1131
- PTPv2
- *ibaClock*
- NTP via Windows

Depending on the use of special interfaces other methods for time synchronization may be offered.

4.7.1 DCF77

With this method, you can configure an external time synchronization of the system by means of the DCF77 signal. The system time of the *ibaPDA* computer is set. Other time sources may also work provided they generate a signal in compliance with the DCF77 convention.

What is DCF77?

The acronym DCF77 is the label for a coded time signal which is transmitted on the standard frequency of 77.5 kHz on long wave. It can be received within a range of 2,000 km (1,243 miles) around Frankfurt/Main, Germany.

DCF77 derives from the international frequency list where D stands for Deutschland (Germany), C for long wave, F for "near Frankfurt" and 77 for the carrier frequency.

The time transmitted is the official time of the Federal Republic of Germany and thus the Central European time or the Central European daylight saving time.

The time base currently consists of three cesium atomic clocks, operated by the National Metrology Institute of Germany (PTB).

There are many devices on the market for receiving and converting the signal.

DCF77 signal and time information

The time information is transmitted as a digital signal on the normal carrier frequency of 77.5 kHz. The digital signal is formed by a negative modulation of the signal (decreasing the amplitude of the carrier frequency down to 25 %) in a one-second cycle.

The modulation starts at the beginning of each second from second 0 to 58 of one minute. In the 59th second, no lowering takes place and the following second mark indicates the beginning of the next minute. The length of the amplitude lowering at the beginning of a second provides the binary value of the digital signal. A 100 ms lowering stands for the value "0" (FALSE); 200 ms for "1" (TRUE).

Thus, with every second, part of the time information is transmitted so that it takes at least one minute for all of the time information to be transmitted.

The time and date are encoded by 58 digits, i.e., 58 seconds.

In addition to the pure time and date information, more information is included, such as operating information (seconds 1 to 14) as well as the transmitter type used, time zone, daylight saving time and switching seconds (second marks 15 to 19).

Connection to ibaPDA

The integrated DCF77 driver decodes the time information from a digital input signal that must be generated in compliance with the DCF77 standard.

If the *ibaPDA* system is installed in the range of the original DCF77 signal, there are several devices on the market that convert the DCF77 radio signal into a discrete digital signal. ASCII strings containing the time information cannot be processed by *ibaPDA*.

In order to be able to transmit the DCF77 time to *ibaPDA*, the discrete pulse outlet of the DCF77 receiver should be connected to a digital input on an *ibaPADU* or ibaNet device.

In the configuration dialog of the I/O Manager of *ibaPDA*, you then just have to select this input as the DCF77 signal.

Devices which have already been used with *ibaPDA*:

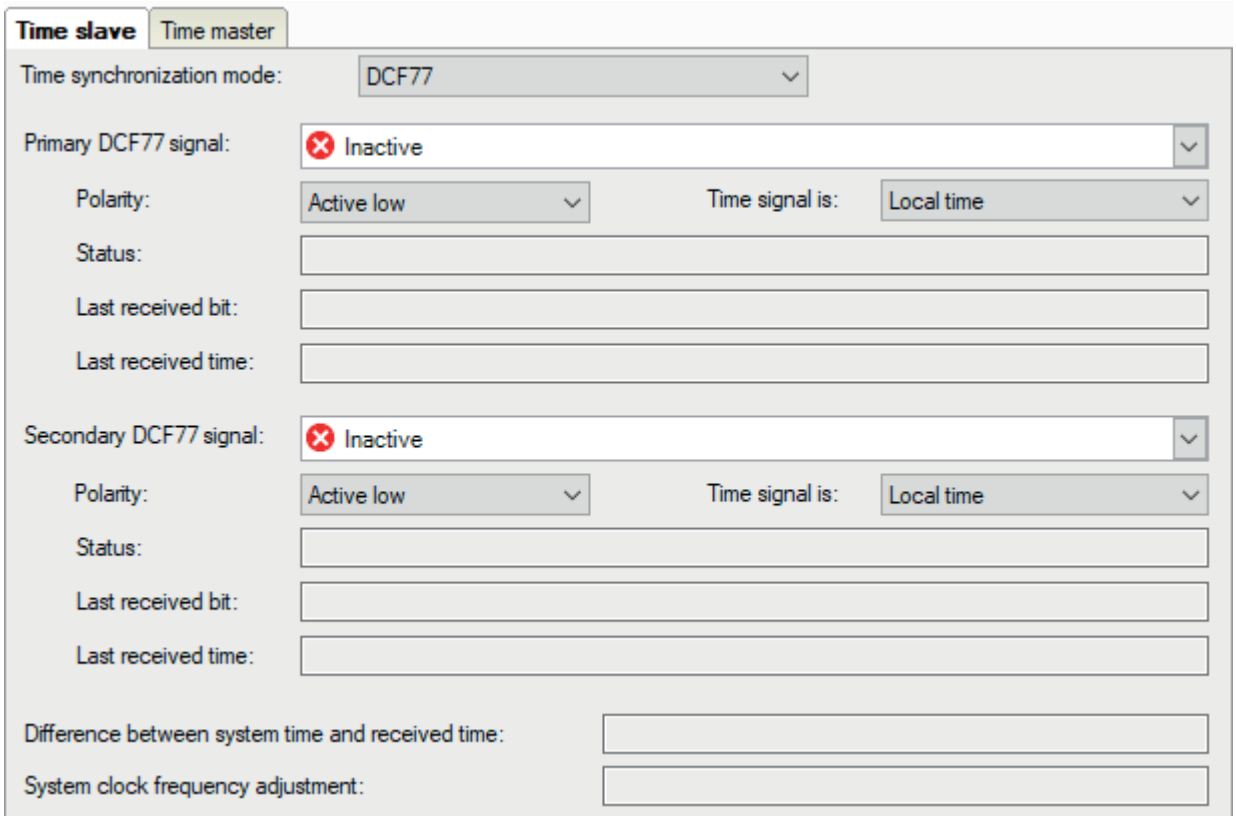
- Meinberg COM52HS DCF77 receiver for DIN-rail mounting
- Meinberg RU226: DCF77 receiver in compact housing

Since using a simple digital input, it is possible to synchronize the *ibaPDA* system time with external time sources, such as clocks.

ibaPDA can thus be synchronized even in regions where the DCF77-signal is not available.

4.7.1.1 DCF77 Configuration

The configuration dialog offers the possibility to configure two DCF77 signals. In this way, a redundant time management can be realized.



Primary DCF77 signal
In this field, select the digital input signal which is used for transmitting the decoded DCF77 signal. Click on the little arrow button in the field. A signal tree showing all available modules and signals opens. Select the appropriate signal by mouse click.

Polarity
In this field, select the correct polarity of the pulse. Depending on the device type you are using, the control pulse may be logical "1" (TRUE) or logical "0" (FALSE). Please refer to the manual of the DCF77 receiver and set the correct polarity.

Time signal is
Open this selection field in order to set the time mode of the DCF77 signal. Available for selection are:

- Local time
- UTC time (Universal Time Coordinated)
- UTC time with DST compensation

Comment: DST stands for daylight saving time. In this mode, *ibaPDA* will convert the time from UTC to local time just like in the normal UTC time mode. If the daylight saving time bit is set in the telegram, *ibaPDA* will subtract 1 hour to compensate the daylight saving time. In all other modes, the daylight saving time bit is ignored.

Status
Clear text messages regarding the DCF77 connection are displayed here.

Status	Meaning
Inactive	DCF77 synchronization is disabled.
Invalid	An invalid sequence of bits was received (parity error).
Valid	*ibaPDA* is synchronized with DCF77, a sequence was received.
Sequence received	A complete sequence (bits 0 to 59) was received.
Signal received	A minute pulse was received.
Signal lost	No minute pulse was received.

Table 4: Status of a DCF77 connection

Last received bit
This line displays the bit no. and the value of the last received bit. Bit no. 0 ... 59, value: 0 (FALSE) or 1 (TRUE). With the DCF77 decoding scheme, it is possible to get the meaning of the bit. For service purposes only.

Last received time
This line displays the last received time as clear text in the format dd/mm/yy hh:mm:ss.ms. This shows the time of the last synchronization of the system.

Difference between system time and received time
The received time is cyclically compared with the system time whereas deviation is determined. Depending on whether the difference is rather positive or rather negative, a corresponding frequency adjustment of the internal system clock is calculated.

System clock frequency adjustment
The value of the frequency adjustment has been determined from the difference between the system time and the received time. Thus, a system clock which is too fast or too slow is corrected.

4.7.2 IEC 1131

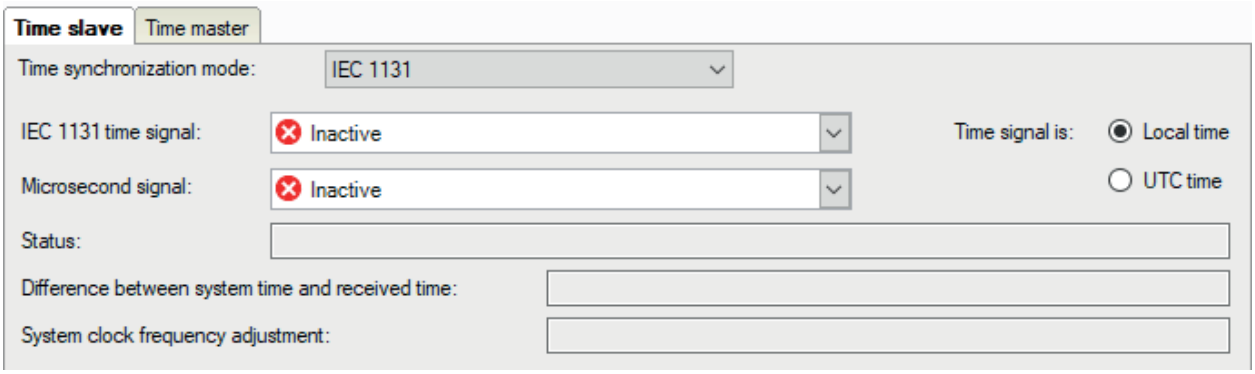
What is IEC 1131 time?

In compliance with the standard IEC 1131 (EN 61131), time can be transmitted by a DWORD value. This time represents the number of seconds that have passed since 1st January 1970, 00:00.

A second DWORD with a microsecond counter that goes from 0 to 999999 is required by *ibaPDA* for higher accuracy. The counter is reset at the start of every second.

The time source can be any system, time server or PLC capable of generating such a time value.

IEC 1131 time settings



IEC 1131 time signal
Select the analog signal carrying the time value from the drop-down list.

Microsecond signal
Select the analog signal carrying the microsecond value from the drop-down list.

Time signal is local time/UTC time
Check the appropriate option according to the time format generated by the time source. If "Local time" was selected, *ibaPDA* will accept that time without any conversions. If "UTC" is selected, *ibaPDA* will convert the time to the local time according to the time zone settings of the computer.

Status
The status field shows the current status of the time synchronization. The time synchronization only works when the acquisition is running.

Difference between system time and received time
The received time is cyclically compared with the system time whereas deviation is determined. Depending on whether the difference is rather positive or rather negative, a corresponding frequency adjustment of the internal system clock is calculated.

System clock frequency adjustment
The value of the frequency adjustment has been determined from the difference between the system time and the received time. Thus, a system clock which is too fast or too slow is corrected.

4.7.3 PTPv2

What does PTP mean?

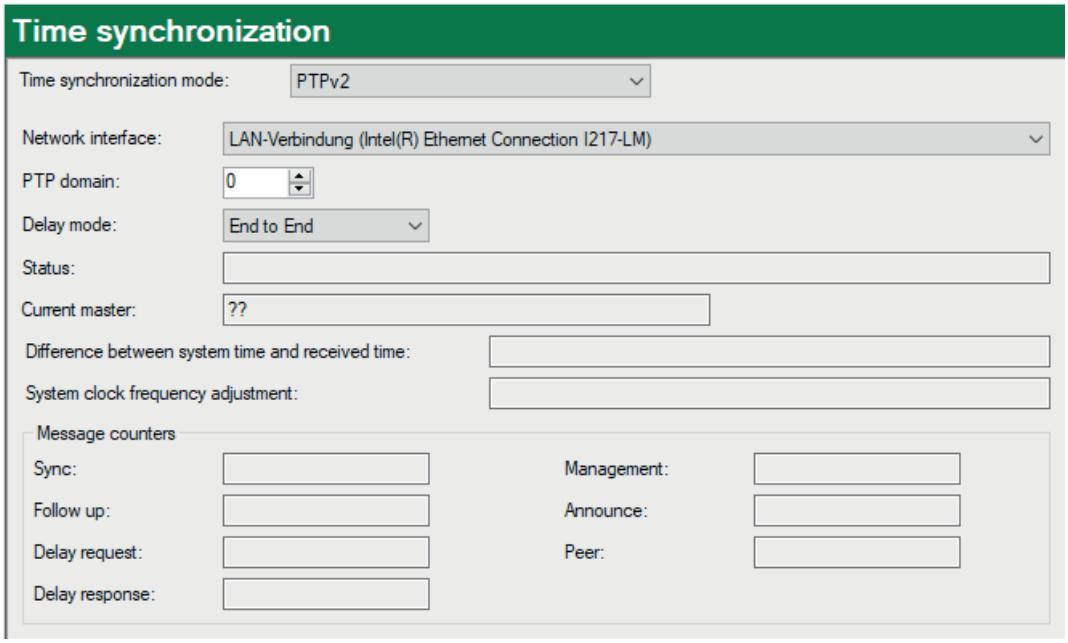
The Precision Time Protocol (PTP) is a communications protocol for synchronizing the system time of different systems (PC, PLC and other devices) in a network. The protocol is defined in accordance with Standard IEEE 1588.

In a typical PTP topology, there is one time master and one or more time slaves that synchronize to the master. The highest synchronization accuracy of software-based PTP applications, as in *ibaPDA*, is 5 ms.

ibaPDA can be configured as slave. When an *ibaPDA* system is configured as a time slave, it synchronizes to a time master in the network, provided one is available.

ibaPDA supports the standard PTPv2 according to IEEE 1588-2008 (or "IEEE 1588-V2"). With the new process, only the multicast address 224.0.1.129 is used in all domains to distribute the time in the network. The domain number can have the value 0 to 255 and is part of the time telegram.

PTP settings, time slave



Network interface
Here, choose the network interface of the *ibaPDA* computer to be used for time synchronization.

PTP domain
If *ibaPDA* is configured as a slave, set the same PTP domain on which the time master is also sending.

If *ibaPDA* is configured as a master, you can set any PTP domain between 0 and 255.

Delay mode
The accuracy of the PTP protocol is based on the continuous computation of delays which can occur at any time when transmitting information in a network and taking these delays into account for the time synchronization of the time slaves. You can choose between two methods which are available for this purpose.

- End-to-End: This method determines the delay between time master and time slave over the entire way through the network. Therefore, the slave sends a "Delay request" message to the master, which in return responds with a "Delay response" message. This is used to calculate the transit time of the message through the network. The method is suitable for any network even if network components such as switches, hubs or routers are not necessarily PTP-compliant. However, the achievable accuracy of the end-to-end method in a network is lower than the accuracy of the peer-to-peer method in a network of the same topology and

with "PTP-transparent clocks" Moreover, the end-to-end method generates a higher network load, particularly if multiple slaves are in the network.

- *Peer-to-Peer*: This method calculates the delay between the ports of adjacent devices by means of so called "Peer-delay-request/-response" messages. All network devices along the way between time master and time slave determine the delay on the section towards the devices they are directly connected with and pass on this information to the next device. The total delay between master and slave is calculated by so called PTP-transparent clocks (switch, hub or router) by writing the received delay value - supplemented with the delay which may be generated by the device itself - into the correction field of a PTP event message (e.g. sync telegram). The peer-to-peer method applies only to networks which are fully PTP-compliant. It generates less network load and it is less sensitive regarding asymmetries or changes in the network compared to the end-to-end method.

Status
The status field shows the current status of the time synchronization. The time synchronization only works when the acquisition is running.

Current master
If *ibaPDA* is configured as a slave, the IP address of the time master is shown here for information purposes.

Difference between system time and received time
The received time is cyclically compared with the system time whereas deviation is determined. Depending on whether the difference is rather positive or rather negative, a corresponding frequency adjustment of the internal system clock is calculated.

System clock frequency adjustment
The value of the frequency adjustment has been determined from the difference between the system time and the received time. Thus, a system clock which is too fast or too slow is corrected.

Message counters
This overview shows different message counters. There is one counter per telegram type. If there is a master-slave connection, the counters usually count up: *sync*, *announce*, *delay request* and *delay response*.

4.7.4 ibaClock

What is ibaClock?

ibaClock is a compact device for the time synchronization of several connected *ibaPDA* systems.

ibaClock sets the time base for all connected *ibaPDA* systems as the central master clock and ensures an exact and timely synchronized acquisition of the data. An accuracy of better than 150 ns can be achieved.

Additionally, *ibaClock* can synchronize more members by using other time protocols:

- PTP
- NTP
- DCF77

The highly accurate time is received via an active GPS antenna.

Alternatively, IRIG-B or PTP can be used as external time source.

In order to be able to use *ibaClock*, an *ibaFOB-… D* card must be inserted into the computer.

You also need to first configure the device in the I/O Manager under the *ibaFOB-… D* interface.

ibaClock settings

Other documentation

A detailed description of how to set up the *ibaClock* can be found in the device manual.

Information is displayed in the *Time slave* tab of the *time synchronization* node about the device, the clock and the system time synchronization. You cannot adjust any settings here. The information is identical to that in the *Diagnostics* tab of the *ibaClock* module under the *ibaFOB-… D* card.

<Write firmware> button
With this button, it is possible to perform firmware updates. Select the update file, `clock_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.

Only perform a firmware update after consulting the iba support team.

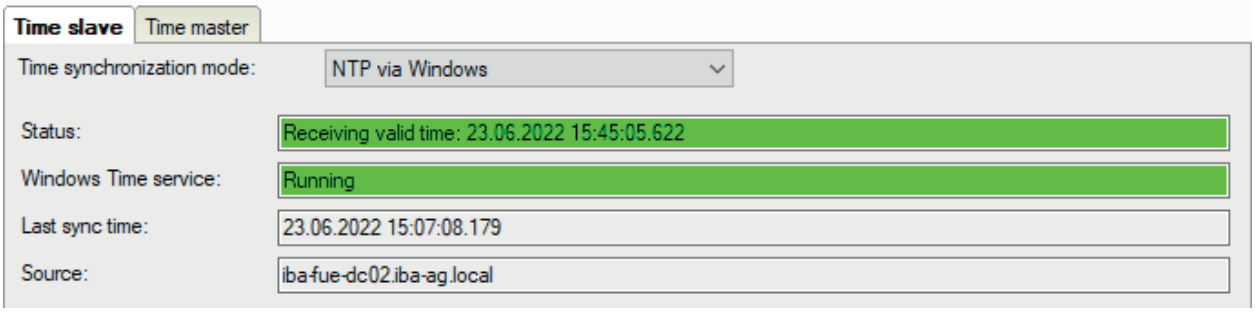
<Reset to factory defaults> button
Click this button to reset all settings to the factory settings after confirming the following prompt with <Yes>.

4.7.5 NTP via Windows

With this setting, *ibaPDA* simply uses the Windows internal time management via NTP.

The fields in this view are for display purposes only. You do not need to configure anything here.

The status information is only displayed when the acquisition is running.



The information is also written to the time synchronization info fields in the data file.

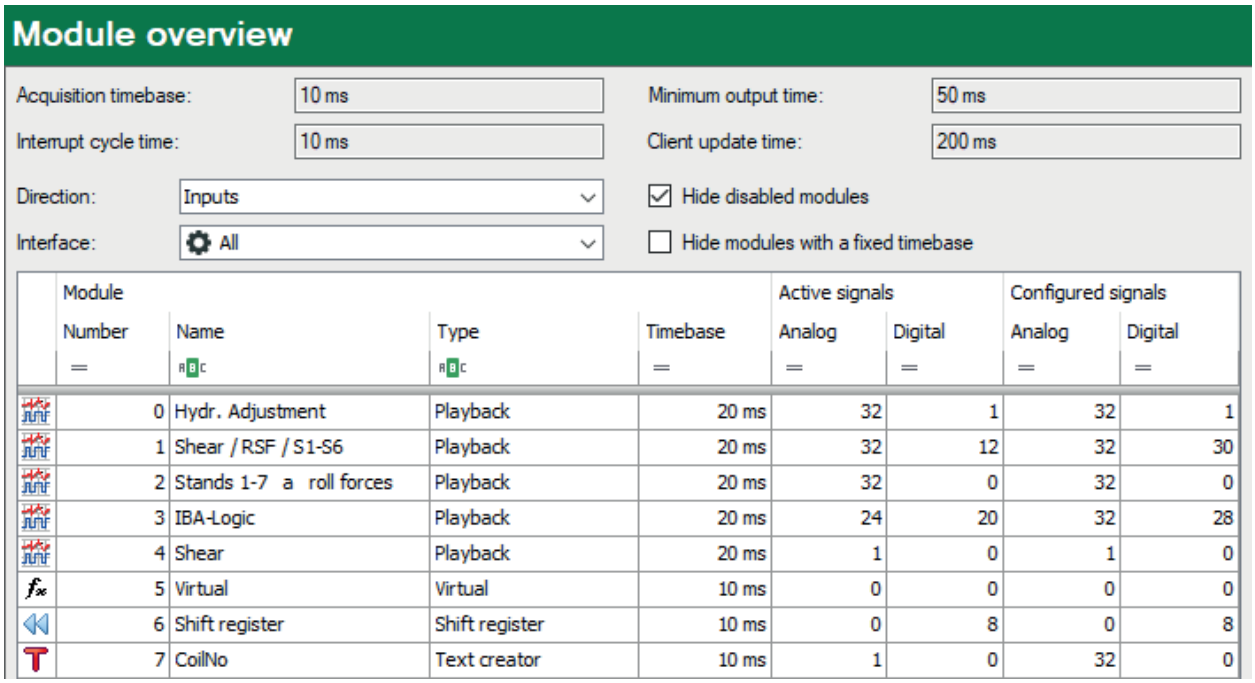
Via the functions *TimeSinceLastSync* and *TimeSyncStatus* in the expression builder, you can read out the interval to the last synchronization and the status, and can additionally display or record this.

Note

An internal *ibaPDA* and an internal Windows time synchronization should not be active in parallel. In such cases, a warning is issued during the validation of the I/O configuration, prompting the user to disable one of the timing modes.

4.8 Module overview

The *Module overview* node offers an overview of the time base settings of the modules and their signals. Here, it's easy to see the effect that changes of module timebases have on the output time and client update time. If you change the module timebases, the resulting values are accordingly updated here immediately.



Module				Active signals		Configured signals	
Number	Name	Type	Timebase	Analog	Digital	Analog	Digital
0	Hydr. Adjustment	Playback	20 ms	32	1	32	1
1	Shear / RSF / S1-S6	Playback	20 ms	32	12	32	30
2	Stands 1-7 a roll forces	Playback	20 ms	32	0	32	0
3	IBA-Logic	Playback	20 ms	24	20	32	28
4	Shear	Playback	20 ms	1	0	1	0
5	Virtual	Virtual	10 ms	0	0	0	0
6	Shift register	Shift register	10 ms	0	8	0	8
7	CoilNo	Text creator	10 ms	1	0	32	0

System time values (only display)

- *Acquisition time base*: Central acquisition time base, as it is set in the Register *General*, *Settings* node.
- *Interrupt cycle time*: Interrupt cycle of the acquisition
- *Minimum output time*: The fastest cycle in which signals that are configured as outputs can be updated and output. The shortest achievable cycle time of the outputs equals the smallest common multiple of all module time bases, but is at least 50 ms.
- *Client update time*: Cycle in which the appearance of the live data is updated on the connected clients. This value too, depends on the smallest common multiple of all module time bases. The minimum here is 200 ms.

Table settings

Hide disabled modules
If this option is enabled, modules which have been disabled in their module settings will not be shown in the table.

Hide modules with set time base
Some virtual modules, e.g. 16 and 32-bit decoders, are permanently bound to the central acquisition time base. If this option is enabled, the affected modules will not be shown in the table.

Direction
Choose which modules should be shown in the table: Inputs or Outputs.

Interface
If you want to show only modules of a certain interface, then you can select the interface here. Multiselection is not possible.

You can setup more filters in the columns.

Table contents
All configured modules are listed in the table. Make a double-click on a module to get to the settings of the respective module. There you may change the module settings.

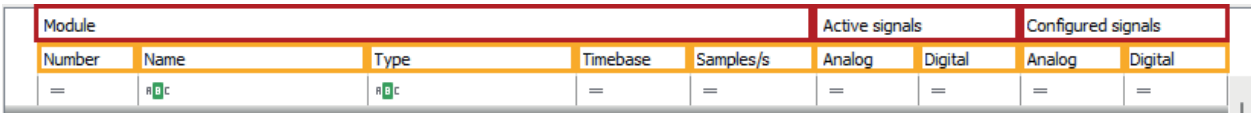
In the column *Timebase* you can change the value right away. You'll see the effects of the change directly on the right side. Additionally, you'll get a note if a change of the output time is the consequence.

The *Active signals* column shows how many analog and digital signals of the respective module are active.

The *Configured signals* column shows how many analog and digital signals are configured in the respective module. Configured signals may also be inactive. Inactive signals do not account for the total number of used and licensed signals (bar graph on the bottom of the I/O Manager).

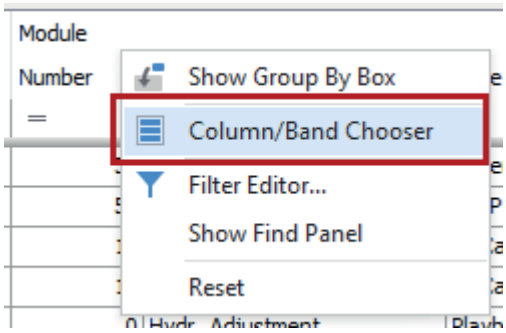
The *Samples/s* column is hidden by default. You can show the column by opening the context menu of the table header and selecting *Column/Band Chooser*. The values in this column show the number of samples per second acquired by each module. The values result from the number of active signals of a module divided by the timebase, given in seconds.

Customize table layout
You can adjust the layout of the table according to your needs by showing or hiding columns or column groups ("bands"). The colum headers have two lines, where the upper line represents the bands and the lower line represents the columns.



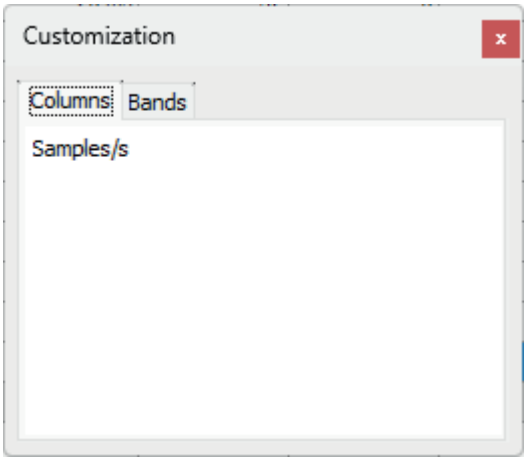
To customize the table layout follow these steps:

1. Open the context menu on the band or column headers. The context menus differ but both have the *Column/Band Chooser* command.



2. Select Column/Band Chooser.

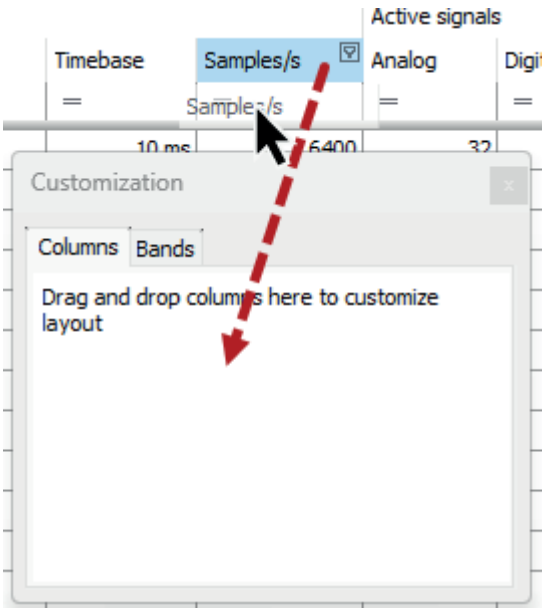
   The *Customization* dialog opens.



   The *Columns* and *Bands* tabs contain the columns and bands which are currently hidden.

3. To show the column or band again just doubleclick on their name.

4. In order to hide columns or bands from the table, drag the column or band in question into the customization window.



Filter the table

You may filter the table entries in different ways:

- Via the filter row:
  Select a search option via the <ABC> button or <=>. Then enter the search string or a value.
- Via the hidden filter button in the header. The button appears when moving the mouse over the respective column header.
  Open the filter window with the filter button. Similar to Microsoft Excel you can now filter the entries in this column.
- Via the search row:
  Via a right-click in the header you can open the search row by choosing <Show find panel> in the context menu. Enter here the search string or value. The entire table will be browsed for your search string or value and filtered.

For detailed information about the filter functions, please refer to part 7 of the manual, chapter *Advanced filters in tables*.

4.9 Knowhow protection

The *Knowhow protection* node offers mechanisms for protecting intellectual property associated with certain calculations and/or settings in *ibaPDA*, which are considered as user knowhow worth protecting.

4.9.1 Application range of know-how protection

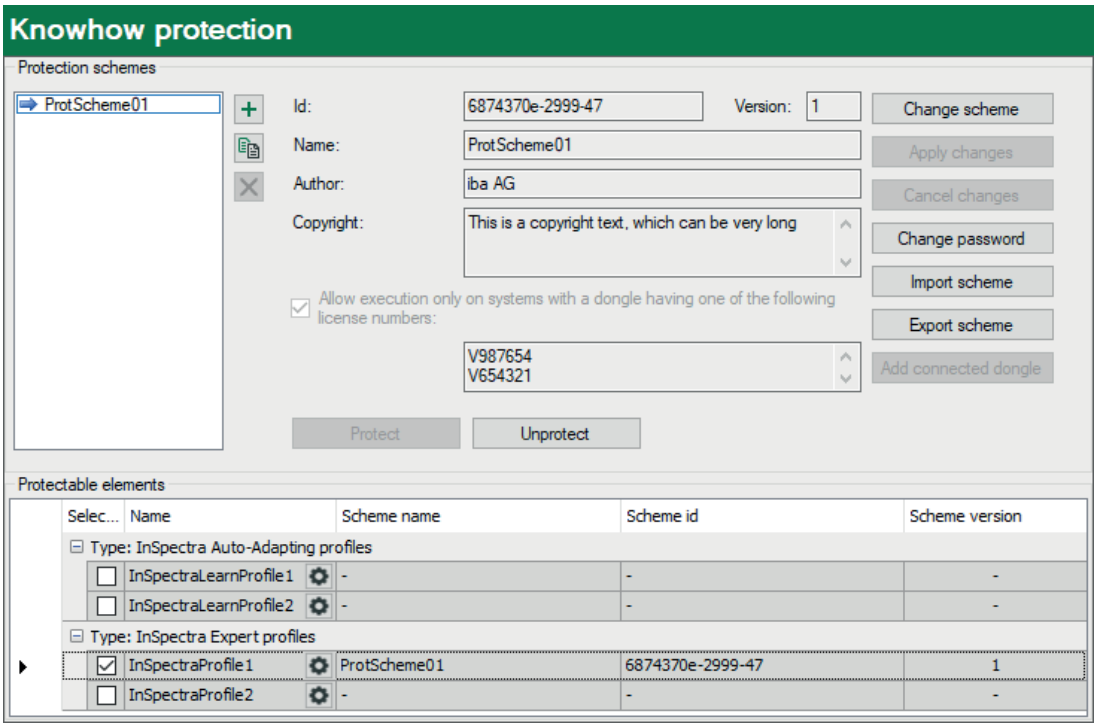
In principle, the know-how protection can be applied to all module types which use profiles, such as

- InSpectra and InCycle modules
- Computation module
- Lookup table
- Parameter set
- Process condition

The following protective functions are realized:

- Protection against change
  The protected elements cannot be changed without entering a password.
- Read protection
  The configuration of the protected elements is not displayed without entering a password.
- License protection
  The protected elements are only executed on systems that run with a previously registered dongle or registered soft license. Several license numbers can be registered.

The protection is realized via so-called protection schemes that, once defined, can always be applied again.



The basic procedure is as follows:

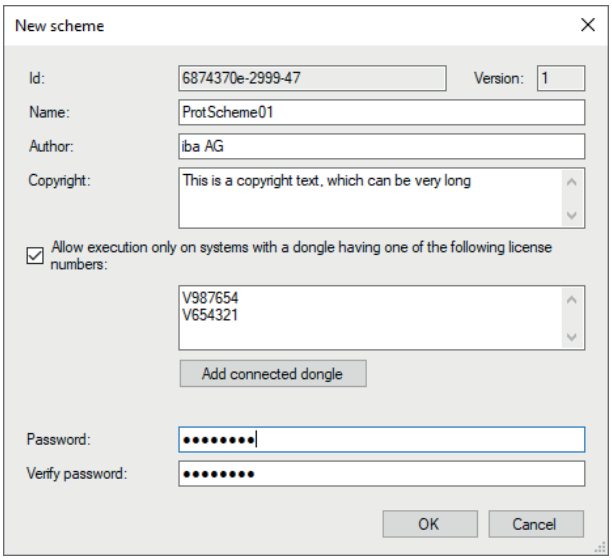
1. Generating a protection scheme
2. Applying a protection scheme to an element

Tip

If you click on the gear icon then you get directly to the configuration dialog of the respective profile.

4.9.2 Creating a protection scheme

1. Open the I/O Manager and highlight the node *Knowhow protection* in the *General* tab.
2. Click the button + to add a new scheme.
   The “New scheme” dialog opens.
   The parameters *ID* and *Version* are automatically generated.



3. Now enter the other parameters and then click on <OK>.

The settings and entries to be made for a protection scheme are specifically as follows:

Author (optional)
Enter the name of the author here.

Copyright (optional)
You can enter a note text here about the copyright of the elements protected by this scheme.

Only permit execution on systems with one of the following license numbers
Enable this option if the elements protected by this scheme are only to be executed on systems with certain license numbers (license protection). Then enter all respective license numbers in the field below. You can easily enter the number of the respective connected dongle or active soft license using the <Add connected dongle> button. If you do not enable this option, there is no execution restriction of the protected elements with respect to the license number.

Password
Enter a password that consists of at least 8 characters. Spaces are not permitted. You will need the password for the following actions:

- Viewing the configuration of a protected element
- Changing the configuration of a protected element
- Changing or removing the protection scheme

Note

What to do, if you don't know the password anymore?

The password of a protection scheme is encrypted and saved in the I/O configuration. Note the password and store it in a save place where you can find it.

If you forgot the password you won't be able to open or edit protected profiles for *ibaInSpectra* or *ibaInCycle*, for instance Because you cannot reset the password by yourself, the only way to fix it is to save the I/O configuration of your system and send the configuration to the iba support desk. You may as well take the project file or simply generate a support file over the *Help* menu and send it to the iba support desk, conjoined with the request for resetting the protection scheme passwords.

iba can only remove the passwords but is not able to retrieve them. iba erases the passwords from the configuration and send it back to you. Then you can load this configuration into your system. Finally, you can define a new password.

4.9.3 Application of a protection scheme

An element can always only be protected by one protection scheme, but a protection scheme can be applied to several elements.

If you have elements in your *ibaPDA* configuration worth protecting, such as InSpectra profiles, then these elements are shown in the table below in the dialog

In order to protect one or more elements, first highlight the desired scheme in the list of the protection schemes (top left).

Then highlight the relevant lines at the bottom by setting a check mark in the selection box and click on the button <Protect>.

Then enter the password for the respective protection scheme and click <OK>.

4.9.4 Removing the protection

In order to remove the protection for one or more elements, highlight the corresponding lines in the table below in the dialog by setting a check mark in the selection box. Then click on the button <Unprotect>.

Then enter the password for the respective protection scheme and click <OK>.

If you want to remove the protection for several elements at the same time that are protected with different schemes, then you have to enter the passwords for all respective schemes in the password dialog.

4.9.5 Importing and exporting protected elements

When elements are protected, they can be exported and imported. The configuration of the elements is encrypted in the export files (*.protectionScheme). You therefore have an easy way of spreading protected know-how to different *ibaPDA* systems.

For an export, highlight the desired element and click <Export scheme>. Then select the desired storage path, enter a file name and close the dialog by pressing <Save>.

If you have highlighted several elements for export, then a prompt dialog appears asking whether all highlighted elements should be saved in the export file or whether you want to select more first.

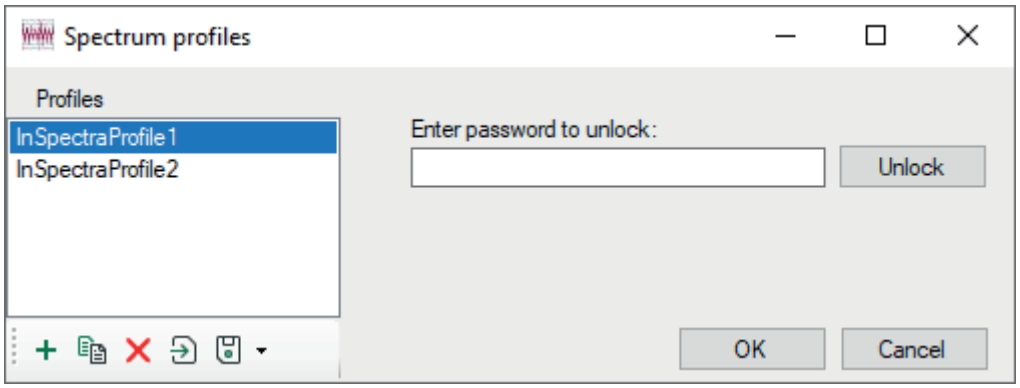
If you want to import a protected element file, click on <Import scheme>, select the desired file (*.protectionScheme) and close the dialog by pressing <Open>.

4.9.6 Unlocking protected items

If you want to access protected items in the application, then you must first unlock the respective item by entering the password for the protective rule.

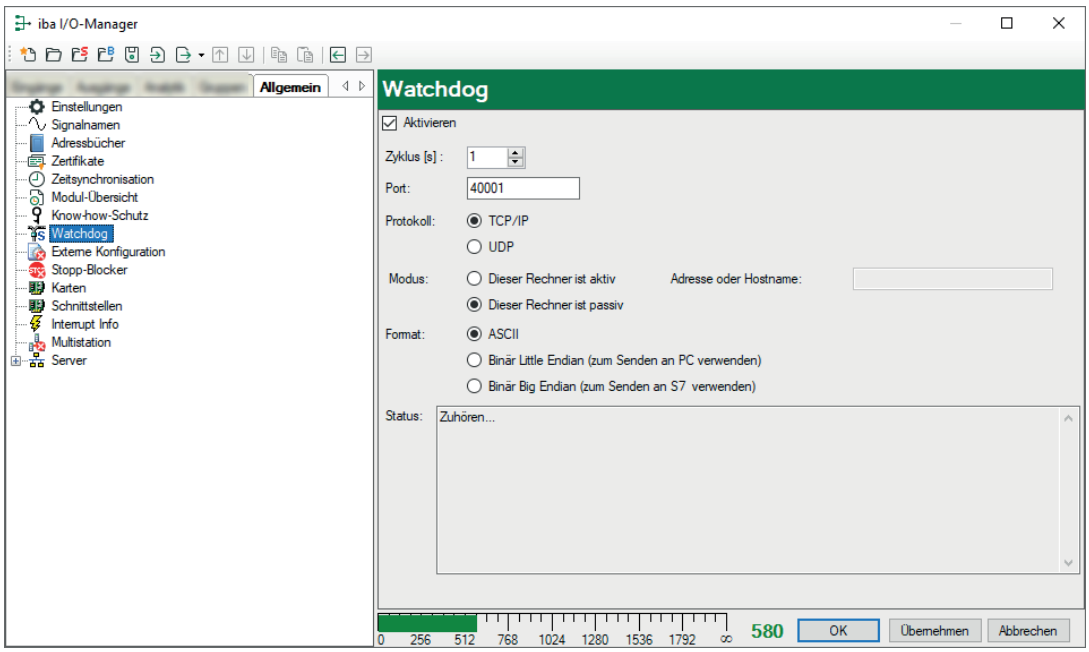
If, for example, you want to view the configuration of the protected InSpectra profiles, then you must first enter the password in the “Spectrum” dialog and click on <Unlock>.

The access remains protected until the I/O manager is closed again.



4.10 Watchdog

The watchdog function is used to monitor the status and operation of *ibaPDA* with the help of another system, e. g. a supervisory PLC.



Enable
Check this box if you want *ibaPDA* to send a watchdog telegram to one or more remote computers. This allows *ibaPDA* to be monitored by other systems. If this option is enabled, the watchdog telegrams will be sent cyclically – irrespective of a running or halted acquisition.

Cycle
Choose or enter the cycle time in seconds for the transmission cycle of the watchdog telegram.

Port
Port address for the watchdog telegram. Default port number is 40001, according to the RFC definition. Adjust if required.

The port address for this connection must be the same in the target computer.

Protocol
Select the protocol for transmitting the watchdog telegram, depending on the receiving supervisory system:

- TCP/IP
- UDP

Generally, the UDP protocol contains less overhead information.

Mode

- *This node is active*
  Select this option if the *ibaPDA* PC is to actively establish a UDP or TCP/IP connection. The IP address or host name of the target computer (host) in the network to which *ibaPDA* is to establish the connection must be entered in the *IP address or host name* field. In active mode, only one target computer is allowed.

- *This node is passive*
  Select this option if the TCP/IP connection initiative should be established by another or several other computers to *ibaPDA*. If *Enable* is also selected for the watchdog telegram, the sign of life is sent to all the computers establishing and maintaining a connection to *ibaPDA*.

Format
Choose between the telegram formats

- ASCII
- Binary Little Endian
- Binary Big Endian

Depending on how the watchdog telegram should be interpreted by other systems or devices, one of the formats must be selected. The ASCII format is recommended for displays or ASCII-based systems whereas the binary format is suitable for computers and PLC systems. If the target system is a SIMATIC S7 controller, choose the Big Endian format for a binary telegram.

In general, the binary format is easier to interpret because it has a fixed size and the information is written at fixed offsets within the telegram.

Status
This field displays status messages which reflect the status of the watchdog connection. If the watchdog is in passive mode, then all the connections are displayed. If in active mode, only one connection is shown.

Further status messages:

- Stopped watchdog is disabled.
- Connected to … xyz: Connection to computer xyz active
- Listening: *ibaPDA* is waiting for another computer to establish the connection
- Error: Error message (text)

4.10.1 Information about the watchdog telegram structure, ASCII format

As described, *ibaPDA* can be configured in a way that it cyclically sends watchdog messages to one or more computers in the network. Besides the information that *ibaPDA* is "still alive", in that it is capable of sending this message, a range of other information is contained in the telegram indicating the current status of *ibaPDA*.

Time stamp
Date and time in ms resolution

PC name
Name of the *ibaPDA* computer in the network

File name (in data storage 1, 2, … n)
Name of the current data file of the relevant data storage on the hard disk.

Status data store 1, 2, … n
Status of measuring and data recording; possible values:

- Idle: no measurement (after start of *ibaPDA* or driver restart)
- WaitForTrigger: Measurement is running, waiting for start trigger, pre-trigger phase
- Recording: Data recording is running
- PostTrigger: Stop trigger has arrived, post-trigger phase

MB free memory for data storage 1, 2, … n
Free memory space on the hard disk

The telegram extends for each further data storage by the last three items.

The entries of information mentioned above are separated by commas inside the message. The end of the telegram is marked by the NUL character (\0). The entries are in ASCII characters. If only one data storage is used, the last entry field is the one with free memory for data storage 1. The message length varies depending on the length of the entries, e.g. path and file names.

If no data storage is enabled, the telegram consists of: *Time stamp,PC name,PDA6,,Idle,-1*

The telegram ends with the NUL character ('\0').

Note

If the system is an *ibaQDR* system, then the telegram is enhanced with further information:

- QDR is running (QDR, Running)

- QDR is stopped (QDR, Idle)

An example of a possible telegram structure can be found in Part 7, *ASCII watchdog telegram structure*.

4.10.2 Information about the watchdog telegram structure, binary format

A list shows the C structure of the telegram in binary format.

An example of the binary code of a watchdog telegram can be found in Part 7, *Binary watchdog telegram*.

Structure of the binary watchdog telegram

Byte offset	Data type	Contents	Remark	
0 1 2 3	int32		Message counter is incremented after each message	Message
4 5 6 7	int32		Version number (currently =1)	
8 9 10 11	int32		Measurement status =1: PDA is measuring	General
12	char	7 6 5 4 3 2 1 0	Bit 0 = 1: Everything is OK Bit 1 = 1: There are disabled signals	
13	char		Reserved	
14	char		Reserved	
15	char		Reserved	
16 17 18 19	int32	MSB LSB	Connection status to ibaCapture-CAM servers connections 1 to 32 1 bit per connection; bit = 1: connection ok	Capture-CAM
20 ...	int32	MSB	Connection status to ibaCapture-CAM servers connections 33 to 64	
24 25 26 27	int32	MSB LSB	Connection status to ibaCapture-HMI servers connections 1 to 32 1 bit per connection; bit = 1: connection ok	Capture-HMI
28 ...	int32	MSB	Connection status to ibaCapture-HMI servers connections 33 to 64	
32 33	short		Status QDR data store	QDR data store
34 35	short		Current directory	
36 37 38 39	int32		Free disk space in MB	
40	char		Reserved	
41	char		Reserved	
42	char		Reserved	
43	char		Reserved	
44 45	short		Status normal data store	1st normal data store
46 47	short		Current directory	
48 49 50 51	int32		Free disk space in MB	
52	char		Reserved	
53	char		Reserved	
54	char		Reserved	
55	char		Reserved	
56 ... 67				2nd data store
68 ... 79				3rd data store
80 ...				more data stores

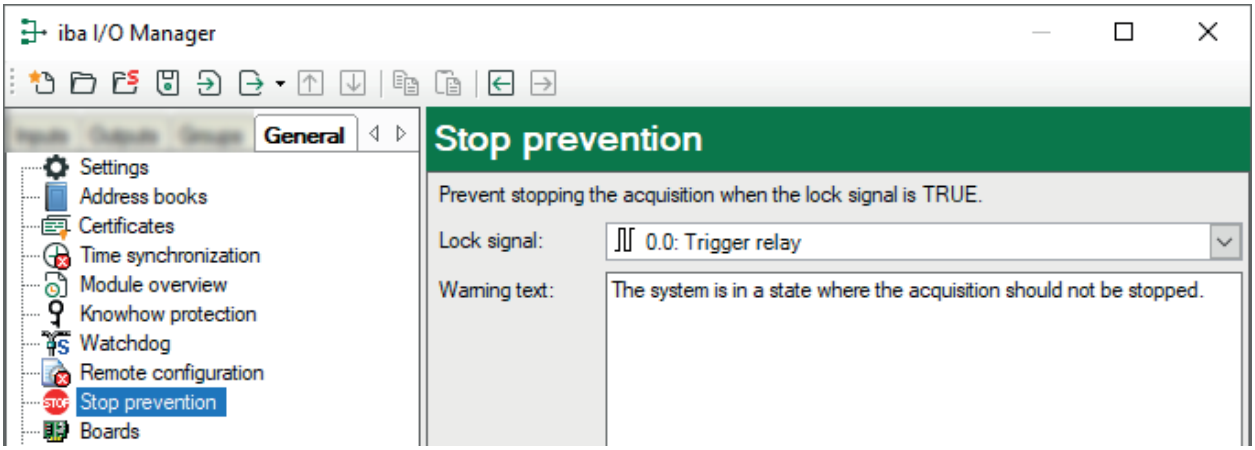
4.11 External configuration

For a detailed description of the remote configuration, see Part 1, *Remote configuration*.

4.12 Stop prevention

By enabling the “Stop prevention function you have the possibility to prevent the acquisition being stopped by accident.

You can enable the stop prevention by selecting any digital signal. If attempts to stop the acquisition occur on an *ibaPDA* client while the signal is TRUE, a message box pops up asking for confirmation. Only if you answer this question with “Yes” the acquisition will be stopped. Using this feature you can protect a running acquisition with signals from the plant, the process or by an operation.



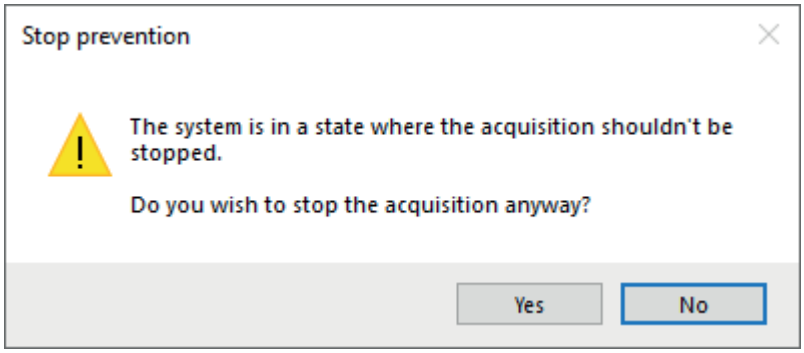
Lock signal
Select the appropriate signal from the signal tree in the drop-down list.

All digital signals can be used as lock signal, e. g.

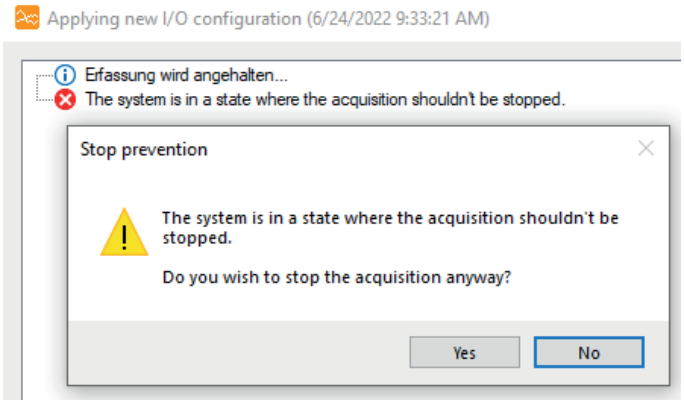
- Digital input signals from sensors or PLC
- *ibaQPanel* inputs
- Virtual signals

Warning text
You can overwrite or modify the standard text in this field. This helps you to include practical advice, directions or contact data in the warning message.

If the stop prevention is triggered by clicking on the < button STOP, the warning message pops up immediately after confirmation of the stop request.



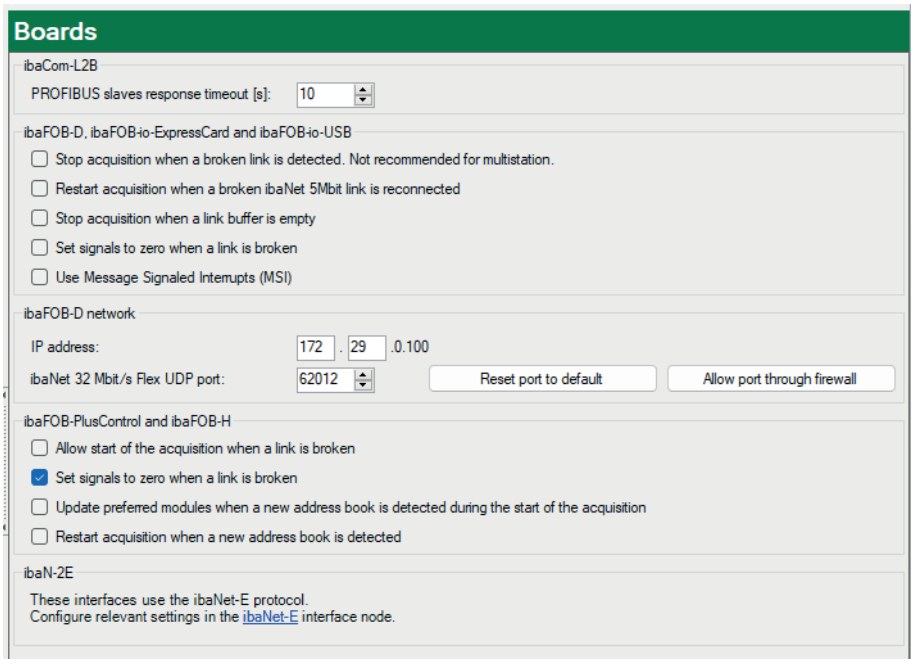
If the stop prevention is triggered by a change of the I/O configuration, the warning message pops up, supplemented by an entry in the validation dialog.



4.13 Boards

Several optional settings regarding iba PC boards can be made at this stage, for example, in order to determine the system's behavior in case of communication errors. Referring to the following boards

- ibaFOB-io-D/-Dexp, ibaFOB-2io-D/-Dexp, ibaFOB-4i-D/-Dexp or ibaFOB-2i-D/-Dexp
- ibaFOB-io-ExpressCard, ibaFOB-io-USB
- ibaFOB-PlusControl, ibaFOB-H
- ibaCom-L2B-4-8 or -8-8
- ibaN-2E



ibaCom-L2B

Profibus slaves response timeout [s]
Enter the waiting time for the connection setup to the DP master here.

When the acquisition starts (Button <GO>), *ibaPDA* tries to establish a connection to the DP master. If the connection is not made within the time set here, the connection is recognized as faulty.

ibaFOB-D, ibaFOB-io-ExpressCard and ibaFOB-io-USB

These settings apply only to either fast measurements (25 kHz) with a higher data transmission rate (5 Mbit/s) or measurements with a fast data transmission rate (32 Mbit/s).

Stop acquisition when a broken link is detected
If you select this option, then all FOB connections of the corresponding card are monitored. If a connection fails, then the measurement is stopped and an error message is output in the event log, provided the report option is also enabled.

Restart acquisition when a broken ibaNet 5Mbit link is reconnected
Select this option if the system is to be put back into operation as soon as the connection has been restored. Any disabled modules will be re-enabled.

Stop acquisition when a link buffer is empty
The link buffer should never be empty. An empty link buffer indicates a malfunction between the card and computer.

Set signals to zero when a link is broken
Enabling this option sets all measured signals of a connection to zero if the connection is interrupted. Otherwise the signal values would show the value at the time of the connection interruption.

Use Message Signaled Interrupts (MSI)
MSI is disabled by default. Enabling MSI on a computer may be useful in case of irregular interrupts or poor interrupt processing.

The firmware version E2 or higher is required.

ibaFOB-D network

IP address and subnet mask
Default setting 172.29.1.100/255.255.0.0

Both during the installation of *ibaPDA* or of the *ibaFOB-D* network as well as during any validation of the I/O configuration, it is automatically checked whether the default IP address is already part of an existing subnet. If this was the case, then the IP address for the ibaFOB-D network would have to be changed. Use the <Check IP address> button to manually trigger this check. A corresponding message is displayed after the check is completed.

ibaNet 32 Mbit/s Flex UDP port
You can set the port number here, which is used for the UDP communication from the ibaNet 32 Mbit Flex protocol. The default port is 62012. If the communication via this port is not permitted by the firewall, you can reconfigure the firewall using the button <Allow port through firewall> so that the port is enabled.

You can always reset the port 62012 using the button <Reset port to default>.

ibaFOB-PlusControl and ibaFOB-H

These settings apply only if an ibaFOB-Plus board or ibaFOB-H board is installed.

Allow start of the acquisition when a link is broken

If one or more connections with the PLUSCONTROL system or Hitachi MACH3 HVDC control system cannot be established at the start of the acquisition, the acquisition will start anyway.

Set signals to zero when a link is broken

See above.

Update preferred modules when a new address book is detected during the start of the acquisition

When establishing the connection with a PLUSCONTROL system or MACH3 HVDC control system, *ibaPDA* reads and analyses the address book. For "preferred signals", "preferred modules" will be created automatically. If a new address book is available, you usually have the possibility to choose how to handle the "preferred modules" (update, replace or no change). If you enable this option, then the existing "preferred modules" will be updated with the "preferred modules"taken from the new address book. Signal-IDs of the already existing "preferred signals" will remain unchanged. If you disable this option then no update will be made

Restart acquisition when a new address book is detected

If you enable this option, then *ibaPDA* checks periodically every 10 s whether the address book has changed or not. If a new address book is detected, the acquisition stops and restarts. The address book will be read at restart of the acquisition. If the option above is enabled too, the "preferred modules" will be updated automatically as well.

ibaN-2E

You can select a time synchronization method here, which applies if the system runs in standalone mode.

4.14 Interfaces

The *Interfaces* node performs two functions:

- Assignment of the physical location of boards and their IDs
- Control of the visibility of the interfaces in the I/O tree

Physical location

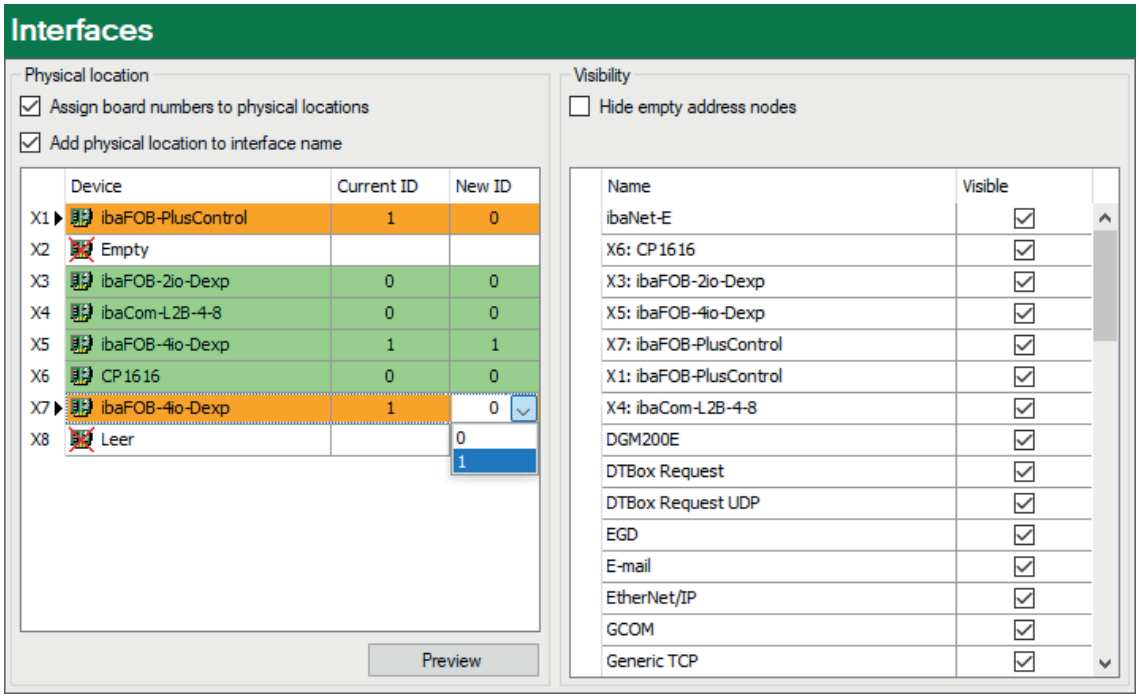
Both a bus and a slot number are automatically assigned by the BIOS of the computer to the plug-in boards in a PC. Based on this information, *ibaPDA* assigns an ID to each iba board. This ID is shown in the 7-segment display of the board. Since a maximum of 8 boards can be inserted into one PC per board type (e.g. *ibaFOB-D/-Dexp* or *ibaFOB-TDC*) , IDs from 0 to 7 are assigned for each board type. The sort order of the boards in the interface tree in the I/O Manager is carried out automatically in ascending order according to the ID. However, this order does not necessarily match the physical order or arrangement of the cards in the computer. So it might also occur that the inserted board in the first slot has a higher ID than the board in the second slot.

If you want the order of the boards in the I/O Manager to match their physical order in the slots then you can make this correction in the "Physical location" table.

Note that this function is only available for certain motherboards, which are supported by *ibaPDA* and used in the *ibaRackline* and *ibaDeskline* computers:

- Avalue EAX-C246P
- Protech BA-0951
- SuperMicro C2SBC-Q
- SuperMicro X10SAT
- SuperMicro X11SPL-F

The table in the "Physical location" area shows the detected boards in slots X1 ... with their current ID.



If you want to change an ID, click in the *New ID* column and select the desired ID number.

Note that the IDs per board type must always be unique (no duplicate IDs).

Boards with altered ID are marked in orange in the table.

Pressing the <Preview> button makes the new ID number flash in the 7-segment display of the board. You can therefore check whether the assignment is correct before you confirm the new configuration with <OK>.

If you have already configured the modules and signals for a board, the modules do not migrate with the ID! You must therefore be sure that the modules are assigned to the correct boards (connectors). If not, you must manually move the modules in the I/O Manager.

ibaPDA ensures that a board's ID is retained if more boards are added.

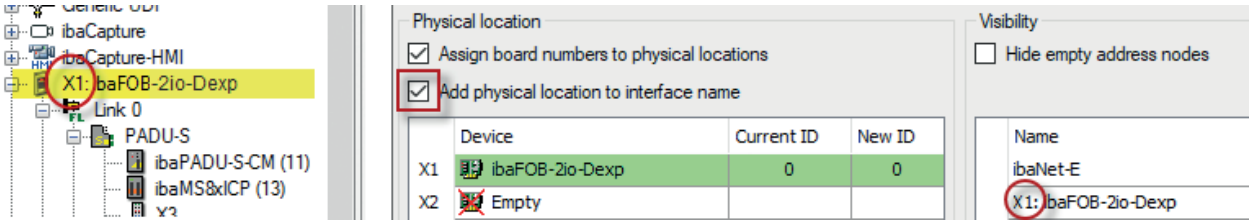
However, if one of multiple boards of the same type is removed, the ID may be changed automatically.

If you, e.g., have installed 3 *ibaFOB-D* boards and the board with ID 1 is removed then the board with ID 2 automatically receives the ID 1 since there cannot be any gaps in the ID numbering.

Assign board numbers to physical locations
If this option is enabled, the board IDs are assigned according to the slot order. Recognized boards are highlighted in green in the table.

If *ibaPDA* is being installed on an *ibaRackline* or *ibaDeskline* computer for the first time, then the option is enabled by default. In all other cases, it is disabled.

Add physical location to interface name
If this option and the option *Assign board numbers to physical locations* are enabled, then the card names in *ibaPDA* receive a prefix with the physical location.



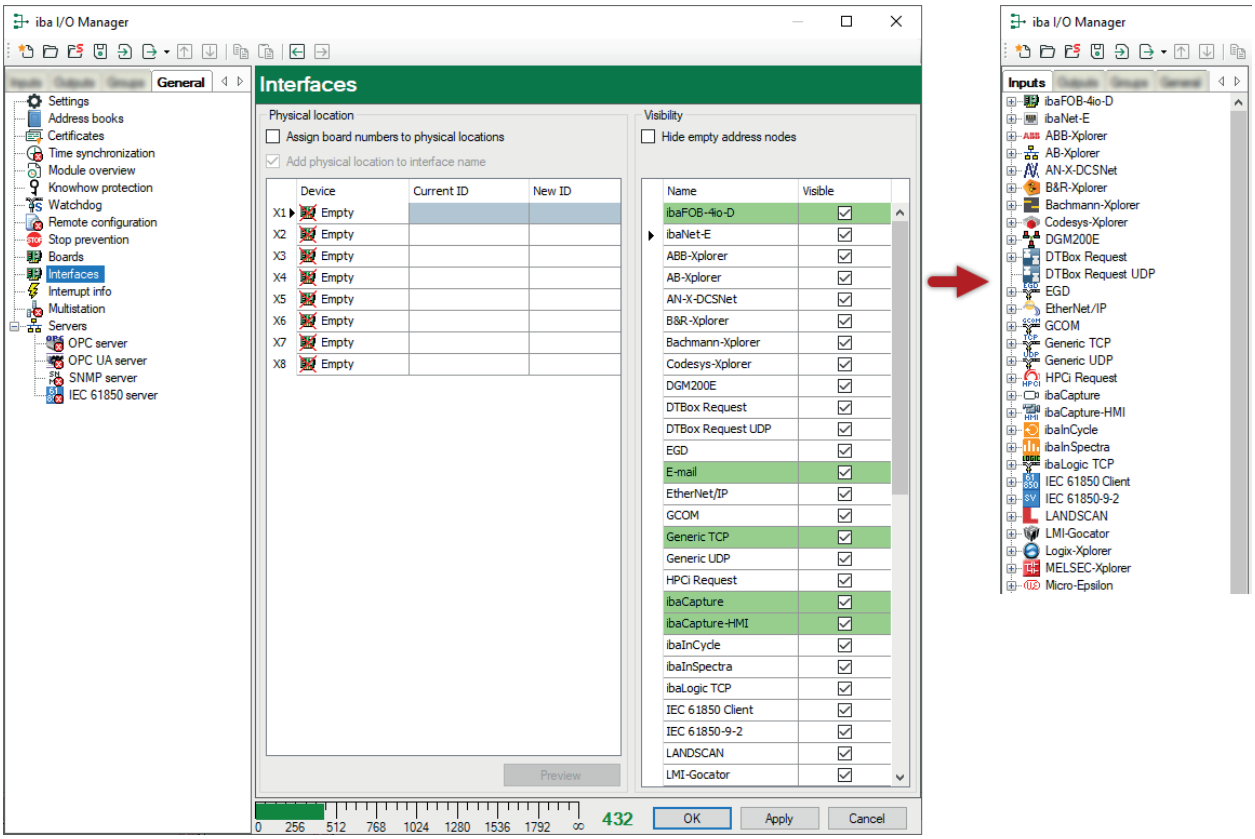
Visibility
The table *Visibility* lists all the interfaces that are available either through licenses or installed cards. These interfaces can also be viewed in the interface tree.

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

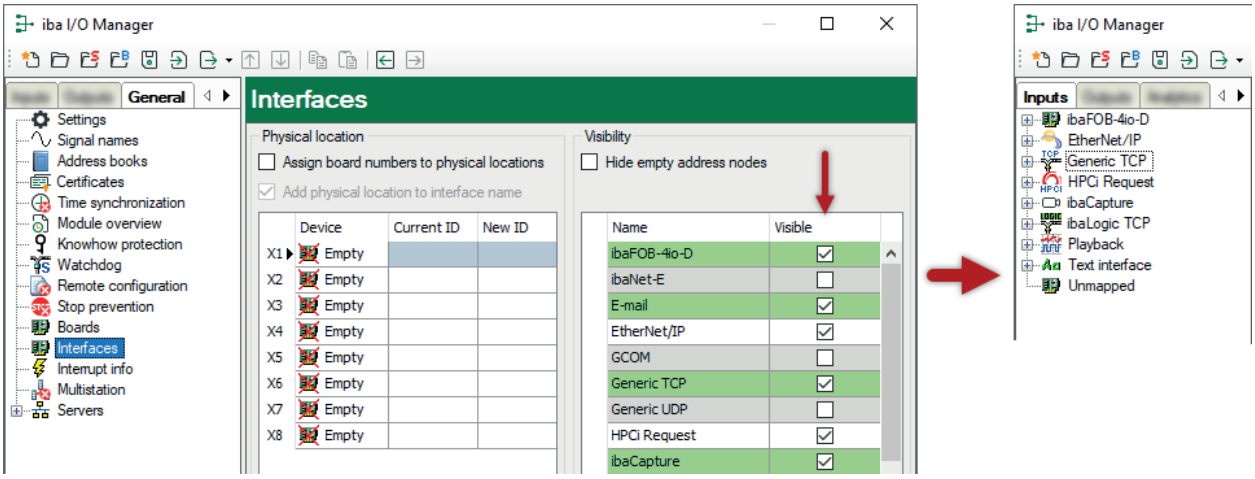
Interfaces with configured modules are highlighted in green and cannot be hidden.

If you double-click on an interface in the table, you will jump directly to this interface in the signal tree in the I/O Manager.

All interfaces visible:



Selected interfaces are visible, the others are hidden:



4.15 Interrupt Info

The display for the driver status (*ibaPDA* driver) must show the "Loaded" status on green background for proper operation. In case of a failure, the field turns red and an error message appears below the start button. If, for example, another process is already running with *ibaPDA*-drivers, the message would indicate that an instance of the program is already running.

In addition, you will find a number of displays that reflect the interrupt behavior. The interrupts are good indicators of the proper function of the system.

Interrupts
The total number of interrupts acquired is displayed here, which is composed of iba's own and external interrupts. When using the internal timer this value equals the "internal time counter”.

iba PCI interrupts
This is an interrupt counter for interrupts that are generated by an iba card (PCI, PCIe or ExpressCard). In proper operation, it counts upwards at a rate of 1000/s (corresponding to an interrupt cycle of 1 ms), if the acquisition time base is 1 ms.

Non-iba PCI interrupts
This counter reading will only change if also other installed PC cards generate interrupts. This could occur in case of “shared interrupts.” If the iba card generating the interrupt is assigned an exclusive interrupt, the counter must not change.

Internal timer counter
The internal timer counter is active when no interrupt source (PC card) is available. In this case the Windows timer is used.

The counter usually also count upwards with 1000 / sec.

Time correction
The value for time correction will only be displayed during acquisition and the internal timer counter serves as interrupt source.
Generally, the internal timer counter has not exactly the same resolution like the basic sample time. Hence, each counter step has a deviation of a few microseconds. These microseconds are added in the time correction counter. As soon as the sum matches the timer resolution, a double counter pulse will be triggered. This method enables the internal timer to match the basic sample time in average.

Acquisition thread CPU usage
This is the CPU time the server needs to read data from the driver and process them, e.g. evaluation of virtual signals, writing to the data file, buffering for transmission to clients etc.

Interrupt buffer (fill level)
This display shows how much data is in the interrupt buffer of the driver.
The *ibaPDA* server periodically reads from this buffer. If the interrupt service routine (ISR) of the driver takes too long, the *ibaPDA* server has not enough time to read all data out of the buffer and the buffer overflows. This would cause a stop of the acquisition. In that case, either a reduction of the number of signals or an increase of the basic sample time is required in order to ensure proper operation. This would reduce the read-out time for the ISR.

In the *I/O-Manager – General* tab – *Interrupt Info* node you may increase the size of the interrupt buffer up to 1024 MB, in order to prevent fast buffer overflow. Default size of the buffer is 50 MB.

Timer resolution
This time value given in ms should approximately correspond to the measurement time base. The timer is used for the internal generation of interrupts.

Interrupt buffer size
Enter here the maximum size for memory-buffered items.

Information for an extended diagnosis is available in 3 sub-tabs.

Interrupt tab

Interrupt times

These values are only displayed when the acquisition is running.

The system measures the actual processing time of the ISR (=interrupt time) and stores the shortest (Min) and longest (Max) time. The ISR reads the requested data from the different PC cards. The ratio of interrupt time to interrupt cycle time shows the percentage of CPU time required to read the data from the cards.

The actual interrupt time should never exceed double the interrupt cycle time. If so, interrupts will get lost and the measurement is not accurate.

Interrupt cycle times

These values are only displayed when the acquisition is running.
The system measures the actual cycle time of the interrupt and stores the shortest (Min) and longest (Max) time.

Asynchronous mode tab

In this tab you will find information about the timing of the DPC routine in asynchronous mode. In fact, information is only available if cards using the asynchronous mode are installed in the system.

Basically, the system works in "synchronous" mode. Synchronous means that the data collected is copied from the card to the PC's working memory by means of the interrupt service routine (ISR). In asynchronous mode, this process takes place outside the ISR with a DPC (deferred procedure call) routine. The DPC routine has a lower priority and thus does not block the system if the processing of routine is not fast enough, e.g. due to a high volume of data. With each interrupt, it is checked whether the DPC routine has stopped. If the DPC routine is still active for an interrupt, the "DPC busy counter" value is incremented.

The switchover to asynchronous mode can be enabled for older cards, such as *ibaFOB-S* or *ibaFOB-X* (I/O Manager, *General* tab, *Boards* node). Other cards, e.g., for reflective memory, can also be used asynchronously. Newer cards, such as ibaFOB-D, do not require asynchronous mode as they write the data directly into the processor memory using DMA technology.

The DPC routine always copies data blocks that are grouped according to acquisition times (time base). Thus, all blocks of data, e.g., that are recorded with 10 ms, are grouped together in one entry. Each of these entries is displayed in the table in the lower part of the tab.

The "Time in DPC routine (ms)" information shows the sums of the copy times of all the entries that were copied at the same time.

The other columns in the table display the times needed to copy the data of the respective entry. The entries to be copied are listed in a waiting list. The maximum number of entries in the queue is normally the max. copy time divided by the cycle time. If the maximum number of entries in the queue is significantly higher then this means that the system could not copy the data fast enough, measured at the set cycle time / acquisition time base.

If the queue contains more than 25 entries, the measurement is stopped with an error message.

Multiprocessor tab

In this tab you will find information about the distribution of the operations or interrupts in multi-core processors. This information is used for the extended diagnostics by iba support.

4.16 Multistation

The multistation operation offers the option to synchronize data acquisition on various *ibaPDA* computers with high precision. The multistation operation is used in cases where the number of measuring signals needed exceeds the maximum capacity of the input cards of a computer or if several *ibaPDA* computers are required due to spatial conditions.

You need a special license to use the multistation function:

ibaPDA-Multistation license (order no. 30.001130)

A multistation configuration can consist of 2 to 33 *ibaPDA* systems.

Perform the configuration of the *ibaPDA* system for operation in a multi-station network with the dialogs in this branch. Here, you determine whether the *ibaPDA* system is active as master or slave in the multistation group.

The standalone mode is set at the factory or if there is no multistation license.

Other documentation

For detailed information on the configuration of the multistation system and how it works, see the separate documentation for the product *ibaPDA-Multistation-License* (order no. 30.001130).

4.17 OPC server

On this node, you can adjust the parent settings for using the OPC DA interface.

There is a separate configurations dialog for OPC UA.

OPC DA ("classic" OPC) uses the DCOM protocol for communication. Please note the following information:

Note

Due to a vulnerability in the DCOM protocol, Microsoft has released security update KB 5004442, which was activated in the final stage on March 14, 2023, and affects the connectivity of OPC components.

With the activation of the new security feature, only DCOM connections (network connections) between OPC servers and clients that support packet-based authentication are possible. Communication between OPC Classic servers and clients running on the same computer is not affected.

Detailed information about the effects of KB 5004442 DCOM security update is available on the Microsoft website (https://support.microsoft.com/en-us/topic/kb5004442-manage-changes-for-windows-dcom-server-security-feature-bypass-cve-2021-26414-f1400b52-c141-43d2-941e-37ed901c769c)

iba AG does not provide support for problems with OPC Classic remote network connections. Depending on the application, there is also the option of communicating via OPC UA instead, thereby avoiding communication via DCOM.

You can enable and disable the OPC server here. The disabled status is indicated by an icon (white cross on red circle) in the tree on the left.

Only the number of the maximum permissible OPC clients that can be connected to *ibaPDA* at the same time must be set at this point.

This setting serves to prevent overloading the resources of *ibaPDA* by too much OPC communication. The default setting is 64. Only increase the value when absolutely necessary.

The other fields show different actual values for OPC communication.

In addition, you have the option to suppress OPC alerts.

Note

The tags in the OPC server will be updated like outputs of *ibaPDA*. Thus, the fastest update cycle derives from the least common multiple of all module time bases or is at least 50 ms respectively.

All other settings for the OPC communication are made in the interface and modules.

Also see part 3, *OPC interface.*

Also see part 3, *Module type OPC client.*

Also see part 3, Module Type *Redundant OPC client.*

Also see part 3, *Module type OPC server.*

4.18 OPC UA Server

Other documentation

You will find an detailed description of the OPC UA server features in the manual about the product *ibaPDA-OPC-UA-Server+*.

4.18.1 General information

By default, *ibaPDA* provides the OPC UA server functionality in order to make data and information about its own status publicly available. This includes information such as

- Issue
- Active licenses
- Status of data acquisition and recording
- Connected OPC UA clients

The OPC UA server function is thus used as an alternative to the SNMP interface or the watchdog telegram in order to inform other systems about the status of *ibaPDA* .

With the *ibaPDA-OPC-UA-Server+* extension, you can also publish all recorded or calculated signals as well as text channels via OPC UA. This allows other systems with OPC UA client functionality to access data collected by *ibaPDA* .

The signals that are published via OPC UA can be conveniently selected using the signal tree in the *ibaPDA* I/O Manager.

The signal values are updated cyclically.

Note

The tags in the OPC UA server, in the *Modules* node, are refreshed at the same rate as the *ibaPDA* outputs. Thus, the fastest update cycle derives from the least common multiple of all module time bases or is at least 50 ms respectively.

The tags in the nodes *Licensing*, *Acquisition*, *Client*, *Data stores* and *Configuration* are updated once every second.

The OPC UA clients can only read (but not write) the tags provided by *ibaPDA* as the OPC UA server.

If required, you may add so called *Writable Tags* (analog and digital) to the OPC UA server data model. By means of an *OPC UA-Server module*, which can be added on the interface node *OPC UA*, it is possible that OPC UA clients can write values into the OPC UA server of *ibaPDA*. In order to use *Writable Tags* and the *OPC UA-Server module* you will also need the license *ibaPDA-OPC-UA-Server+*.

You can imported or generate the certificates required for communication between the OPC UA server (*ibaPDA*) and an OPC UA client in *ibaPDA*.

4.19 SNMP server

ibaPDA has a built-in SNMP server.

SNMP (simple network management protocol) was designed to be able to monitor and control network elements (e.g., routers, switches, printers, computers, etc.) from a central station. By using the SNMP server, the *ibaPDA* system can be monitored by a network-monitoring tool, e. g., Paessler PRTG or Nagios. The SNMP versions, V1, V2c and V3, are supported. The authentication as well as the data transmission is secured by current encryption algorithms.

By default, the SNMP server provides a set of diagnostic signals that give feedback about the state of the system and the various data stores. With the additional license (*ibaPDA-SNMP-Server +*, order no. 30.670050), you can also make any signal acquired via *ibaPDA* available as an object in the SNMP server.

Note

The tags in the SNMP server will be updated like outputs of *ibaPDA*. Thus, the fastest update cycle derives from the least common multiple of all module time bases or is at least 50 ms respectively.

Other documentation

You will find a detailed description of the SNMP server features in the manual about the product *ibaPDA-SNMP-UA-Server+*.

4.20 IEC61850 server

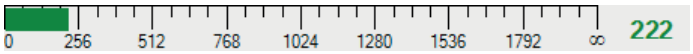
ibaPDA has an integrated IEC61850 server, which can be used with a corresponding license (*ibaPDA-IEC61850-Server*, art. no. 30.670052).

Up to 16 IEC61850 client connections can be configured. The configuration can be exported into a CID file.

Other documentation

You will find a detailed description of the IEC61850 server features in the manual about the product *ibaPDA-IEC61850 server*.

4.21 Signal usage display



The bar chart at the bottom of the I/O manager dialog indicates the current usage of signals. The scale limit corresponds to your *ibaPDA* license, i.e. the maximum number of signals. Only the signals actually used, i.e. the signals with active checkboxes in the signal tables, are taken into account.

5 Groups and vector signals

The signal group feature provides more clarity from a technological point of view.

Open the group area by clicking on the *Groups* tab in the I/O Manager.

An arbitrary number of signals from different data sources, modules or interfaces can be assigned to a group. This provides a more comprehensive view on technological relationships. However, it is your choice whether signals of the same type are to be grouped, e.g. all temperatures, all pressures etc., or to group all signals of a machine or technological device. Of course, a signal can be assigned to multiple groups.

Once you have created groups, you have the choice to get the signals displayed in the signal tree of the *ibaPDA* client, sorted by groups.

The group assignment of the signals and the group names are stored in the data file (*.dat). Hence, they are available in *ibaAnalyzer* too and you may have the signals displayed in the signal tree, sorted by groups.

A group can also be configured as a so called vector signal. Vector signals consist of a number of related signals, such as the signals of a flatness measurement system or the different zones of a temperature scanner. Vector signals are stored in the DAT file and can be used in *ibaAnalyzer* for 2D-top view or 3D representation.

In *ibaQPanel*, vector signals can be used for online 2D top view representation in the trend graph.

5.1 Adding groups

In order to add a new group, click *Click to add group* in the group tree. Alternatively, right-click in the group tree area and select *Add group* from the context menu.

You can also form a divided structure consisting of groups and subgroups. To create a subgroup, right-click on a group and select “*Add group*” from the context menu that appears.

You can give the group a name in the "Group name" field.

If you are using *ibaInSpectra* modules then you can adjust the configuration of the *ibaInSpectra* interface that groups should automatically be created for the *ibaInSpectra* modules. All signals of the *ibaInSpectra* modules are included in this group.

After the text, "Group linked to:", the interface or module name will appear as a hyperlink for certain group types.

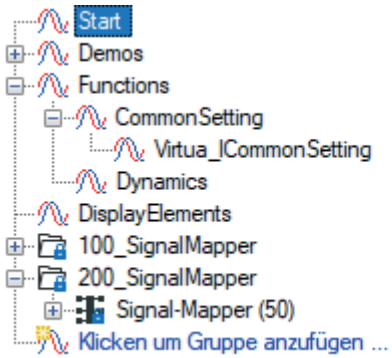
For groups of *ibaInSpectra* modules, for example, you can select a desired group and then click on the blue hyperlink. You are then directed to the module settings of the relevant *ibaInSpectra* module.

If you create folders in the interface tree of the I/O Manager, the folders are shown among the groups by default too. You can disable this function for each folder. Folders are not shown among the groups unless they have a content.

These folders have a lock symbol because they were created outside of the *Groups* tab and cannot be deleted here. However, it is possible to drag and drop signals or other folders inside.

Note

Names of groups, vectors and folders which are on the same level or sub-level have to be unique!



The ID of a group, vector or folder consists of path and name, e. g. ID = Functions/CommonSettings.

5.2 Renaming a group

To rename a group, you have the following options:

- Right-click on group name in the tree and select *Rename* in the context menu.
- Select group in the tree and change the name in the *Group name* field
- Select the group in the tree and press the <F2> key.

5.3 Deleting groups

To delete a group, you have the following options:

- Select the group name in the tree and press the <Del> key.
- Right-click on group name in the tree and select *Remove* in the context menu.

5.4 Move groups

To move a group up or down within the group tree, use the arrow buttons in the I/O Manager.

If you move a group to another group by dragging and dropping it, it will become a subgroup.

5.5 Assigning signals to a group

In the right part of the dialog you see all available data sources, modules and signals as configured in the I/O manager (Inputs) before.

You can assign signals to groups easily by drag and drop.

A signal can be assigned to multiple groups.

You may select multiple signals with the usual key-mouse-combinations <Ctrl> + mouse click or <Shift> + mouse click and assign them together to a group.

You may even assign entire modules to a group when you select the module name and drag it into the group.

By using the <Swap order> button, you can reverse the order of the signals within the vector. The group must be selected (highlighted) to do so. If you like to change the position of one or more signals, select the signal(s) in question and use the up/down arrow buttons in the group tree window. Furthermore, cameras from an *ibaCapture* module can be added to groups as well.

5.6 Deleting signals from a group

In order to delete a signal from a group, click on the corresponding signal in the group tree and press <DEL>. Alternatively, make a right mouse click on the signal name and choose *Remove* from the context menu.

You may select multiple signals with the usual key-mouse-combinations <Ctrl> + mouse click or <Shift> + mouse click and delete them together in the same manner from a group.

5.7 Remove inactive signals from a group

If signals of a group have been disabled and are no longer needed in the group, they can easily be removed from the group. Disabled signals in the group tree are greyed out.

To do this, highlight the group in question. If disabled signals are included in this group then the context menu on this group shows the option to remove disabled signals.

Click on *Remove disabled signals* and the signals in question will be removed from the group.

For information about enabling/disabling signals see part 3, *Notes on working with signal tables*.

5.8 Creating vector signals

In order to create a vector signal, first add a group as described above. Select the group in the tree view and enable the *Vector* option. The group is converted into a vector signal indicated by a yellow-red 2d icon. You may add signals to the vector from the signal tree by drag and drop.

An existing group can be transformed into a vector and vice versa at any time. Enable/disable only the option *Vector*.

By using the <Swap order> button, you can reverse the order of the signals within the vector. This is useful if, among other things, the order of the signals in the I/O Manager is not suitable for the spatial view of the visualized measurement, e.g., for 2D false color representations of material profiles.

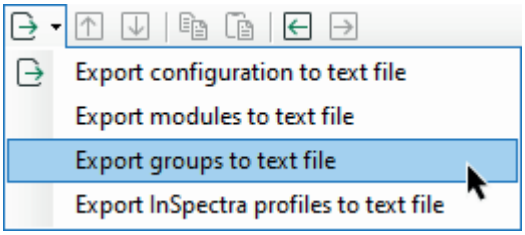
5.9 Export groups

You can export the configuration of your groups to a text file.

So you can, e.g, efficiently edit and extend your group tree by using a suitable text editor or MS Excel. You can then import the edited file again.

The text file can also be used for documentation or transferred to other *ibaPDA* systems.

1. Click the export button in the I/O Manager toolbar and select the groups to export.



2. Use the browser to select the desired directory (drive and folder) in which to store the file, and specify a prefix for the file name. The .txt extension is added by *ibaPDA* automatically.
3. Click on <Save>.

After two headers, the text file contains a line per signal with tab-delimited entries for the group name, signal ID and signal name.

Example

Configuration	Export
Stand_F7 Positions [0:24]: 024 F7 Pos. DS Entry [0:25]: 025 F7 Pos. DS Exit [0:26]: 026 F7 Pos. OS Entry [0:27]: 027 F7 Pos. OS Exit [2:31]: 095 F7 roll force OS [2:30]: 094 F7 roll force DS [2:16]: 072 F7 RPM [2:17]: 073 F7 current [3.17]: F7 Stand loaded [3.26]: Thickn. behind F7 [3.27]: Width behind F7	GROUPS Group→SignalId→SignalName Stand_F7→[2:31]→095 F7 roll force OS Stand_F7→[2:30]→094 F7 roll force DS Stand_F7→[2:16]→072 F7 RPM Stand_F7→[2:17]→073 F7 current Stand_F7→[3.17]→F7 Stand loaded Stand_F7→[3.26]→Thickn. behind F7 Stand_F7→[3.27]→Width behind F7 Stand_F7\Positions→[0:24]→024 F7 Pos. DS Entry Stand_F7\Positions→[0:25]→025 F7 Pos. DS Exit Stand_F7\Positions→[0:26]→026 F7 Pos. OS Entry Stand_F7\Positions→[0:27]→027 F7 Pos. OS Exit

6 Text signals and text processing

ibaPDA is able to acquire, display and record alpha-numerical data in addition to purely numerical information.

Note

With version v7.0.0 of *ibaPDA*, the processing of text information was fundamentally reorganized as regards program technology. Instead of the earlier “technostrings,” text signals are now used, which are fully integrated in the configuration of the modules, analog signals area.

There is no more “technostring” category in the I/O Manager.

6.1 Using the text signals

In addition to pure numerical measured values, processing text information also makes it possible to acquire process-relevant information, such as technological information, specifications about the product, recipes, primary data, reference values, process information or customer data. Interactive text input by the user can thus be included in the analysis of the measurement data.

Some examples:

- Save text signals in the data file

  All text or text parts that were defined as text signals are available in the signal selection during the data recording configuration and can be saved in the data file or in the *ibaHD-Server*. Text signals can be shown with other measurement signals in *ibaAnalyzer* and used in the report generator or extracted to databases.

- Text signal for naming the data file

  Every text signal can be used to name the data file. This must be configured under *Files* in the data storage configuration dialog.

- Dragging numerical values from text signals

  Every numerical character or numerical section of a text can be directly converted into a numerical value and is then available for the numerical calculation of virtual signals.

- Linking and processing texts

  Special text operations, such as ConcatText, TrimText, ReplaceText, etc. are available in the expression builder in order to increase the flexibility of text processing and to increase the informational content of the display and record.

- Display text signals

  Any text signal can be displayed live in the *ibaPDA* client and in *ibaQPanel*.

6.2 Properties of the text signals

Text signals are analog signals of the STRING or STRING[x] type. STRING[x] is a text (string) with a maximum length of x characters.

Text signals can be transmitted to *ibaPDA* through various interfaces. Depending on the type of interface, the text signals are configured in the normal modules or in separate text modules.

A text signal can contain the entire source text or parts of it. The information of a text can be split into several sections in the I/O Manager, whereby each section shows a text signal and can be processed as individual pieces of information.

A text containing a lot of different information may be split up into any number of text signals.

With some interfaces, a text can be split directly in the interface module. With other interfaces, another module *Text splitter* must be added to split a text. The newly created sections are configured in the same module as new text signals of the data type STRING or a numerical data type.

Text signals are used wherever numerical signals are used.

The start/stop index of the characters (for fixed text length) or different delimiters (for changeable text lengths) serve as criteria for splitting the text. In many cases, the JSON format can also be used.

Text signals that only contain numbers can be interpreted as a numerical value with or without decimal marks and can be used for calculations.

A text (signal) can contain up to 10,000 characters.

Every text signal can be enabled or disabled individually.

The different types of texts are classified by their source or the interface which is used for transmission.

Some of the interfaces depend on a corresponding release in the dongle.

Text source	Note
TCP/IP text UDP Text	The source text is transmitted as a TCP/IP telegram or UDP datagram to *ibaPDA* via the computer network. You can configure this in the I/O Manager under the node *Text interface*. In the *TCP text or UDP text* module, *Analog* tab, the source text with the STRING data type and the text signals derived from the source text with the STRING data type or a different data type are configured. For numerical data types, the conversion is carried out automatically. A *TCP text or UDP text* module must be configured for each source text or telegram.

Text source	Note
Serial text	The source text is transmitted via a serial interface to *ibaPDA* (COM1...COM4). You can set the configuration in the I/O Manager under the node *Text interface*. In the *Serial text* module *Analog* tab, the source text with the STRING data type and the text signals derived from the source text with the STRING data type or a different data type are configured. For numerical data types, the conversion is carried out automatically. Depending on the source text or telegram, a module of the type *Serial text* must be configured.
File text	The source text is contained in a text file (.txt, .csv, .json) that is provided on a dedicated drive in a defined path (local or network) by an external system. You configure this in the I/O Manager under the node *Text interface*. In the *File text* module, *Analog* tab, the source text with the STRING data type and the text signals derived from the source text with the STRING data type or a different data type are configured. For numerical data types, the conversion is carried out automatically. Depending on the source text or file, a module of the type *File text* must be configured.
Text creator	The source text is created on the *ibaPDA* server in the *Text creator* module, *Text creation* tab. It can be a typed in fixed text, the value of a signal or the text of another text signal. It can also be a combination of all three forms. Furthermore, XML and JSON data structures can be used for text creation. You can set the configuration in the I/O Manager under the node *Virtual*. In the *Analog* tab of the same module, you will then configure the derived text signals with the data type STRING or a different data type. For numerical data types, the conversion is carried out automatically. Depending on the source text, a module of the type *Text creator* must be configured.
ibaQPanel text input	The text can be entered manually via a text input control of *ibaQPanel*. You can set the configuration in the I/O Manager under the node *Virtual*. A module of the type *ibaQPanel text input* can receive several text signals. In the process, every text signal corresponds to exactly one text input control in the *ibaQPanel* view. Optionally, these text signals can be broken down into sections with a *Text splitter* module and, if necessary, converted into other data types.

Text source	Note
Text shift register	Management of existing text signals in the form of a shift register. It provides the last 1 to 64 values of other text signals. The current value of a text signal is applied to the shift register with a trigger signal (rising edge). You can set the configuration in the I/O manager under the node *Virtual*. The required text signals must have been created beforehand with the different modules.
OPC / OPC UA	The source text is transmitted to *ibaPDA* via the PC network (Ethernet TCP/IP) by an OPC or OPC UA server. You can set the configuration in the I/O Manager under the node *OPC* or *OPC UA*. In the *OPC client* or *OPC UA client* module, *Analog* tab, the source text is configured with the data type STRING. To divide the text from the OPC tag onto individual text signals, a *Text splitter* module must be configured per source text under the *Virtual* node.
B&R-Xplorer	The source text is transferred via the B&R-Xplorer interface (Ethernet TCP/IP) to *ibaPDA*. The processing of symbols of the data type STRING or WSTRING is supported. The configuration can be done in the I/O Manager in a *B&R-Xplorer* module. The source text is formed in the *Analog* tab from the symbols with a string data type. To divide the source text into individual text signals, a module *Text splitter* must be configured under the *Virtual* node.
Codesys-Xplorer	The source text is transferred via the Codesys-Xplorer interface (Ethernet TCP/IP) to *ibaPDA*. The processing of symbols of the data type STRING or WSTRING is supported. This can be configured in the I/O manager in a *Codesys V2 or V3* module. The source text is formed in the *Analog* tab from the symbols with a string data type. To divide the source text into individual text signals, a module *Text splitter* must be configured under the *Virtual* node.
Logix-Xplorer	The source text is transferred via the Logix-Xplorer interface (Ethernet TCP/IP) to *ibaPDA*. The processing of symbols of the data type STRING is supported. This can be configured in the I/O manager in a *Logix-Xplorer* module. The source text is formed in the *Analog* tab from the symbols with a string data type. To divide the source text into individual text signals, a module *Text splitter* must be configured under the *Virtual* node.

Text source	Note
S7-Xplorer	The source text is transferred via the *S7-Xplorer* interface (Ethernet TCP/IP) to *ibaPDA*. The SIMATIC S7-300, S7-400, S7-1200, S7-1500 and WinAC controllers are supported. The configuration can be done in the I/O Manager in a *S7-Xplorer* module. One or more source texts are formed in the *Analog* tab from S7 operands or symbols with the string data type. To divide the source text into individual text signals, a module *Text splitter* must be configured under the *Virtual* node.
TwinCAT-Xplorer	The source text is transferred via the TwinCAT-Xplorer interface (Ethernet TCP/IP) to *ibaPDA*. The processing of symbols of the data type STRING or WSTRING is supported. The configuration can be done in the I/O Manager in a *TwinCAT-PLC or BC/BX controller* The source text is formed in the *Analog* tab from the symbols with a string data type. To divide the source text into individual text signals, a module *Text splitter* must be configured under the *Virtual* node.
ibaVision	The source text is sent from *ibaPDA* output modules, which are configured in *ibaVision*, to *ibaPDA*. The text signals are configured in the I/O Manager under the node *ibaCapture* in so-called *ibaVision text input* modules. To divide the text into individual text signals, a module *Text splitter* must be configured under the *Virtual* node.
EtherNet/IP	The source text is transmitted to *ibaPDA* as a TCP/IP telegram via the EtherNet/IP connection. The configuration can be done in the I/O Manager in an *EtherNet/IP* module. The source text is formed in the *Analog* tab from the symbols with the STRING data type. To divide the text into individual text signals, a module *Text splitter* must be configured under the *Virtual* node.
Generic-TCP	The source text is transmitted to *ibaPDA* as a TCP/IP telegram via the Generic-TCP interface. You can configure this in the I/O Manager in a *Generic TCP* module. The source text is formed in the *Analog* tab from the symbols with the STRING[32] data type. To divide the text into individual text signals, a module *Text splitter* must be configured under the *Virtual* node.

Text source	Note
Generic-UDP	The source text is transmitted to *ibaPDA* as a UDP datagram via the Generic-UDP interface. This can be configured in the I/O Manager in a *Generic unicast UDP, Generic multicast UDP, HiPAC Request or TwinCAT Request* module. The source text is formed in the *Analog* tab from the symbols with the STRING[32] data type. To divide the text into individual text signals, a module *Text splitter* must be configured under the *Virtual* node.
Modbus TCP Server	The source text is transmitted to *ibaPDA* as a TCP/IP telegram via the Modbus TCP server interface. This can be configured in the I/O Manager in a *Modbus Generic* module. The source text is formed in the *Analog* tab from the symbols with the STRING[32] data type. To divide the text into individual text signals, a module *Text splitter* must be configured under the *Virtual* node.
Reflective memory	The source text is transmitted via a Reflective Memory (RM) connection to *ibaPDA*. A special RM interface card is required for the *ibaPDA* computer. Perform the configuration in the I/O Manager under the node *Reflective Memory* in a *Reflective Memory Text* module. The source text and the text signals with the data type STRING derived from the source text or a different data type are configured in the *Analog* tab of the module. For numerical data types, the conversion is carried out automatically.
ScramNet+	The source text is transferred to *ibaPDA* via a ScramNet+ connection. Therefore, a special interface card SC150 (manufacturer: Curtiss Wright) must be installed in the *ibaPDA* computer. Perform the configuration in the I/O Manager under the node *ScramNet+* in a *ScramNet Text* module. The source text and the text signals with the data type STRING derived from the source text or a different data type are configured in the *Analog* tab of the module. For numerical data types, the conversion is carried out automatically.
Sisteam TCP	The source text is transmitted to *ibaPDA* as a TCP/IP telegram via the Sisteam-TCP interface. You can configure this in the I/O Manager in a *Sisteam TCP Generic* module. The source text is formed in the *Analog* tab from the symbols with the STRING[32] data type. To divide the text into individual text signals, a module *Text splitter* must be configured under the *Virtual* node.

Text source	Note
SIMADYN D	The source text is transmitted to *ibaPDA* via a connection with the SIMADYN-D system (CS12/13/14). This requires a special ibaFOB-SD/-SDexp interface card for the *ibaPDA* computer. Perform the configuration in the I/O Manager under the node *ibaFOB-SD/-SDexp* in a *Simadyn-D text* module. The source text and the text signals with the data type STRING derived from the source text or a different data type are configured in the *Analog* tab of the module. For numerical data types, the conversion is carried out automatically.
SIMATIC TDC	The source text is transmitted to *ibaPDA* via a connection with the SIMATIC TDC system (CP52M0 in GDM). This requires a special ibaFOB-TDC/-TDCexp interface card for the *ibaPDA* computer. Perform the configuration in the I/O Manager under the node *ibaFOB-TDC/-TDCexp* in a *TDC text* module. The source text and the text signals with the data type STRING derived from the source text or a different data type are configured in the *Analog* tab of the module.
TDC TCP/UDP	The source text is transmitted to *ibaPDA* as a TCP/IP telegram or UDP datagram via the TDC-TCP/UDP interface. This can be configured in the I/O Manager in a *TDC TCP/UDP Generic* module. The source text is formed in the *Analog* tab from the symbols with the STRING[32] data type. To divide the text into individual text signals, a module *Text splitter* must be configured under the *Virtual* node.
VIP TCP/UDP	The source text is transmitted to *ibaPDA* as a TCP/IP telegram or UDP datagram via the VIP-TCP/UDP interface. This can be configured in the I/O Manager in a *VIP TCP/UDP Generic* module. The source text is formed in the *Analog* tab from the symbols with the STRING[32] data type. To divide the text into individual text signals, a module *Text splitter* must be configured under the *Virtual* node.

Table 5: Sources and interfaces for text signals

Note

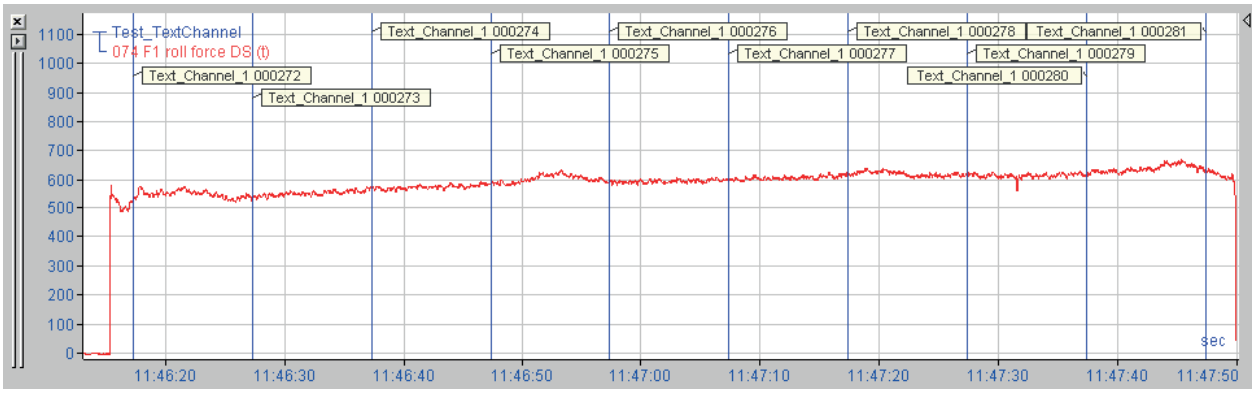
Different information structures are supported for parsing the texts:

- Fixed width: The information is always located at the same place in the text and is determined by the start and stop index of the characters.
- Delimited: The information units within a text are separated by delimiters (e.g., semicolons, commas, tabs, any character).
- JSON: If the text is transmitted in JSON format (e.g., a text file), the information units are interpreted as elements of a JSON object.

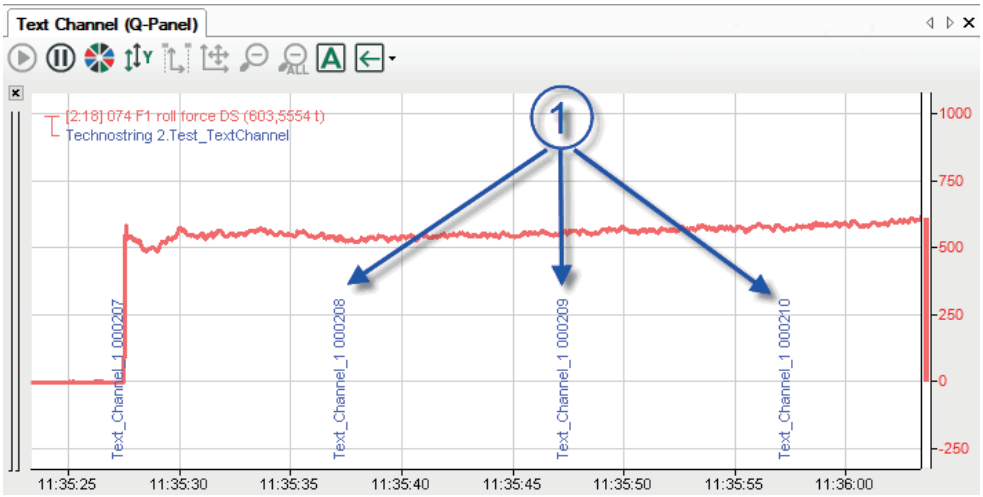
6.3 Using text signals

6.3.1 General

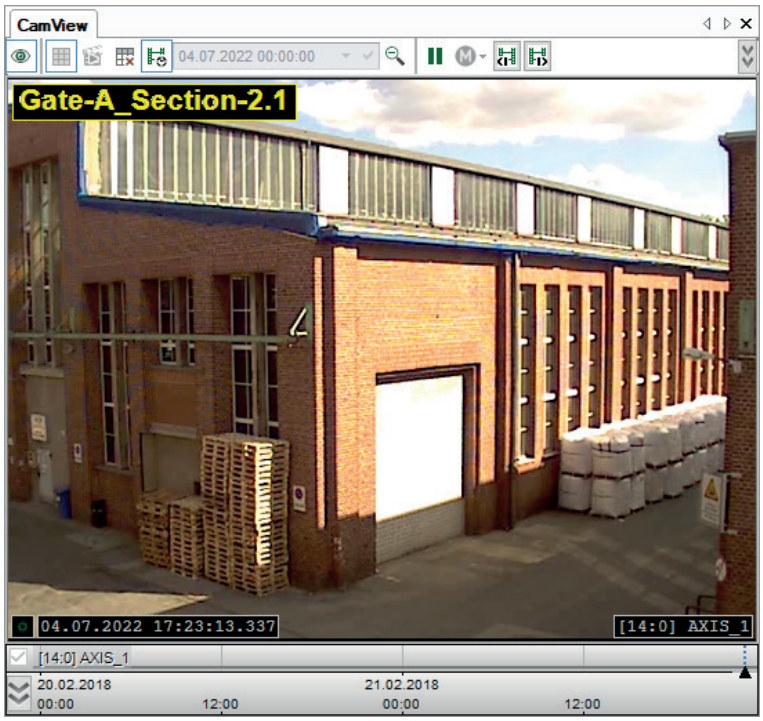
Text signals can be written in a data file like other analog signals. *ibaAnalyzer* shows the text signals as labels at the time of occurrence or their change. They can be added to the graph like usual signals.



If you use *ibaQPanel* for online display on your *ibaPDA* client, the text signals can be displayed along with the *ibaQPanel* trend graph. An example for this is shown in the figure below.



If you use an *ibaCapture* camera view either on your normal *ibaPDA* client or on an *ibaQPanel*, you can use text signals for overlay text displays. An example for this is shown in the figure below.

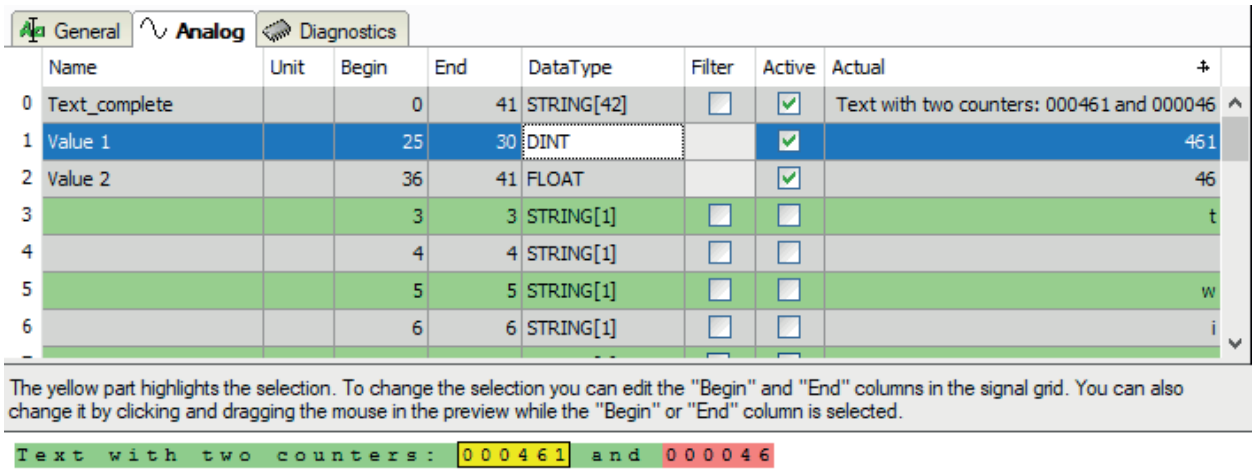


6.3.2 Configuration

In order to measure and record text signals together with other signals, the text signals are located in the respective module as analog signals with the STRING data type. They therefore receive a standard signal ID ([module number: signal number]).

When using a text splitter module, the created text signals receive the ID from the text separator module and not from the source module, which was used to acquire the text.

The example in the figure shows the analog signal table of the text splitter module.



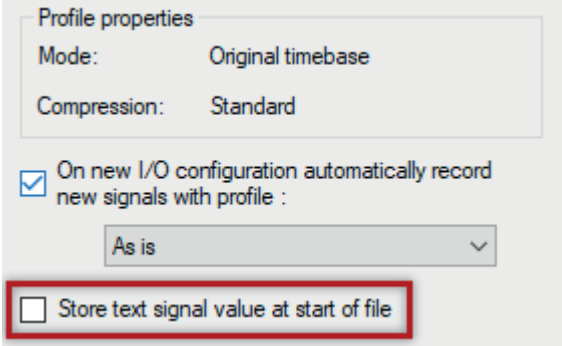
The "Filter" property (see configuration table in the illustration above) controls the acquisition of the text signals. If the filter is disabled, then each time a new source text is received the text signal value is written to the data file.

If the filter is enabled (checkmark), the text signal value is only written to the data file if it differs from the previous values.

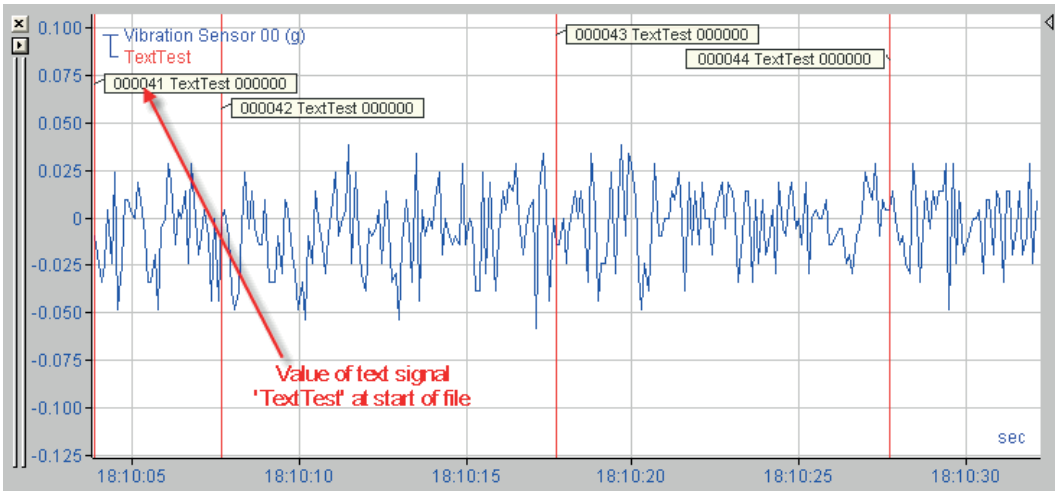
Tip

Usually, the values of the text signal are only saved in the data file when a value changes. So if the value of a text signal does not change during the recording, no text appears in the file. If the value changes during the file recording, only the text signal value from that time point onwards is known.

When a text signal already has a value at the beginning of the data storage, the option *Store text signal values at start of file* can be selected for ordinary data storage in the *Signal selection* dialog. If this option is enabled then the actual values of the assigned text signals are always stored at the beginning of each data file.



So, you can look up the value of the text signal for the whole file. The following screenshot shows the result in *ibaAnalyzer*.



6.3.3 Creating text signals

Text signals are created in the *Analog* tab of a suitable module.

Text source / type	Interface	Module	+ Text splitter module
TCP/IP text	Text interface	TCP text	No
UDP text		UDP text	No
COM (serial text)		Serial text	No
Text file		File text	No
Custom	Virtual	Text creator	No
ibaQPanel text input		ibaQPanel text input	Optional
Shift register		Text shift register	No
ibaVision	ibaCapture	ibaVision text input	Yes
OPC DA	OPC	OPC	Yes
OPC UA client	OPC UA	OPC UA Client	Yes
S7-Xplorer	S7-Xplorer	S7-Xplorer	Yes
B&R-Xplorer	B&R-Xplorer	B&R-Xplorer	Yes
Codesys-Xplorer	Codesys-Xplorer	Codesys V2 or Codesys V3	Yes
Logix-Xplorer	Logix-Xplorer	Logix-Xplorer	Yes
TwinCAT-Xplorer	TwinCAT-Xplorer	TwinCAT PLC or BC/BX controller	Yes
EtherNet/IP	EtherNet/IP	EtherNet/IP	Yes
Generic TCP	Generic TCP	Generic TCP	Yes
Generic UDP	Generic UDP	Generic unicast UDP, Generic multicast UDP, HiPAC Request or TwinCAT Request	Yes
Modbus TCP Server	Modbus TCP Server	Modbus Generic	Yes
Reflective Memory	Reflective Memory	Reflective Memory Text	No
SIMATIC TDC	ibaFOB-TD-C/-Dexp	TDC text	No
SIMADYN D	ibaFOB-SD/-Dexp	SIMADYN-D Text	No
ScramNet+	ScramNet+	ScramNet Text	No
Sisteam TCP/IP	Sisteam TCP/IP	Sisteam TCP/IP Generic	Yes
TDC TCP/UDP	TDC TCP/UDP	TDC TCP/UDP Generic	Yes
VIP TCP/UDP	VIP TCP/UDP	VIP TCP/UDP Generic	Yes

Table 6: Interfaces and module selection for text signals

Depending on the module type, first enter the text signal(s) containing the received text. Some module types, such as TCP/IP text, only allow a source text. To do this, you offer the option of directly deriving additional text signals by splitting the source text into individual text signals and, if necessary, converting them into other data types.

With other modular types, such as some Xplorer modules, several text signals can be received. However, you also need a *Text splitter* module with the source text signal as the reference signal to split the text and generate text signals for further use.

The configuration steps are described in the following chapters for each module type.

6.3.4 Delete the text signals

Delete a text signal like any other signal from the signal table.

When using a *Text splitter* module, you can completely delete the sections derived from a source text signal by removing the *Text splitter* module.

Note

By deleting the source text signal, all text signals linked with it and any *Text splitter* modules become invalid.

6.3.5 Defining and assigning text signals

You can configure text signals even if no text has been received. You will need information about the exact structure of the string.

It is much easier (and more reliable) if you can already receive a text. Then the received text is shown in the preview area of the text modules in the *Analog* tab. The prerequisite for this is that the connection to the text source is established and the acquisition is running.

- For fixed width texts, the desired sections for the derived text signals can easily be selected with the mouse.
- For texts whose informational units are separated by delimiters, the individual text parts can be detected and selected.
- For texts in the JSON format, the values of the detected objects are listed.

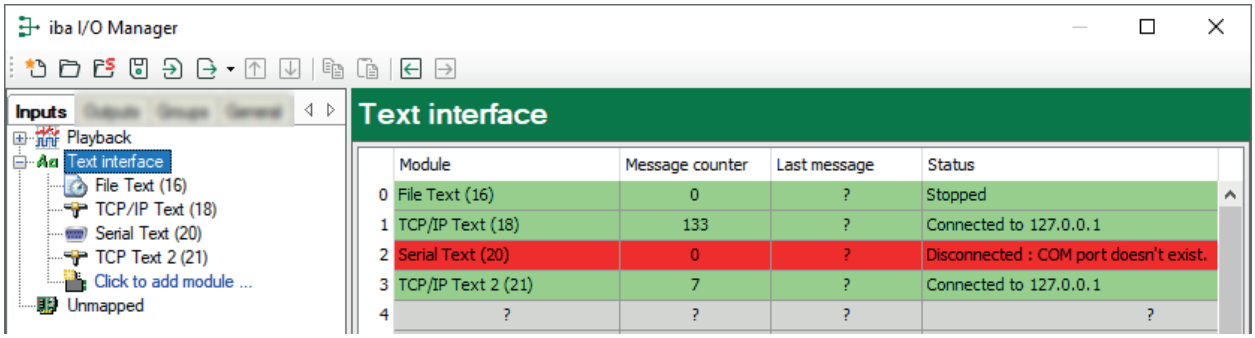
So, any number of text signals can be defined and provided with information from the source text.

6.4 Text interface

You can add the following module types under the node *Text interface* in the I/O Manager:

- TCP Text
- UDP Text
- Serial Text
- File Text

If you select the node *Text interface*, then you will see an overview of all configured modules with the associated telegram counters and status information.



6.5 Text splitter module

The *Text splitter* module is required to generate text signals for some interfaces and can generally be used to generate text signals from existing text signals.

A *Text splitter* module contains a received or generated text signal as an input signal and then provides the complete received text or parts of it as text signals or signals of other analog data types.

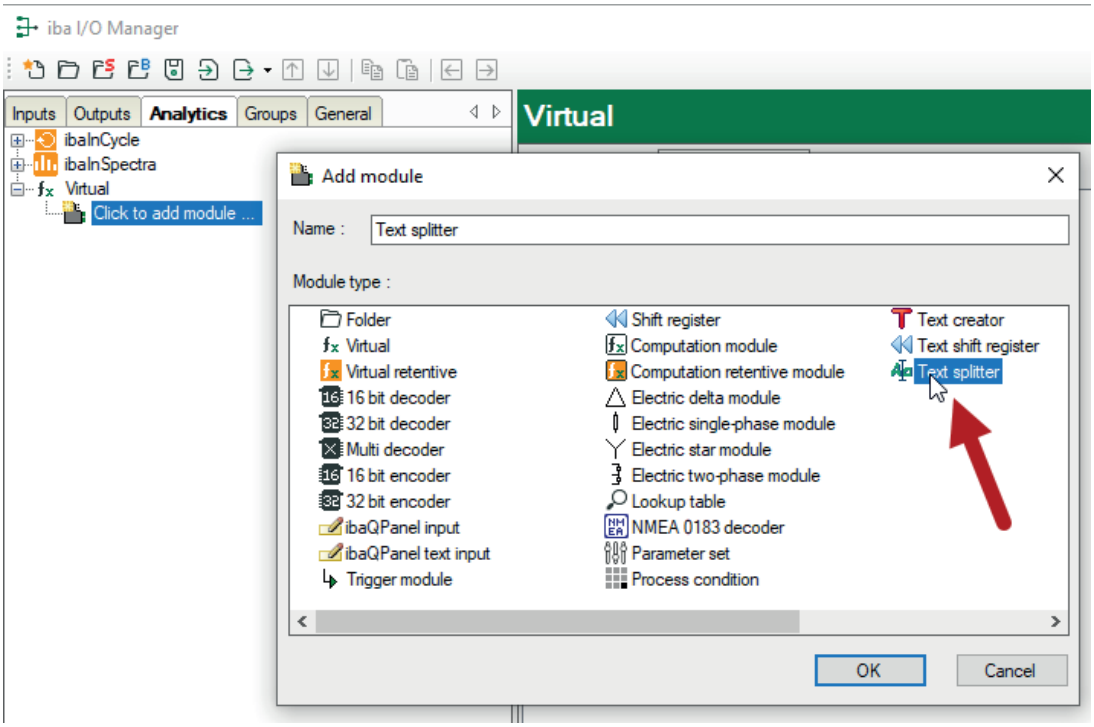
The principle of the text signal definition as carried out in the *Text splitter* module can also be found with other module types.

You need a *Text splitter* module for text signals of the following interfaces:

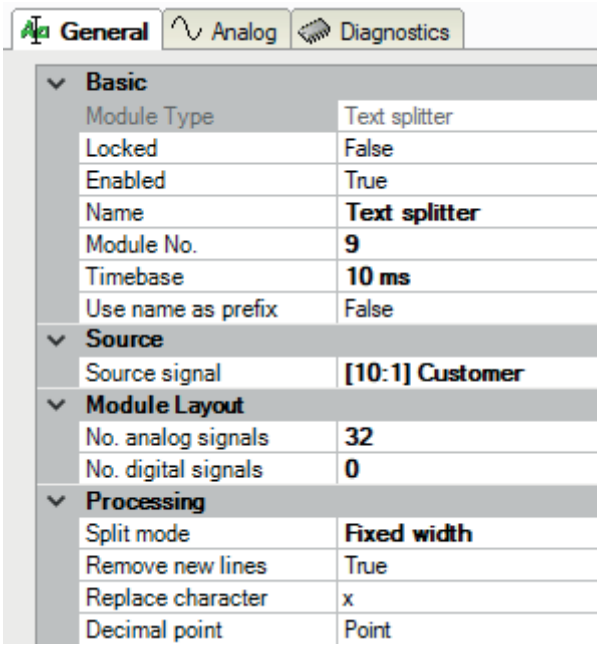
- OPC / OPC UA
- S7-Xplorer
- B&R-Xplorer
- Codesys-Xplorer
- Logix-Xplorer
- TwinCAT-Xplorer
- EtherNet/IP
- Generic TCP
- Generic UDP
- Modbus - TCP - Server
- Sisteam TCP/IP
- TDC - TCP/UDP
- VIP-TCP/UDP
- ibaVision

6.5.1 Add text splitter module

You can add a module by clicking under the *Virtual* interface.



6.5.2 Module settings



Basic settings

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

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

Enabled
Enable the module to record signals.

Name
You can enter a name for the module here.

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

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

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

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

Source

Source signal
Enter the signal from the interface module here, with which the text was received.

Module structure

The number of analog signals is preset to 32, the number of digital signals to zero. If required, you can change the number. Permissible range: 1 to 1000.

Processing

Split mode
Select the suitable mode here that corresponds to the input text structure in order to always correctly read out the information contained:

- Fixed width
- Delimiter
- JSON:

The different settings and their meaning are described in the following chapters.

Remove new lines
If you activate this option (true), then all line feeds (‘new lines”) will be removed when parsing the input text.

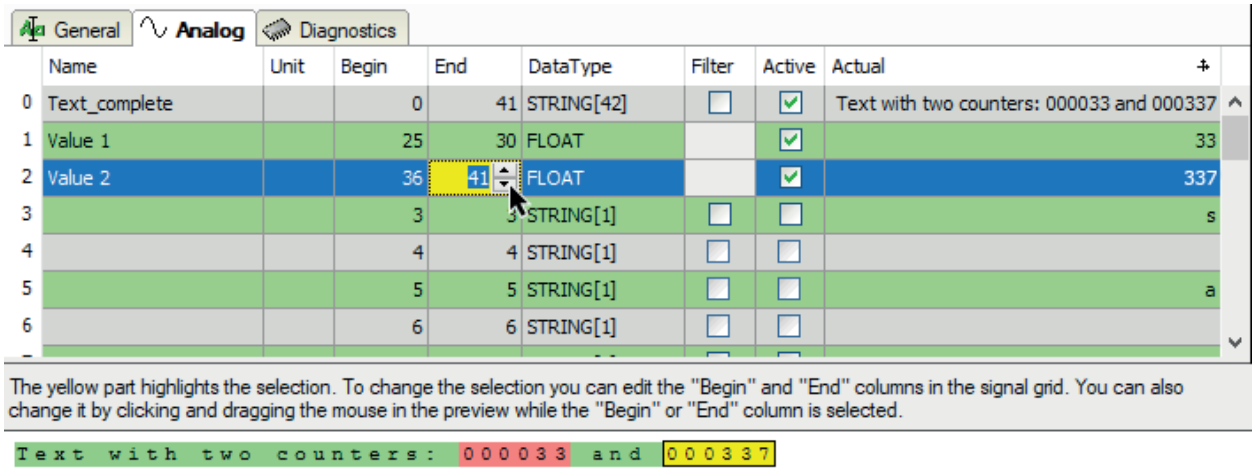
Replace characters
Enter a character here that is to replace all non-printable characters in the input text. Default: x.

Decimal point
Select the decimal separator here so that numerical values with decimal points in the text are correctly interpreted: Period or comma.

6.5.2.1 Fixed width split mode

The *Fixed width* mode is best selected when the input text always has the same length and all information within the text is always at the same place, and therefore always has the same number of characters.

In this mode, you must set a start and stop index for the first and last character for each (text) signal that you want to derive from the input text. You can set this in the *Analog* or *Digital* tab of the module.



If a text has already been received, then it is shown in the preview area under the signal table.

To determine the character area, you can manually enter the start and stop values in the *Start* or *Stop* columns or set them with the spinner buttons. The index for the first character in the text is zero (0). You can also apply this method if no text has been received yet, but you already know the structure.

If an input text is already visible, then you can alternatively use the mouse: Click in the line of the desired signal with the mouse in the column *Start* or *Stop*. Then click in the preview area on the first character in the text for the desired signal, keep the mouse button depressed and drag the mouse to the last character. The selected area will be highlighted in yellow and the indexes will automatically be updated in the *Start* and *Stop* columns.

Then set the desired signal data type. This can either be a text (STRING[x] type) or a numerical data type. In the latter case, the text is parsed in order to generate a numerical value. If a decimal mark is used within the numbers in the text, ensure that the decimal mark (period or comma) is correctly set in the *General* tab.

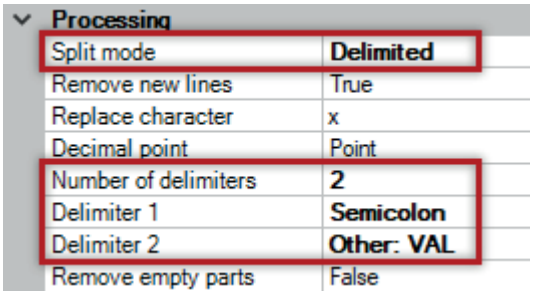
The different colors refer to the following states in the preview area:

Color	Description
Green (1)	All characters of the text associated with a text signal are highlighted in green.
Red (2)	The red color indicates characters in the text which are assigned to multiple text signals.
Yellow (3)	The characters of a text signal selected in the table are highlighted in yellow.

Table 7: Color code in the preview area for text signals (fixed width split mode)

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

6.5.2.2 Delimited split mode



Select this mode if the information units in the input text are separated by defined delimiters. In this case, the length of the text does not have to be constant any more and the character positions are irrelevant. Only the sequence of the information units must always be the same, because the sections from delimiter to delimiter is counted from 0 to *n*.

A delimiter may consist of an individual or several characters. The input text determines what you need.

In addition, several delimiters can be defined. Enter the number of possible delimiters and then select the desired character or character combination for each delimiter. Whenever one of the delimiters appears in the text, this will be interpreted as a new section.

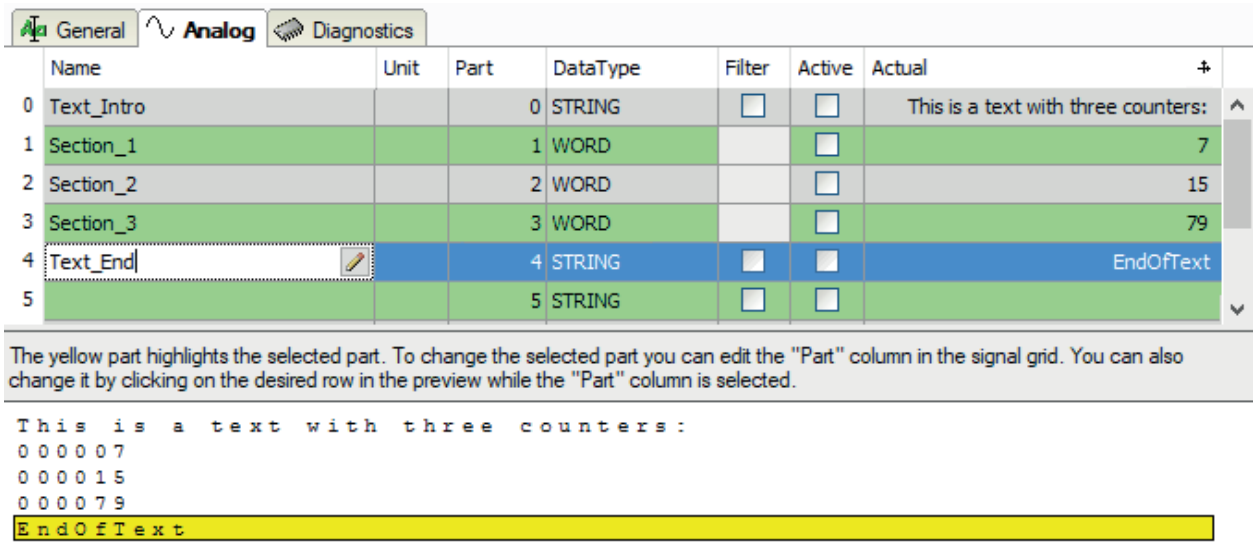
The most common characters are offered in a drop-down list. However, you can define any string as a delimiter (“Other…").

Example with several delimiters
Input text:

This is a text with three counters:**VAL**000007**;**000015**;**000079**VAL**EndOfText

Delimiters are the character string VAL and the semicolon.

The text therefore consists of five parts (0 to 4).

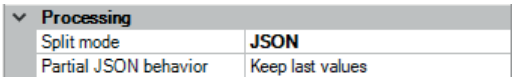


In the delimiter mode, the input text is displayed split in the preview area. Each line corresponds to one part of the input text. In the *Part* column of the signal table, you can now assign the text parts to the signals in case you do not want to use all of the recognized parts or just intend to build a different sequence.

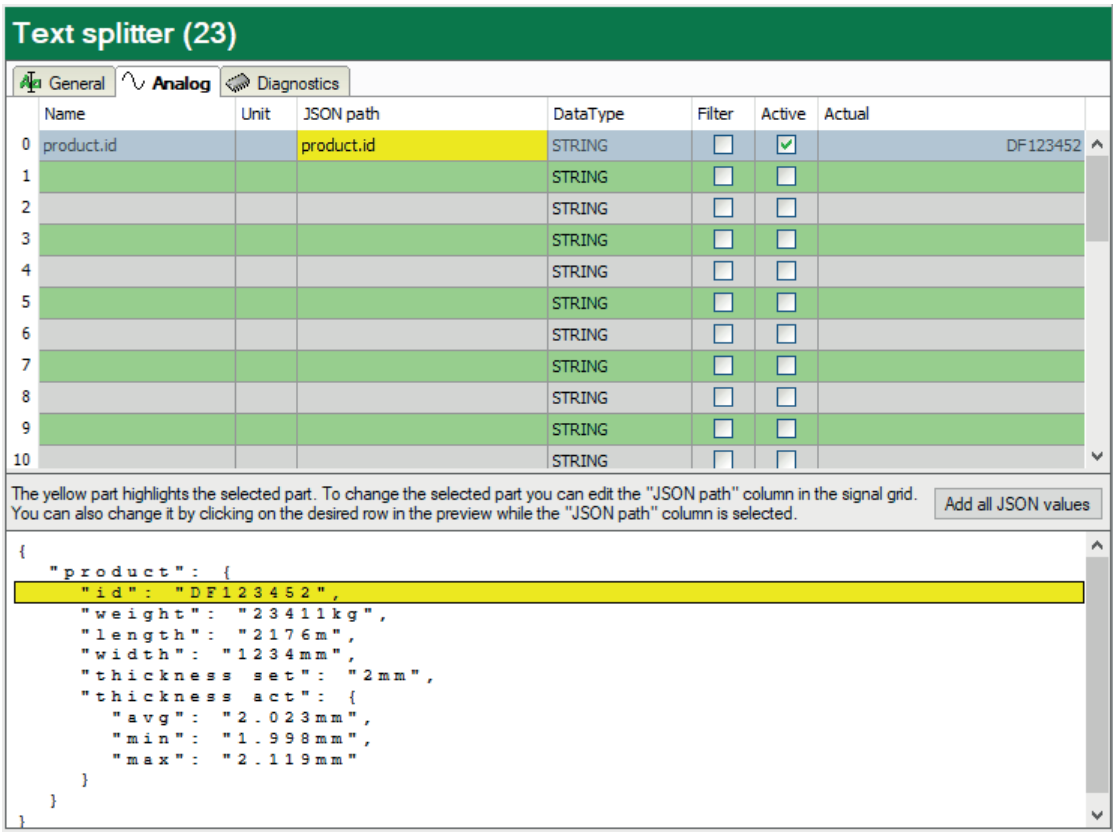
Either enter the number of the part manually or use the spinner buttons. The section set is highlighted in yellow in the preview area.

Alternatively, click in the row of the respective signal in the *Part* column and then in the preview area on the desired text part. The part number will then be updated automatically.

6.5.2.3 Split mode JSON



The text is interpreted as a JSON object in JSON mode. If *ibaPDA* has received a text in the JSON format, the preview area of the *Analog* tab will show a value per line. The text received in the JSON format will be shown already broken down.



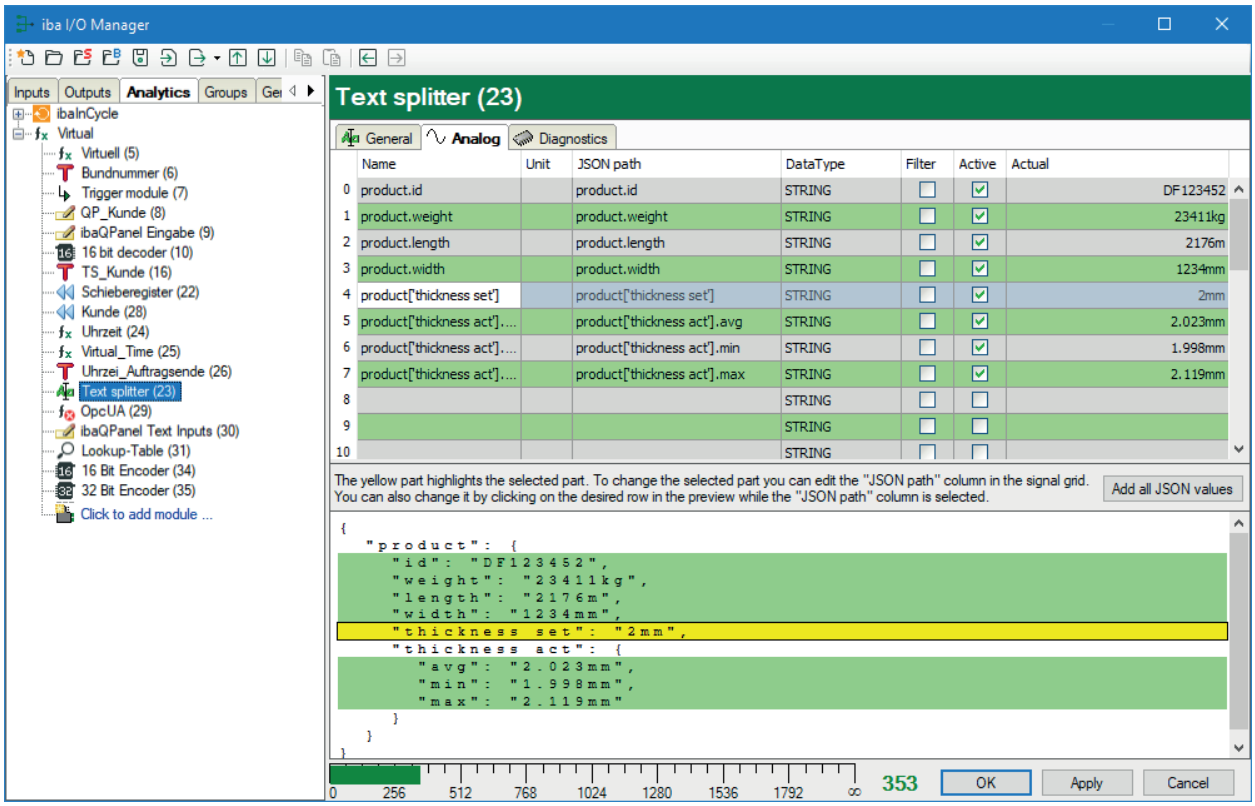
You must now set the JSON path for the desired signals in the signal table. If you want to configure individual values specifically or in a different sequence, proceed as follows:

Click in the signal table in the JSON path column and then on the line with the desired value in the preview area. The correct path will automatically be entered in the signal table. The JSON path will be adopted as the signal name. If necessary, however, you can change the signal name as you wish. Then select the correct data type, which the signal should receive.

To apply the detected values, click on the <Add all JSON values> button.

According to the JSON definition, values in quotation marks will be interpreted as text and values without quotation marks will be interpreted as numerical values. The data type will accordingly automatically be set to STRING or FLOAT.

The following figure shows JSON text signals after all values have been added. The signal names partially have been changed.



In the module settings you can also specify the behavior in case only a JSON part was received. You can choose between *Keep last values* and *Reset values*.

The setting *Reset values* sets the numeric and analog signals not contained in the received messages to zero, text signals to an empty string and digital signals to False.

6.5.3 Signal configuration

Configure the text signals in the *Analog* or *Digital* tab as described in the chapters on the split modes.

Note

The value of a digital text signal can only be 0 or 1. All characters in the input text including the 0 (zero) result in the value 0. All numerical values (single or multi-digit) that are not equal to 0 result in the value 1. Alphanumeric combinations also result in 0.

6.6 Text signals via TCP/IP and UDP

The source text is transmitted via TCP/IP telegram or UDP datagram over the computer network to *ibaPDA.* You can set the configuration in the I/O Manager under the node *Text interface*. In the *TCP Text or UDP Text* module's *Analog* tab, the source text with the STRING data type and the text signals derived from the source text with the STRING data type or a different data type are configured. For numerical data types, the conversion is carried out automatically.

A *TCP Text or UDP Text* module must be configured under the *Text interface* node for each source text or telegram.

Please note the limitation of the receive buffer when configuring the text size.

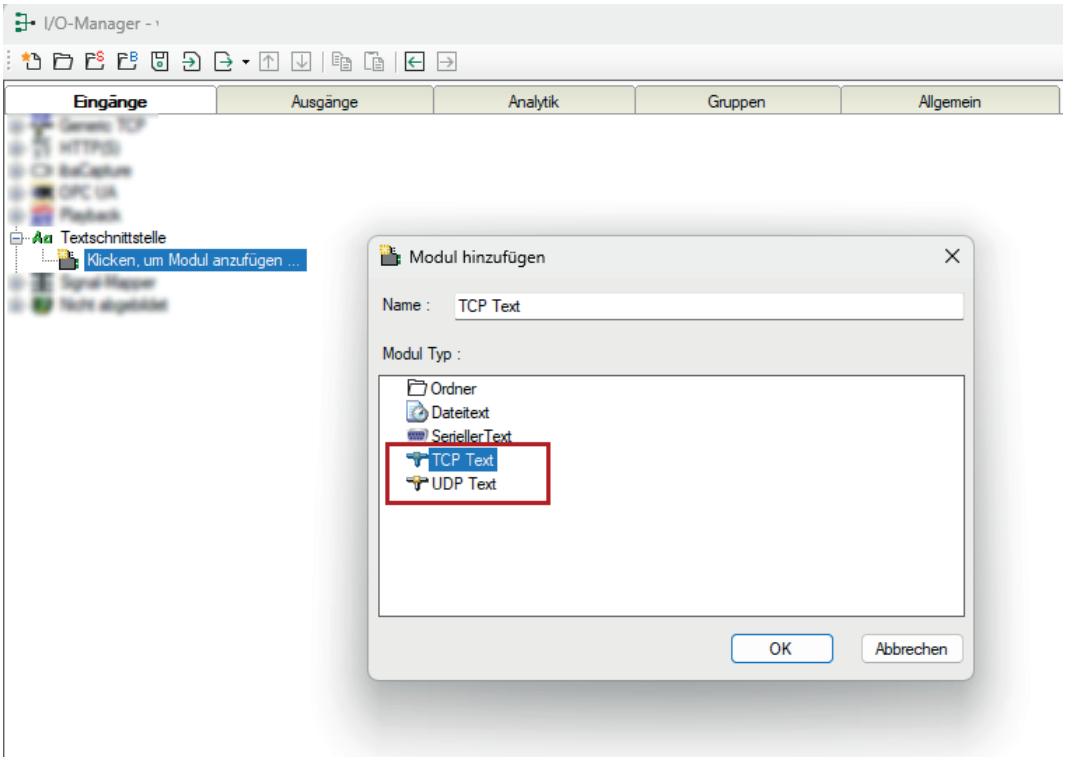
Transmission protocol/module type	Receive buffer size
TCP Text module	20,000 Byte
UDP Text module	64 kB

Thereby, the quantity of data to be processed per message is limited too.

One character needs 1 Byte or 2 Byte, depending on the charcater set or encoding.

6.6.1 Add module

You can add a module *TCP Text or UDP Text* under the node *Text interface*.



6.6.2 Module settings

Basic settings

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

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

Enabled
Enable the module to record signals.

Name
You can enter a name for the module here.

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

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

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

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

Text encoding
You can select the type of text encoding or the code page here for a correct interpretation and display of the received text data for inputs as well as of the text data to be sent for outputs. Available for selection are, beside system locale according to the Windows system settings (default) and UTF-8 Unicode, all other encodings.

Source

Terminator
This is the agreed termination character for the text, which is expected by *ibaPDA* and which should be used by the transmitting system. The terminator could be any ASCII-character, e. g. Carriage Return or another decimal coded character. You may alter the terminator according to your needs. The default value is 13 ("carriage return”).

Port
The port number must match the port number used by the transmitting system in order to establish a TCP/IP connection and receive the text telegrams.

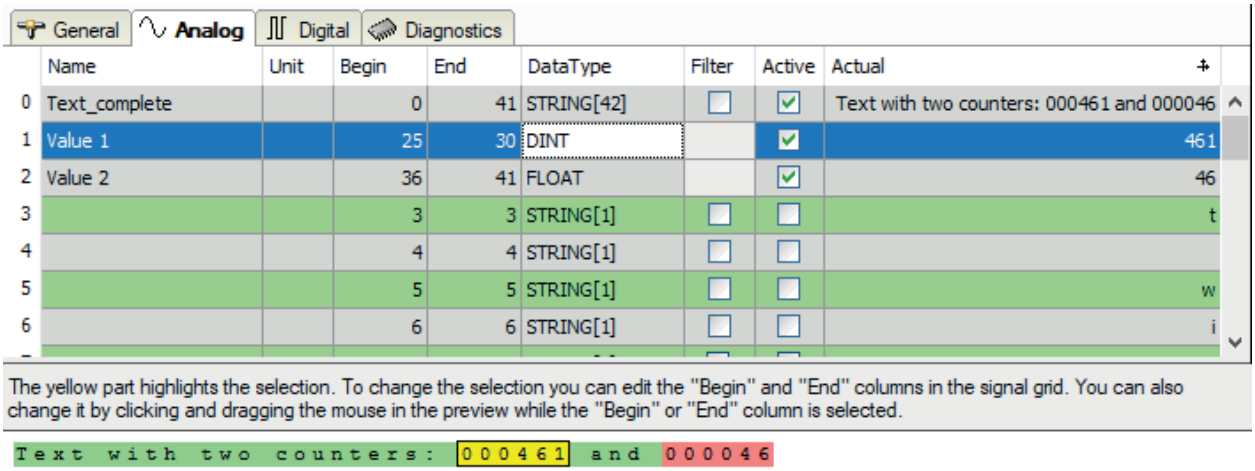
Active connection (TCP Text only)
This option is set to “False” by default. This means that *ibaPDA* waits for a connection to be established by the source computer of the text while listening on the specified port.

Enable this option (true) if you want *ibaPDA* to establish the connection and request data from a host computer. In "Active connection = true" mode, it is necessary to enter a valid IP address or computer name. Entering an alternative address is recommended.

Module structure and processing

See the *Text splitter* module. ↗ *Module settings, page 101*

6.6.3 Signal configuration



In the *Analog* tab, select the desired characters, parts or JSON values for the text signals, depending on the split mode. In the figure, the *Fixed width* split mode is selected.

You can freely assign the signal name. If numbers are to be numerically interpreted in the text, simply select a corresponding data type for the respective text signal.

In the *Digital* tab, there is only one signal that displays whether there is a connection to the text source.

Tip

A good possibility to test the function of the TCP/IP text is using the `TcpIpTest.exe` and `TextSend.exe` help programs, which come along with the *ibaPDA*. This means that process computer presets can be simulated as part of a system test or commissioning even if the original systems are not yet available.

6.7 Text signals via serial (COM) interface

The source text is transmitted via a serial interface to *ibaPDA* (COM1...COM4). You can set the configuration in the I/O Manager under the node *Text interface*. In the *Serial text* module *Analog* tab, the source text with the STRING data type and the text signals derived from the source text with the STRING data type or a different data type are configured. For numerical data types, the conversion is carried out automatically.

Depending on the source text or telegram, a module of the type *Serial text* must be configured.

6.7.1 Add module

You can add a module *Serial text* under the node *Text interface*.

6.7.2 Module settings

Basic settings

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

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

Enabled
Enable the module to record signals.

Name
You can enter a name for the module here.

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

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

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

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

Text encoding
You can select the type of text encoding or the code page here for a correct interpretation and display of the received text data for inputs as well as of the text data to be sent for outputs. Available for selection are, beside system locale according to the Windows system settings (default) and UTF-8 Unicode, all other encodings.

Source

Terminator
This is the agreed termination character for the text, which is expected by *ibaPDA* and which should be used by the transmitting system. The terminator could be any ASCII-character, e. g. Carriage Return or another decimal coded character. You may alter the terminator according to your needs. The default value is 13 (‘carriage return”).

COM interface
Here, select the correct COM interface (COM1...COM4) of the server computer which is connected to the source computer.

Baud rate
Select the appropriate baud rate (600...115200) for data transmission from the selection list. Source-system-dependent.

Data bits and stop bits
Select the appropriate settings from the dropdown list. Source-system-dependent.

Parity
Select the parity (none, odd, even, mark, space) from the selection list. Source-system-dependent.

Character spacing
If you enable this option (true), then the telegram will automatically be closed 500 ms after receiving the last character. This is an alternative to the termination character or is a solution if there is no terminator.

Module structure and processing
See the text splitter module. ↗ *Module settings, page 101*

6.7.3 Signal configuration

Configure the text signals in the *Analog* tab as described for the text splitter modules, see ↗ *Text splitter module, page 100*.

In the *Digital* tab, there is only one signal that displays whether a connection is set to the text source.

6.8 Text signals via text file (file text)

The source text is in a text file (.txt, .csv, .json) that is provided by an external system on a dedicated drive in a defined path (local or network). You can set the configuration in the I/O Manager under the node *Text interface*. In the file text module, *Analog* tab, the source text with the STRING data type and the text signals derived from the source text with the STRING data type or a different data type are configured. For numerical data types, the conversion is carried out automatically.

Depending on the source text or file, a module of the type *File text* must be configured.

6.8.1 Add module

You can add a module *File text* under the node *Text interface*.

6.8.2 Module settings

Basic settings

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

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

Enabled
Enable the module to record signals.

Name
You can enter a name for the module here.

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

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

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

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

Text encoding
You can select the type of text encoding or the code page here for a correct interpretation and display of the received text data for inputs as well as of the text data to be sent for outputs. Available for selection are, beside system locale according to the Windows system settings (default) and UTF-8 Unicode, all other encodings.

Source

Update time
Set the update time here in which*ibaPDA* is to search for and load the specified file in the specified directory.

Default: 1000 ms

Use terminator
If you disable this option (false), then the entire text in the file is used.

If you enable this option, then another line appears for specifying the termination character.

This is the agreed termination character for the text, which is expected by *ibaPDA* and which should be used by the transmitting system. The terminator could be any ASCII-character, e.g., Carriage Return or another decimal coded character. You can alter the terminator according to your needs. The default value is 13 (‘carriage return”).

File path
Enter a valid path and the full file name of the file containing the text in this field. If *ibaPDA* finds the relevant file, it will be immediately read. The file content is interpreted as a source text.

Wildcards are supported in file name. This allows you to use, e.g., a combination of prefix and dynamic part in the file name.

Therefore, the source system generating the text files must ensure that a text file with the source text is copied **in time** into the specified path whenever a new text should be available for *ibaPDA*.

User name, password
When the file path is located on a network drive, enter the login information here for the access right to the drive.

File mode
By selecting the file mode, you specify how to proceed with the file after it has been acquired and read.

- Delete file
  The file is read and deleted immediately. If the same text is to be processed again or another text is to be processed, a new file must be provided again. This setting is suitable if it has been ensured that a file with the correct text will always be available at the right time.
- Keep file
  The file is read and not deleted. The file is read again if its modification date changes. Thus, you can transmit new texts to *ibaPDA* by overwriting the file. This setting is suitable if it is possible or likely that there is not always an up-to-date file and the old text is the "lesser evil" compared to an empty text.

Module layout and processing

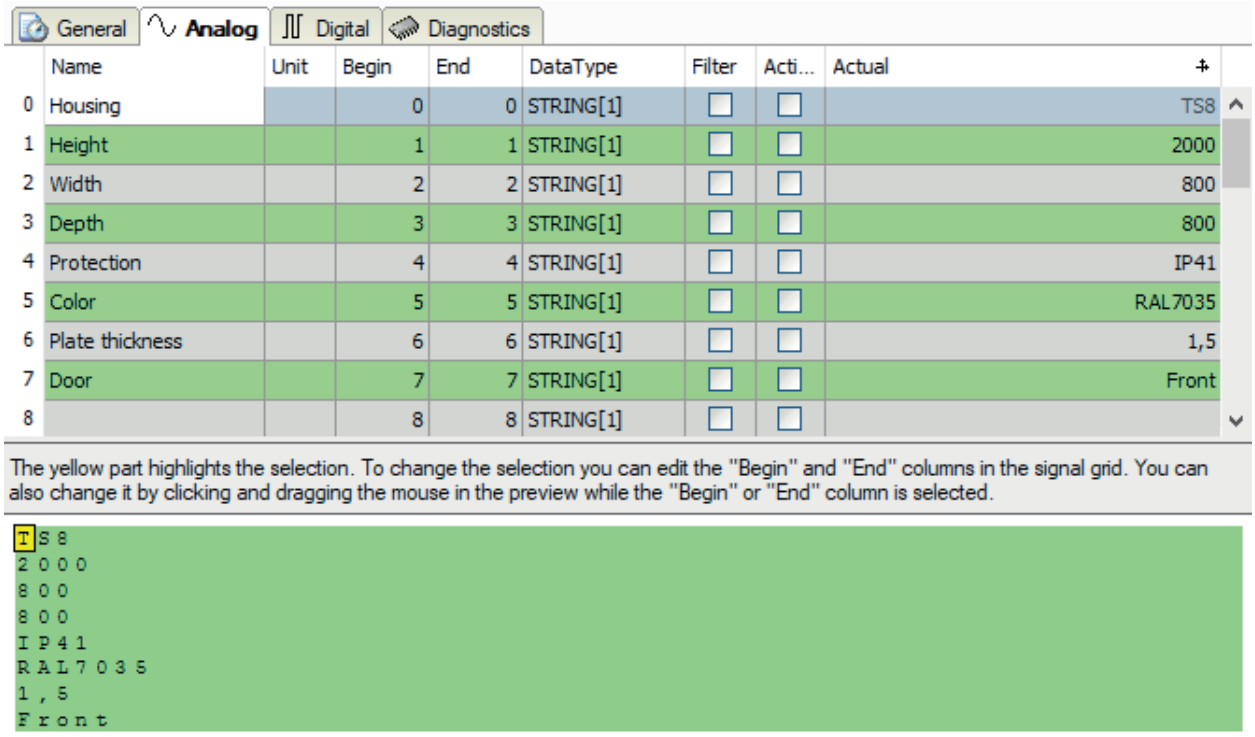
See the text splitter module. ↗ *Module settings, page 101*

6.8.3 Signal configuration

Configure the text signals in the *Analog* tab as described for the text splitter modules, see ↗ *Text splitter module, page 100*.

In the *Digital* tab, there is only one signal that displays whether a connection is set to the text source.

The following figure shows an example for text coming out of a text file in “delimited” split mode.



6.9 Custom text (text creator)

The source text is created on the *ibaPDA* server in the module *Text creator*, tab *Text creation*. It can be a typed in fixed text, the value of a signal or the text of another text signal. It can also be a combination of all three forms. Furthermore, XML and JSON data structures can be used for text creation. You can set the configuration in the I/O Manager under the node *Virtual*. In the *Analog* tab of the same module, you will then configure the derived text signals with the data type STRING or a different data type. For numerical data types, the conversion is carried out automatically.

Depending on the source text, a module of the type *Text creator* must be configured.

6.9.1 Add module

Select the *Analytics* tab in the I/O Manager and add a *Text creator* module under the *Virtual* node.

6.9.2 Module settings

Basic settings

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

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

Enabled
Enable the module to record signals.

Name
You can enter a name for the module here.

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

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

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

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

Format

Mode

The mode determines the format which is used to create the source text.

You can choose between the following options:

- Plain text (default): Create a simple, plain text
- XML: Create an XML structure
- JSON: Create a JSON structure

Compact text (XML/JSON mode only)

If you disable this option (false), then the text of the output signal is formatted with indentation for better readability (as it is done in the preview). If you let this option enabled (true, default), the text is formatted without indentation.

Format Booleans as integers (XML/JSON mode only)

In the XML/JSON format, Booleans are always resolved to *true* or *false*.

Enable this option if you want to use Boolean values as numbers 0 and 1.

Note

Changing the plain text setting to either JSON or XML will remove the configured signals, derived from the plain text. A warning is displayed to prevent unintentional changes.

Module structure (Plain text only)

The number of signals is set to 32 analog signals and 0 digital signals. You can change the numbers, if required. Permissible range: 1 to 1000.

Processing

Split mode (Plain text mode only)

Select the appropriate mode, which corresponds to the structure of the source text in order to read the information correctly:

- Fixed width (description see ↗ *Fixed width split mode, page 103*)
- Delimited (description see ↗ *Delimited split mode, page 104*)
- JSON (description see ↗ *Split mode JSON, page 106*)

Note

The split mode set here applies to the formation of text signals in the *Analog* tab based on the static texts, process signals and text signals defined in the *Text creation* tab. If you select the *Delimited* split mode, for example, then a new part will be recognized whenever a corresponding delimiter appears in the created text.

Remove new lines (Plain text mode only)
If you enable this option (True), then all line feeds will be removed when parsing the source text.

Replace character (Plain text mode only)
Here, you can enter a character which should be used as replacement for non printable characters. Default: x.

Decimal separator (Plain text mode only)
Select the appropriate decimal separator in order to read numbers with decimals correctly: point or comma.

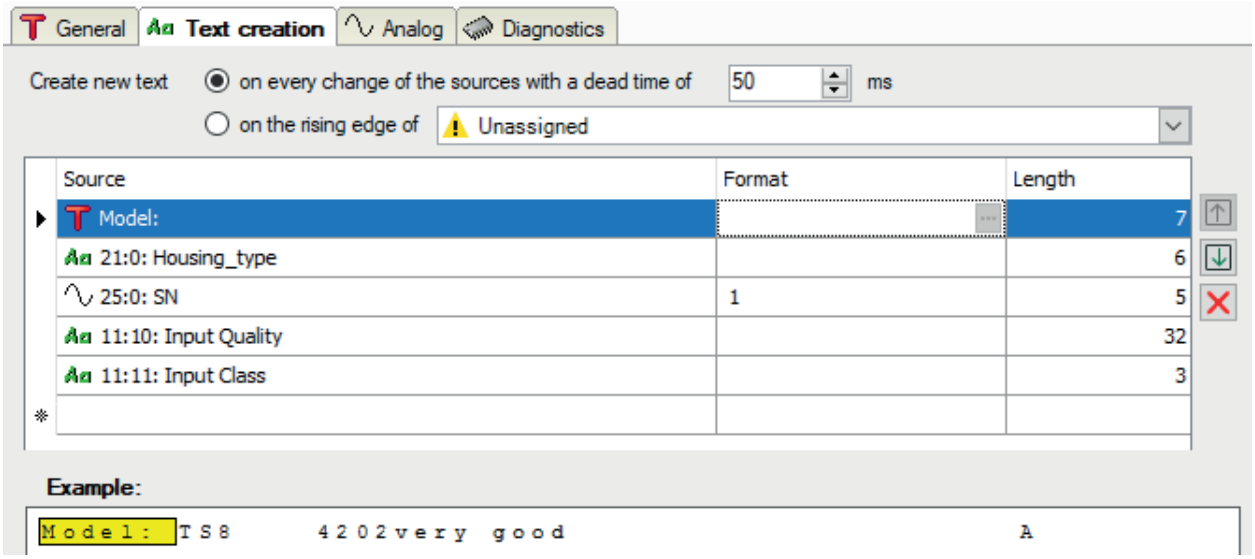
→ Continue with *Plain text* text creation: ↗ *Text creator (Plaintext), page 118*

→ Continue with *XML/JSON* text creation: ↗ *Text creation XML/JSON, page 121*

6.9.3 Text creator (Plaintext)

In the *Text creation* tag, you construct the text for the text signals to be created in the next step.

The following figure shows an example for text creation with static text, text signals and measurement signal.



If you want to use the signal values of other signals in the new text, this must have been defined beforehand.

Above the table, you can define when the new text is to be created.

Create new text in the event of any change…/rising edge…
The choice between these two options determines whether the text is to be considered a new text as soon as its content changes or with the rising edge of a digital signal.

If you choose the first option, then a new text is automatically generated if the values of the source signals change that are part of the text. Changes that occur within the period of the set dead time are ignored.

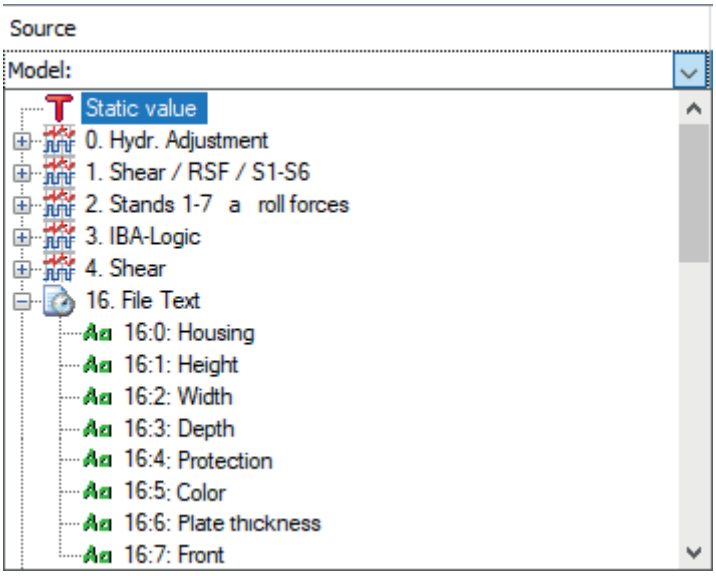
If you intend to control the text creation with a signal, enable the second option and select a digital signal from the dropdown list in the tree structure.

Configuration table for the text
Configure the text using the table. Each row corresponds to a source delivering alpha-numeric data. One or more sources, or rows, can be configured for a text. In total, the resulting text may not contain more than 10,000 characters.

As soon as writing starts in one row, the table automatically expands by a row.

The sequence of the sources (top-down) determines the sequence of their contents in the text (left-right). The arrow keys on the right next to the table allow you to change the arrangement of the sources. If you want to remove a source from the table, select the corresponding row and click on the button ✕.

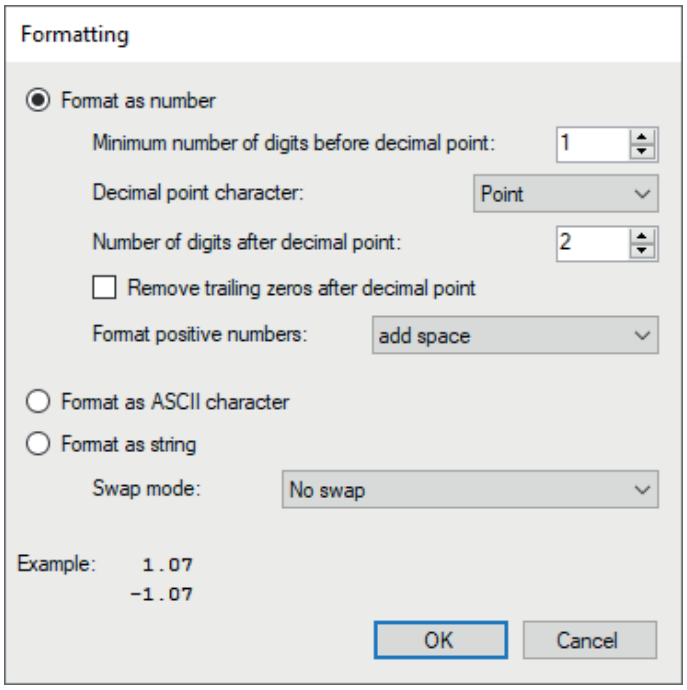
There are different types of sources available. You can select the source type or the source signal in the first column of the table using the dropdown list in the cell.



You have the following options:

Static value
Select this type if you want to enter fixed text, which does not change. Enter the text in the *Source* column. The *Format* column is not relevant. The *Length* column shows the character length of the text. It is updated automatically when you click outside the *Source* column.

Signals from different interfaces
Select any signals from the signal tree to use the values of measured or virtual signals in your text. Optionally, you can specify the format of the signal value. To do so, click the <...> browser button in the *Format* field.



If you like to use the value in decimal format specify the number of digits before and after the decimal point. Consider the possible size of the signal value and allow for enough digits. If negative values are permitted, decide how to deal with positive numbers; either choose from the selection list whether a space or a "+" should be added or no operation is required.

If you enable the Option *Format as ASCII character* option, the signal value will be interpreted as ASCII code and the corresponding ASCII character will be used for the text, e.g., "65" --> "A."

If you select the option *Format as string*, the raw bytes of the signal value are interpreted as characters. A floating point value is 4 bytes which are interpreted as 4 characters. Int32 is also 4 characters and Int16 is 2 characters. You can choose between different swap options. The general result is shown in the example area of the dialog window.

Text signal

Simply select already existing text signals (recognizable by the green Aa icon) if you want to reuse existing texts in the custom text. The *Format* column is not relevant. In the *Length* column, you will initially find the length of the selected text signal (number of characters). You can shorten the length if you only want to use part of the original text signal.

Once you have set up the source(s), you will see the complete text in the preview area under the table.

The preview area only shows the apposition of the text from the sources and does not take any delimiters into consideration. The latter first occurs in the next step, in the *Analog* tab, provided the split mode *Delimited* was selected in the *General* tab.

6.9.4 Text creation XML/JSON

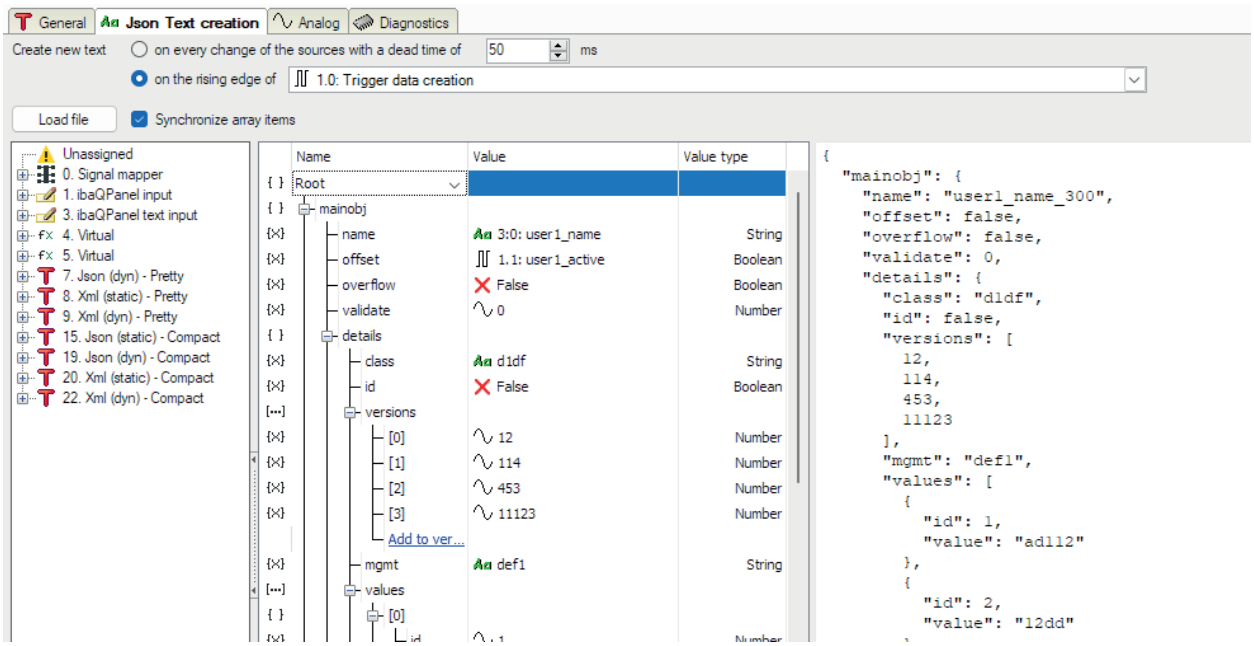
If you select the mode XML or JSON under *Format* in the module settings, then you will get the tab *Xml text creation* or *Json text creation*.

In this tab, you can define the data structure, which is then to be used in the next step as text source.

Note

The module is able to translate XML-specific structures into valid JSON and vice versa. However, it is not recommended to define an XML-targeted structure (e.g. with XML attributes) and then use it as JSON.

The following example explains the text creation with JSON.



On top, you can define when the new text is to be created.

Create new text in the event of any change…/rising edge…
The choice between these two options determines whether the text is to be considered a new text as soon as its content changes or with the rising edge of a digital signal.

If you choose the first option, then a new text is automatically generated if the values of the source signals change that are part of the text. Changes that occur within the period of the set dead time, are ignored.

If you intend to control the text creation with a signal, enable the second option and select a digital signal from the dropdown list in the tree structure.

<Load file> button
If you already have a file (*.json or *.xml) at hand, which you want to use as a template for the data structure, you can import the file over this button.

Synchronize array items
If you enable this option, then all structural changes you make for one array item will be copied to all other items of the same array.

For more information see chapter ↗ *Synchronize arrays, page 128*

Configuration area
The dialog is divided in 3 sections:

- Signal tree (left)

  The signal tree contains all applicable signals, see ↗ *Signal tree, page 126*

- Data structure tree (center)

  In the data structure tree you define the data structure. ↗ *Data structure tree, page 122*

- Preview pane (right)

  The preview shows the result of the data structure. Whenever you change the data structure, the preview is refreshed automatically. The preview does not display signal values. Instead, it shows a combination of signal name and signal ID.

6.9.4.1 Data structure tree

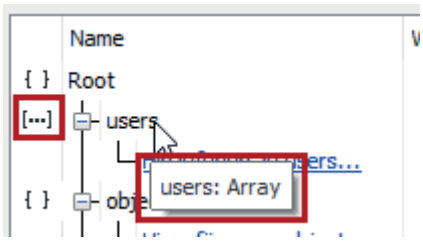
Learn here, how you can set up a data structure.

The data structure is represented by nodes. Depending on the node type more nodes can be nested infinitely in terms of number and depth.

Node types
The following node types are available.

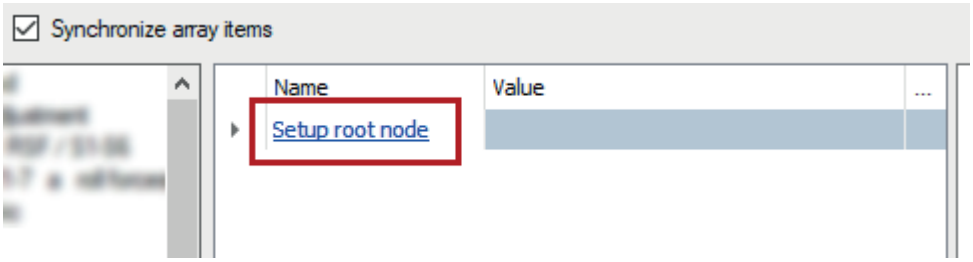
Symbol	Name
{ }	Object
[•••]	Array
{X}	Property
	Attribute (XML mode only)

The node type is indicated by an icon in front of the name or as a tooltip when hovering the mouse over the node.

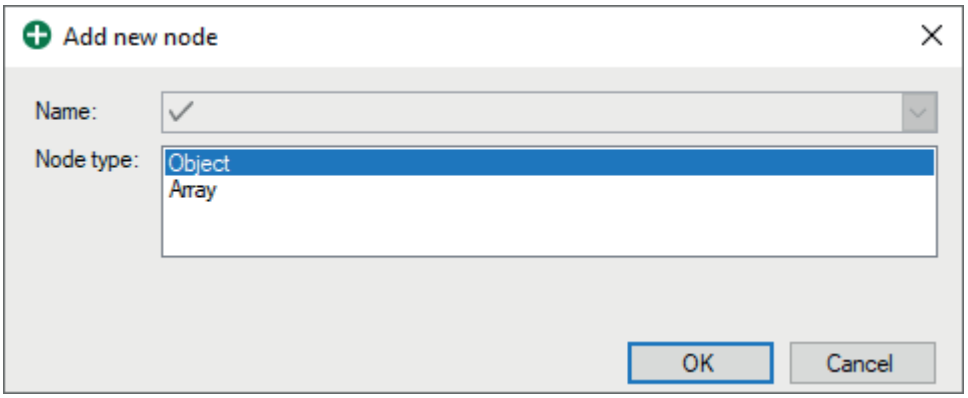


To build a data structure proceed as follows:

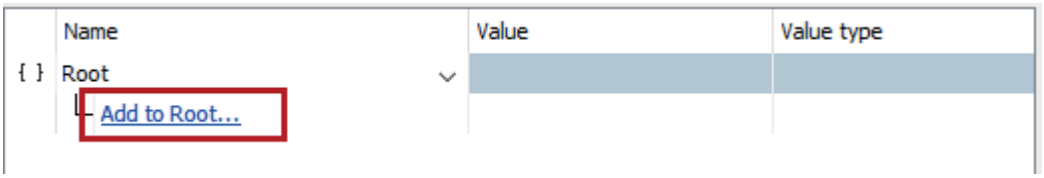
1. Before configuring a data structure the root node must be defined. Therefore, click on *Set up root node* in the *Name* column.



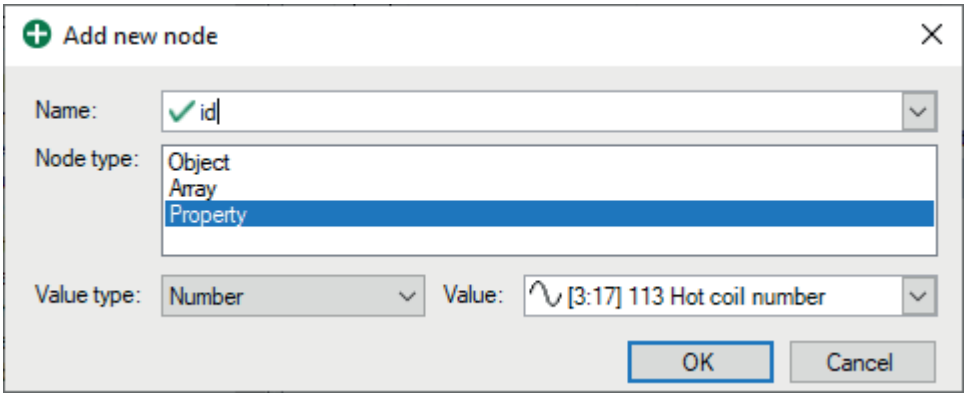
2. A dialog opens, providing the option to either add the root node as an object or as an array. In most cases, the root node is an object. Select *Object*, for example, and click <OK>. You cannot enter a name because the *Name* field is not available.



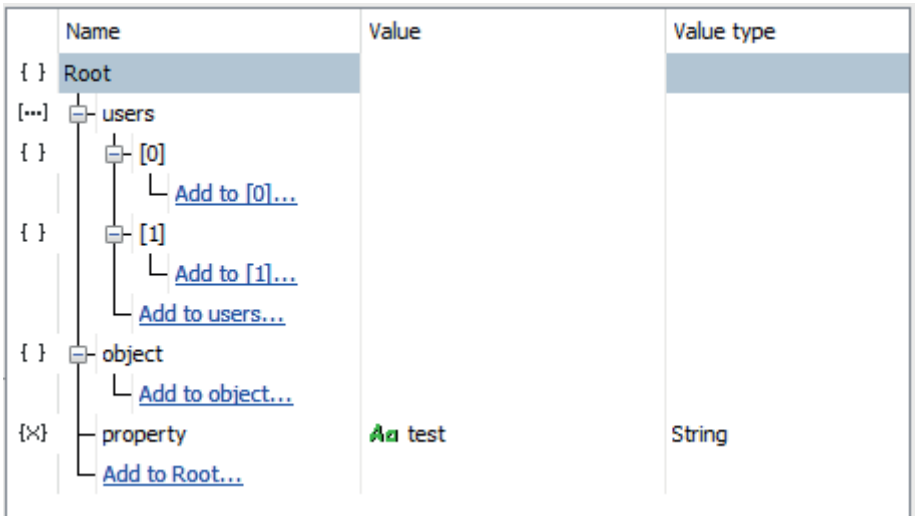
3. Click on *Add to root...* to add more nodes.



4. In the following dialog select the desired node type for each, enter or select the additional information and add the node by clicking on <OK>.



5. In this way, arrange your data structure.



Properties of node types

Note

You can always add nodes of the types *Object*, *Array* or *Property* to objects. In XML mode you can additionally choose the *Attribute* type.

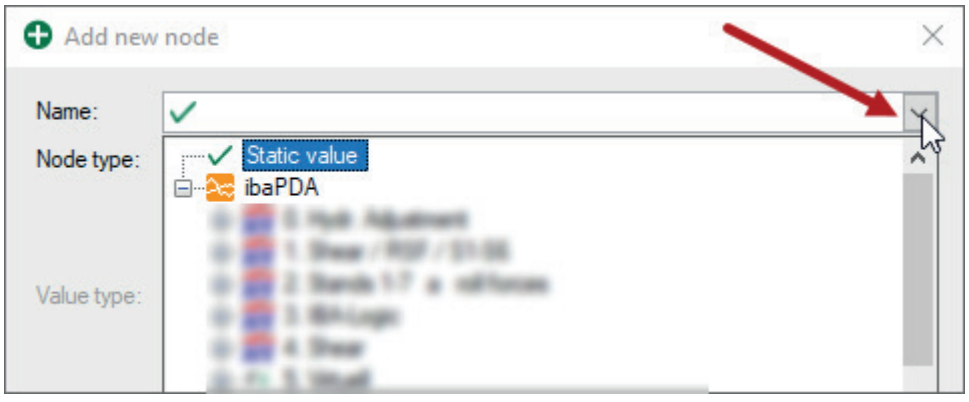
Name

The node name can be a static value or a signal.

Enter the static name manually in the field.

The node names on the same tree level must be unique. If you enter duplicate names, then an error message pops up and the entry is rejected.

If you want to use a signal, then click on the arrow icon for the drop-down list and select the desired signal from the signal tree. The signal name will be used for node name.



Note

Nodes whose parent node is an array cannot have a name. Nodes or items in an array will always be added with their value.

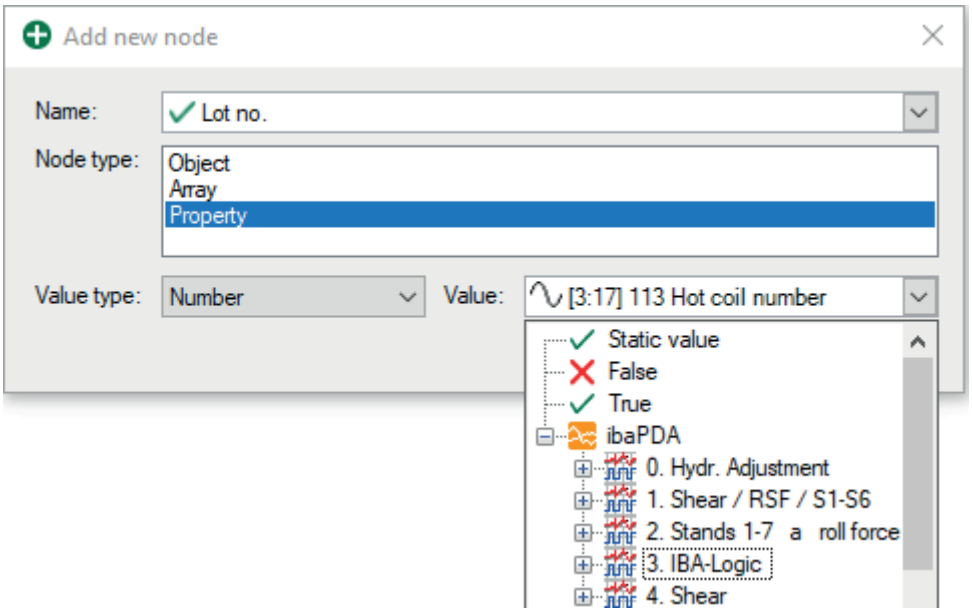
To be able to define a name for a new subordinated node, first add an object to the array and then add the new node to that object.

Value type and value

These two parameters are available only for node type *Property*. *Object* and *Array* do not offer these settings.

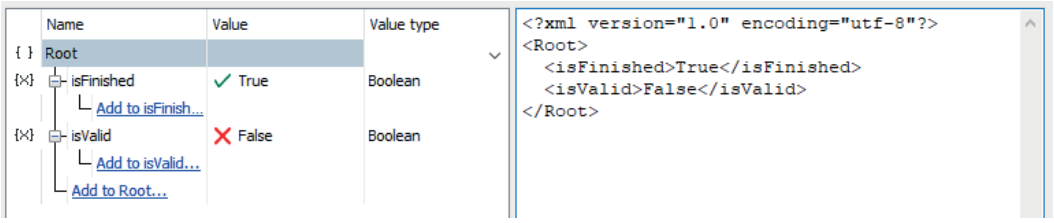
The following value types are available for *Property*:

Value type	Static value	Value from signal
String	Select *Static value* and enter text	Select text signal
Number	Select *Static value* and enter number	Select analog signal
Boolean	Select True or False	Select digital signal



Note

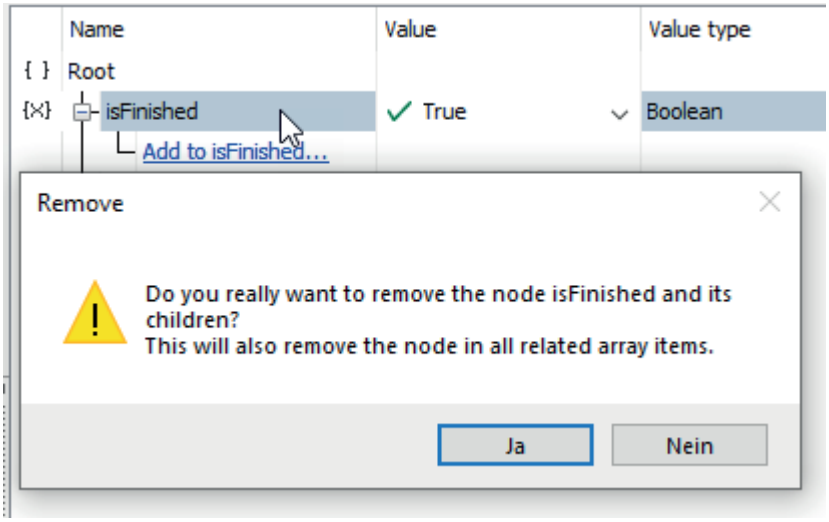
Because XML does not have the concept of data types, Booleans are always resolved to *true* or *false* in the XML mode.



Remove nodes

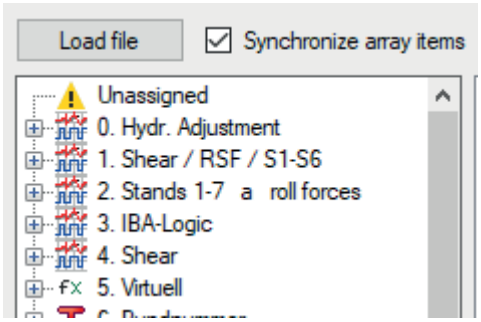
In order to delete a node from the data structure, make a right mouse click on the node in question and select *Remove [x]* in the context menu.

Alternatively, you can press the <Del> key while the node in question is selected in the data structure tree. To prevent unintentional removal a warning pops up.



6.9.4.2 Signal tree

The signal tree in the left pane of the dialog XML/JSON text creation provides all applicable signals for the data structure.



You can add one or more signals by drag & drop. Depending on where you drop the signal from the signal tree, different actions are taken. When dragging the mouse over the nodes, a tooltip shows the action to be taken when dropping the signal.

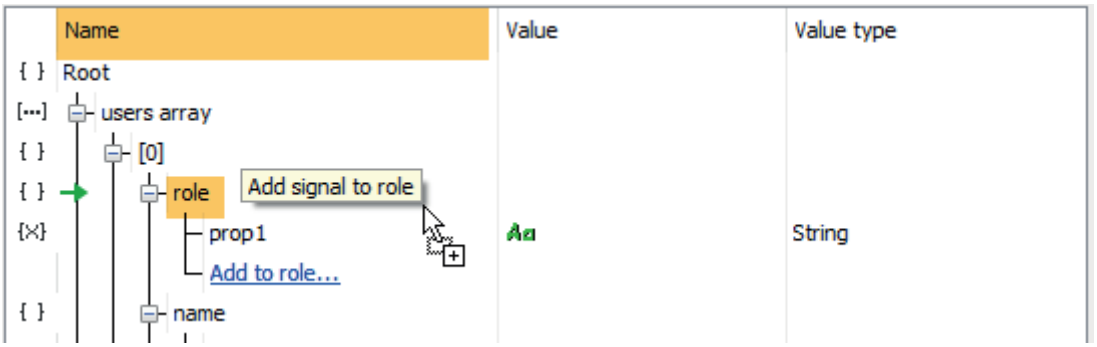
The following table shows the different actions depending on node type and column:

Drop signal on...	... in name column	... in value column
Object	Signal will be added as a new node on the same level.	Value type and value will be replaced by the signal.
Array	Signal will be added as a new node on the same level.	Value type and value will be replaced by the signal.
Property	Name, value type and value will be replaced by the signal.	Value type and value will be replaced by the signal.

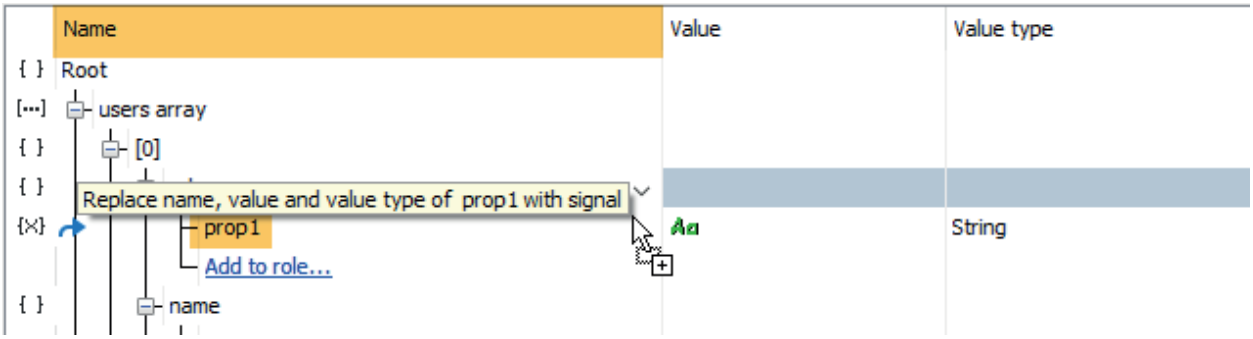
If you select multiple signals in the signal tree at once and drag them on the links *Add to...* or on an object/array, they are added.

Examples

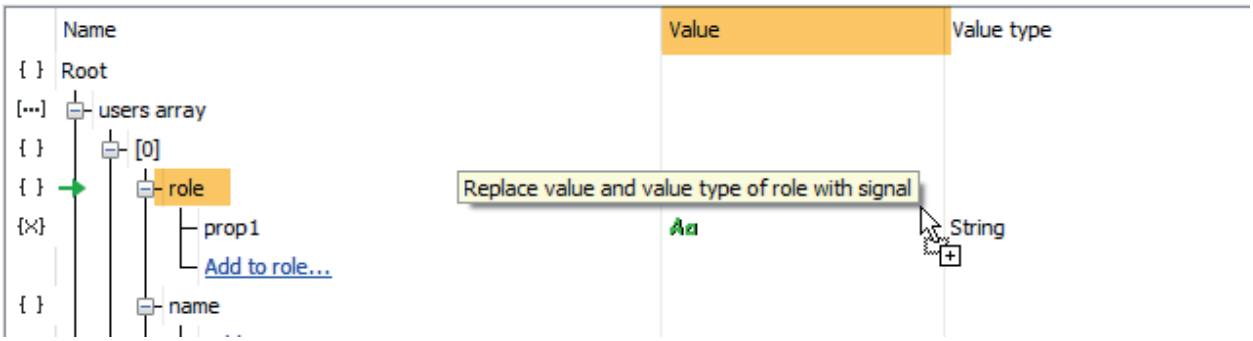
Drag signal on an object, *Name* column



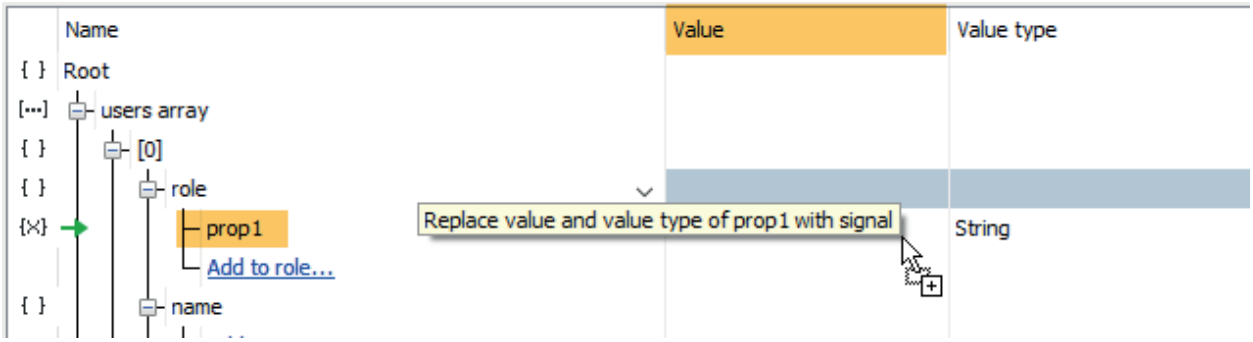
Drag signal on a property, *Name* column



Drag signal on an object, *Value* column



Drag signal on a property, *Value* column

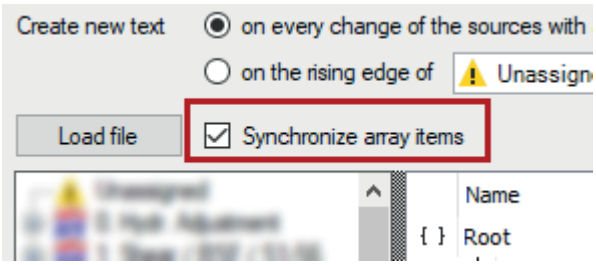


6.9.4.3 Synchronize arrays

When working with larger and more complex data structures, it takes a lot of effort to change the structures of arrays and keep them consistent.

For example, when renaming a property in the first array item, you would also have to rename it in every other array item of the same array to keep the structure the same.

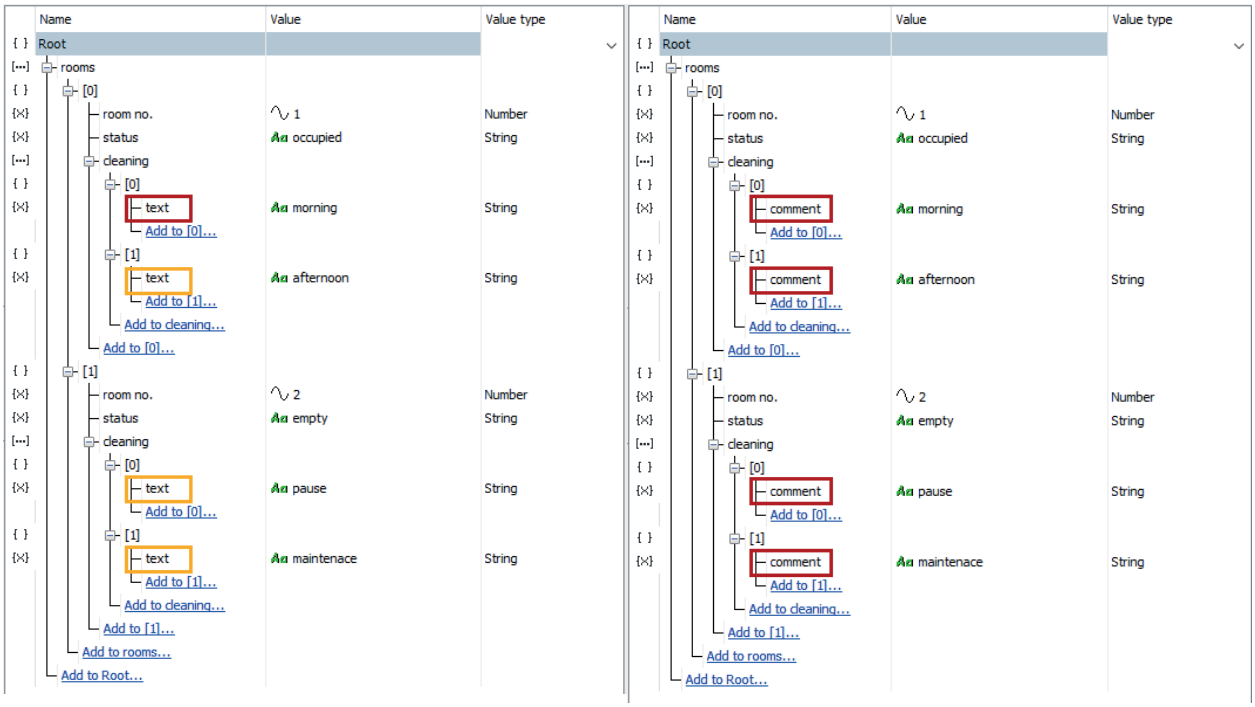
For your convenience, there is the option *Synchronize arrays* in the dialog of XML/JSON text creation.



This function only applies to changes which you currently make. Differently structured arrays and items which are already existing will not be synchronized retroactively by enabling this option.

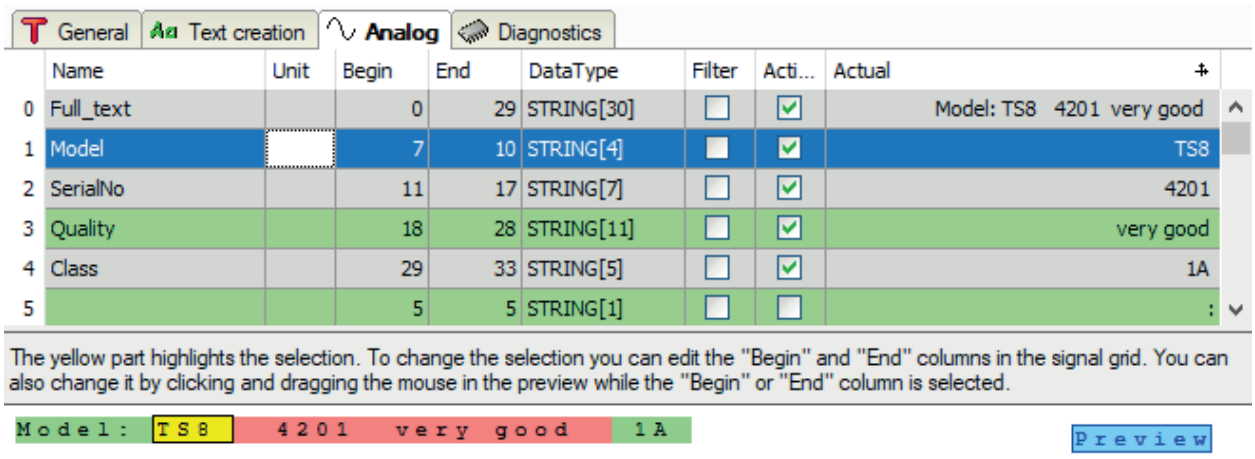
Array items which have been created by the synchronization option can be altered individually after disabling the option.

Example
The following structure shows renaming the node “text” to “comment” in the first array item (left picture). Due to the synchronization option all other corresponding array items are renamed, too (right picture).



6.9.5 Signal configuration

Configure the text signals in the *Analog* tab as described for the text splitter modules, see ↗ *Text splitter module, page 100*.



6.10 Text via ibaQPanel text input

The text can be entered manually via a text input control of *ibaQPanel*. You can set the configuration in the I/O Manager under the node *Virtual*.

A module of the type *ibaQPanel text input* can receive several text signals. In the process, every text signal corresponds to exactly one text input control in the *ibaQPanel* view. Optionally, you can break down these text signals into sections with a *text splitter* module and, if necessary, convert them into other data types.

6.10.1 Add module

You can add a module *ibaQPanel text input* under the node *Virtual*.

6.10.2 Module settings

Basic settings

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

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

Enabled
Enable the module to record signals.

Name
You can enter a name for the module here.

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

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

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

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

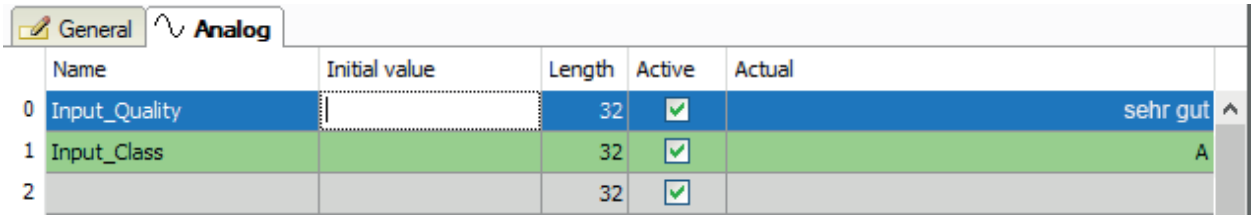
Text encoding
You can select the type of text encoding or the code page here for a correct interpretation and display of the received text data for inputs as well as of the text data to be sent for outputs. Available for selection are, beside system locale according to the Windows system settings (default) and UTF-8 Unicode, all other encodings.

ibaQPanel input

Remember last values
If you enable this option, then in the case of a new I/O configuration, the last known values are loaded instead of the default values *Analog*, Column *Default*).

Module structure
See the text splitter module. ↗ *Module settings, page 101*

6.10.3 Signal configuration



Each line corresponds to a text signal. Each text signal can be assigned a text input control in *ibaQPanel*. In the figure, the *ibaQPanel* text input module has for instance two inputs.

If necessary, you can specify a default text, which is a value in the text signal, if no input has occurred via the *ibaQPanel* element, e.g. “Text input required.”

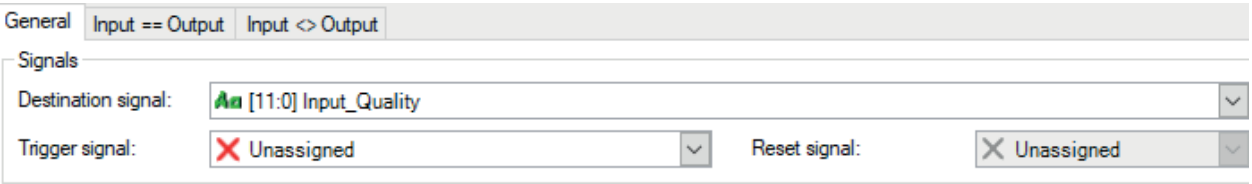
Tip

The default text is also shown in the *ibaQPanel* text input control if you set the text mode “Input == Output” in the properties of the element.

Since the text input control does not have any length restriction for the text, you can define the number of characters in the *Length* column. Before the text input, only the first *n* characters are used as the value for the text input control.

Example for configuration of the ibaQPanel text inputs

Objects in the Qpanel view for the input of quality information. Clicking on <Apply> will apply the input text as a value of the assigned text signal.



The desired text signal from the *ibaQPanel text input* module must be selected as the target signal in the properties of the *ibaQPanel* text input element.

6.11 Text shift register

Using the *Text shift register* module, the last 1 to 64 values of other text signals can be saved. The current value of one or more text signals is applied to the shift register with a trigger signal (rising edge). You can set the configuration in the I/O Manager under the node *Virtual*.

The required text signals must have been created beforehand with the different modules.

The memory depth of the shift register ([1]... [n]) can be adjusted from 1 to 64.

The shift register module provides all tab contents indexed as new text signals (text a [1], text a [2], ... text a [n], with n = depth of the shift register)

The values in the shift register are moved with a trigger signal (rising edge of a digital signal).

All texts contained in the shift register can be used independently of each other, e. g., for display purposes.

6.11.1 Add module

You can add a module *Text shift register* under the node *Virtual*.

6.11.2 Module settings

Basic settings

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

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

Enabled
Enable the module to record signals.

Name
You can enter a name for the module here.

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

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

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

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

Text encoding
You can select the type of text encoding or the code page here for a correct interpretation and display of the received text data for inputs as well as of the text data to be sent for outputs. Available for selection are, beside system locale according to the Windows system settings (default) and UTF-8 Unicode, all other encodings.

Shift register

Source signals
Select the text signal(s) here that are to be processed in the shift register.

Trigger signal
Select the digital signal with the rising edge of which the current value of the source texts is stored and the values in the tab are shifted one position.

Reset signal
Select the digital signal here with whose rising edge all values in the shift register are cleared.

Depth
Select the desired memory depth ([n]) of the shift register. The values can be adjusted from 1 to 64.

Remember last values
If you enable this option, then in the case of a new I/O configuration or when restarting the acquisition, the last known values are loaded instead of the default values (*Analog*, Column *Default* tab).

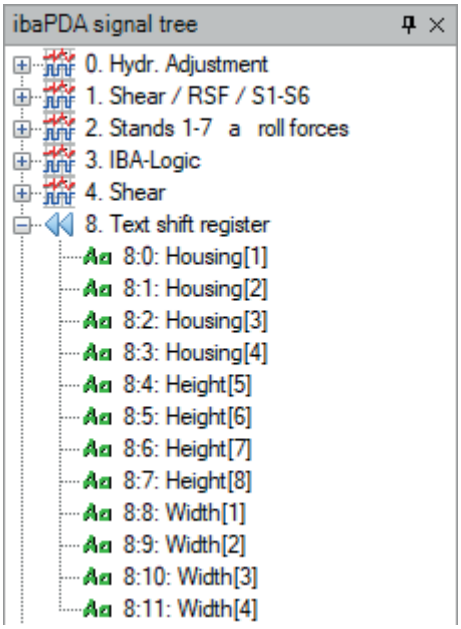
6.11.3 Signal configuration

General | Analog

	Name	Unit	Initial value	Act...	Actual
0	Housing[1]			☑	
1	Housing[2]			☑	
2	Housing[3]			☑	
3	Housing[4]			☑	
4	Height[5]	inch		☑	
5	Height[6]	inch		☑	
6	Height[7]	inch		☑	
7	Height[8]	inch		☑	
8	Width[1]	inch		☑	
9	Width[2]	inch		☑	
10	Width[3]	inch		☑	
11	Width[4]	inch		☑	

The number of resulting text signals is determined by the number of source signals and the register depth. The figure shows the text signals of a text shift register with a depth of 4.

The text signals provided by the text shift register are available in the signal tree:

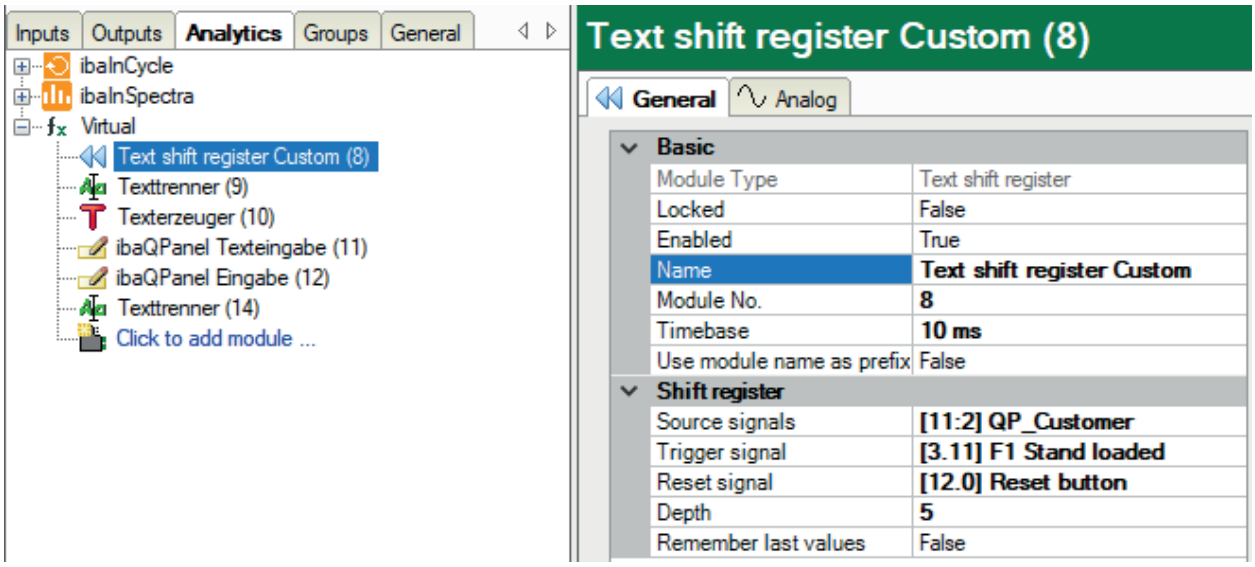


Tip

For analog signals, there is a corresponding function with the "Shift register" module in the *Virtual* node, *Hardware* area in the I/O Manager.

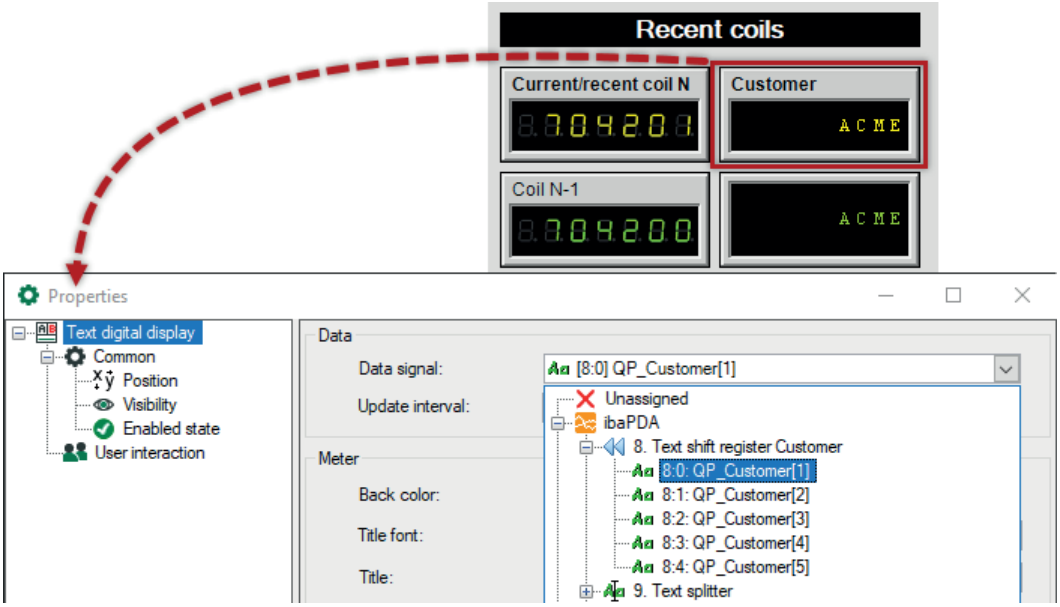
6.11.4 Application example for text shift register

For a facility that produces steel sheets for the automotive industry, an on-screen display is to be made with *ibaQPanel* showing the customer names of the last 5 coils produced in addition to the batch numbers. The “customer” text signal is created in a text shift register. The already existing "QP_Customer” text signal always contains the appropriate customer name for the production. This is manually entered into *ibaQPanel* but it can also be any other automatically-generated text. The trigger signal is fired each time a new coil enters the plant. Since the last 5 coils are supposed to be displayed, the register depth is set to 5.

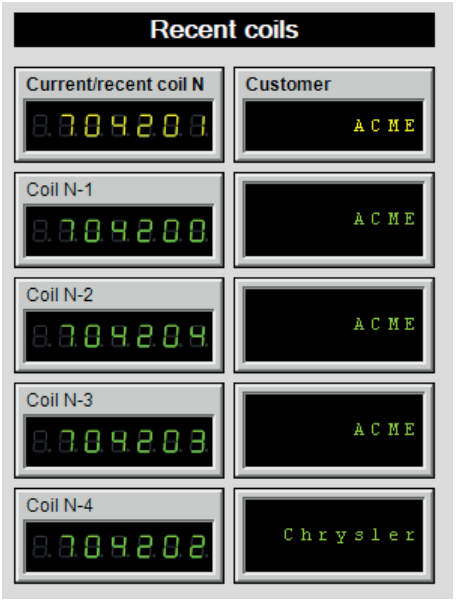


Five digital text displays are configured in *ibaQPanel*, each of which accesses an indexed shift register text signals.

[1] = latest register, [5] = oldest register



The result: The customer names for the last 5 coils now appear in the right row. The most recent or current coil is at the top. The left row was implemented with digital numeric displays and analog signals in the (numeric) shift register module.



6.12 Text signals via OPC / OPC UA

The source text is transmitted to *ibaPDA* via the PC network (Ethernet TCP/IP) by an OPC or OPC UA server. You can set the configuration in the I/O Manager under the node *OPC* or *OPC UA*. In the *OPC client* or *OPC UA client* module, *Analog* tab, the source text is configured with the data STRING. To divide the text from the OPC tag onto individual text signals, a module *Text splitter* must be configured per source text under the *Virtual* node.

6.13 Text signals via other interfaces

Other documentation

Text signals are supported for a number of additional interfaces. For a description of the settings, please refer to the manual for the interface product.

- Reflective Memory
- HiPAC Request
- ScramNet+
- SIMATIC TDC/SIMADYN D
- EtherNet/IP
- S7-Xplorer
- B&R-Xplorer
- TwinCAT-Xplorer
- Codesys-Xplorer
- Logix-Xplorer
- Generic TCP
- Generic UDP
- TwinCAT Request
- Modbus TCP Server
- Sisteam TCP
- TDC TCP/UDP
- VIP TCP/UDP
- S7 TCP/UDP
- ibaVision

7 Outputs

In the *Outputs* tab of the I/O Manager, you have the option to output digital and analog signals. These output signals are primarily intended to signal events, or to trigger alarms or alerts. Since *ibaPDA* offers a wide range of process signals, signals can also be compared, computed and issued to other systems.

Below you will find a list of the interfaces that can be used for digital and analog outputs and their corresponding output modules.

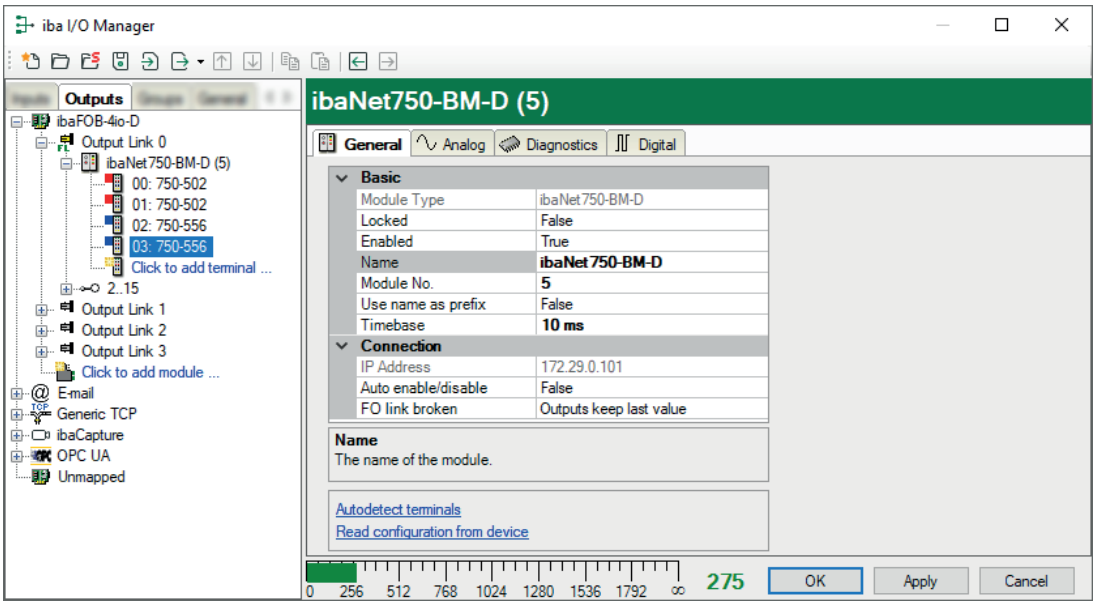
Interface	Output modules	Comment
ibaFOB-io-S, -X, -D/-Dexp (incl. FOB-2io-, 4i+4o) ibaFOB-io-ExpressCard	FOB Alarm ibaNet750-BM	Only available when the card is inserted FOB-link in 3 Mbit mode
ibaFOB-io-D/-Dexp (incl. FOB-2io-, 4i+4o) ibaFOB-io-ExpressCard ibaFOB-io-USB	ibaPADU-S-CM ibaPADU-S-IT-2x16 ibaPADU-S-IT-16 HAICMON CMU ibaCMU-S iba PQU-S ibaBM-DP ibaBM-ENetIP ibaBM-PN ibaLink-io-embedded ibaLink-VME ibaNet750-BM/-BM-D ibaFOBalarm	Only available when the card is inserted FOB-Link in 32 Mbit Flex mode FOB-link in 3 Mbit mode
ibaCom-L2B-x-8	L2B Integer I/O L2B Real I/O L2B S7 Real I/O	Only available when the card is inserted
ibaCapture	ibaCapture	Only available with ibaCapture license; for recording control and PTZ cameras
	ibaVision output ibaVision V2 output	for the data supply of ibaVision
ibaNet-E	ibaW-750 + terminals ...	Standard, always available, but appropriate devices required
OPC (standard)	OPC output module	Standard, always available

Interface	Output modules	Comment
OPC UA	OPC UA client OPC UA server	Only available with license OPC UA client
E-mail	E-mail	Standard, always available up to 256 configurable e-mails
Generic TCP	Generic TCP Output	Only available with ibaPDA-Interface-Generic-TCP license; passive mode is supported.
Generic UDP	Generic UDP Output	Only available with ibaPDA-Interface-Generic-UDP license; passive mode is supported.
EtherNet/IP	EtherNet/IP I/O Scanner EtherNet/IP I/O module	Only available with ibaPDA-Interface-EtherNet/IP
Reflective memory	Reflective Memory Output	Only available with RM board and ibaPDA-Reflective Memory Access license
Modbus TCP Client	Modbus client	Only available with ibaPDA-Interface-Modbus over TCP-Client license
ABB-Xplorer	ABB Xplorer MMS	Only available with ibaPDA-Interface-ABB-Xplorer license
OMRON-Xplorer	OMRON FINS	Only available with ibaPDA-Interface-OMRON-Xplorer license
S7-Xplorer	S7-Xplorer	Only available with ibaPDA-Interface-S7-Xplorer license
TwinCAT-Xplorer	TwinCAT PLC	Only available with ibaPDA-Interface-TwinCAT-Xplorer license
MQTT	MQTT	Only available with ibaPDA-Interface-MQTT license
SQL database	SQL command	Only available with at least one of the following licenses: ibaPDA-Interface-MySQL, -Oracle, -PostgreSQL, -SAP-HANA or -SQL-Server

Table 8: Interfaces with output function and corresponding output modules

Note

The calculation timebase is a time base specifically for the calculation of output values. You can set this time base independently of the general timebase and the timebase of the input modules. Output values are sent with a low priority and can always be delayed or displaced by the higher prioritized tasks of the measured value acquisition, since they represent the core function of *ibaPDA*. The shortest achievable cycle time of the outputs is independent of the calculation timebase and is 50 ms or equal to the smallest common multiple of all input time bases. Check the current output cycle time of the system in the I/O Manager under *General* tab, *Module overview* node.



7.1 Create outputs

The basic steps to configure alarms and output signals are:

- Installation of a suitable interface card in the *ibaPDA*-server PC and/or licensing of a suitable interface.
- Adding the appropriate output module in the I/O Manager
- Definition and activation of digital and/or analog output signals

The configuration of the output signals follows the same concept as for the measured signals.

7.2 Adding output modules

The method of adding output modules is basically the same for all types. Only the module and signal settings vary.

If the required hardware is installed (see above), follow these steps:

1. Open the I/O manager and select the *Outputs* tab.
   The tree structure shows an overview of the available data interfaces with their output links.

2. By *Click to add module ...* you add a new module to the respective interface.
   A dialog opens where you can name the new module. Select the appropriate module type.
3. Enter a name for the module and press <OK> to close the dialog.
4. The new module will be added to the next free link of the interface.

Of course, you can also right-click on the link of your choice in order to add a module via *Add module* in the context menu.

You can now configure the module settings and signals.

7.3 Setting up output modules

7.3.1 General module settings

The general module settings are usually defined when adding a module in the *Inputs* area in the I/O Manager and mostly apply to input and output modules.

Nevertheless, there are some specific settings that are different for the outputs. In the basic settings, for example, there is the "Calculation time base" instead of the "Time base".

Basic settings

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

Locked
A module can be locked in order to prevent change of module settings by accident or unauthorized users.

Enabled
Disabled modules are excluded from the signal acquisition.

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

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

Calculation timebase
Timebase (in ms) for the calculation of the output values.

The calculation timebase and update timebase are NOT the same!

For interfaces whose modules have an input and an output side (same module number), the calculation timebase is identical to the module timebase on the input side. Hence, changing the calculation timebase also changes the module timebase on the input side and vice versa.

For interfaces with separate input and output modules (different module numbers), the calculation timebase can be changed independently.

The smallest possible calculation time base, i.e., the fastest possible update of the outputs, results from the smallest common multiple of all module time bases, and is at least 50 ms. .

This setting does not exist for the E-mail, OPC output and SQL command output modules.

Minimum output timebase (information only)
Smallest timebase for updating the outputs.

This value is calculated automatically by the system considering the current I/O configuration. It is displayed for information only. The minimum output timebase depends on the least common multiple of all module timebases, with a minimum of 50 ms.

Use name as prefix
Puts the module name in front of the signal names.

Module structure

Number of analog/ digital signals
Defines the number of configurable analog and digital signals in the signal tables. The default value is 32 for each. You can change the number. The maximum value is 1000.

7.3.2 OPC output module

The OPC output module can be used to write values in OPC signals provided by an OPC server. The OPC output module in *ibaPDA* is an OPC slave.

7.3.2.1 General tab

It is not possible to set the time base, since this results from the smallest common multiple of all module time bases and is at least 50 ms. Higher time bases will be calculated automatically.

In the *General* tab, you should

- select the OPC server computer
- select the OPC server (service)
- enter user account data (if required)
- enter a group name (optional)

For notes on the general module settings see ↗ *General module settings, page 141*.

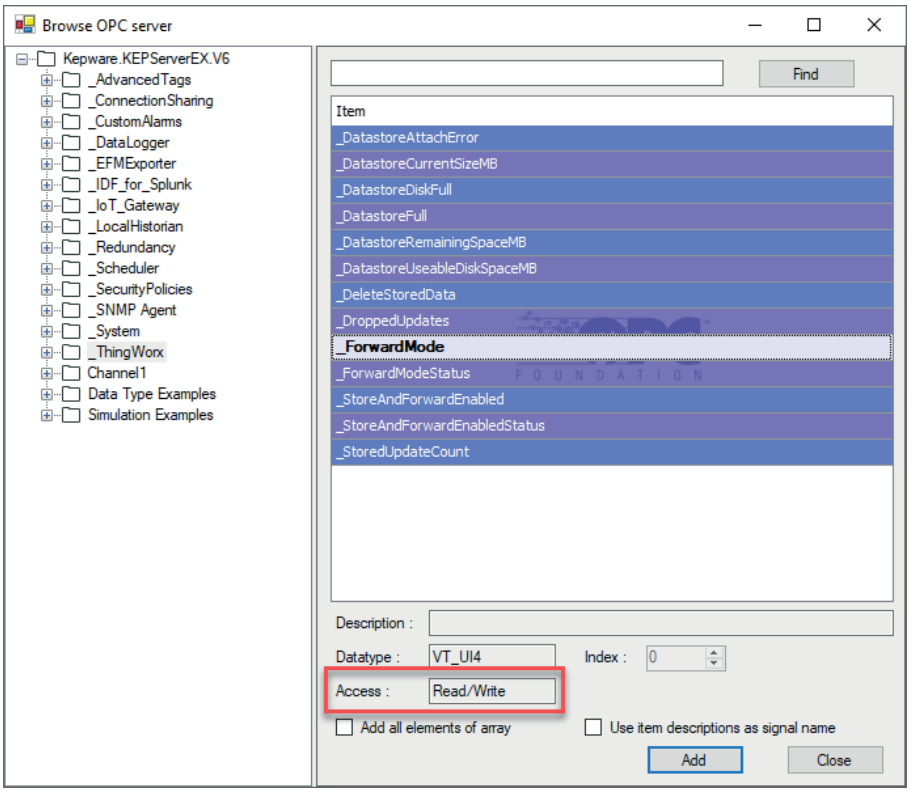
In addition, you can still configure the following settings:

Force data type
If this option is enabled (True), then the analog signals are requested with data type R4 (4 byte floating point value) and the digital signals with data type BOOL. Use this option if the OPC server can dynamically change the data type of its signals.

Do initial read
When this option is enabled (True) *ibaPDA* will wait until the first read of data from the OPC server is finished before the acquisition starts.

Add item attempts
Certain OPC servers do not accept items until they have loaded their complete configuration. Via this setting you can determine how often *ibaPDA* should try to add the requested OPC items to the group. There is a waiting time of 1 s between two attempts.

After you have made the settings, click on the blue hyperlink *Connect* at the bottom of the tab. As soon as the connection is established, you can click on the available hyperlink *Add signals* in order to browse the OPC signals. Only items with read/write or write access apply for selection. Selecting OPC items at this stage will put them automatically into the correct signal table (analog/digital).



7.3.2.2 Analog and Digital tab

In the signal tables on the *Analog* and *Digital* tabs, you can define the output signal in the *Expression* column.

The result of the expression will be written to the item of the OPC server selected in the "*Item ID*" column next to it.

For selection of OPC items, you can open the browser also from the *Item ID* column.

7.3.3 Email

Using the email output interface, emails can be sent each time a trigger signal is triggered and whenever acquisition is started and stopped.

Therefore, users can be notified automatically, e.g., in the case of an alarm. You can also use current signal values and text in the email as well as send attachments (still images from an iba-Capture system or other files).

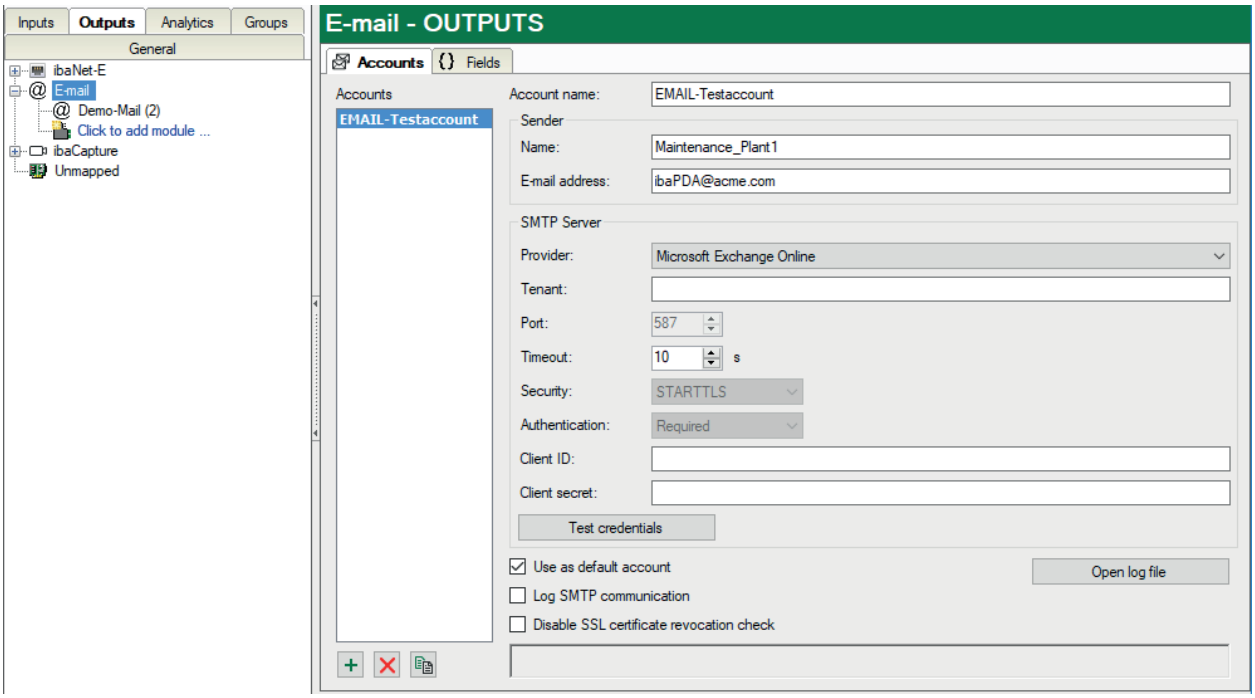
The email interface is part of the standard scope of *ibaPDA* and does not require an additional license. *ibaPDA* has an integrated email client that can connect to any SMTP server on the network or the Internet. The system supports TLS communication to SMTP servers as well as OAuth 2.0 authentication.

The interface supports up to 256 different emails that use different accounts and can be addressed to various addressees and/or have different content. Various emails can be sent with the same trigger signal or individual trigger signals. A module must be configured for each email.

The accounts, as well as some placeholder fields, must be set up on the “Email“ interface node beforehand to transmit selected signal values.

7.3.3.1 Basic settings (accounts and fields)

Accounts tab (interface)



In this tab, you configure the email accounts to be used by *ibaPDA* to send emails.

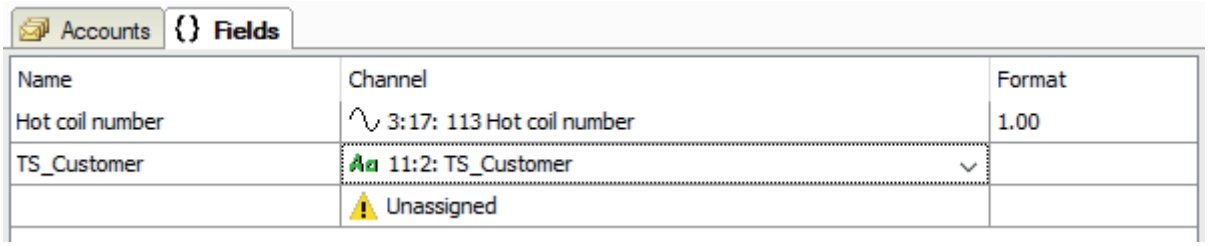
Add an account using the button + and enter the required information. If available, multiple accounts can be configured and used. Which account will be used for which email is determined when you configure the email module. You can duplicate an account in the list by using the copy button. This is useful if, e.g., you want to set up different senders.

If you want to configure a standard SMTP server, select "Other" in the *Provider* field. Alternatively, you can select other servers, like for instance *Microsoft Exchange Online*. The input mask changes accordingly to enable configuring the selected provider. For such providers *ibaPDA* supports the authentication protocol OAuth 2.0.

Optionally, you can declare an account as the default account so that this account is automatically entered when a module is added.

You can also enable logging of SMTP communication for each individual account.

Fields tab (interface)



In this tab, you can define fields for signal values (numeric and text) that can be used as placeholders in emails. Thus, with the fields, current analog and digital signals or calculated values can be inserted into the message. Accordingly, you can insert values of text signals into the message.

First enter the name of the desired placeholder in the *Name* column. Then you can open a drop-down list in the *Channel* column, from which you select the desired signal or text channel.

In the numeric fields, you can even specify the number format in the *Format* column. Click on the <...> button to open the settings dialog.

The names of the fields must **not** contain the following characters:

- Square brackets **[ ]** (indicate a placeholder)
- **IT** at the beginning of the name (reserved for image triggers, see below)
- **DS** at the beginning of the name (reserved for data records, see below)

Names may not be given twice.

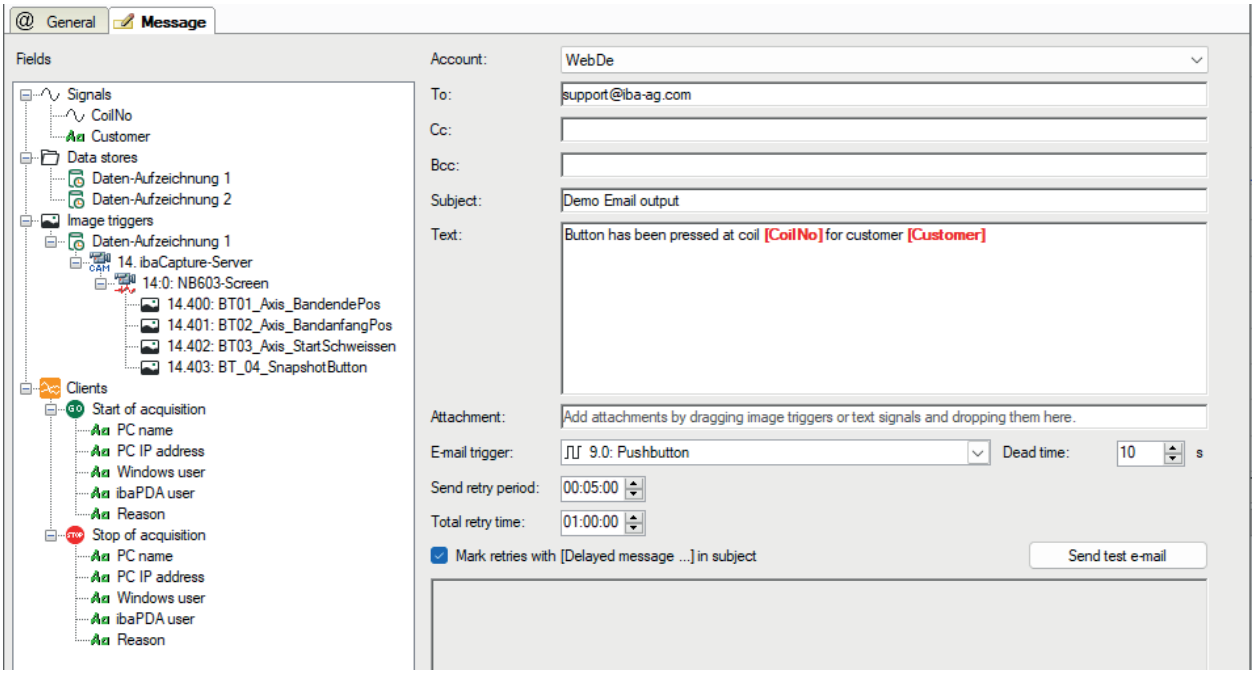
7.3.3.2 Email module settings

General tab

For email modules, only the standard information must be entered in the general module settings.

For notes on the general module settings see ↗ *General module settings, page 141*.

Message tab



In this tab, you will create the actual message.

In the window on the left, the fields for the signals defined in the *Fields* tab are displayed in the form of a signal tree.

Below this, all projected data stores and image triggers are displayed. These data records and image trigger fields always refer to the path of the last data file or the last images.

These data stores and image trigger fields always refer to the path of the last data file or the last images.

Select the desired account and enter the standard information such as *recipient*, *CC* and *subject* as well as the message text. You can also use fields with text signals in the lines *To*, *Cc* and *Bc*, and can thus enter the addressees dynamically. For example, if an email is to be sent to a different recipient on weekdays than on weekends.

For the *email trigger*, select whether the sending of email should be triggered at the start or stop of acquisition or by a digital signal.

A dead time (default 10 seconds) prevents the email from being sent multiple times should the trigger signal be triggered several times in quick succession.

With the *Send retry period* and *Total retry time* settings, you can determine the intervals at which, and over how long in total, a failed transmission of an email is repeated.

If you enable the option *Mark retries with [Delayed message...] in subject*, then this note is automatically inserted in the subject line if the email could only be sent after several failed attempts.

With the <Send test email> button, you can send the email without the trigger signal being triggered or the measurement running. If the measurement is not running and the configured fields have no values then the names of the fields are entered in the email.

Using the fields

In addition to static input, you can add fields as placeholders in various places, thus dynamically shaping your email.

The following table shows where you can add which fields and what impact this has.

Field	Use in...	Result when sending the email
Analog and digital signals	Subject	Value is applied
	Message text	Value is applied
	Attachment	Value is applied and interpreted as part of a path or file name. If the file exists, it is attached.
Text signals	To/Cc/Bcc:	Text is applied
	Subject	Text is applied
	Message text	Text is applied
	Attachment	Text is applied and interpreted as part or all of a path or file name. If the file exists, it is attached.
Data storages	Subject	Name of the data file (full path) is applied.
	Message text	Name of the data file (full path) is applied.
Image trigger	Subject	Name of the image file (full path) is applied.
	Message text	Name of the image file (full path) is applied.
	Attachment	Image file is attached
Clients	Message text	The information of the selected fields referring to the last start and/or last stop of the acquisition is applied.

Table 9: Use of fields / placeholders in email

A field can be added at the desired location in the entry fields, *Subject*, *Text* and *Attachment*, in the following ways:

- By double-clicking on the field; the field will be inserted in the text at the cursor position
- By dragging and dropping
- By manual entry of the field name in square brackets

The fields are highlighted in the text with red, bold characters to indicate that these points are later replaced by actual values.

Data storages cannot be added as an attachment as data files are often too large for emailing and have not yet been concluded by *ibaPDA* when the email is sent.

If you want to send multiple attachments, they must be separated by semicolons.

Entries in the *Attachment* field must always be the full path and file name of the files that you want to attach.

The path and file name can be formed in different ways:

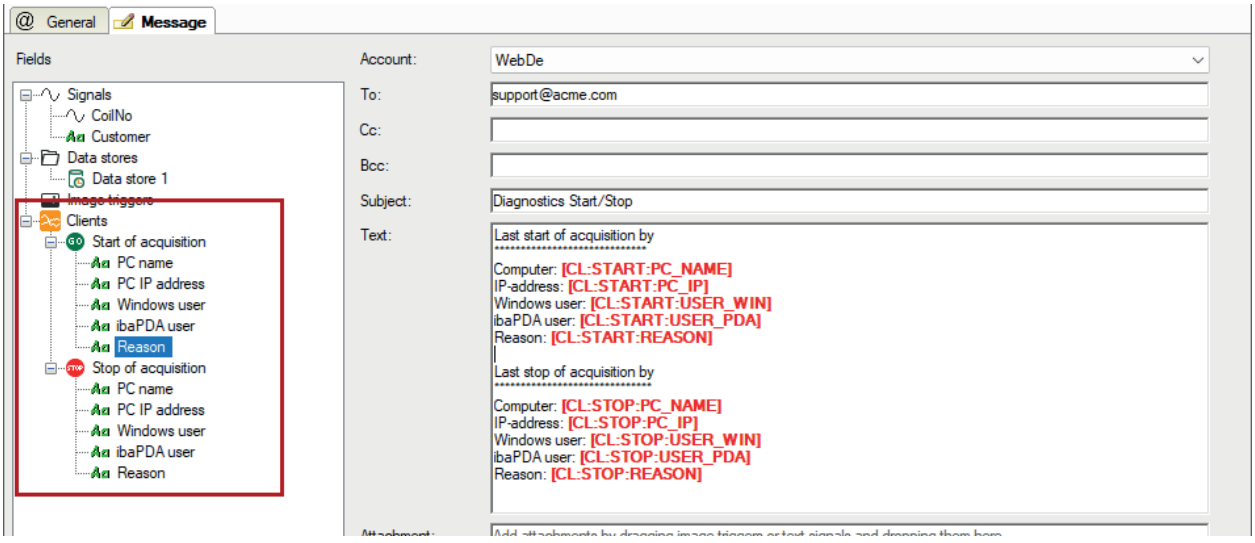
Type	Example	Comment
Entered fully without placeholders	D:\xyz\agb.pdf	Only useful if the path and file-name never change.
Only a text field	[Report] [Imagetrigger_ABC]	Full path and file name must be included in the text box. (Automatically the case for image triggers.)
Combination of static text and text field	\\DATASRV\reports\[Shiftreport]	Text field only contains the file name.
Combination of text and numeric fields	[Destination] [Prefix] [Number] [Extension]	[Number] is a numeric field with a consecutive number, [Destination], [Prefix] and [Extension] are text fields.
Combination of static text and text and numeric fields	D:\abc\[PathA]\[Reportname]_[No].pdf	-

Table 10: Examples of entries in the attachment line

Example for a message about the last start and last stop of acquisition

The message can be sent at the time when the acquisition was started or stopped, in order to send an immediate notification, or at any time.

Dedicated fields related to the client where the last start or stop of acquisition was initiated provide more detailed information.



- PC name: Name of the client computer
- IP address: IP address of the client computer
- Windows user: Windows user name that was used to log into Windows on the client
- ibaPDA user: *ibaPDA* user name that was used to log into *ibaPDA*
- Reason: Reason, why the acquisition started or stopped

Possible reasons are:

- Interactive start (start by user operation)
- Interactive stop (stop by user operation)
- Interactive apply new configuration (I/O configuration changed and applied by user)
- Automatic start (e. g. when a disconnected PLC returns)
- Start after remote configuration
- Start triggered by other synchronized system (e. g. in a Multistation network)
- ibaPDA service is shutting down
- PC is going to sleep (computer switches to power saving or standby mode)
- Error occurred (unknown error)
- License changed (license container disconnected or license changed)

7.3.4 ibaNet750-BM, -BM-D output modules

For each link of an ibaFOB output card, you can add one ibaNet750 module with up to 8, 16 or 32 analog and digital output signals, depending on the address method selected.

When using a ibaFOB…-D card or ibaFOB-io-ExpressCard, the device ibaNet750-BM-D can also be used in 32 Mbit flex mode.

7.3.4.1 General tab

Basics

For notes on the general module settings see ↗ *General module settings, page 141*.

Note

You may add an ibaNet750 module either on the input side (Inputs tab) or on the output side (Outputs tab) and add input terminals as well as output terminals. The module number of the ibaNet750 module will always be the same on both sides.

Depending on the connection mode of the module (see description below), the module is available on just one or both sides.

ibaNet750

Address mode
The address mode setting must comply with the switch setting (S1) on the device.

Mode	Switch positions on device	Addresses	Number of analog and digital signals
1	1– 8	1– 8	8/8
2	C – F	1 - 2, 3 - 4, 5 - 6, 7 - 8	16/16
4	A, B	1 - 4, 5 - 8	32/32

Table 11: ibaNet750 address modes

ECO mode
The ECO mode setting (TRUE or FALSE) must comply with the switch (S2) position on the device.

Connection mode
With the connection mode setting, you can control the mapping of the input and output fiber optical links of an *ibaFOB* card to an *ibaNet750-BM* device.

Mode	Display	Comment
I	Inputs	Module is only visible in the *Inputs* tab of the I/O manager, even with output terminals, if configured. Module is not visible in the *Outputs* tab. FO output link is not mapped to *ibaNet750-BM* device.
O	Outputs	Module is only visible in the *Outputs* tab of the I/O manager, even with input terminals, if configured. Module is not visible in the *Inputs* tab. FO input link is not mapped to *ibaNet750-BM* device.
IO	Inputs and Outputs	The module will be visible in the *Inputs* and *Outputs* tabs. FO input and output links are mapped to the *ibaNet750-BM* device.

Table 12: ibaNet750 connection modes

7.3.4.2 Analog and Digital tab

The signal tables for analog and digital signals are available as soon as a corresponding output terminal has been added to the ibaNet750 module.

The signals are divided into groups according to the terminal name.

In the signal tables on the *Analog* and *Digital* tabs, you can define the output signal in the *Expression* column.

7.3.5 ibaL2B... output modules

3 output module types are available for ibaL2B:

Output modules	Data type	GSD file
L2B integer I/O (5)	32 inputs/outputs - integer	iba_0F08.gsd
L2B real I/O (6)	32 inputs/outputs - real	iba_0F09.gsd
L2B S7 real I/O (7)	28 inputs/outputs - real (S7-400) 28 inputs/outputs - real (S7-300)	ibaF0b_4.gsd ibaF0b_3.gsd

Table 13: L2B output modules

With ibaL2B i/o modules, there are 32 analog (28 for S7 Real only) and 32 digital signals available per module, corresponding to the capacity of a PROFIBUS slave.

7.3.5.1 General tab

In the *General* tab, you should

- enter a calculation time base (optional)
- set the slave number according to your PROFIBUS project
- set the timeout (optional)
- select byte order

For notes on the general module settings see ↗ *General module settings, page 141*.

7.3.5.2 Analog and Digital tab

In the signal tables on the *Analog* and *Digital* tabs, you can define the output signal in the *Expression* column.

7.3.6 Reflective Memory output module

7.3.6.1 General tab

For notes on the general module settings see ↗ *General module settings, page 141*.

No further general settings are required. The main setup of the Reflective Memory interface must have been made before in the *Inputs* tab of the I/O manager.

7.3.6.2 Analog and Digital tab

In the signal tables on the *Analog* and *Digital* tabs, you can define the output signal in the *Expression* column.

Remember to set the address of each analog signal or address and bit number for digital signals.

7.3.7 EtherNet/IP I/O output modules

Modules that have already been defined at the input level can also be seen in the tree of the output interfaces and can be used for output signals.

The general module settings are identical to those on the input side and are retained. Only the number of analog and digital signals can be set differently.

For notes on the general module settings see ↗ *General module settings, page 141*.

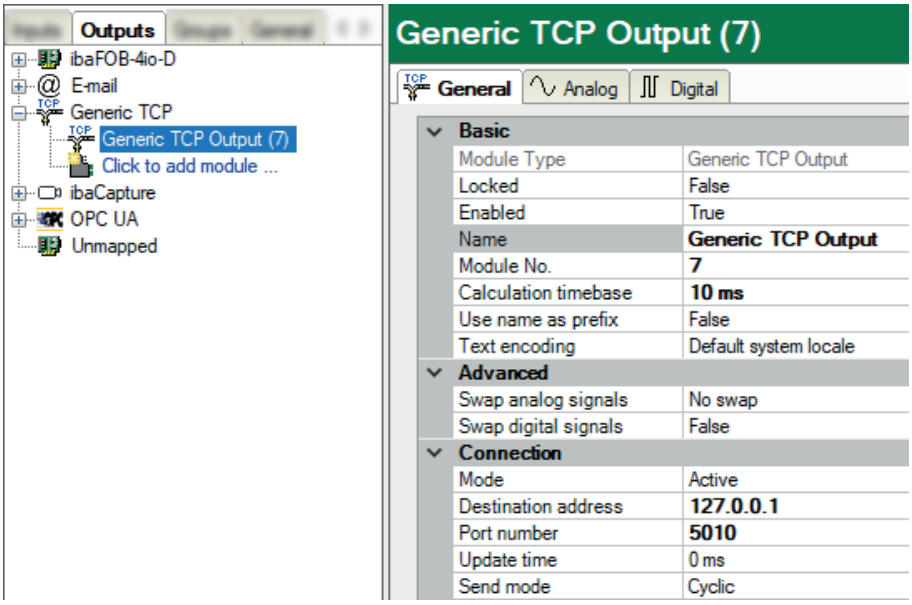
Configure the desired output signals in the *Analog* and *Digital* tabs. Enter a name, expression and address and select a data type for analog signals. The following data types are available for analog signals:

Data type		Description	Value 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

Table 14: Data types (Big and Little Endian)

7.3.8 Generic TCP output module

By using the *Generic TCP Output* module, you can transmit analog and digital signals via TCP/IP from *ibaPDA* to other systems.



7.3.8.1 General tab

The basic settings and the settings under *Advanced* can be adjusted as usual.

For notes on the general module settings see ↗ *General module settings, page 141*.

Text encoding
ibaPDA supports different text encodings for the output of texts or strings.

- Default system locale As specified in the Windows settings
- Western European (1252): Code page for Western Europe
- UTF-8: Unicode

Select the desired encoding here.

Connection settings
The following settings must be made for the connection:

Mode
Connection setup mode

- Active: *ibaPDA* establishes the connection to the port at the specified address.
- Passive: *ibaPDA* listens on the specified port and waits for a request to establish the connection. Only one connection is allowed in this mode.

Destination address
Here specify the destination address of the system with which the connection should be established.

Port number
Here specify the port via which the target system should communicate.

Update time
Update time determines in which cycle *ibaPDA* sends telegrams. If the update time is set to 0 ms, then *ibaPDA* sends a telegram as soon as output data is available. As for all output modules, the shortest cycle is 50 ms (see calculation time base).

Send mode
The selected send mode determines when *ibaPDA* should send the data.

- Cyclic: A telegram is sent at the set update time, at least every 50 ms.
- On change: A telegram is sent each time the signal data is changed.
- On trigger: A telegram is sent each time a rising edge is detected on the trigger signal.

7.3.8.2 Analog and Digital tab

You configure the analog output signals in the Analog tab.

In the Expression column, enter a signal or, if necessary, using the expression editor, a mathematical expression, the result of which is to be displayed.

The address you enter in the Address column is the address within the telegram where the data is written.

A data type must be set for each analog signal. The following data types are available:

Data type	Description	Value range:
SINT	8 bit with positive or negative sign	-128 … 127
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	±1.175495E-38 … ±3,403E+38
DOUBLE	IEEE754; double precision; 64 bit floating point;	±2,225E-308 ... ± 1, 798E+308
STRING[32]	String with up to 32 characters	

Table 15: Data types

Digital signals are processed in packets of 16 bits (integer). Consecutive addresses therefore have a distance of 2 bytes and the individual digital signals are identified by the bit number (0 to 15).

For a signal to actually be processed, a check mark must be placed in the *Active* column. If a signal is not active, then a null is entered at the appropriate place in the telegram.

7.3.9 Generic UDP output module

By using the *Generic UDP Output* module, you can transmit analog and digital signals via UDP from *ibaPDA* to other systems.

7.3.9.1 General tab

The basic settings and the settings under *Advanced* can be adjusted as usual.

For notes on the general module settings see ↗ *General module settings, page 141*.

Text encoding
ibaPDA supports different text encodings for the output of texts or strings.

- Default system locale As specified in the Windows settings
- Western European (1252): Code page for Western Europe
- UTF-8: Unicode

Select the desired encoding here.

Connection settings
The following settings must be made for the connection:

Destination address
Here specify the destination address of the system with which the connection should be established.

Port number
Here specify the port via which the target system should communicate.

Update time
Update time determines in which cycle *ibaPDA* sends telegrams. If the update time is set to 0 ms, then *ibaPDA* sends a telegram as soon as output data is available. As for all output modules, the shortest cycle is 50 ms (see calculation time base).

Send mode
The selected send mode determines when *ibaPDA* should send the data.

- Cyclic: A telegram is sent at the set update time, at least every 50 ms.
- On change: A telegram is sent each time the signal data is changed.
- On trigger: A telegram is sent each time a rising edge is detected on the trigger signal.

7.3.9.2 Analog and Digital tab

You configure the analog output signals in the Analog tab.

In the Expression column, enter a signal or, if necessary, using the expression editor, a mathematical expression, the result of which is to be displayed.

The address you enter in the Address column is the address within the telegram where the data is written.

A data type must be set for each analog signal. The following data types are available:

Data type	Description	Value range:
SINT	8 bit with positive or negative sign	-128 … 127
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	±1.175495E-38 … ±3,403E+38
DOUBLE	IEEE754; double precision; 64 bit floating point;	±2,225E-308 ... ± 1, 798E+308
STRING[32]	String with up to 32 characters	

Table 16: Data types

Digital signals are processed in packets of 16 bits (integer). Consecutive addresses therefore have a distance of 2 bytes and the individual digital signals are identified by the bit number (0 to 15).

For a signal to actually be processed, a check mark must be placed in the *Active* column. If a signal is not active, then a null is entered at the appropriate place in the telegram.

7.3.10 Modbus-TCP Client output module

The output module is not an autonomous module, but rather an extension of the *Modbus Client* module. With the output module, you can write data from *ibaPDA* to a controller.

You can configure the module in the *Outputs* tab. You do not have to add it separately. The module will be available as soon as a *Modbus Client* module has been added.

The settings correspond to those in the *Inputs* tab, apart from the module-specific settings, and can also be configured there. The connection settings also correspond to those in the *Inputs* tab.

For notes on the general module settings see ↗ *General module settings, page 141*

Please note the following differences to the settings of the input modules:

- For Analog type only "Holding register", single or multiple is possible
- For Digital type only "Holding register" and "Coils", each single or multiple are possible
- The transmission cycle of the messages is determined by the following parameters:
  - Update time (in ms): If you enter 0 or a value <50, the transmission cycle equals the *ibaPDA* task cycle of approximately 50 ms
  - Send only when changed:
    False: the output message will be sent in the above mentioned cycle (update time).
    True: the output message will only be sent when one of the signals changes its value, but at the latest after 1 min.

7.3.11 ABB-Xplorer output module

Please consider to check the *Enable ABB-Xplorer outputs* option in the I/O Manager on the root node *ABB-Xplorer* in order to use outputs.

Modules that have already been defined at the input level can also be seen in the tree of the output interfaces and can be used for output signals.

Select the *Outputs* tab in the module tree to configure the module. You do not have to add it separately. The module will be available as soon as an *ABB-Xplorer-MMS* module has been added.

Except for the calculation timebase and the minimum output timebase the general module settings are identical to those on the input side and are retained.

The number of analog and digital signals (module layout) can be set differently.

For notes on the general module settings see ↗ *General module settings, page 141*.

MMS
Unlike to the input side there is an additional setting *Write mode*.

Send mode
Determines when new data is written to the controller:

- Cyclic: Data is written cyclically at the set update time.
- On change: Data is written each time the signal data is changed.
- On trigger: Data is written with every rising edge of the trigger signal.

All signals of a module are always written, regardless of the write mode.

Trigger signal
This field only appears when the "on trigger" send mode is selected. Select here a digital signal. A rising edge on this digital signal writes the signal values taken at the time of the rising edge.

The connection settings are identical for inputs and outputs.

Configure the desired output signals in the *Analog* and *Digital* tabs. Enter name and expression and select the desired symbol in the PLC using the symbol browser.

If you make your selection of analog signals in the symbol browser, the corresponding data type will be applied as well. Without symbol browser you can enter the symbol name an select the data type of analog signals. The following data types are available for analog signals:

SINT, BYTE, INT, WORD, DINT, DWORD, FLOAT, DOUBLE

7.3.12 OMRON-Xplorer output module

The output module is not an autonomous module, but rather an extension of the *OMRON FINS* module. With the output module, you can write data from *ibaPDA* to a controller.

Note

Depending on the configured protection-level access of the OMRON CPU, writing values to the CPU may be not possible.

You can configure the module via the *Outputs* tab. You do not have to add it separately. The module will be available as soon as an *OMRON FINS* module has been added.

The settings correspond to those in the *Inputs* tab, apart from the module-specific settings, and can also be configured there. The connection settings also correspond to those in the *Inputs* tab.

For information about the general module settings refer to ↗ *General module settings, page 141*

Module-specific settings

Update time
With the update time, you determine the speed with that *ibaPDA* tries to request the data from the OMRON PLC. The actual resulting update time may be higher, depending on the load of the device. Check the diagnostic overview for measured update rates, because an overload will cause loss of samples.

Send mode
Determines when new data is written to the controller:

- Cyclic: Data is written cyclically at the set update time.
- On change: Data is written each time the signal data is changed.
- On trigger: Data is written with every rising edge of the trigger signal.

All signals of a module are always written, regardless of the write mode.

Trigger signal
This field only appears when the "on trigger" send mode is selected. Select here a digital signal. A rising edge on this digital signal writes the signal values taken at the time of the rising edge.

Output signals

The OMRON-Xplorer output module does not support digital signals.

You can configure the signals to be output in each case via the expression editor. Open the expression editor via the <fx> button in each signal row.

For each signal, enter a name and select memory type, address and data type. Supported data types are SINT, BYTE, INT, WORD, DINT, DWORD, LINT, QWORD, FLOAT, DOUBLE.

Note

ibaPDA reads and writes all signals for an Xplorer module via a common connection. Therefore, the total number of configured signals influences the update time.

7.3.13S7-Xplorer output module

Please consider to check the *Enable S7-Xplorer outputs* option in the I/O Manager on the root node *S7-Xplorer* in order to use outputs.

Only the module type *S7-Xplorer* can be used for outputs.

Modules that have already been defined at the input level can also be seen in the tree of the output interfaces and can be used for output signals.

Except for the calculation timebase and the minimum output timebase the general module settings are identical to those on the input side and are retained.

The number of analog and digital signals (module layout) can be set differently.

For notes on the general module settings see ↗ *General module settings, page 141*.

S7
Unlike to the input side there is an additional setting *Write mode*.

Send mode
Determines when new data is written to the controller:

- Cyclic: Data is written cyclically at the set update time.
- On change: Data is written each time the signal data is changed.
- On trigger: Data is written with every rising edge of the trigger signal.

All signals of a module are always written, regardless of the write mode.

Trigger signal
This field only appears when the "on trigger" send mode is selected. Select here a digital signal. A rising edge on this digital signal writes the signal values taken at the time of the rising edge.

The connection settings are identical for inputs and outputs.

Configure the desired output signals in the *Analog* and *Digital* tabs. Enter name and expression and select the desired symbol in the PLC using the symbol browser or enter the S7-operand.

If you make your selection of analog signals in the symbol browser, the corresponding data type will be applied as well. Without symbol browser you can enter the symbol name an select the data type of analog signals. The following data types are available for analog signals:

BOOL, CHAR, BYTE, WCHAR, WORD, DWORD, SINT, INT, DINT, LINT, USINT, UINT, UDINT, REAL, LREAL, S5 REAL, TIME, LTIME, TIMER, COUNTER, STRING[], WSTRING[], CHAR[], WCHAR[]

7.3.14 TwinCAT-Xplorer output module

Please consider to check the *Enable TwinCAT-Xplorer outputs* option in the I/O Manager on the root node *TwinCAT-Xplorer* in order to use outputs.

Only the module type *TwinCAT PLC* can be used for outputs.

Modules that have already been defined at the input level can also be seen in the tree of the output interfaces and can be used for output signals.

Except for the calculation timebase and the minimum output timebase the general module settings are identical to those on the input side and are retained.

The number of analog and digital signals (module layout) can be set differently.

For notes on the general module settings see ↗ *General module settings, page 141*.

PLC
Unlike to the input side there is an additional setting *Write mode*.

Send mode
You can choose between three different send modes:

- Cyclic: Cyclic writing according to the calculation timebase
- On change: Data is written whenever the value of an analog signal changes
- On trigger: Data is beeing written each time a rising edge is detected on "send trigger" signal.

The connection settings are identical for inputs and outputs.

Configure the desired output signals in the *Analog* and *Digital* tabs. Enter name and expression and select the desired symbol in the PLC using the symbol browser.

If you make your selection of analog signals in the symbol browser, the corresponding data type will be applied as well. Without symbol browser you can enter the symbol name an select the data type of analog signals. The following data types are available for analog signals:

FLOAT

7.3.15 ibaCapture output module

By using the *ibaCapture* output module, signals can be sent from *ibaPDA* to an *ibaCapture* server in order to control video recording and/or cameras.

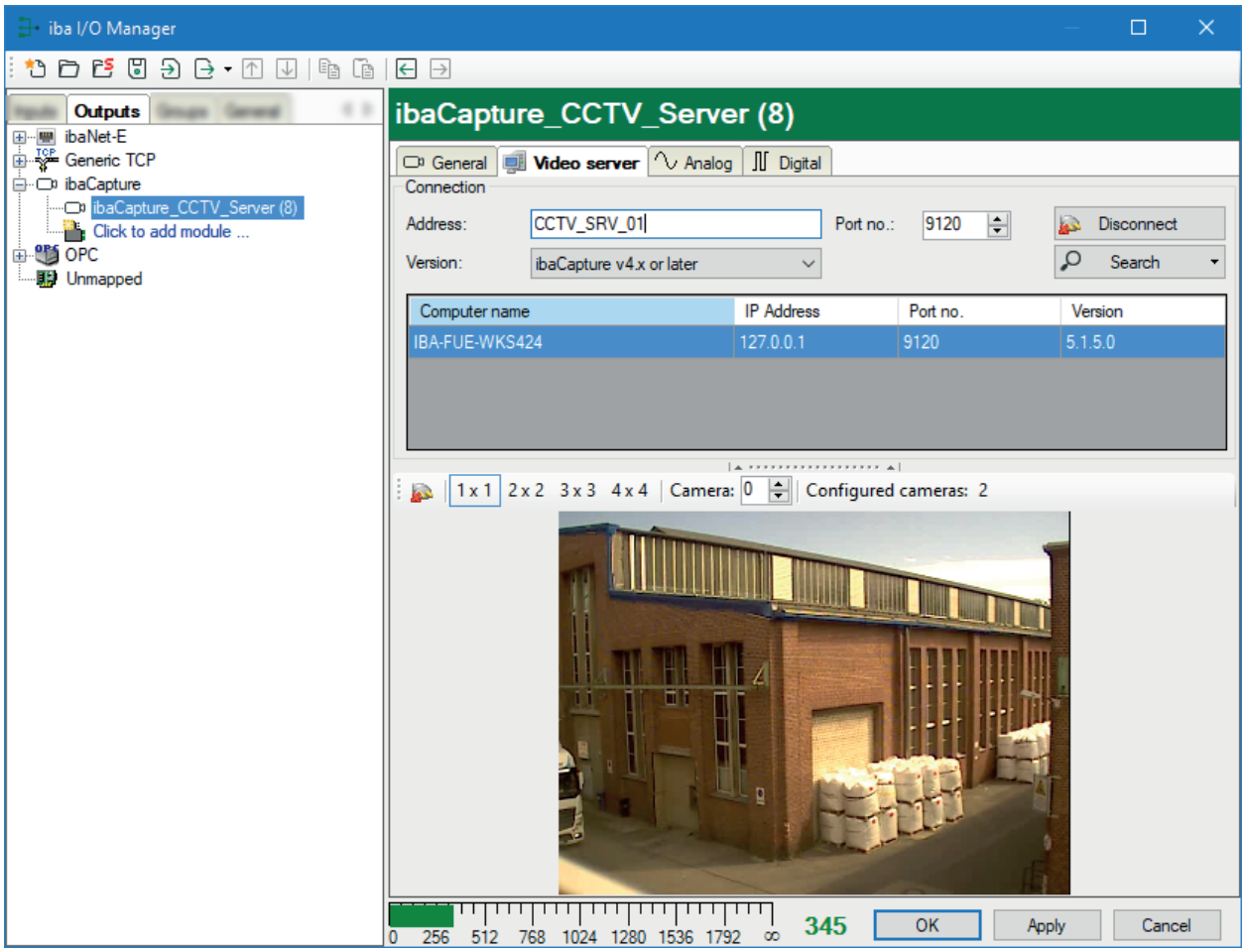
7.3.15.1 General tab

You can configure the usual basic settings in the *General* tab.

For notes on the general module settings see ↗ *General module settings, page 141*

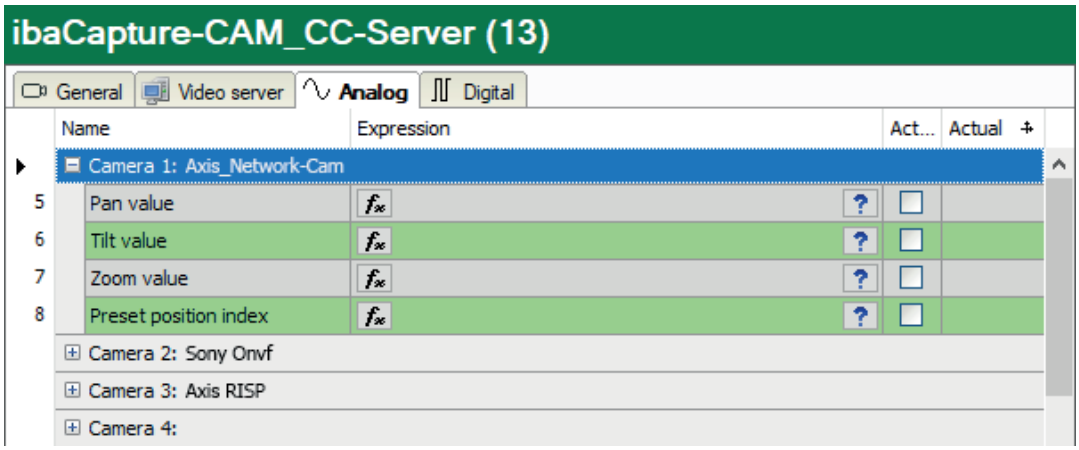
7.3.15.2 Video server tab

Each *ibaCapture* output module must first be connected to the appropriate *ibaCapture* server. You can make the settings in the *Video server* tab of the module.



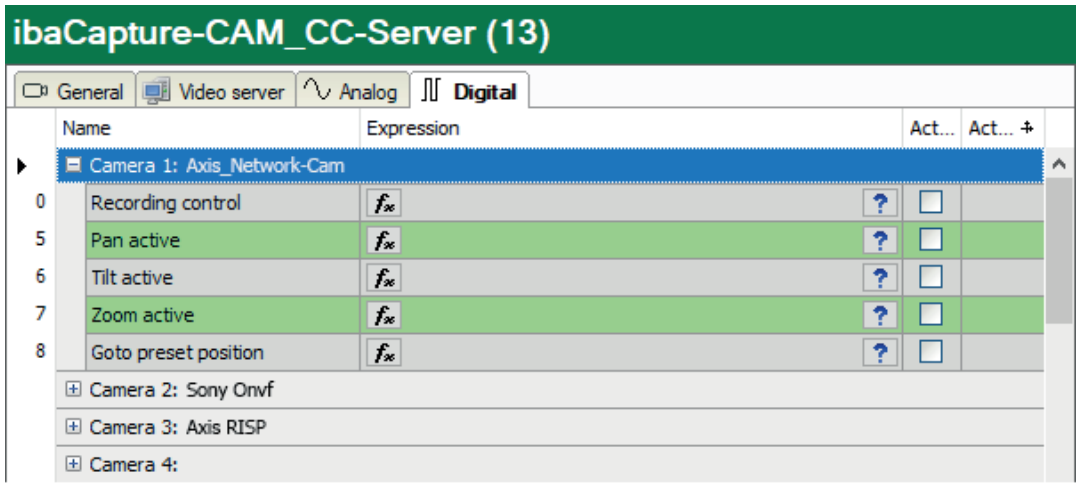
7.3.15.3 Analog and Digital tab

Controlling PTZ cameras with analog signals



If you use PTZ (pan tilt zoom) cameras and want to control their settings in terms of pan, tilt and zoom with *ibaPDA*, you can do this by using analog output signals. You can define the values for the pan and tilt angle as well as for the zoom factor by using the expression editor. Presets stored in the camera can also be accessed directly. A data set is available for each of an *ibaCapture* server's 16 cameras.

Controlling video recording and PTZ cameras with digital signals



A data set with digital output signals is available for each camera of an *ibaCapture* server. You can use the first signal"Camera recording control ..." to start and stop a camera video recording. You can use the expression builder to configure the signal depending on the process. To control PTZ cameras, you will find the digital output signals to enable or disable the control functions (pan, tilt, zoom and preset) in this tab.

Other documentation

Detailed information on the use and configuration of output signals can be found in the *ibaCapture* product manual.

7.3.16 ibaVision output module

With this type of module, you can transfer signal values from *ibaPDA* to an *ibaVision* system.

For notes on the general module settings see ↗ *General module settings, page 141*

Other documentation

For a description of the *ibaVision* functions and configuration options for this module, see the *ibaVision* product manual.

7.3.17 FOB alarm output module

One FOB alarm module each with 64 analog/digital output signals is possible per link of an ibaFOB output card. The FOB alarm module only supports the 3 Mbit mode. If you add an FOB alarm module to a link of an ibaFOB-D card, then the link automatically switches to the 3 Mbit mode. Per FOB alarm module, you can configure up to 64 analog and 64 digital signals and send to iba-capable devices with 3 Mbit mode, such as ibaLink-SM-64-io, ibaLink-SM128-i-2o, ibaPA-DU-8-O or ibaNet750-BM.

7.3.17.1 General tab

In the *General* tab, you can change the connection mode to “Real” if the receiving system specifies real values. The default setting is “integer.”

Setting the calculation time base is not possible because the FOB alarm outputs are always processed with the lowest time base that corresponds to the smallest common multiple of all module time bases (min. 50 ms). Higher time bases will be calculated automatically.

For notes on the general module settings see ↗ *General module settings, page 141*.

7.3.17.2 Analog and Digital tab

You can configure the desired output signals in the column *Expression* on the tabs *Analog* and *Digital* in the signal tables, just like for the virtual signals. To define the output signals, use the expression builder, which will again provide you with all options for the mathematical and logical calculation. You may enter simple numerical or Boolean equations or complex mathematical expressions. The expression builder is the supporting tool for the creation of mathematical expressions. If you know the syntax, you may also manually enter the expressions in the table. Use the button <fx> to open the expression builder.

For more information about the expression builder, please refer to part 4 *Expression builder (virtual signals)*.

Remember that the result of the expression must match the signal's data type.

Virtual analog signals are always REAL type values. If the connection mode is set to “Integer,” the analog output signal will be rounded to the nearest whole number (integer). In order to get a virtual digital signal or alarm, the result of the expression must be a Boolean quantity (TRUE or FALSE).

7.3.18 Outputs to the iba modular system (ibaPADU-S)

If *ibaPDA* is connected to the central unit of an iba modular system via an *ibaFOB-D* card then analog and digital signal values can also be output to the modular system.

The following central units are supported:

Name	Function
ibaPADU-S-CM	Setting discrete analog and digital output signals using appropriate output modules
ibaPADU-S-IT-16	Setting discrete analog and digital output signals using appropriate output modules; Data supply of the application on the device, e.g., ibaLogic
ibaPADU-S-IT-2x16	Setting discrete analog and digital output signals using appropriate output modules; Data supply of the application on the device, e.g., ibaLogic
HAICMON CMU	Data supply of the application on the device
ibaCMU-S	Setting discrete analog and digital output signals using appropriate output modules; Data supply of the application on the device
ibaPQU-S	Setting discrete analog and digital output signals using appropriate output modules; Data supply of the application on the device

Table 17: Central units of the modular system with the support of outputs

Other documentation

For information about the configuration of the outputs for modular system devices, see the relevant manual for the device.

7.3.19 Outputs to iba bus modules (ibaBM-...)

If *ibaPDA* is connected to a bus module in the 32 Mbit flex mode via an ibaFOB-D card, analog and digital signal values can also be written on the connected fieldbus.

The following bus modules are supported:

Name	Function
ibaBM-DP	Sending output signals to a DP master via an active slave configured in the device.
ibaBM-PN	Sending output signals to a PN controller via a device slot configured in the device.

Table 18: Bus modules with output support

Other documentation

For information about the configuration of the outputs for bus modules, see the relevant manual for the device.

7.3.20 Outputs to iba system interconnections (ibaLink-...)

If *ibaPDA* is connected to a card for system interconnection in 32 Mbit flex mode via an ibaFOB-D card with a module, analog and digital signal values can also be written in the memory of the target system.

The following system interconnections are supported:

Name	Function
ibaLink-VME	Sending output signals to a VME-based automation system (via the card's dual-port RAM)
ibaLink-io-embedded	Depending on implementation in an external system

Table 19: System interconnections with output support

Other documentation

For information about the configuration of the outputs for system interconnections, see the relevant manual for the device.

7.3.21 OPC UA client output module

The *OPC UA client output* module is also available with the license for the OPC UA client interface.

Other documentation

Note the extensive explanations about the prerequisites and the connection set-up in the manual about the product *ibaPDA-Interface-OPC-UA-Client*.

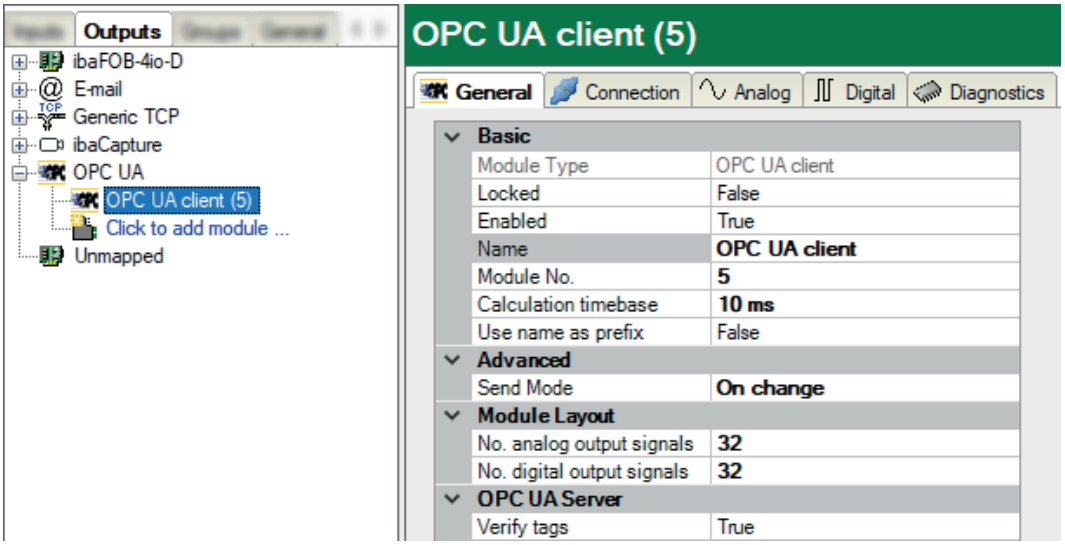
Modules that have already been defined at the input level can also be seen in the tree of the output interfaces and can be used for output signals.

If you want to use the output signals, suitable writable tags must have been configured on the corresponding OPC UA server.

As with all output modules, the time base for the output cycle is at least 50 ms.

General module settings

The general module settings are partially identical to those on the input side and are retained.



No settings with respect to the OPC server can be made for the output module.

In the *Advanced* section, you can use the Send mode to set when the output data should be written:

- On change
  The output data is written when the signal values change, but not faster than the general output cycle time of the system allows (at least 50 ms).
- On trigger
  The output data is written on each rising edge of the selected trigger signal (write trigger), but not faster than the general output cycle time of the system allows (at least 50 ms).

The number of analog and digital output signals is preset to 32. If necessary, you can change the number (max. 1000).

Clicking on <Select symbols> opens the OPC UA symbol browser. The prerequisite here is that the right address book of the OPC UA server was created. See all available tags here. By double-clicking on a tag or selecting the tag and clicking on <Add>, the tag is entered in the next open line of the appropriate signal table (*Analog* or *Digital*) as *node ID*.

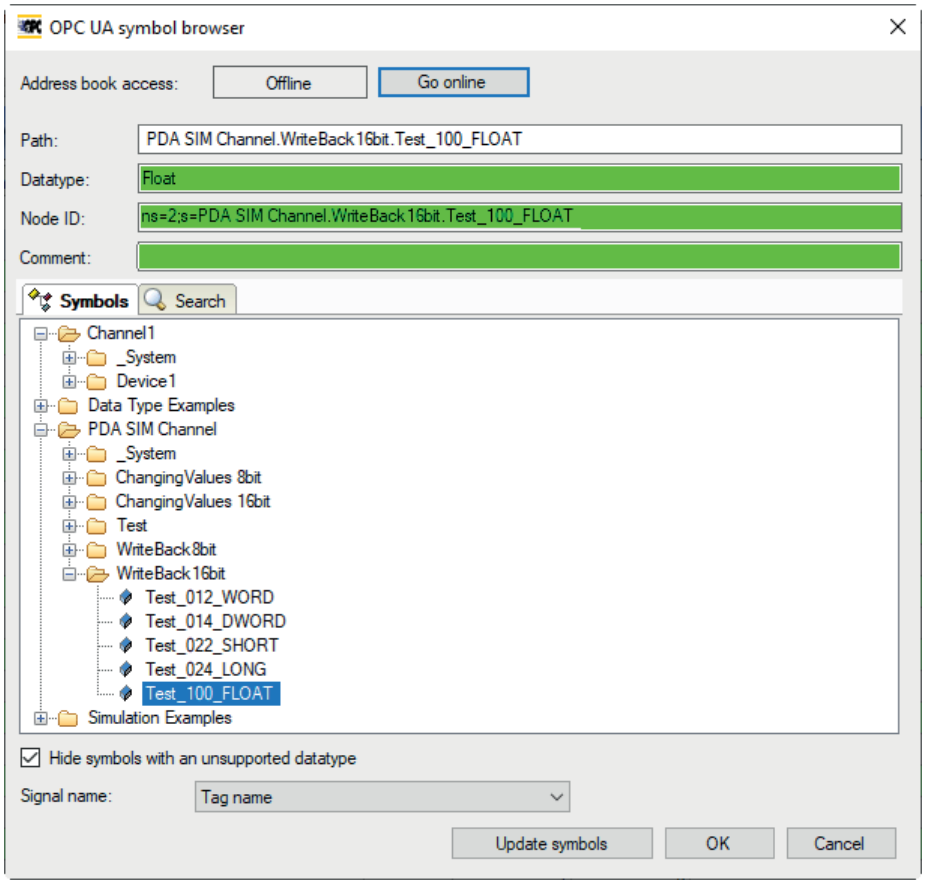
Connection

The connection settings are the same as those on the input side (OPC UA client module, I/O Manager area *Inputs*).

Analog and digital signals

In the *Analog* and *Digital* tabs, you can either select measurement signals from the signal tree in the *Expression* column and thus make them available as OPC UA output signals, or generate separate (virtual) signals using the expression builder.

To allocate the signals to the OPC UA tags of the OPC UA server, proceed in the *Node ID* column. Click on the button <...> in a table cell to open the OPC UA symbol browser, which you can use to select the desired tag.



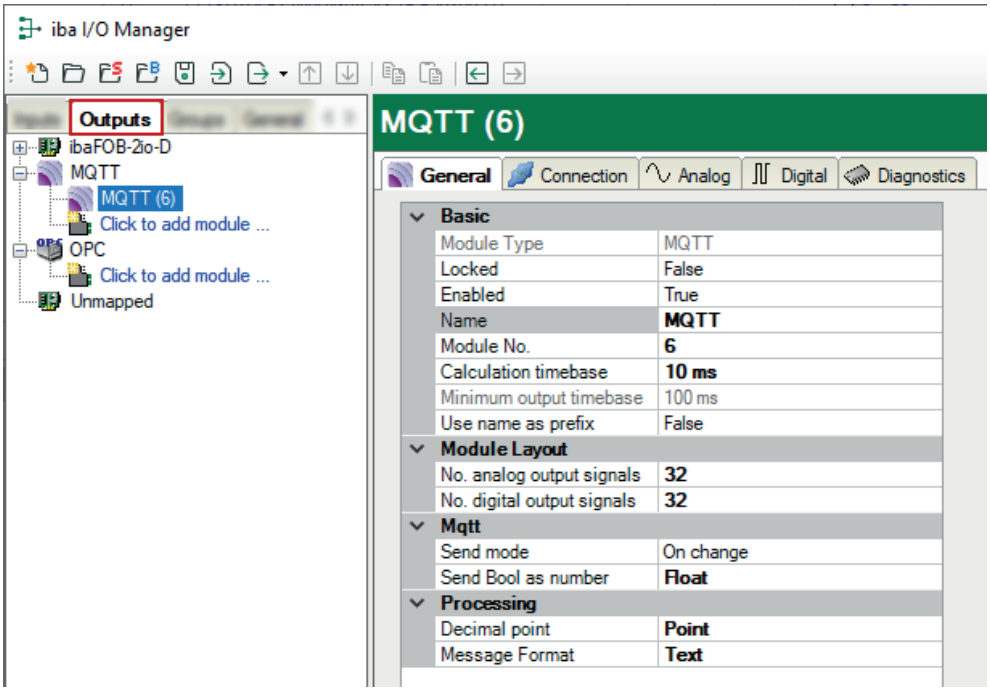
If you open the symbol browser from the signal tables instead of via the hyperlink in the *General* tab, then you will already be shown the tags with the corresponding data types.

7.3.22 MQTT output module

The output module is not an autonomous module, but rather an extension of the MQTT module. With the output module, you can write data from *ibaPDA* to an MQTT-Broker.

Select the *Outputs* tab in the module tree to configure the module. You do not have to add it separately. The module will be available as soon as an MQTT module has been added.

The following module settings can be configured on the *General* tab:



Basic settings

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

Locked
A module can be locked in order to prevent change of module settings by accident or unauthorized users.

Enabled
Disabled modules are excluded from the signal acquisition.

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

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

Calculation timebase
Timebase (in ms) for the calculation of the output values.

The calculation timebase and update timebase are NOT the same!

For interfaces whose modules have an input and an output side (same module number), the calculation timebase is identical to the module timebase on the input side. Hence, changing the calculation timebase also changes the module timebase on the input side and vice versa.

For interfaces with separate input and output modules (different module numbers), the calculation timebase can be changed independently.

The smallest possible calculation time base, i.e., the fastest possible update of the outputs, results from the smallest common multiple of all module time bases, and is at least 50 ms. .

This setting does not exist for the E-mail, OPC output and SQL command output modules.

Minimum output timebase (information only)
Smallest timebase for updating the outputs.

This value is calculated automatically by the system considering the current I/O configuration. It is displayed for information only. The minimum output timebase depends on the least common multiple of all module timebases, with a minimum of 50 ms.

Use name as prefix
Puts the module name in front of the signal names.

Module layout

No. of analog signals/ No. of digital signals
Determines the number of analog/digital output signals in this module. The default value is 32 for each. Maximum value is 1000.

Mqtt

Send Bool as number
Decide whether Boolean values in text mode should be written as FLOAT values (1 and 0) or as Boolean values (TRUE and FALSE).

Send mode
Determines when new data is written to the controller:

- Cyclic: Data is written cyclically at the set update time.
- On change: Data is written each time the signal data is changed.
- On trigger: Data is written with every rising edge of the trigger signal.

All signals of a module are always written, regardless of the write mode.

Trigger signal
This field only appears when the "on trigger" send mode is selected. Select here a digital signal. A rising edge on this digital signal writes the signal values taken at the time of the rising edge.

Processing

Decimal point

Configure, which character should be used as decimal point.

Message format

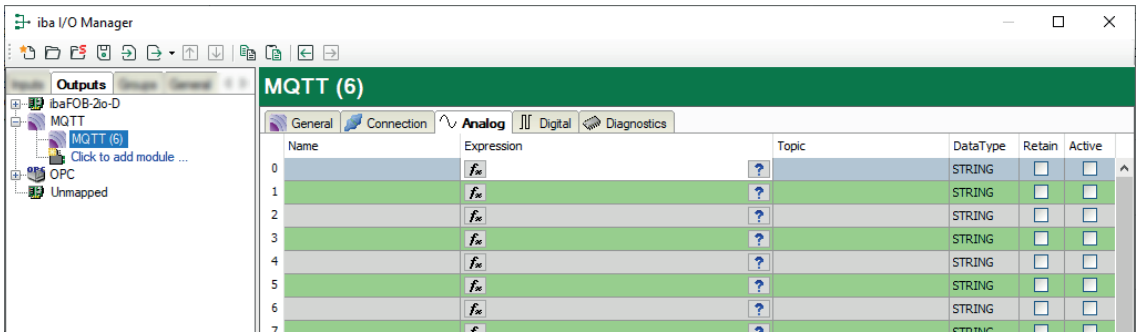
Select if values in MQTT broker are stored in text format or as binary values.

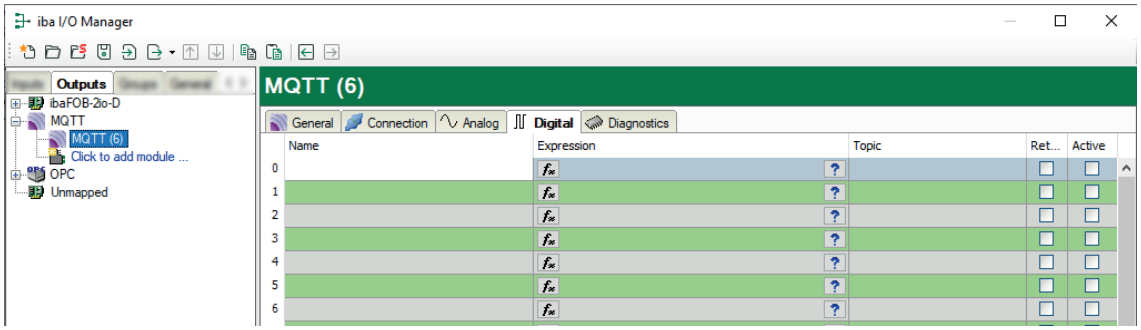
Output signals

The analog and digital signals be outputs, are configured in the expression editor. The expression editor can be opened by the <fx> button from each signal row.

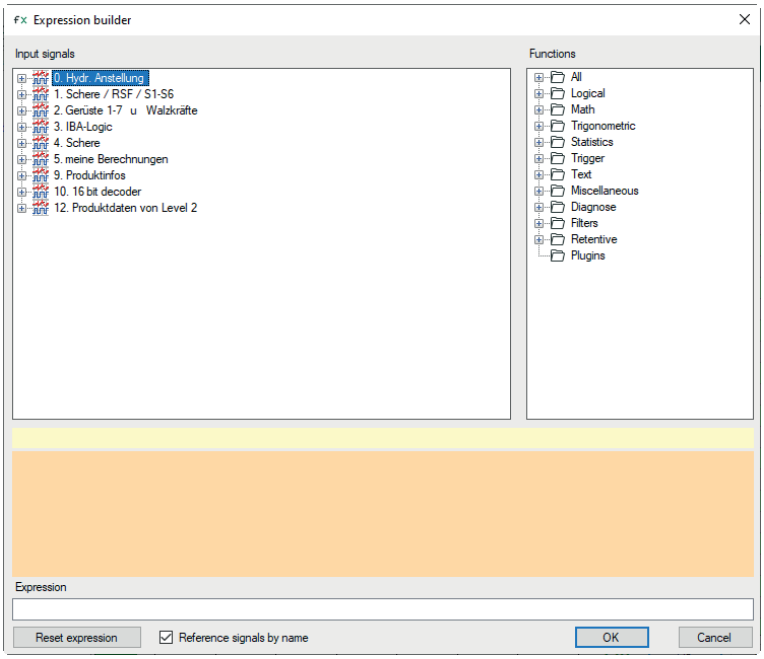
In the *Analog* and *Digital* tabs enter a name for each signal and a data path to the broker. The following data types are supported for analog signals: SINT, BYTE, INT, WORD, DINT, DWORD, FLOAT, DOUBLE, STRING.

If you enable the *Retain* option, the signals are stored on the broker until they are overwritten.





Expression builder:



For more detailed information about the expression builder please refer to part 4 of the *ibaPDA* manual.

7.3.23 Output module for SQL commands to SQL 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 from different manufacturers are supported (Maria DB, MySQL, Oracle, PostgreSQL, SAP HANA, SQL Server).

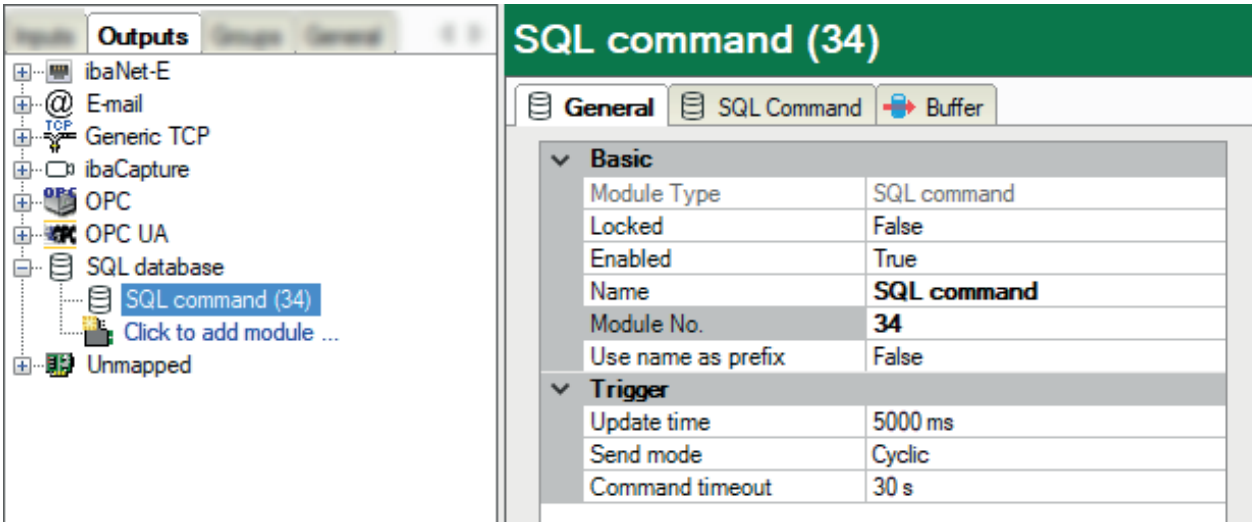
The data exchange with the databases occurs via corresponding SQL modules, which are configured according to the specific database type.

The SQL commands are configured in the *Outputs* tab. An *SQL command* module must be configured for each SQL command.

General module settings

For notes on the general module settings see ↗ *General module settings, page 141*.

The "Calculation timebase" parameter is not available here.



Trigger

- Update time
  In the "Cyclic" send mode, the period in which the command is issued; in the "On change" or "On trigger" send mode, the dead time for suppression of closely successive changes/triggers, if not desired.
- Send mode
  - Cyclic: Cyclic sending of the command after the update time
  - On change: Command is sent when the value of the "Send trigger" signal changes
  - On trigger: Command is sent when "Send trigger" has a rising edge.
- Send trigger
  Only available with send mode "On change" or "On trigger"
  Select a suitable signal (analog/digital) with which you want to control the execution of the command.
- Command timeout
  Time limit (in seconds) for the execution of a command. If a command cannot be completely executed within this time after triggering, it is aborted.

SQL command tab
You enter the SQL command in this tab.

Buffer tab
In this tab you can configure a buffer in case the database is not accessible. The memory buffer is always enabled, the file buffer can be additionally enabled and configured.

Commands that could not be executed because the connection to the database was interrupted are stored here and executed sequentially after the connection is restored.

Other documentation

A detailed description of the SQL interface and SQL modules can be found in the manual for the corresponding software product:

- ibaPDA-Interface-MySQL
- ibaPDA-Interface-Oracle
- ibaPDA-Interface-PostgreSQL
- ibaPDA-Interface-SAP-HANA
- ibaPDA-Interface-SQL-Server

7.3.24 ibaNet-E output modules

You can use the *ibaNet-E* interface to establish a connection to an *ibaNet-E*-compatible device.

If you use *ibaPDA* on a PC, the connection is established over the network adapter of the computer or over a special *ibaNet-E* network adapter (*ibaM-2E*). In this case you can connect to devices as follows:

- *ibaW-750* + output terminals of the WAGO I/O-System 750
- *ibaM-COM* + output modules of the *ibaMAQS* system

If you use *ibaPDA* on an *ibaM-DAQ*, then you configure the *ibaM-DAQ* device itself with its own outputs as well as attached modules of the *ibaMAQS* system on the *IO bus* (internal *ibaNet-E*). Additionally, several modules are available for selection, which can be configured as output modules:

- *ibaW-750* + output terminals of the WAGO I/O-System 750
- *ibaM-COM* + output modules of the *ibaMAQS* system or + *ibaM-FOB-2io* with *ibaPADU-/ibaBM* devices with output modules

The configuration of outputs for *ibaW-750* follows the same way like for *ibaNet-750-BM*.

When configuring the combination of *ibaM-COM* + *ibaM-FOB-2io* + *ibaPADU-/ibaBM* devices with output modules, the rules and settings of the connected devices apply.

7.3.25 ibaM-DAQ outputs

The device *ibaM-DAQ* provides one digital output, which can be configured for the *ibaM-DAQ-IO* module.

For notes on the general module settings see ↗ *General module settings, page 141*

No more module settings need to be made for the outputs.

Output signal

There is only one digital output signal.

Name

Here you can enter a signal name and if you click on the symbol in the *Name* field two more comments.

Expression

By means of the expression builder you can assign signals to the output or create logical and mathematical signal expressions.

Active

Enable/disable the signal

8 The I/O configuration file

All configuration data is stored in an XML file.

The *ibaPDA* system uses three different configuration files for I/Os, data storage and display (layout).

For the I/O configuration, *ibaPDA* always uses the *CurrentIoConfig.io* file. The file is stored in a sub-folder of the `ProgramData` folder (path: `c:\ProgramData\iba\ibaPDA...`).

This file is loaded with each start of the system. The file is updated with each configuration change and when exiting the I/O Manager by pressing the <OK> button. Thus, the current configuration is always stored in this file without the need for explicit saving. Each time a new configuration is applied, the previous one is automatically saved in the `...\Backup` folder. The last ten configurations are always available in the backup folder.

If the most recently used configuration file is not found, the server automatically loads the most recent backup configuration.

If you have applied a valid and functional configuration, it is recommended to also store the configuration file under another name in a protected folder or disk of your choice in order to avoid unintended modifications or loss of configuration data.

8.1 Saving an I/O configuration

1. Open the I/O Manager after you have made sure that the current configuration is correct.
2. Click the button (save configuration) in the toolbar of the I/O Manager.
3. In the dialog, select the folder where you want to store the file. We recommend choosing the path `c:\ProgramData\iba\ibaPDA`, as this is the default path of the I/O configuration files. Of course, however, you may also choose any other folder.
4. Enter a file name (prefix).
5. Choose the file name extension (default: .io)
   - .io: A simple XML file will be saved.
   - .zip: The XML file will be saved in ZIP file.
6. Click on <Save>.

8.2 Loading an existing I/O configuration

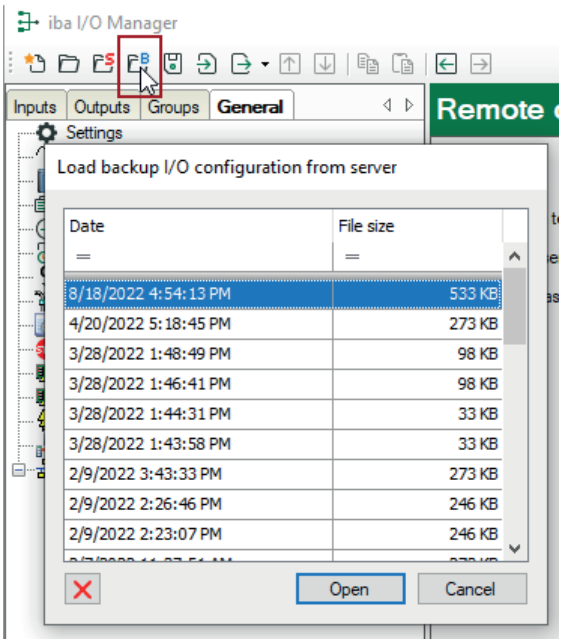
1. Open the I/O manager.
2. Click the button (open) in the toolbar of the I/O Manager.
3. Using the browser, select the configuration file you would like to use. Usually, you will find the configuration files in the folder `c:\ProgramData\iba\ibaPDA\Server`. The browser shows only files with the extension *.io*.
   The *CurrentIoConfig.io* file contains the latest configuration.
4. Click on <Open>. The configuration will be loaded and verified with the devices connected and the license information in the dongle. If no conflict occurs, the configuration is considered as valid and the acquisition will be started.

Tip

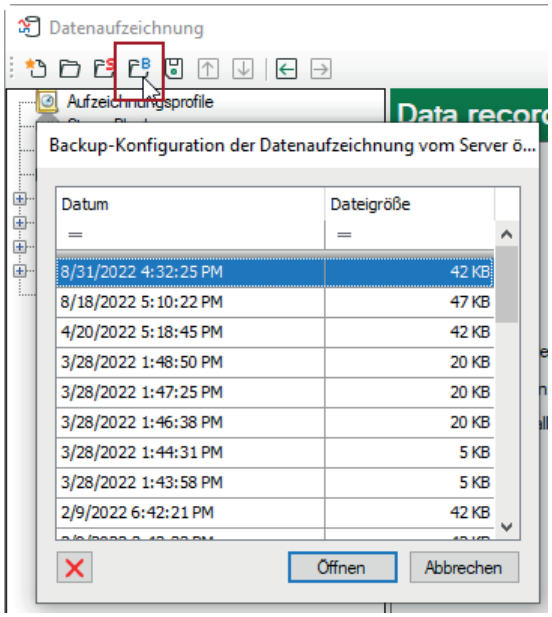
Alternatively, you can store the current configuration in a project. For more information, please refer to the "ibaPDA project management" chapter.

8.3 Loading a backup configuration file

Each time a new I/O configuration or datastore configuration is applied, *ibaPDA* automatically creates a backup of the previous configuration. These backups are stored in the path `%ProgramData%\iba\ibaPDA\Backup` on the *ibaPDA* server computer. In the toolbar of the I/O Manager or data storage manager there is a dedicated button which provides direct access to the backups.



Open backup in I/O Manager



Open backup in datastore manager

If you click on this button, a dialog window will open, which shows a list of the available backups with their date and time of creation and file size.

By selecting a row and clicking on <Open> or by doubleclicking on a row you can open the corresponding backup in the I/O Manager or data storage manager respectively.

If you want to apply the configuration from the backup, then click on <Apply> or <OK>.

In the dialog window with the backup list there is a button to remove a selected backup, if needed.

8.4 Exporting an I/O configuration to a text file

The export function generates a text file which contains modules, groups and profiles. If you have configured profile-based modules (e.g., InSpectra modules), then their profiles will also be exported.

You can only obtain a complete I/O configuration, e.g. including the settings in the *General* branch in the I/O Manager, by saving as an XML file (*.io) or using the project save function.

Modules, groups and profiles can also be exported separately.

You can edit all of the exported files and then reimport them into *ibaPDA*.

This step is sometimes necessary if you update from older *ibaPDA* versions (< v6.0/v7.0) to newer versions.

Even in the case of an initial configuration where many signals need to be configured, editing a text file (e.g., in MS Excel) is usually more convenient and efficient than using the I/O Manager dialogs, especially if you already have the necessary signal information in list form.

Simply create an empty sample configuration which consists of one module of each type.

Next, export this configuration into a text file and then open it with a standard text editor or MS Excel (for example).

You can see the structure of the information in the text file. Copy the module parts as often as needed and adjust the module numbers. You can then edit the signals for each module or insert parts of signal lists from your engineering environment via copy and paste.

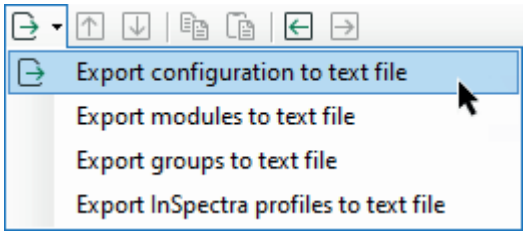
Note

When saving the text file, make sure that a Unicode or UTF-8 data format is selected so that umlauts (mutated vowels) and special characters in signal names and comments are correctly displayed in the I/O manager afterwards.

Once you re-import this text file into *ibaPDA*, the I/O configuration for the modules and signals is complete.

Proceed as follows:

1. Open the I/O manager.
2. Load or create a configuration.
3. Click on the export button and select *Export configuration to text file*.



4. Use the browser to select the desired directory (HDD and folder) in which to store the file, and specify a prefix for the file name. The file suffix *.txt* will be added automatically by *ibaPDA*.
5. Click on <Save>.

Note

The other export options only include the elements specified as part of the overall configuration.

Tip

Export of single modules is possible, too. If there is no need to export an entire I/O configuration, you can export a single module. Simply right-click on the module in the I/O Manager tree and choose *Export* from the context menu.

If you want to export multiple modules from one interface, right-click the interface in the I/O Manager.

Other documentation

More information on exporting InSpectra profiles can be obtained from the *ibaInSpectra* manual.

8.5 Importing a configuration text file

You may import files which have been created by older ibaPDA versions (V5, V6), ibaScope or by other tools (e.g. MS Excel), provided they match the ibaPDA-format.

The files must be simple text files. No XML files!

In case of ibaScope configuration files you need files with the extension ".setup".

1. Open the I/O manager.
2. Click the button (Import)
3. By using the browser, select the corresponding configuration text file from its storage location (local drive, network drive, floppy disk, CD-ROM etc.). By default, the browser only shows files with the extension *.txt*. If you are looking for ibaScope files, select the appropriate file type in the lower right corner of the dialog.
4. Click <Open>. The configuration will be loaded and verified with the devices connected and the license information in the dongle. If no conflict occurs, the configuration is considered as valid and the measurement will be started.
5. Store the configuration in the usual way, i.e. as an ibaPDA configuration file, as described above.

Note

Please note that the module and channel settings are taken over from ibaPDA-V5, however, not the archive profiles! The new data storage profiles should be defined after the import is done and the signals should be assigned to the new profiles.

9 Support and contact

Support

Phone: +49 911 97282-14

Email: support@iba-ag.com

Note

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

Product Security Incident Response Team (PSIRT) at iba

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

Email: psirt@iba-ag.com

Contact

Headquarters

iba AG
Gebhardtstrasse 10–20
90762 Fuerth
Germany

Phone: +49 911 97282-0

Email: iba@iba-ag.com

Mailing address

iba AG
Postbox 1828
D-90708 Fuerth, Germany

Delivery address

iba AG
Gebhardtstrasse 10
90762 Fuerth, Germany

Regional and worldwide

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

www.iba-ag.com

ibaPDA

Data interfaces and modules

Manual Part 3

Issue 8.14

Measurement Systems for Industry and Energy

www.iba-ag.com

Manufacturer

iba AG

Gebhardtstrasse 10-20

90762 Fuerth

Germany

Contacts

Main office	+49 911 97282-0
Support	+49 911 97282-14
Engineering	+49 911 97282-13
E-mail	iba@iba-ag.com
Web	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 2026, 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 and can be found in the iba help center docs.iba-ag.com.

Version	Date	Revision	Author	Version SW
8.14	06-2026	Playback: new playback options; update of Virtual retentive module	rm	8.14.0

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

1 About this documentation 9

1.1 Target group and previous knowledge 9

1.2 Notations 9

1.3 Used symbols 10

1.4 Documentation structure 11

2 Basics of data interfaces 12

2.1 Introduction 12

2.2 Handling and availability of interfaces 13

3 Basics of modules 15

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 22

3.6.3 Columns in tables with analog and digital signals 23

3.6.4 Filling signal tables 28

3.6.5 Notes on working in the signal tables 29

4 Standard interfaces (always available) 30

4.1 ibaCapture 31

4.1.1 Module types ibaCapture and ibaVision 33

4.2 OPC interface 34

4.2.1 OPC Client module 37

4.2.2 Redundant OPC client module 41

4.2.3 OPC Server module 42

4.3 Playback 43

4.4 Signal mapper 48

4.4.1 Signal mapper module 48

4.4.2 Add and configure signal mapper profiles 49
4.4.3 Signal mapper example 52
4.5 Text interface 55
4.6 Virtual 56
4.6.1 Diagnosis of the processor load by virtual signals 58
4.6.2 16 bit decoder 59
4.6.3 32 bit decoder 61
4.6.4 Multidecoder module 62
4.6.5 16-bit encoder 68
4.6.6 32-bit encoder 69
4.6.7 Electric modules 70
4.6.8 Energy aggregation module 74
4.6.9 ibaQPanel input 76
4.6.10 ibaQPanel text input 78
4.6.11 Trigger module 79
4.6.12 Virtual 84
4.6.13 Virtual retentive 87
4.6.14 Shift register 88
4.6.15 Computation module 90
4.6.16 Computation retentive module 99
4.6.17 Lookup table 100
4.6.18 Process condition 105
4.6.19 Parameter set 111
4.6.20 NMEA 0183 decoder 115
4.6.21 Vector calculation module 118
4.6.22 Phasor module 121
4.7 Unmapped 123
5 ibaNet data interfaces (for iba devices) 124
5.1 ibaNet protocols 126
5.2 ibaFOB-io-, 2io-, 4io- etc. 128
5.2.1 Interface settings 129
5.2.2 PCI Info 130

5.2.3 Card display 130
5.2.4 Info tab (card level) 132
5.2.5 Memory view (card level) 133
5.2.6 ibaFOB-io on link level 133
5.2.7 Info tab (link level) 133
5.2.8 Configuration tab (link level) 136
5.2.9 Memory view tab (link level) 139
5.3 ibaFOB-io-ExpressCard 140
5.4 ibaNet-E 141
5.5 Modules for ibaPADU-series devices 143
5.5.1 Example: ibaPADU-8 144
5.6 Modules for ibaPADU-S-series devices (modular system) 144
5.6.1 Example: ibaPADU-S-CM 146
5.7 Modules for ibaBM series devices (bus modules) 149
5.7.1 Example: ibaBM-DP with active slave 152
5.8 Modules for ibaLink-series devices (system interfaces) 161
5.8.1 Example: ibaLink-VME in 32Mbit Flex mode 162
5.9 ibaNet750/ibaNet750-D module type 166
5.10 FOB Fast module 168
5.10.1 FOB Fast – General tab 169
5.10.2 FOB Fast – Analog tab 170
5.10.3 FOB Fast – Digital tab 172
6 Ethernet-based interfaces 173
6.1 General and common settings 175
6.2 Connection table 176
6.3 AMEPA 176
6.4 AN-X-DCSNet 178
6.5 CAN (Controller Area Network) 179
6.6 EGD (Ethernet Global Data) 181
6.7 EtherNet/IP 182
6.8 FBGS 184
6.9 GCOM 185

6.10 Generic TCP 186
6.11 Generic UDP 188
6.12 ibaLogic TCP 189
6.13 IEC 61850 Client 190
6.14 IEC 61850-9-2 192
6.15 LAND pyrometer 192
6.16 LANDSCAN 193
6.17 LMI-Gocator 194
6.18 Micro-Epsilon 195
6.19 OPC UA 196
6.20 Modbus TCP client 197
6.21 Modbus TCP Server 198
6.22 Raw Ethernet 200
6.23 Raytek 201
6.24 S7 TCP/UDP 201
6.25 Sisteam TCP 203
6.26 TDC TCP/UDP 204
6.27 VIP TCP/UDP 206
6.28 Modules for Ethernet-based interfaces 207
7 PLC-Xplorer interfaces 211
7.1 AB-Xplorer 211
7.2 ABB-Xplorer 212
7.3 Bachmann-Xplorer 214
7.4 B&R-Xplorer 215
7.5 Codesys-Xplorer 216
7.6 Logix-Xplorer 218
7.7 MELSEC-Xplorer 219
7.8 OMRON-Xplorer 220
7.9 S7-Xplorer 221
7.10 Sigmatek-Xplorer 223
7.11 TwinCAT-Xplorer 224

8 Siemens-specific interfaces 226

8.1 FOB-SD/-SDexp and FOB-TDC/-TDCexp 227

8.2 TDC Request and Simadyn Request 229

8.3 CP1616 230

8.4 CP1626 231

8.5 MMC request 233

8.6 ibaFOB-PlusControl 234

8.7 SINAMICS-Xplorer 235

8.8 SIMOTION-Xplorer 236

8.9 SINUMERIK-Xplorer 238

9 Other manufacturer-specific interfaces 240

9.1 DTBox-Request 242

9.2 DGM200E 243

9.3 DGM200P 245

9.3.1 DGM200P – Board information 246

9.3.2 Module type DGM200P 248

9.3.3 Module type DGM200P dig512 251

9.3.4 Module type HPCi Lite 252

9.3.5 HPCi Lite 255

9.4 GH180-Xplorer 256

9.5 Hitachi MicroSigma 257

9.6 HPCi Request 258

9.6.1 HPCi Request – Overview 259

9.6.2 HPCi Request – Diagnostics 261

9.7 ibaFOB-H 261

9.8 Modbus serial 262

9.9 PC Link 264

9.9.1 PC Link module 266

9.9.2 PC Link Dig512 module type 267

9.9.3 PC Link Symbolic module 268

9.10 Reflective memory 270

9.11 ScramNet+ 271

9.11.1 Interface configuration 272
9.11.2 ScramNet module type 272
9.11.3 ScramNet dig512 module type 275
9.12 Toshiba ADMAP JAMI1 276
9.13 X-Pact 277
9.13.1 X-Pact lite module (Reflective Memory) 278
9.14 X-Pact Request 280
9.14.1 X-Pact Request module 282
10 Cloud, database and message broker interfaces 285
10.1 SQL database interface 285
10.2 MQTT interface 287
10.3 HTTP(S) interface 288
11 Other interfaces and ibaPDA add-ons 289
11.1 Audio 290
11.2 ibaInCycle 291
11.3 ibaInSpectra 292
11.4 Snapshots 293
12 Diagnostic modules 294
13 Support and contact 299

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 they are capable of assessing safety and recognizing possible consequences and risks on the basis of their specialist training, knowledge and experience and knowledge of the standard regulations.

1.2 Notations

In this manual, the following notations are used:

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

1.3 Used symbols

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

Danger!

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

Observe the specified measures.

Warning!

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

Observe the specified measures.

Caution!

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

Observe the specified measures.

Note

A note specifies special requirements or actions to be observed.

Tip

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

Other documentation

Reference to additional documentation or further reading.

1.4 Documentation structure

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.

Part 1	Introduction and installation	General notes, license policy and add-ons Installation and program start User interface, system architecture, client server User management, printing
Part 2	I/O Manager	Basic information about the I/O Manager, general settings Groups and vector signals, text signals, outputs, configuration files
Part 3	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.
Part 4	Expression builder	All functions for calculating virtual signals
Part 5	Data storage	Types of data recording, recording profiles, signal selection
Part 6	Data visualization	All display modes for live data, their operation and settings
Part 7	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 30*.
- 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 124*.
- 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 173*.
- 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 211* 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 211* 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, ibaFOB-SD/-SDexp, CP1616/CP1626 and MMC Request.
  See ↗ *Siemens-specific interfaces, page 226* 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 240*, ↗ *Other interfaces and ibaPDA add-ons, page 289* 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 285* 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!

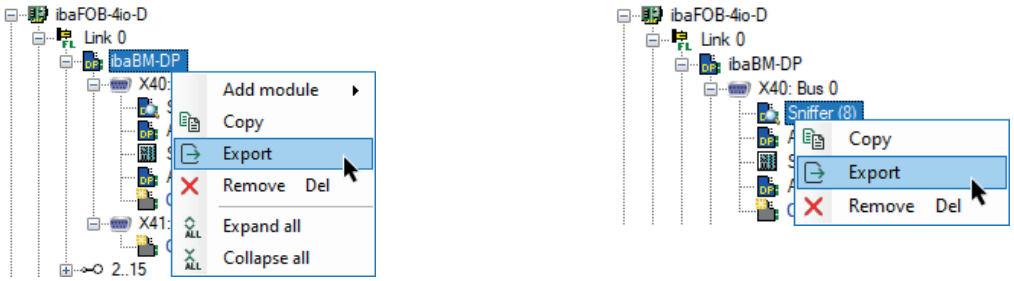
If an existing module cannot be used, e.g. because the corresponding hardware device is temporarily not in operation, it is recommended to shift the module by drag and drop to the "Unmapped" interface node. This will remove the module from the active configuration, but keep the module and signal settings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

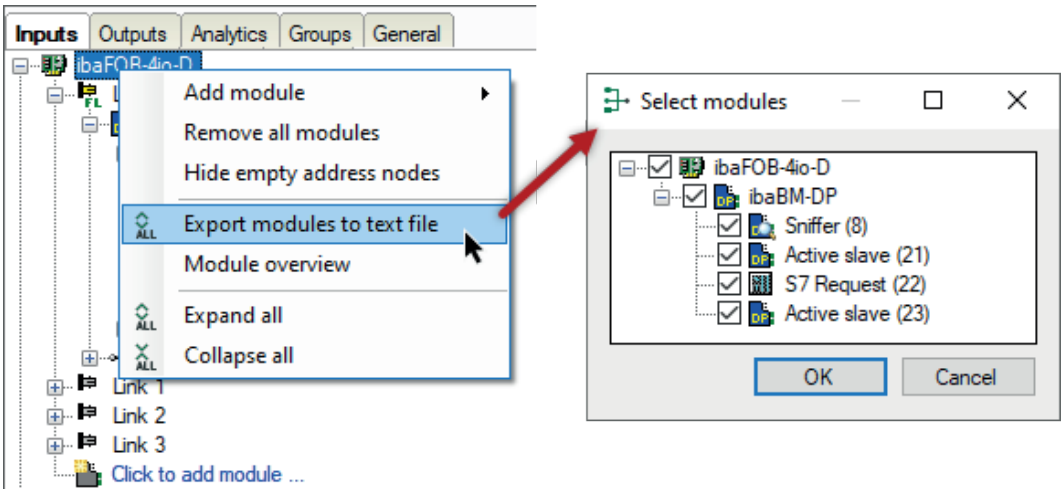
Export of single modules is possible. If there is no need to export an entire I/O configuration, you can export a single module. Just right-click on the module in the I/O Manager tree and select *Export* from the context menu. In case of modules and sub-ordinated sub-modules, the export consists either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

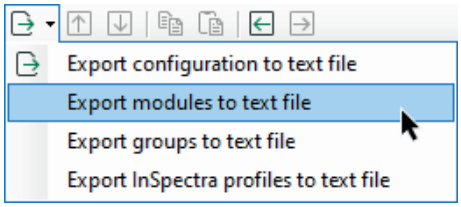
In the example below, on the left side, the ibaBM-DPM-S, Sniffer, S7 Request modules and two Active slave modules are exported. The export on the right side includes just the Sniffer module.



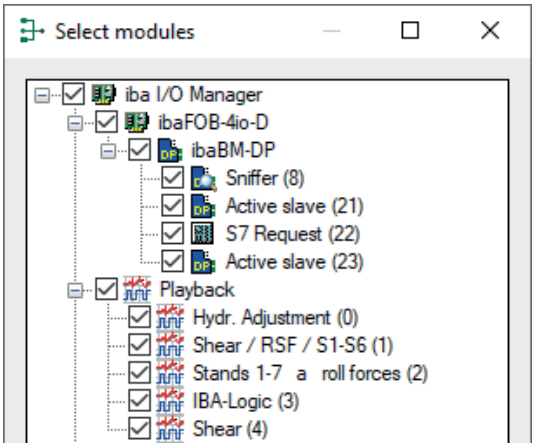
If you call up the export function on the link or interface level, you first see a dialog in which you can deselect modules that you do not wish to export.



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.

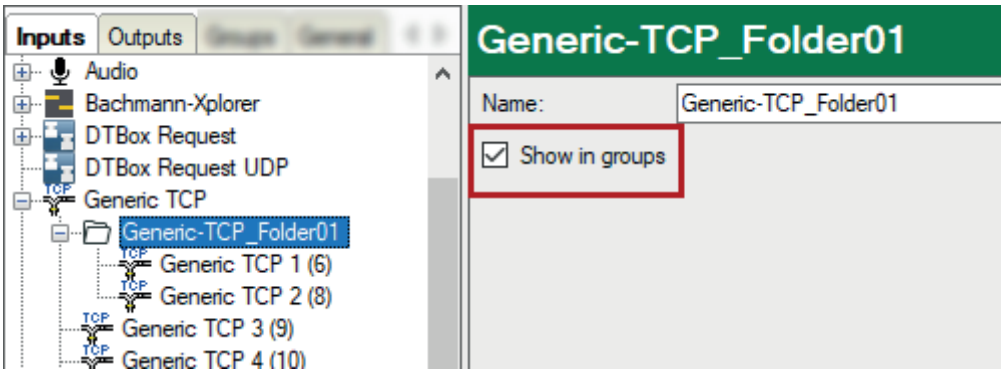
Copy folders
You can copy folders and their sub-folders you have created for one interface to other interfaces. Thus, you can easily transfer folder structures you've created once to other interfaces.

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*.

Comment
You can enter a comment or a description of the module here. Inside *ibaPDA* this will be displayed as a tool tip in the signal tree on mouse-over. The information is also written to the data file where it can be found in *ibaAnalyzer* in the info branch of the module (info field "comment").

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.

When using *ibaPDA* together with *ibaCMC*, the module number range from 100000 to 100500 is reserved for *ibaCMC*.

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:

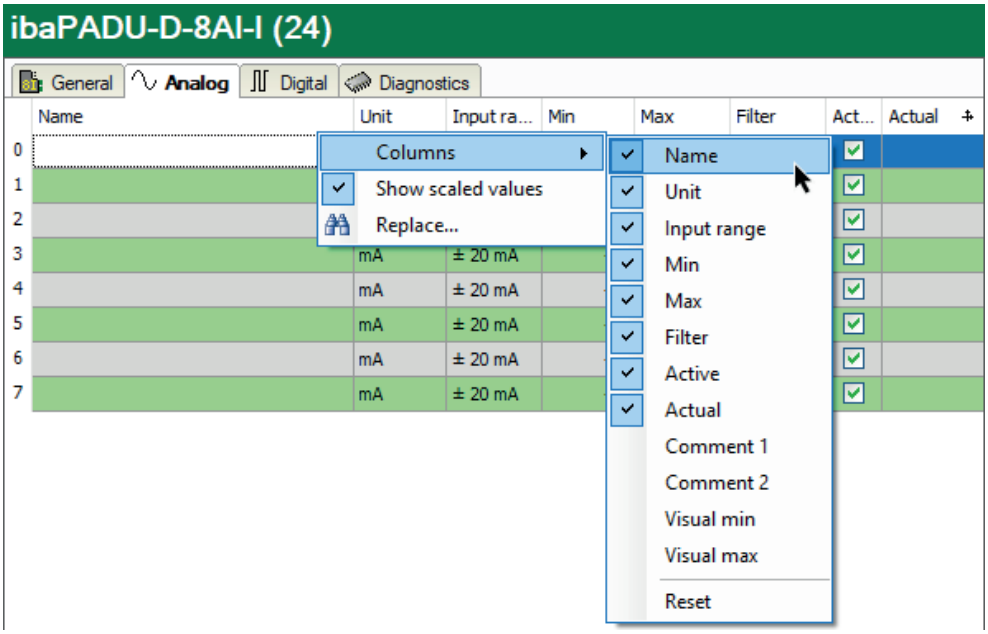
- Column visibility
- Column order
- Column width

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.

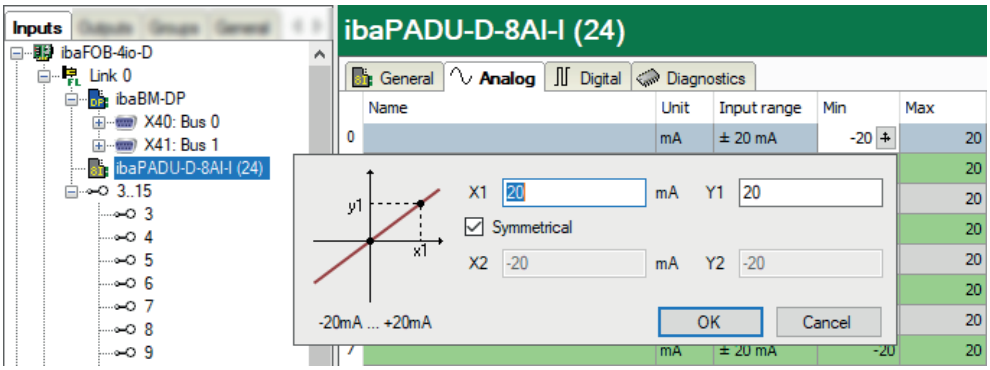
The analog voltage level of +10 V is assigned to the temperature 43 °C.

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.

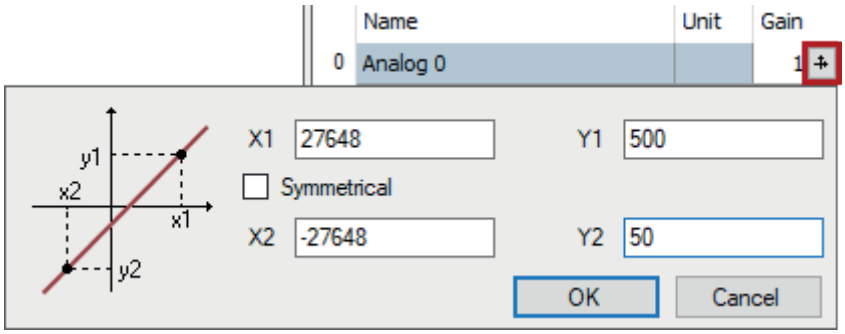
Gain and offset can be entered directly or set by means of the two-point scaling with two pairs of applicable values.

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 ±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 overwitten.

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	↗ *ibaNet-E, page 141*
ibaCapture	Connection to ibaCapture server, ibaVision process	↗ *ibaCapture, page 31*
OPC	OPC (DA) interface for OPC Client and Server	↗ *OPC interface, page 34*
Playback	Program interface for playback of iba data files	↗ *Playback, page 43*
Signal mapper	Program interface for assigning input signals to internal signals	↗ *Signal mapper, page 48*
Text interface	Connection to sources of text signals, such as files, serial interface, TCP, UDP	↗ *Text interface, page 55*
Virtual	Program interface for computation and processing of virtual signals by means of mathematical, technological and many more functions	↗ *Virtual, page 56*
Unmapped	Storage for unused/unassigned modules	↗ *Unmapped, page 123*

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. [1)]

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

[1)] 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 33* 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.

Note

Due to a vulnerability in the DCOM protocol, Microsoft has released security update KB 5004442, which was activated in the final stage on March 14, 2023, and affects the connectivity of OPC components.

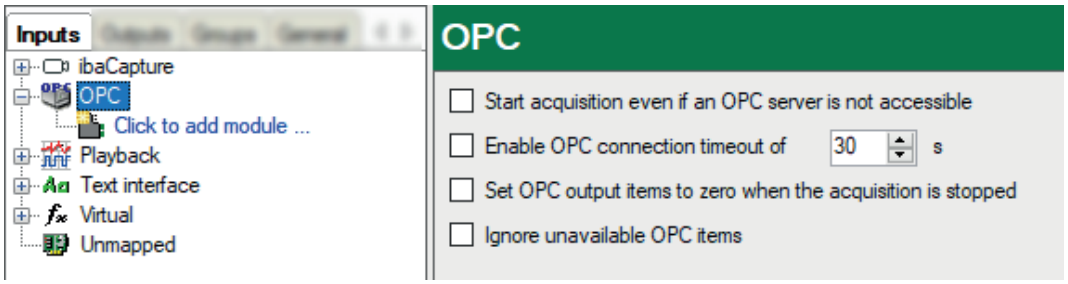
With the activation of the new security feature, only DCOM connections (network connections) between OPC servers and clients that support packet-based authentication are possible. Communication between OPC Classic servers and clients running on the same computer is not affected.

Detailed information about the effects of KB 5004442 DCOM security update is available on the Microsoft website (https://support.microsoft.com/en-us/topic/kb5004442-manage-changes-for-windows-dcom-server-security-feature-bypass-cve-2021-26414-f1400b52-c141-43d2-941e-37ed901c769c)

iba AG does not provide support for problems with OPC Classic remote network connections. Depending on the application, there is also the option of communicating via OPC UA instead, thereby avoiding communication via DCOM.

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 37*
- Module type OPC server, see ↗ *OPC Server module, page 42*
- Module type redundant OPC client, see ↗ *Redundant OPC client module, page 41*

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

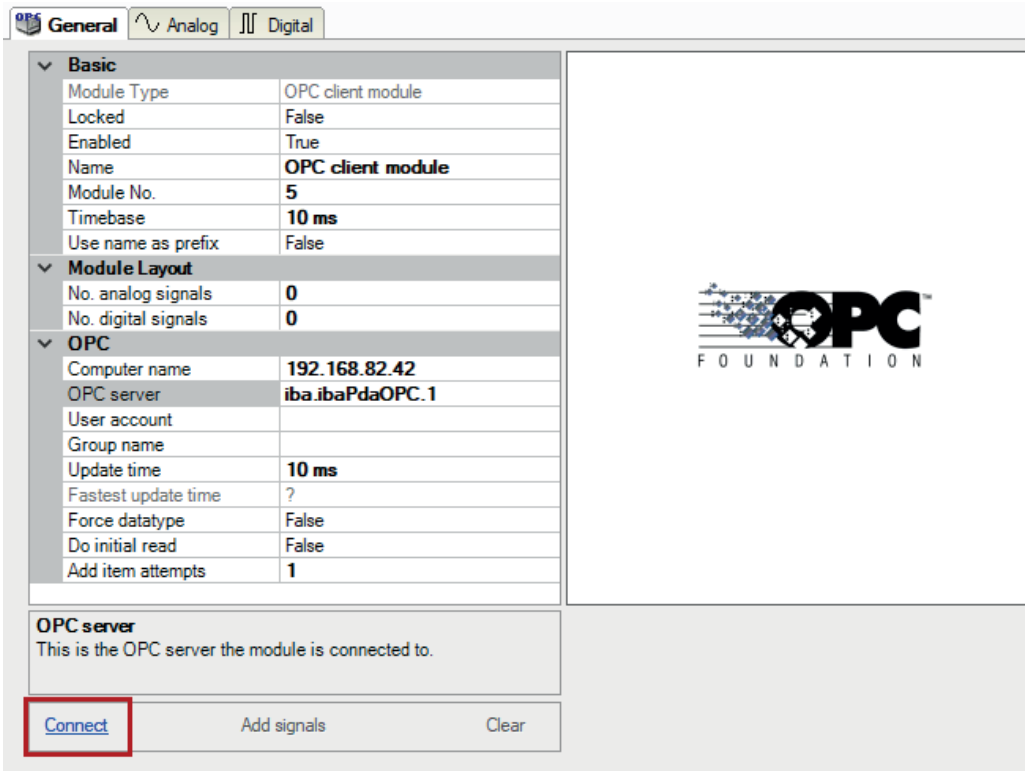
At the bottom of the *General* tab, you can find three hyperlinks:

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

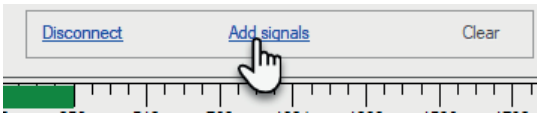
As long as there are not entries in the signal tables, the hyperlinks "Add signals" and "Clear" are disabled.

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

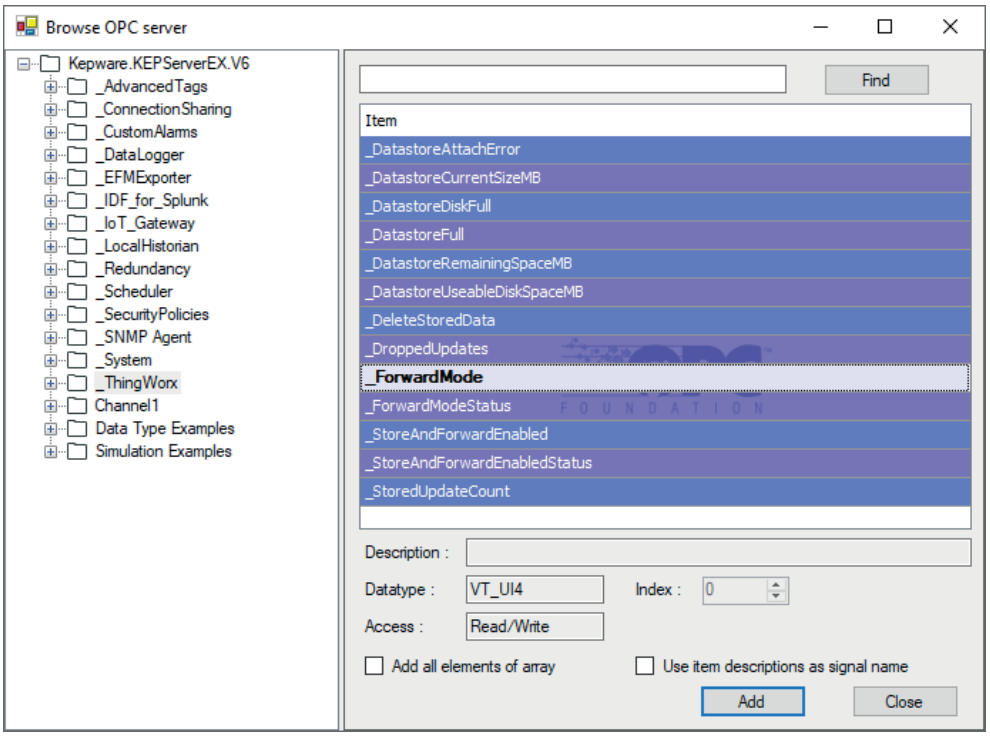
1. After you have made the settings for the OPC server in the *General* tab, click on "Connect".



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 23.

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 23.

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 23.

4.3 Playback

This interface is to be used to replay one or multiple iba data files (*.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. 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.

The replay can be controlled by digital signals (pause/reset).

Possible applications are, for example:

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

The replay speed is adjustable. If needed, you may replay a data recording with a higher or reduced speed, compared to the original recording speed of the file.

Furthermore, you can add a diagnostic module. The diagnostic module provides information about replay speed, current position in the file, total playback time and the full path and file name of the playback file as analog signals. The pause status is provided by a digital signal.

Using playback

1. Open the *ibaPDA* I/O Manager.
2. Click on the *Playback* node in the I/O Manager tree.
3. Check the *Active* option.
4. Select the data file(s).
   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(s) has been protected with a file password, enter the password in the field *File password*.
   The data files will be entered in the *Dat files* field in the order of their selection. If you select multiple files in the browser, the order will follow the sorting of the files in the browser (by name, by date etc.). You may change the order of the files in the *Dat files* field afterwards by editing the file names in this field.
5. Click button <Read dat files>.
   This instructs the *ibaPDA* server to read out the data file and generate the modules and signals that are available in the data file. If you select multiple data files, these will be appended without gaps and displayed in the preview window.

6. Now choose how to proceed.
   There are multiple options available, depending on whether or not playback modules already exist in the configuration:

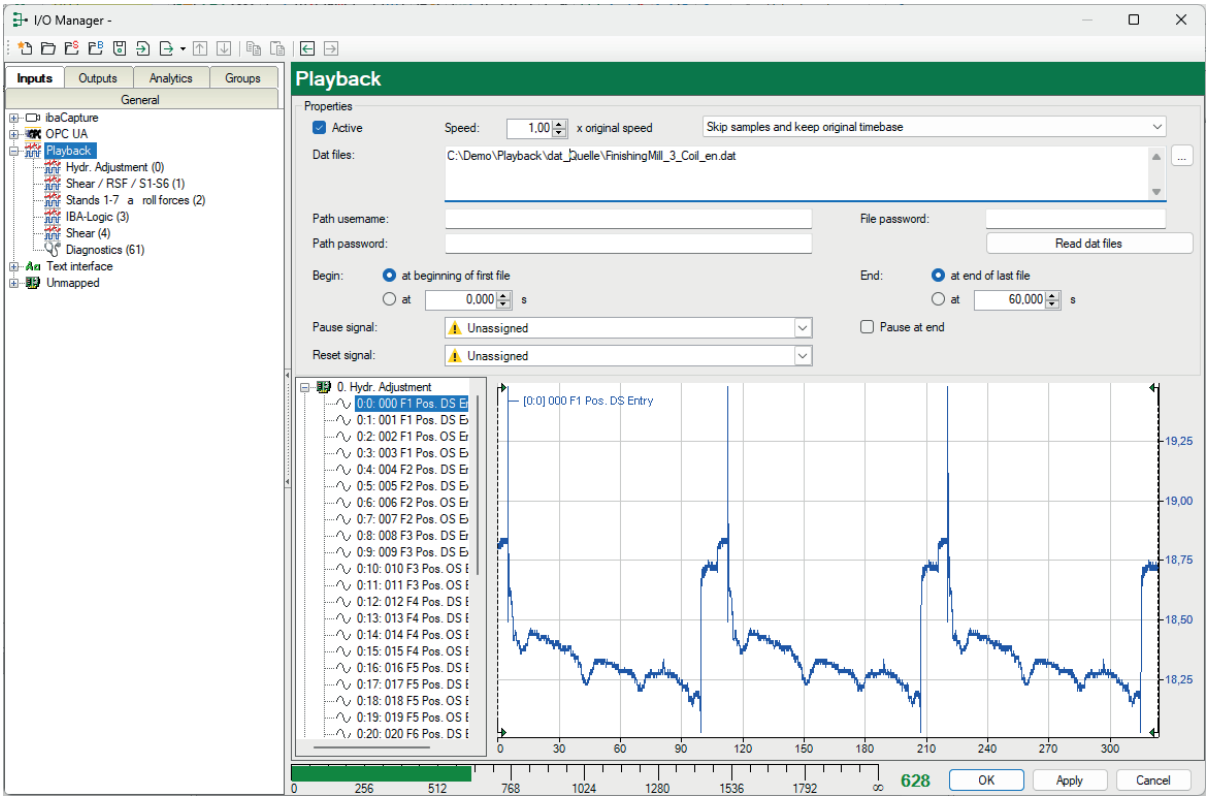
   If no playback modules are available, choose from:

   - 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 modules
     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(s), 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(s) from start to end. You can change the start and stop position by moving the green start and stop marker with the mouse or by entering the start and stop position manually.
8. Select the control signals if you want to control the replay. If you don't select control signals, the range between the two markers will be replayed continuously without interruption as long as the acquisition runs. You can select two signals for controlling the replay:
   - Pause signal
     The replay stops when the selected digital signal is true and it runs when the signal is false.
   - Reset signal
     The replay starts again from the beginning i.e. from the configured starting point when the selected digital signal gets a rising edge.

9. 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.
10. 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.

    If you set up a higher replay speed, e. g. factor 2.00, then choose one of the two options from the drop-down list nearby:

    - Skip samples and keep the original timebase
      Samples may get lost. For example, if you set the replay speed to 2 x original speed, then only every second sample will be returned.
    - Keep all samples and change timebase
      This ensures, that every sample in the data file will be acquired. In exchange, the timebase will be reduced, e. g. halved at doubled speed.

    The following example shows the differences between varying playback speed settings.
11. 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 different playback speeds

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

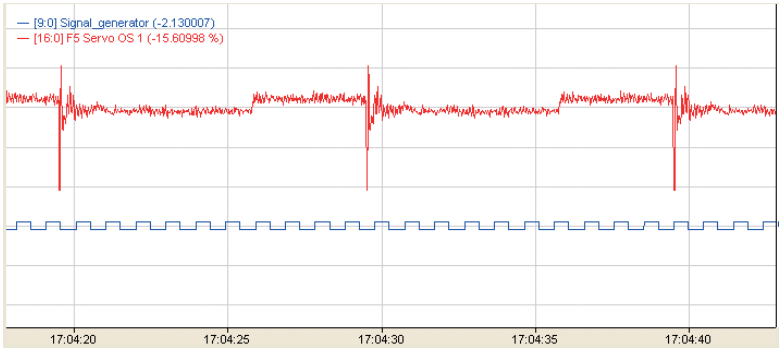
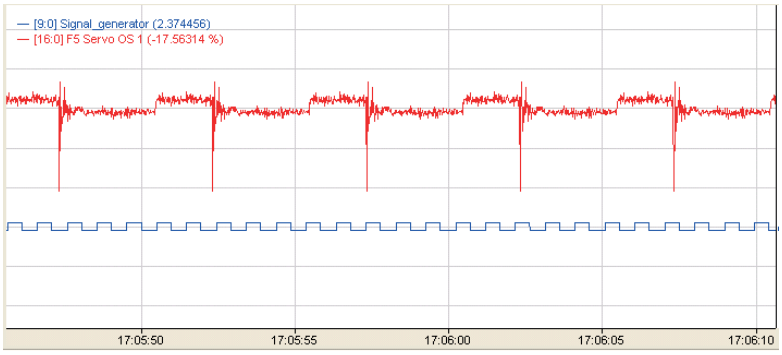
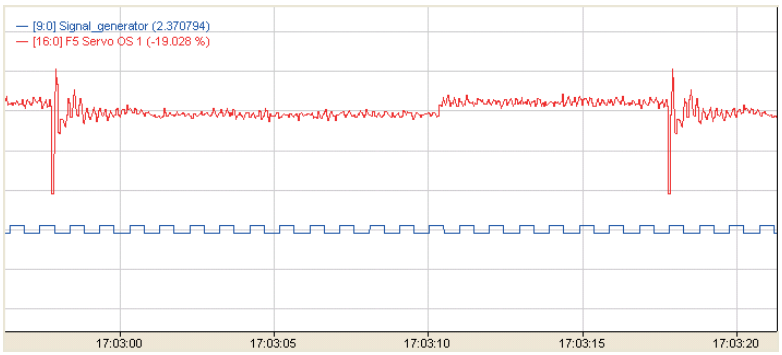
a) original speed

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.

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

Note

Please note, that measured samples from the replayed file can get lost when increasing the replay speed while the option *Skip samples and keep the original timebase* is enabled. For example, if you set the replay speed to 2 x original speed, then only every second sample will be returned. If you would create a new data file based on the replayed data, the data file would be half as long as the original file.

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↗ *Signal mapper module*, page 48	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.

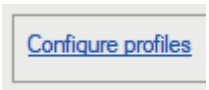
Always when you create a module, you can configure a profile. But you can also configure the required profiles first and create the modules later.

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 **.signalMapperProfile* file
	Export selected profile to a **.signalMapperProfile* 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

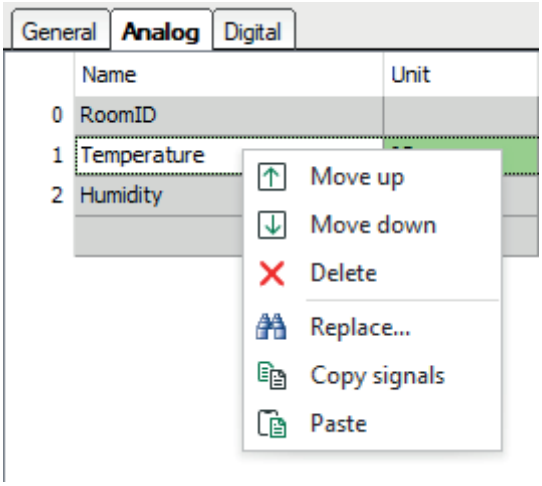
Here, define the analog and digital signals which any signal mapper module will provide after having assigned this profile. If you have filled one row (at least the signal name) and click in the empty space below, a new row will be added.

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 rows up
	Move selected rows down
	Delete selected rows
	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.

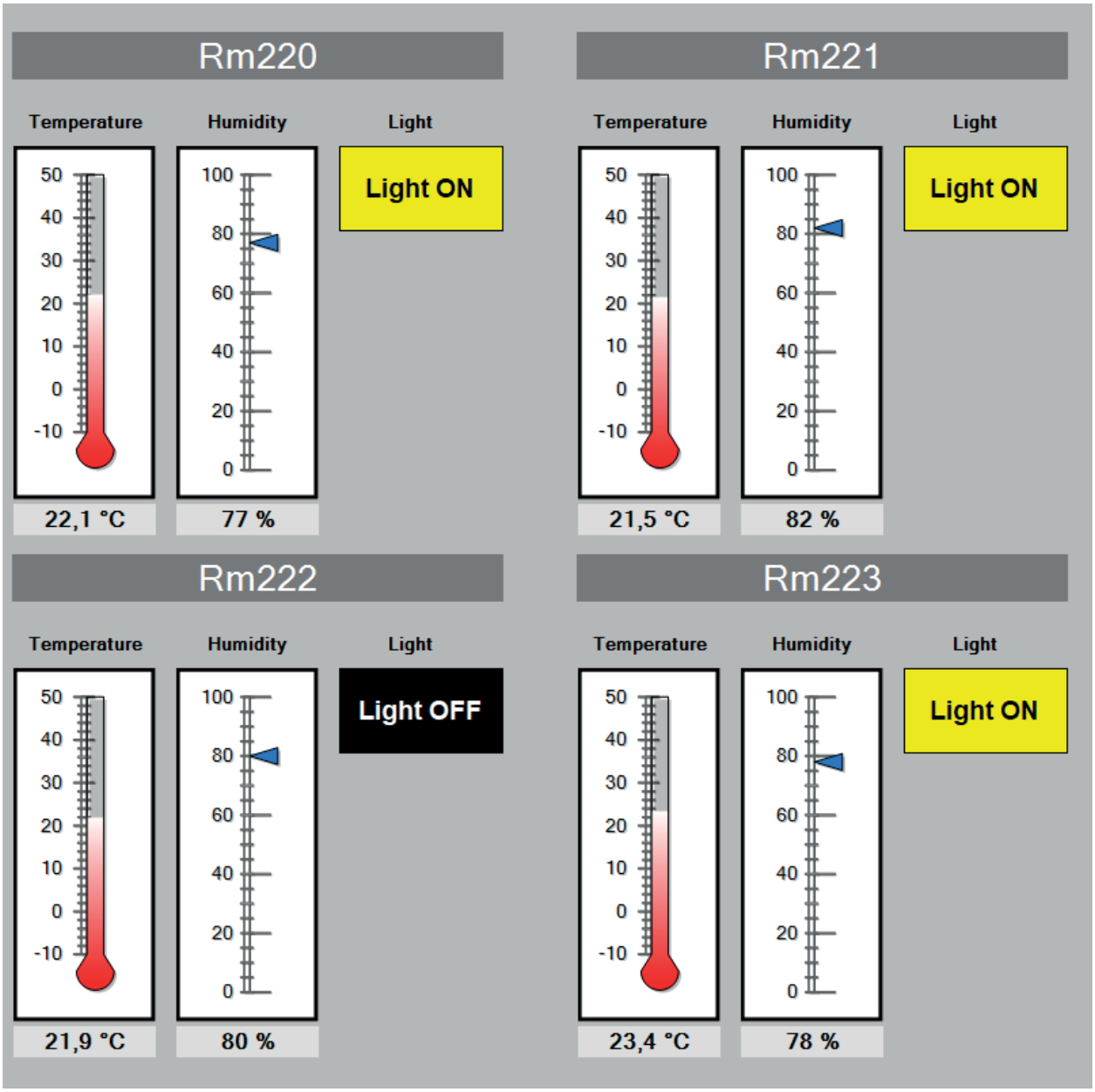


4.4.3 Signal mapper example

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.

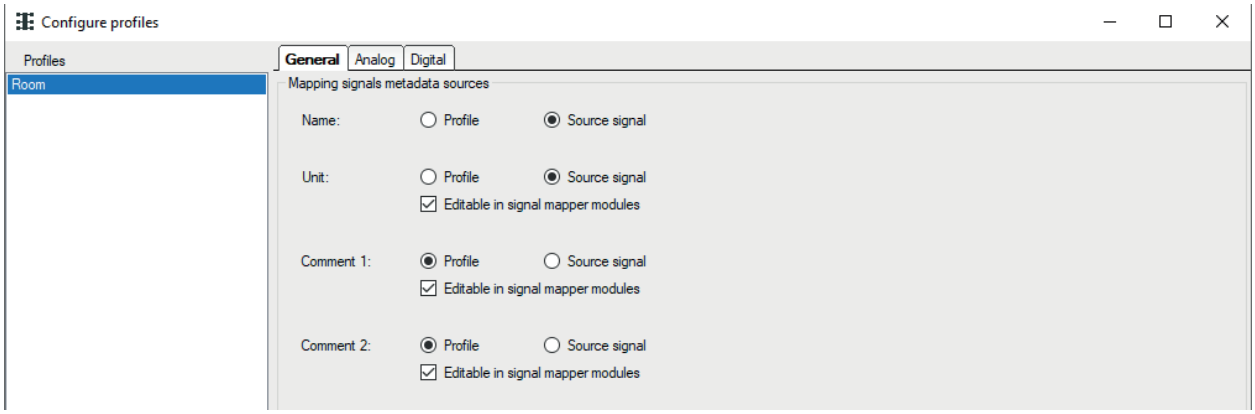
The QPanel layout should be designed before the real input signals are available. Hence, signal mapper modules will be used for the data supply of the QPanel elements.



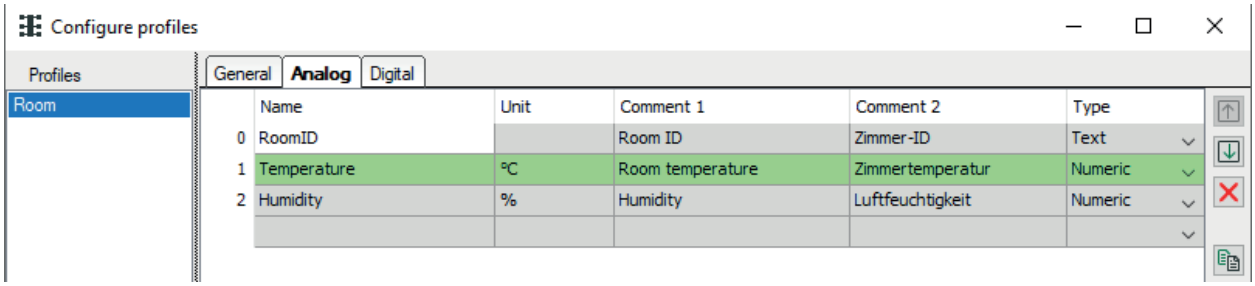
Implementation

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")

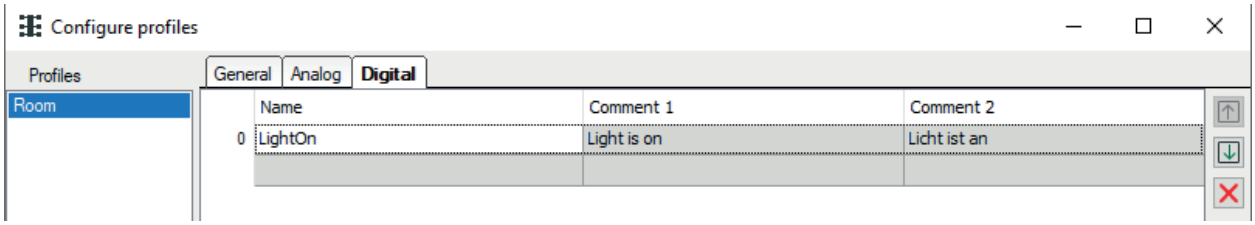
4. On the General tab of the profile decide which metadata should be adopted from the source signal and which should be retained by the profile. In this case, signal name and unit will be adopted from the source signal.



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

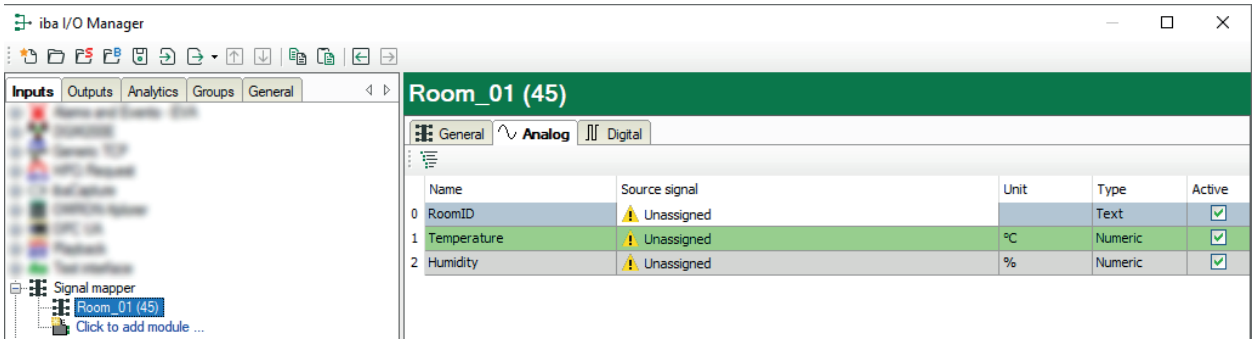


6. On the *Digital* tab define the digital signals, which are needed for the visualization (LightON).



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

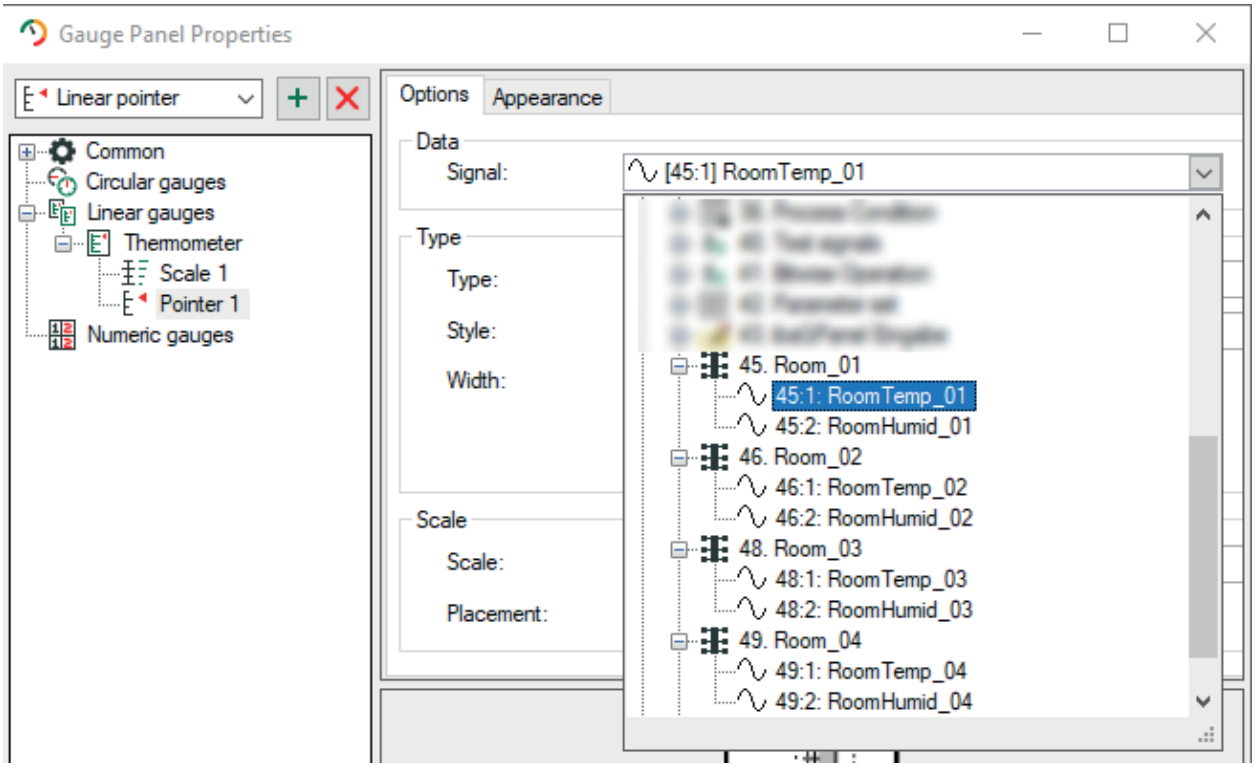
8. Together with the profile the module automatically gets its *Analog* and *Digital* tabs containing the defined signals. The source signals are not assigned yet.



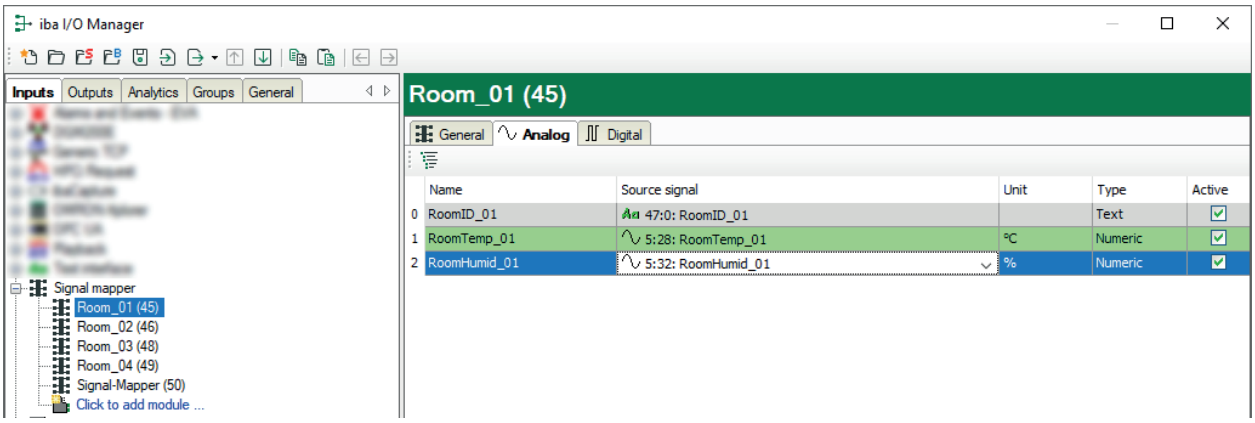
9. Copy the signal mapper module three times and rename each of them, resulting in four modules "Room_01"..."Room_04". All four modules still have the same signal names.

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.



11. 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	*ibaPDA* receives the text as a TCP/IP telegram via the network interface.
UDP text	*ibaPDA* receives the text as a UDP datagram via the network interface.
Serial text	*ibaPDA* 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 *ibaPDA*.

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
↗ *16 bit decoder, page 59*	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)
↗ *32 bit decoder, page 61*	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)
↗ *Multidecoder module, page 62*	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)
↗ *16-bit encoder, page 68*	Packs multiple digital signals into one 16-bit integer analog signal	Transmission of multiple binary data in one analog value (e.g. status word)
↗ *32-bit encoder, page 69*	Packs multiple digital signals into one 32-bit integer analog signal	Transmission of multiple binary data in one analog value (e.g. status word)
↗ *Electric modules, page 70*	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.
↗ *Energy aggregation module, page 74*	Calculates the energy based on a power signal and various time ranges.	Energy
↗ *Trigger module, page 79*	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
↗ *Virtual, page 84*	Provides customized virtual signals created with the expression builder.	Any mathematical or logical calculation

Module types	Function	Application fields
↗ *Virtual retentive, page 87*	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.
↗ *Shift register, page 88*	Stacks actual values of a signal, controlled by a trigger.	Visualization of the last n values
↗ *ibaQPanel input, page 76*	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.
↗ *ibaQPanel text input, page 78*	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, *Text shift register*.	Stacks actual values (texts) of a text signal, trigger-controlled.	Visualization of the last n values (texts)
Text creator See part 2, *Custom text (text creator)*.	Used to configure custom text signals	Creation of static or dynamic texts
Text splitter See part 2, *Text splitter module*.	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
↗ *Computation module, page 90*	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.
↗ *Computation retentive module, page 99*	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.
↗ *Lookup table, page 100*	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.
↗ *Parameter set, page 111*	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.

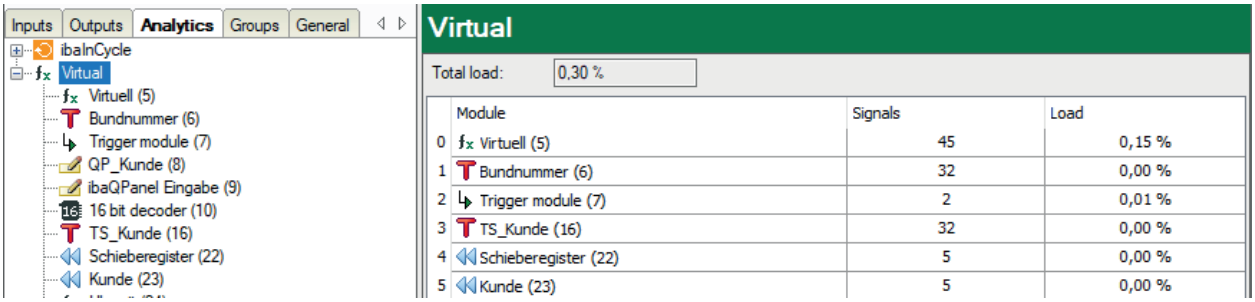
Module types	Function	Application fields
↗ *Phasor module, page 121*	Computation of RMS value, phase angle, effective frequency and ROCOF of a current or voltage signal	Acquisition of charcteristic values of a signal in the field of electrical power.
↗ *Process condition, page 105*	Definition of different 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.
↗ *NMEA 0183 decoder, page 115*	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
↗ *Vector calculation module, page 118*	Calculation of statistical values 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.

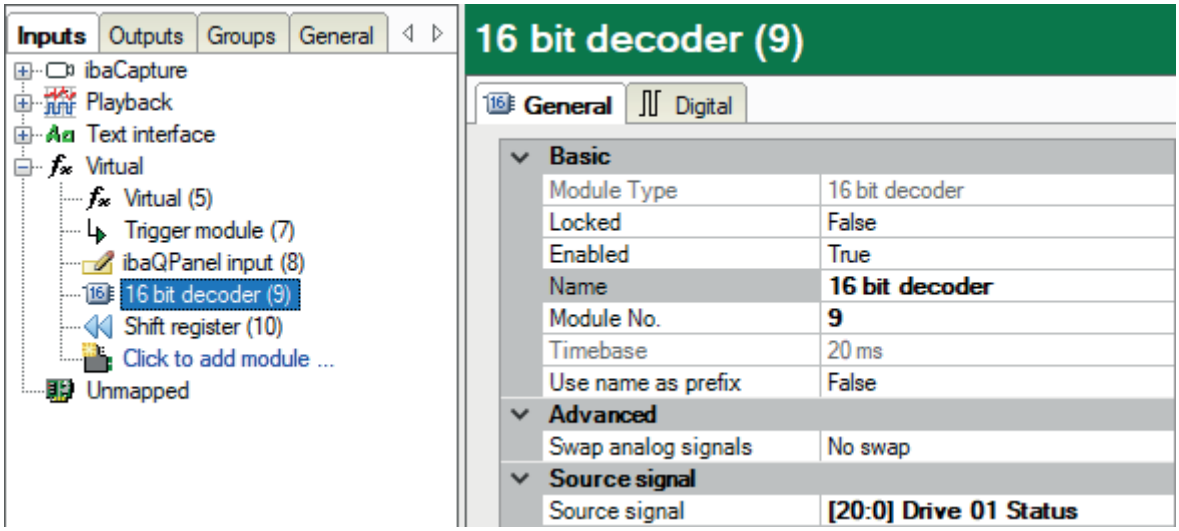


The number of signals contained and the proportional load are displayed next to the name of the module.

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

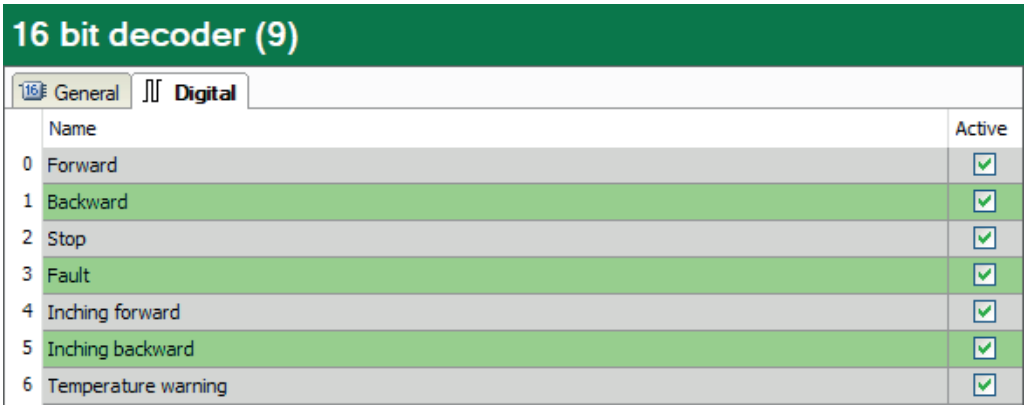
Select the source signal to be decoded from the signal tree.

4.6.2.2 16 bit decoder – Digital tab

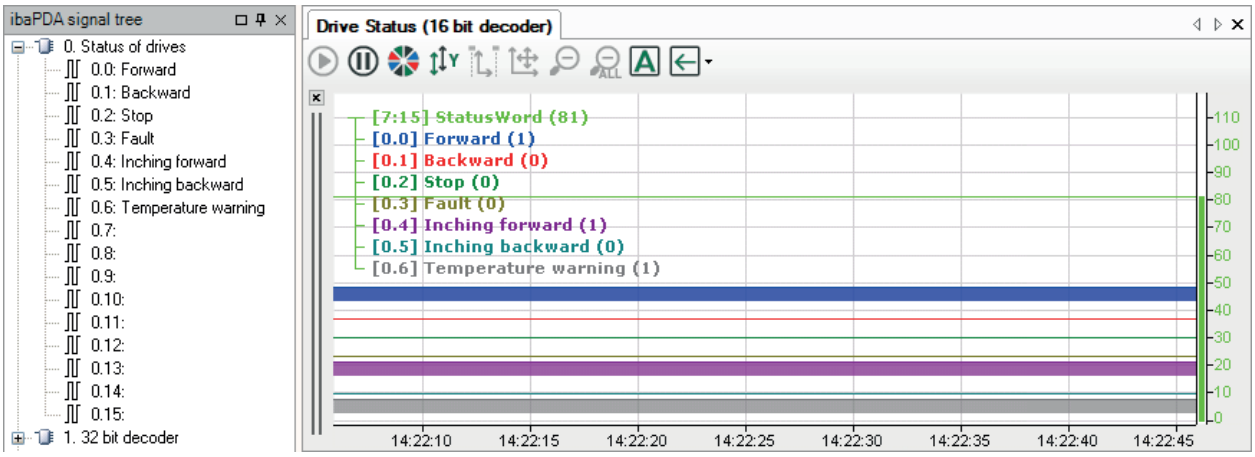
In the *Digital* tab, you can name the 16/32 digital signals.

Example

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



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.

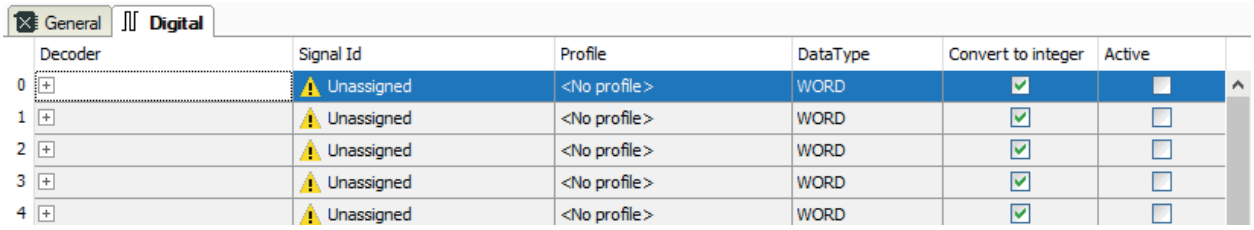
The discrete decoders can be found in the *Digital* tab.

"Configure profiles" link
Clicking on this link opens the configuration dialog for profiles.

It's an 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.

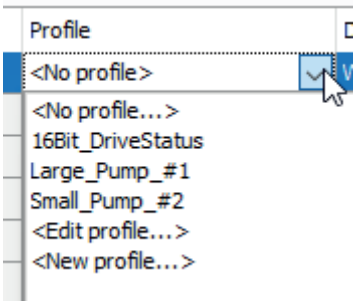


Each group line stands for one decoder. You do the basic set up of the decoders in the columns of each group line.

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 endianess 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.

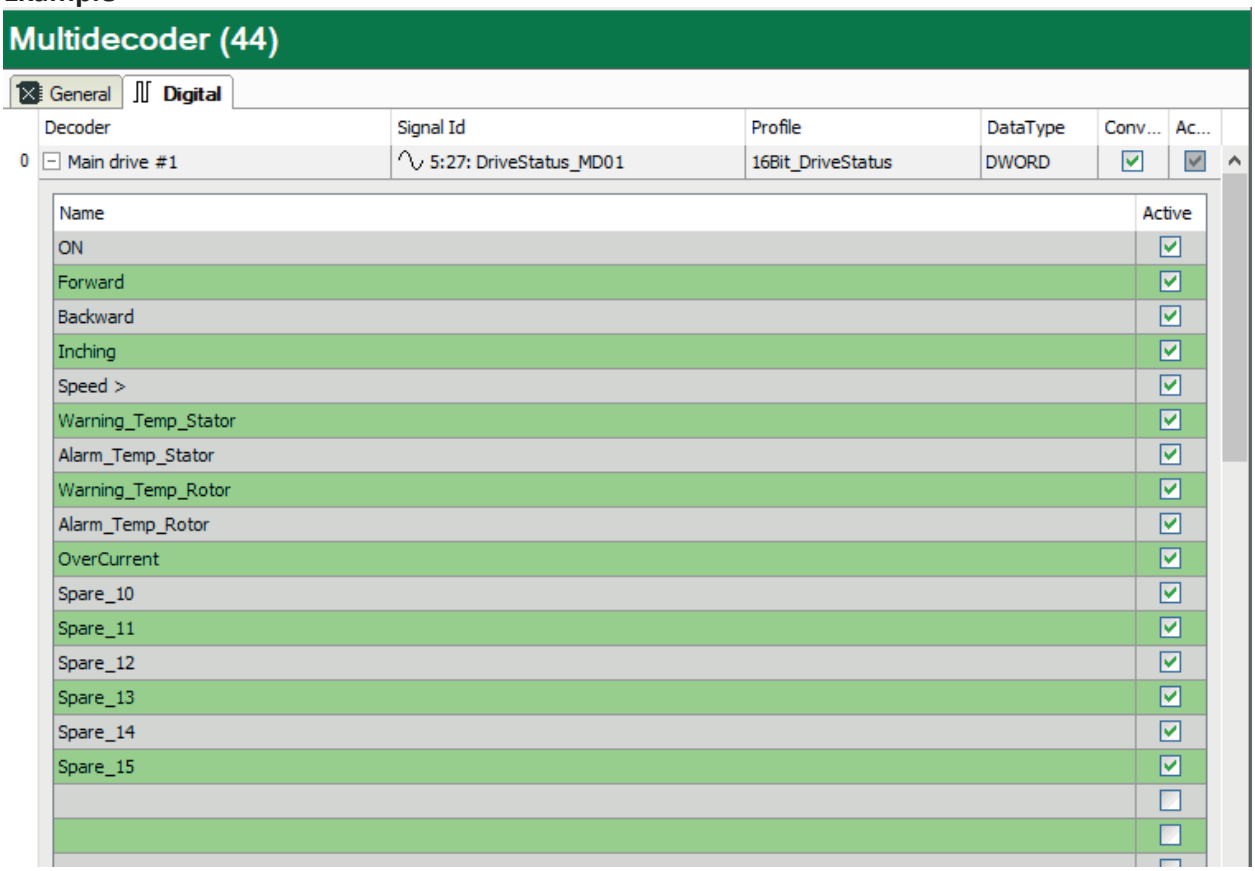
If no profile has been assigned yet, the table is still empty.

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.

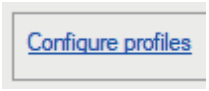
If a profile is assigned, the bits cannot be changed.

Example



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 **.multiDecoderProfile* file
	Export selected profile to a **.multiDecoderProfile* 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.

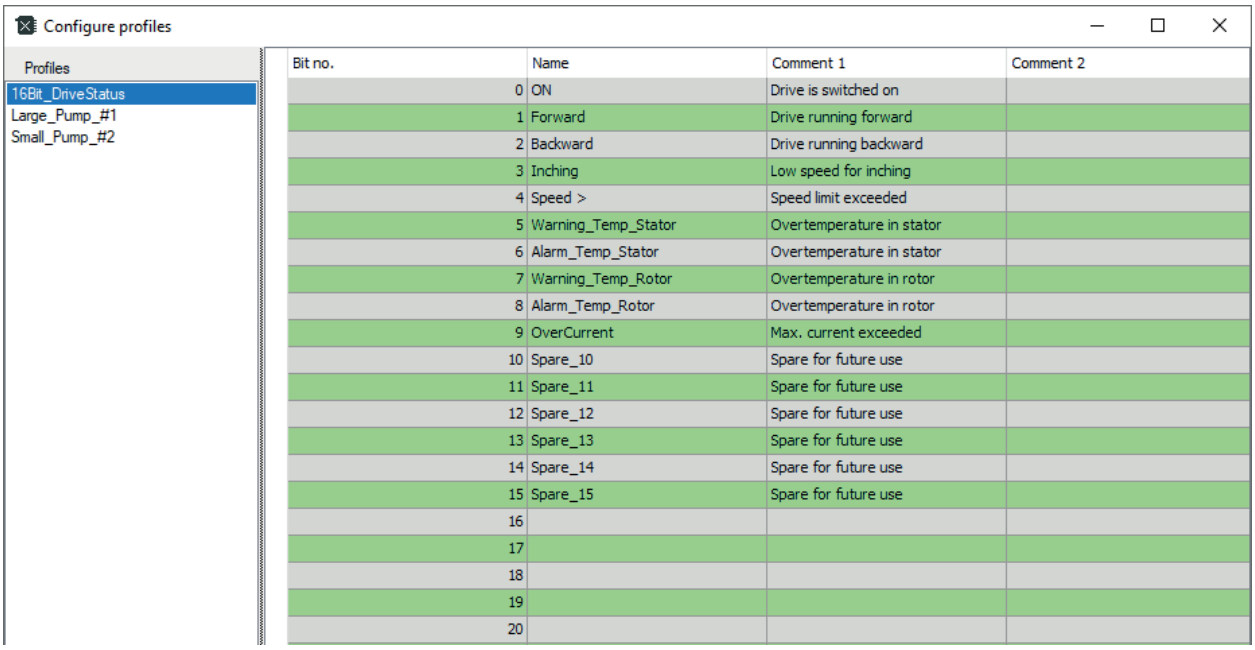
These names will be the signal names of the digital signals. If needed, add some comments. The bit number in the first column is for information only and cannot be changed.

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

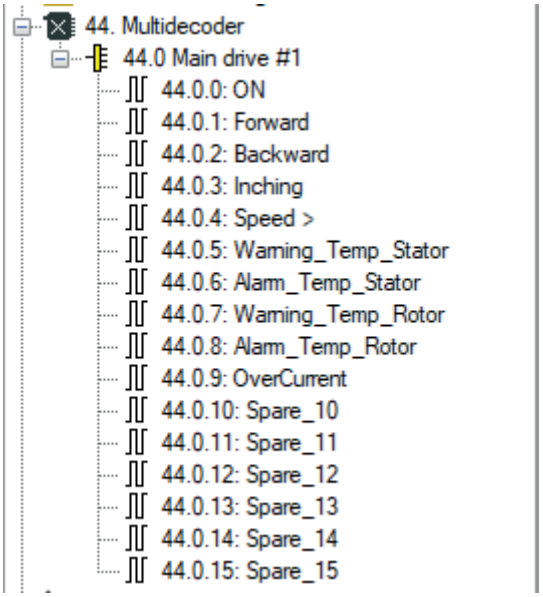


Bit no.	Name	Comment 1	Comment 2
0	ON	Drive is switched on	
1	Forward	Drive running forward	
2	Backward	Drive running backward	
3	Inching	Low speed for inching	
4	Speed >	Speed limit exceeded	
5	Warning_Temp_Stator	Overtemperature in stator	
6	Alarm_Temp_Stator	Overtemperature in stator	
7	Warning_Temp_Rotor	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).

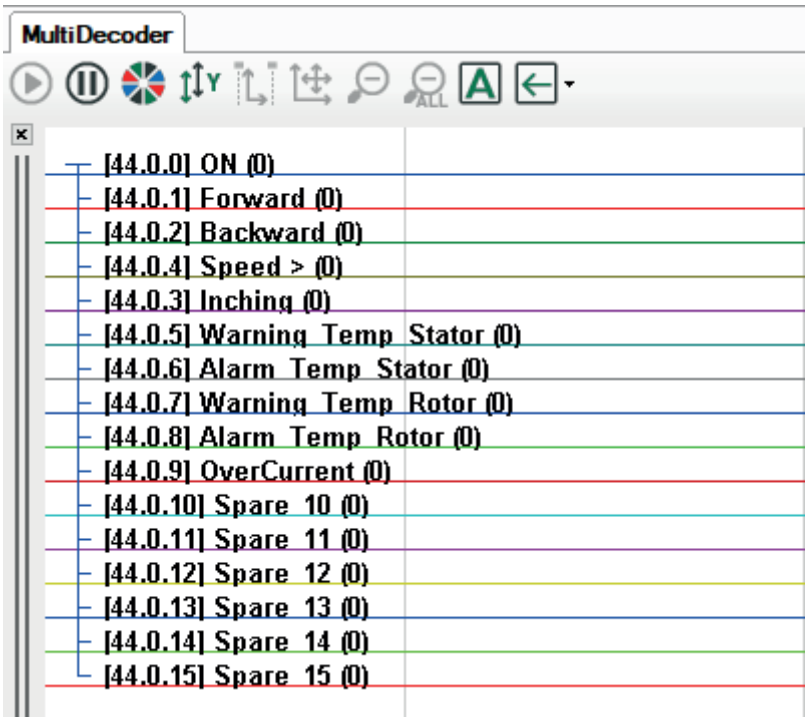


You can drag and drop the signals into a trend graph.

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.



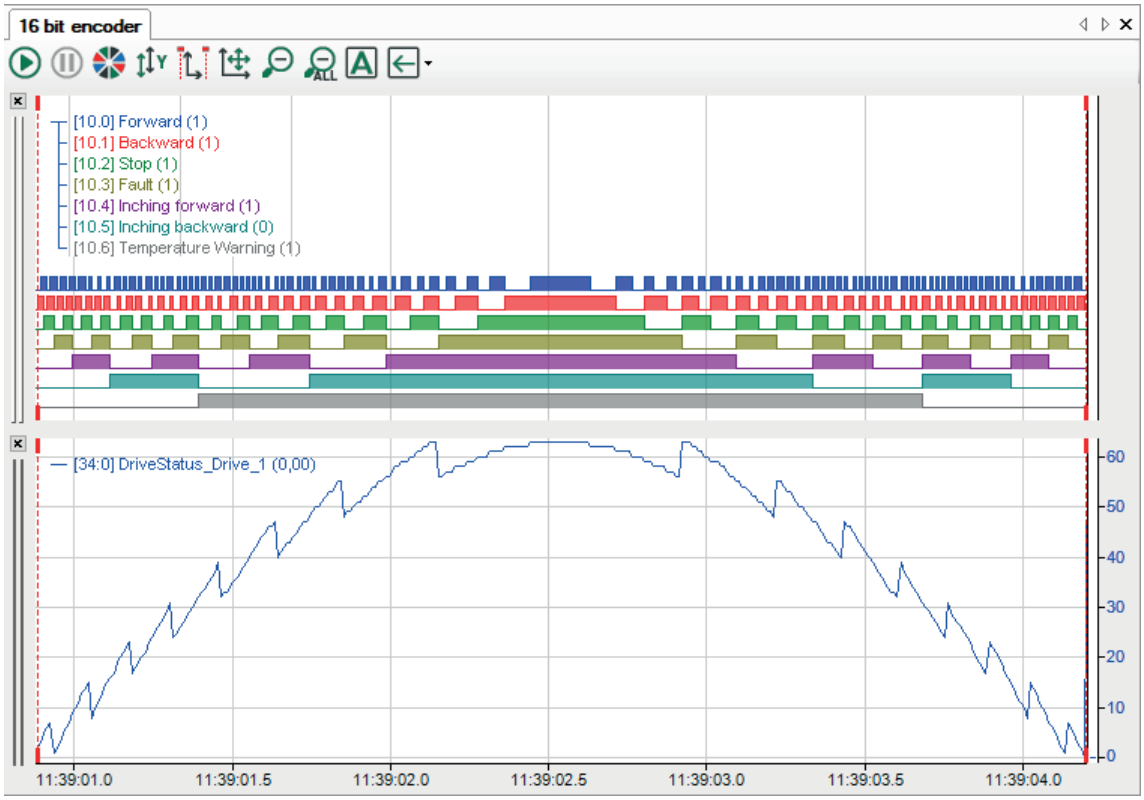
4.6.5 16-bit encoder

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

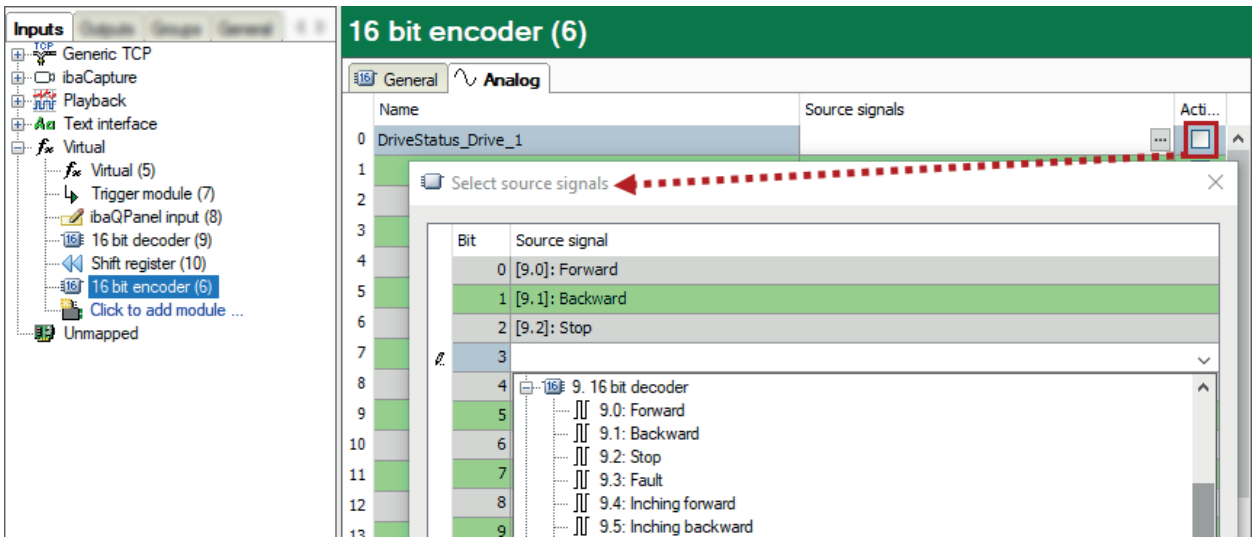
Set the desired number of analog signals that this module should supply here.
Value range: 1 … 1000 (default: 32)

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.

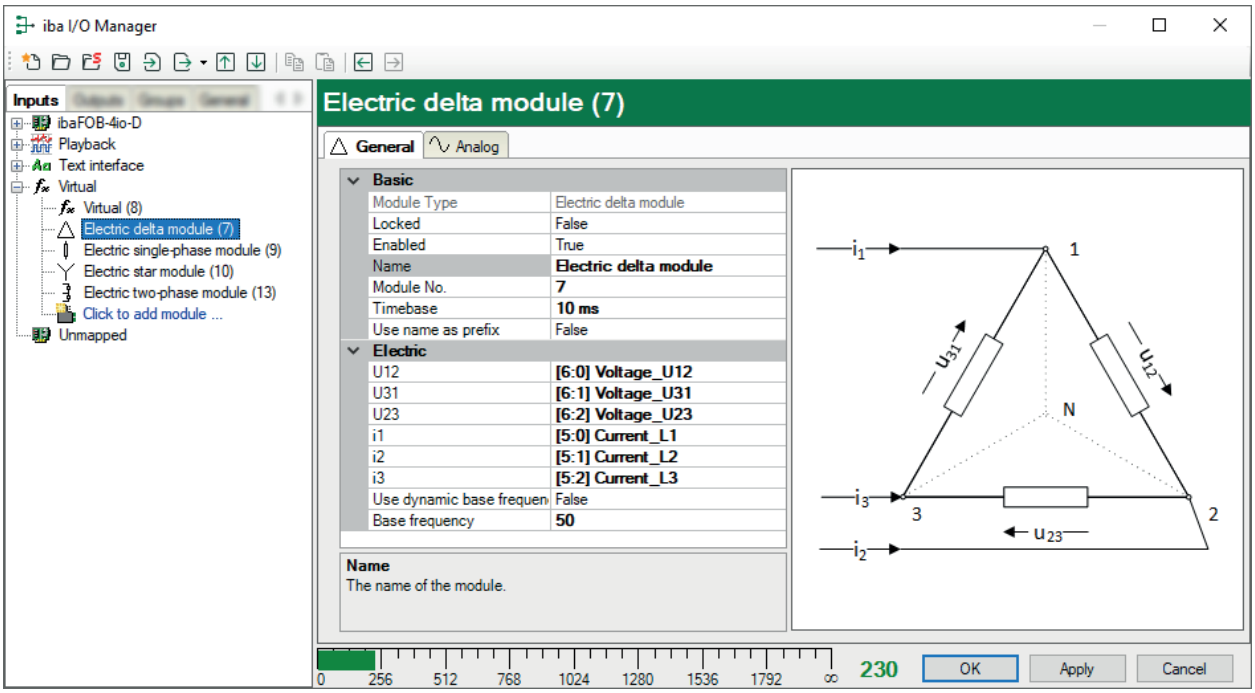
It is configured in the same way as the 16-bit encoder module.

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 u2, u3 and i2, i3, i4 equal to zero. The following figure shows an example for a delta module.



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 / base frequency, as only one new value will be available in each period.

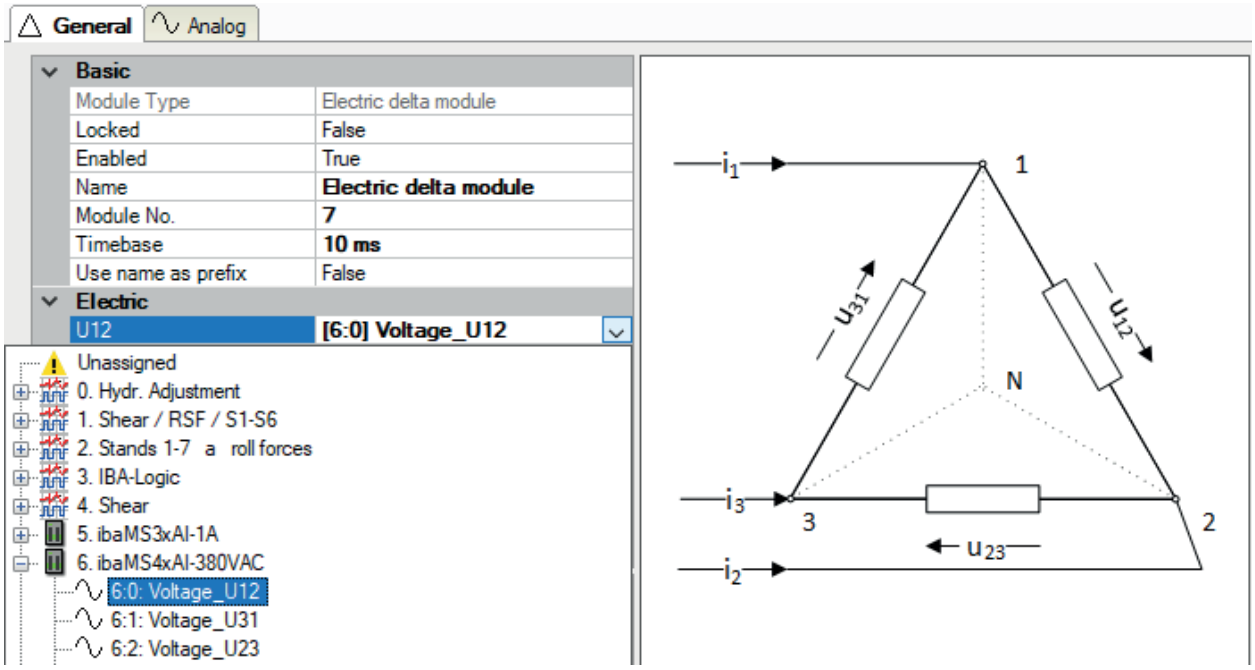
Voltages and currents

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

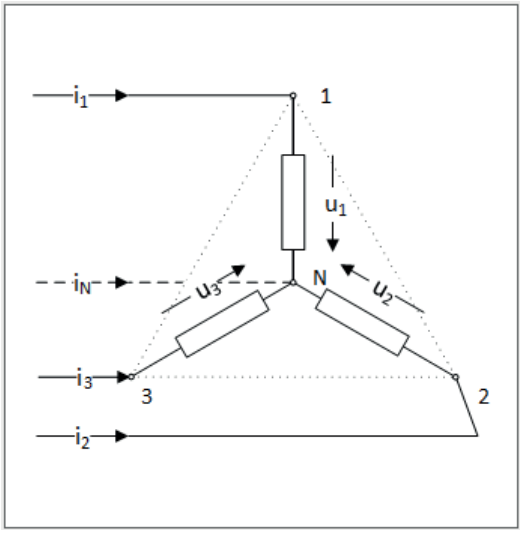
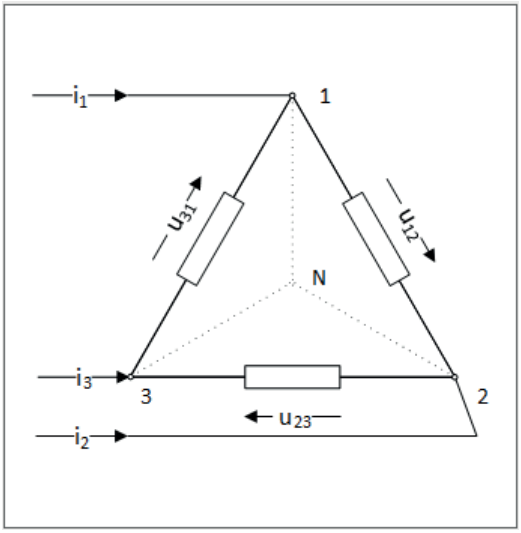
If a certain base frequency has been specified, set the “Use dynamic base frequency” option to FALSE and enter the base frequency in the field below.

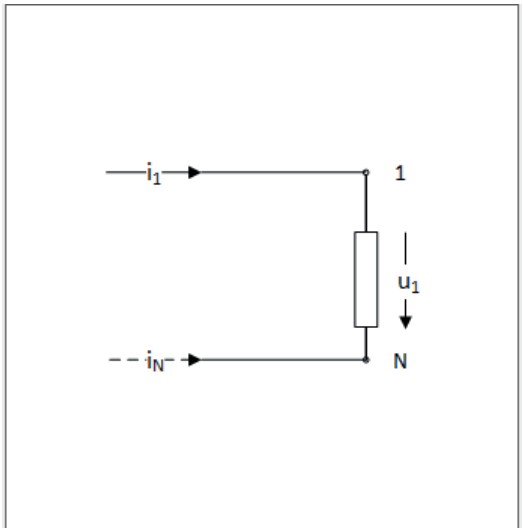
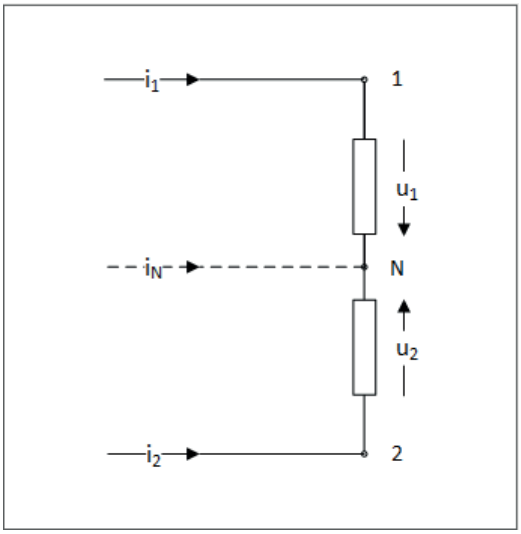
If the base frequency varies and the frequency value should be a separate signal, set the “Use dynamic base frequency” option to TRUE and select the appropriate frequency signal from the selection list in the field below. The following figure shows the assignment of the first voltage signal.



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
	
effective voltage U_1	effective voltage U_{12}
effective voltage U_2	effective voltage U_{31}
effective voltage U_3	effective voltage U_{23}
effective current I_1	effective current I_1
effective current I_2	effective current I_2
effective current I_3	effective current I_3
effective current I_N	total effective voltage
total effective voltage	total effective current
total effective current	real power
real power	apparent power
apparent power	reactive power
reactive power	reactive power, signed
reactive power, signed	power factor
power factor	reactive power factor
reactive power factor	reactive power factor, signed
reactive power factor, signed	

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

4.6.8 Energy aggregation module

By using an *Energy aggregation* module you can calculate the amount of energy out of power over certain time intervals. In case of electrical energy this would be kWh, for example.

This helps, for example, to set up a daily energy consumption profile with one value every 15 minutes.

The source signal for this module is a power input signal. The energy is calculated over a selectable time interval between 1 min and 1 h.

The module is also retentive, i.e. it remembers the last calculated values when the acquisition is stopped and builds further on them when the acquisition is restarted.

The following cases are distinguished :

- When the acquisition is stopped and immediately restarted within the same interval, the initial results will be the ones from the interval previously fully completed. The next results will make use of the power values in the previous and current acquisition.
- When the acquisition is stopped in one interval and restarted in the next interval, the partial results of the first interval will be used for the initial results after restart. The system will then start aggregating to calculate partial results for the current interval.
- When acquisition is stopped in one interval and restarted after the end of the next interval, all remembered last values are discarded and the calculation starts from scratch.

4.6.8.1 Energy aggregation module – General tab

Basic settings

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

Configuration

Source signal

Select the desired power signal for which you want to calculate the transmitted energy in the time interval selected below.

Aggregation interval

Select an interval for the energy calculation. Available for selection are:

- 1 minute
- 5 minutes
- 10 minutes
- 15 minutes
- 1 hour

The timestamps of the aggregation interval are synchronized against the local time, i.e. the 5 minutes interval for example will cause the module to generate new energy values at 9:00 9:05, 9:10, etc.

4.6.8.2 Energy aggregation module – Analog tab

In the *Analog* tab you'll find the results of the energy calculation.

Coverage
This is the percentage of the interval that is used to calculate the current values. The unit is always %. In steady state, this value will always be 100%.

Net energy
The difference of positive energy minus negative energy.

Positive Energy
This is the energy transmitted during positive power values. The value is always positive.

Negative Energy
This is the energy transmitted during negative power values. The value is always positive.

Energy transmitted
This is the sum of the positive energy and the negative energy, so it is the sum of the energy that goes out and goes in.

The units of the signals are generated automatically, based on the unit of the configured power signal.

4.6.9 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*.

4.6.9.1 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.

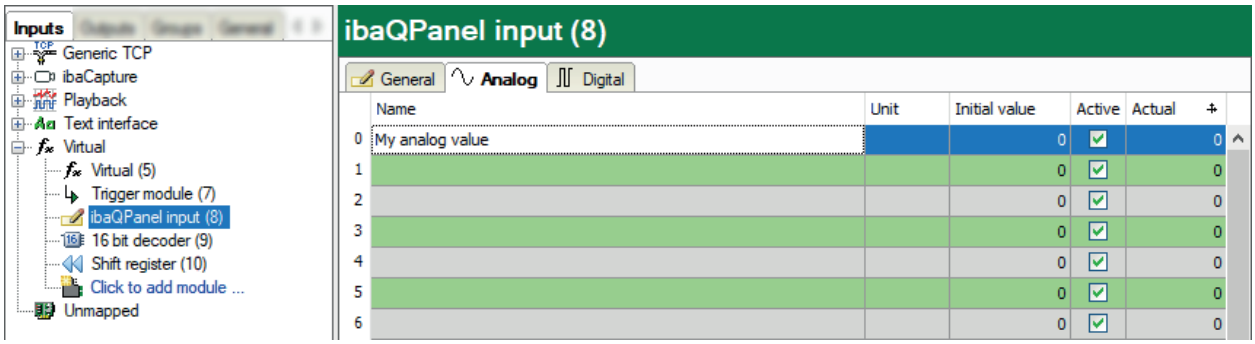
Module layout

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.

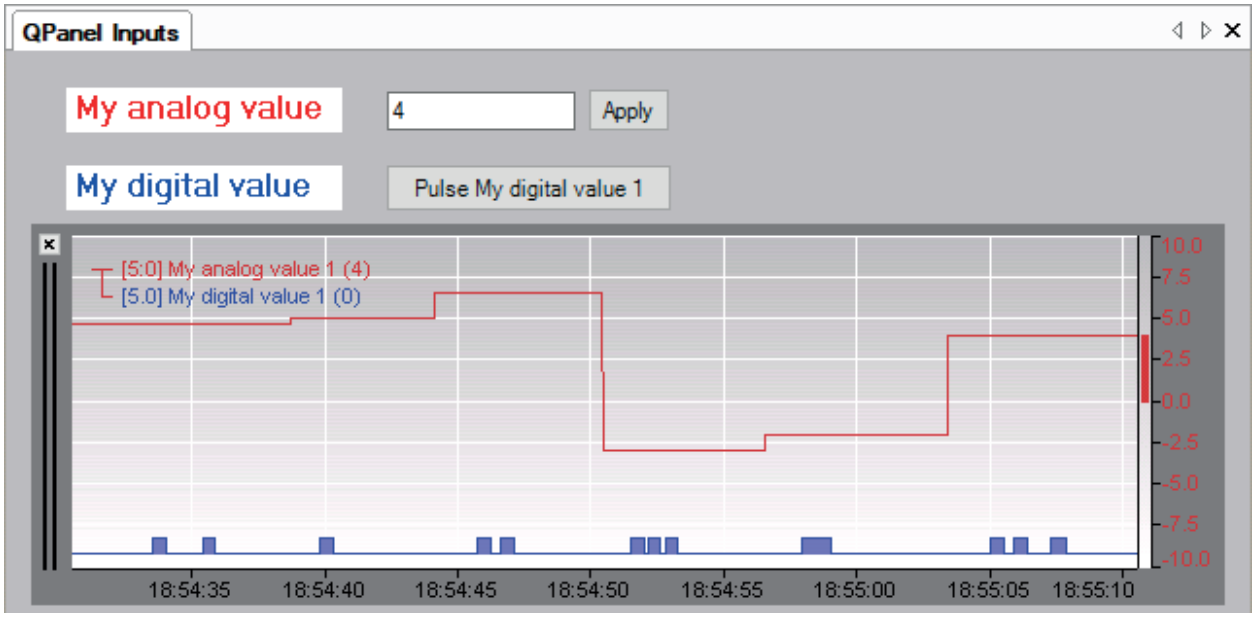
4.6.9.2 ibaQPanel input – Analog and Digital tabs

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.



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.

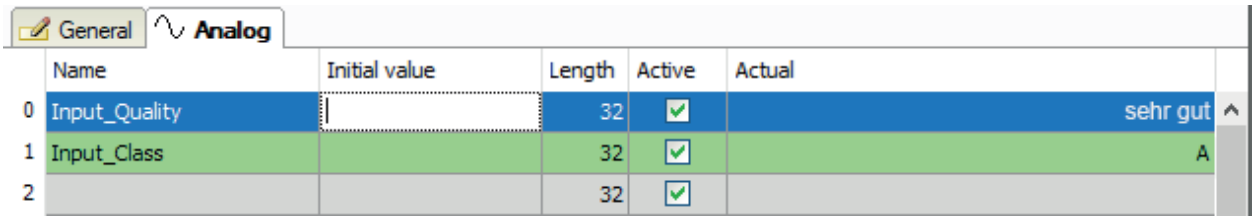
4.6.10 ibaQPanel text input

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.



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*.

4.6.11 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.

4.6.11.1 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

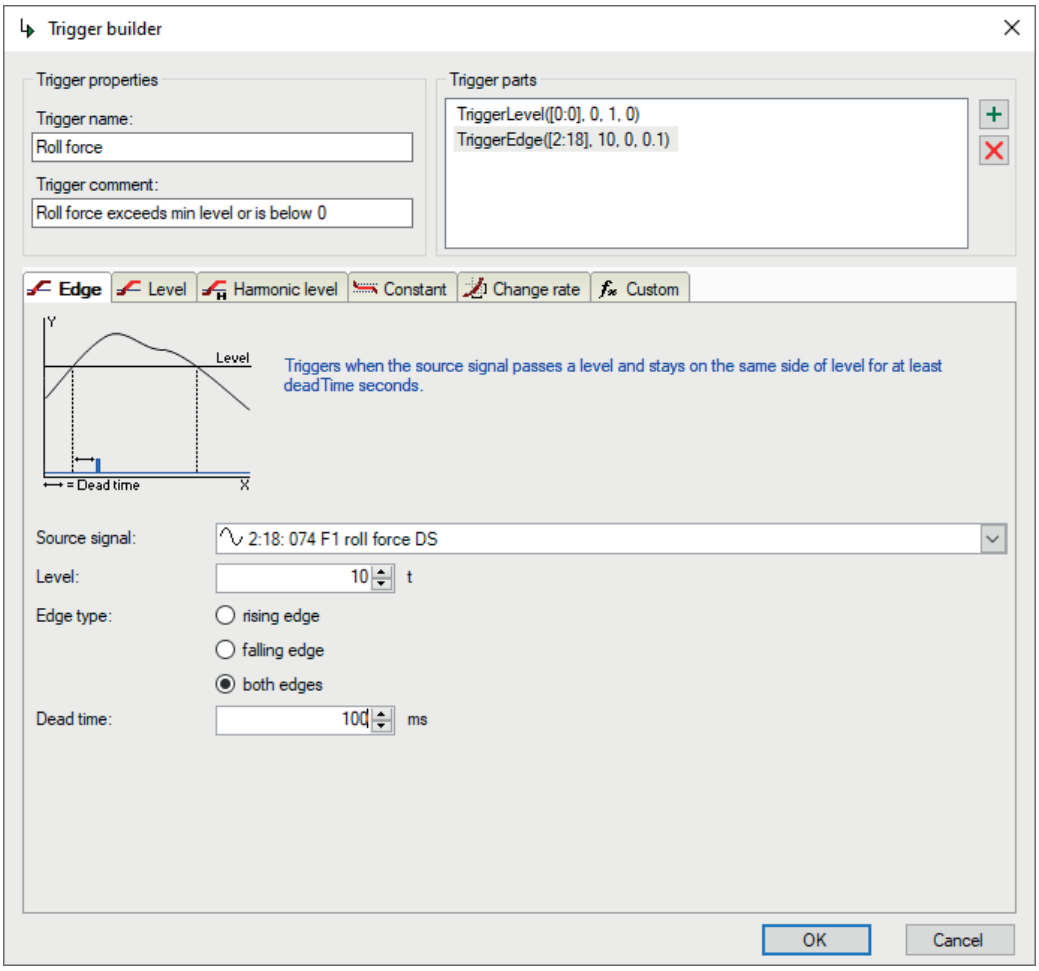
Select which information will be entered in the HD store via the corresponding digital signal (in the *Digital* tab). Available for selection:

- 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.

4.6.11.2 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.



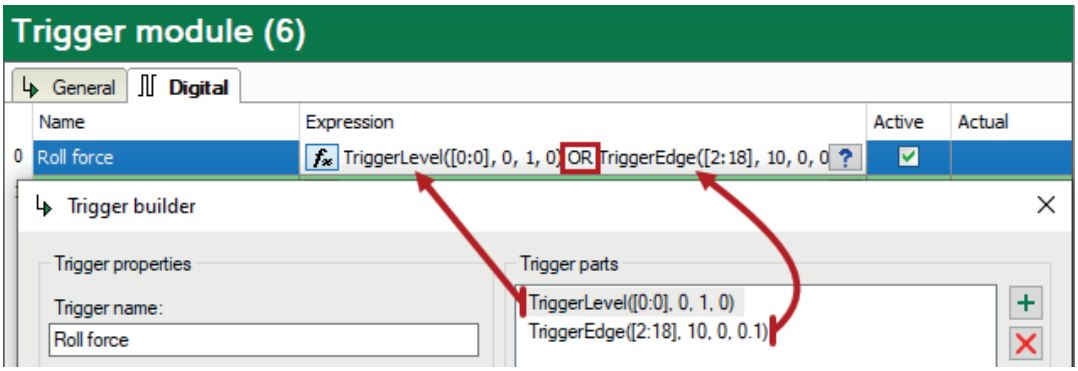
Trigger properties

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.

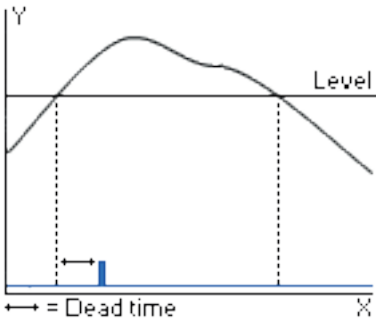
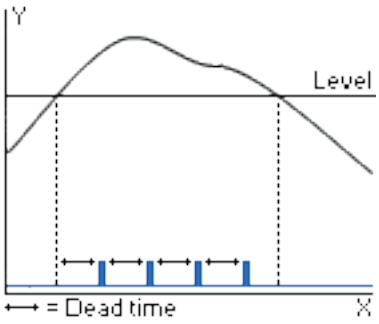
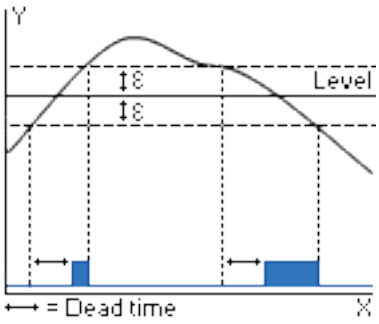
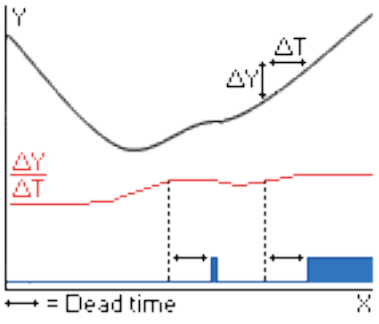
If you want to remove a trigger part, click the button showing the red X symbol.

Trigger configuration tabs

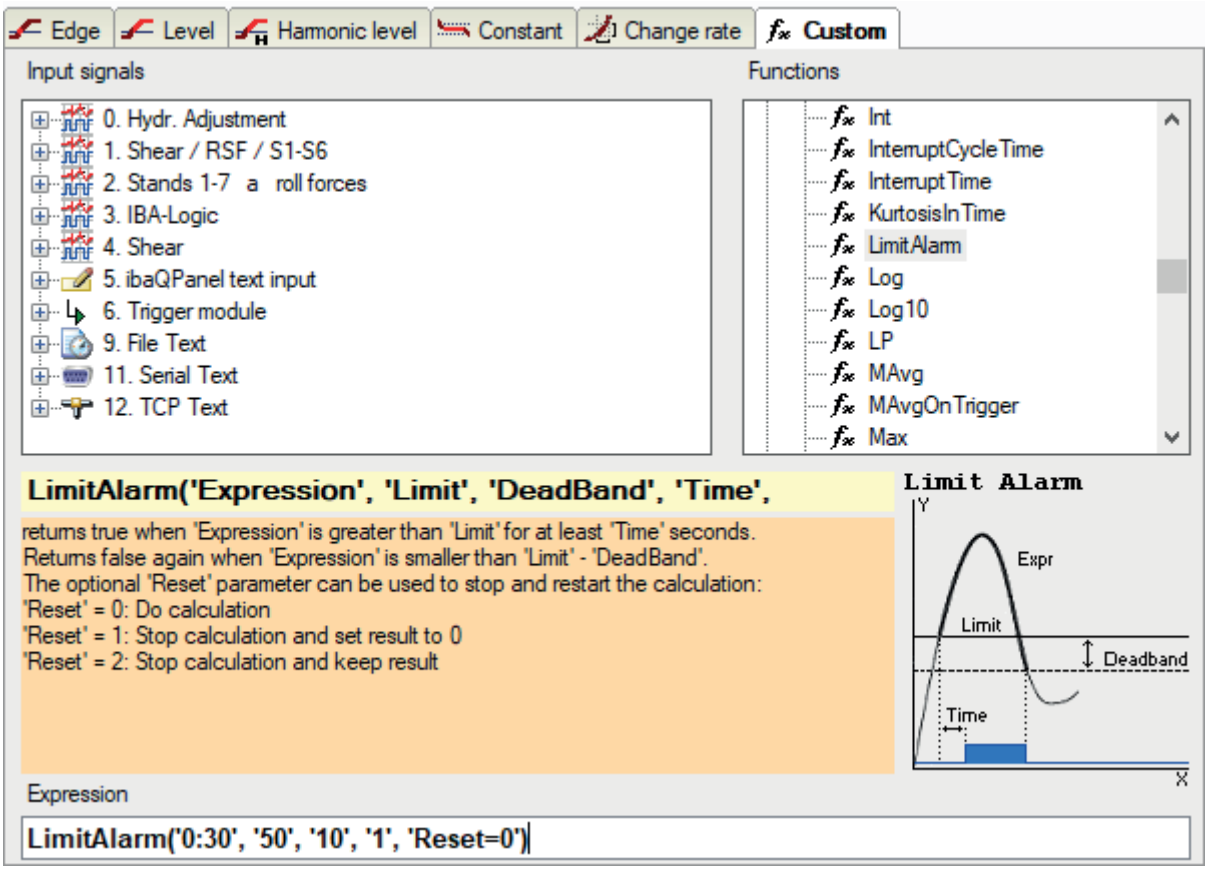
Those expressions can be more or less complex, depending on the nature of the process or machinery. Five different trigger types are available for the most common situations.

- 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.



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

4.6.12 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)*.

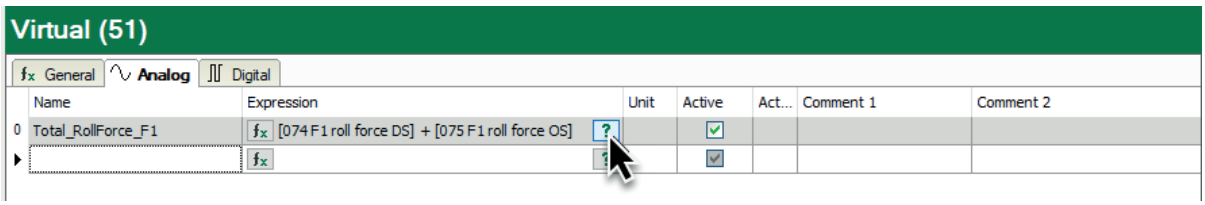
The number of signals (analog or binary) per module is not limited.

4.6.12.1 Cross reference between expression and signal

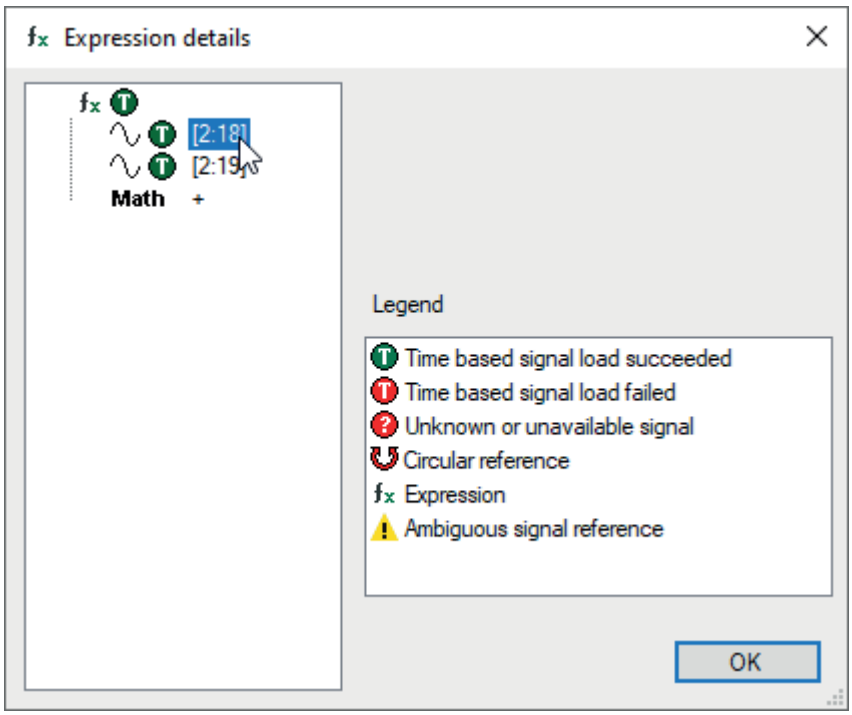
You can swiftly find a source signal of any expression in the virtual signals by using the cross reference function. Particularly, when using the signal name as reference in the expressions instead of the signal number the conclusion on the origin of the signal is not easy.

In order to find the signal source follow these steps:

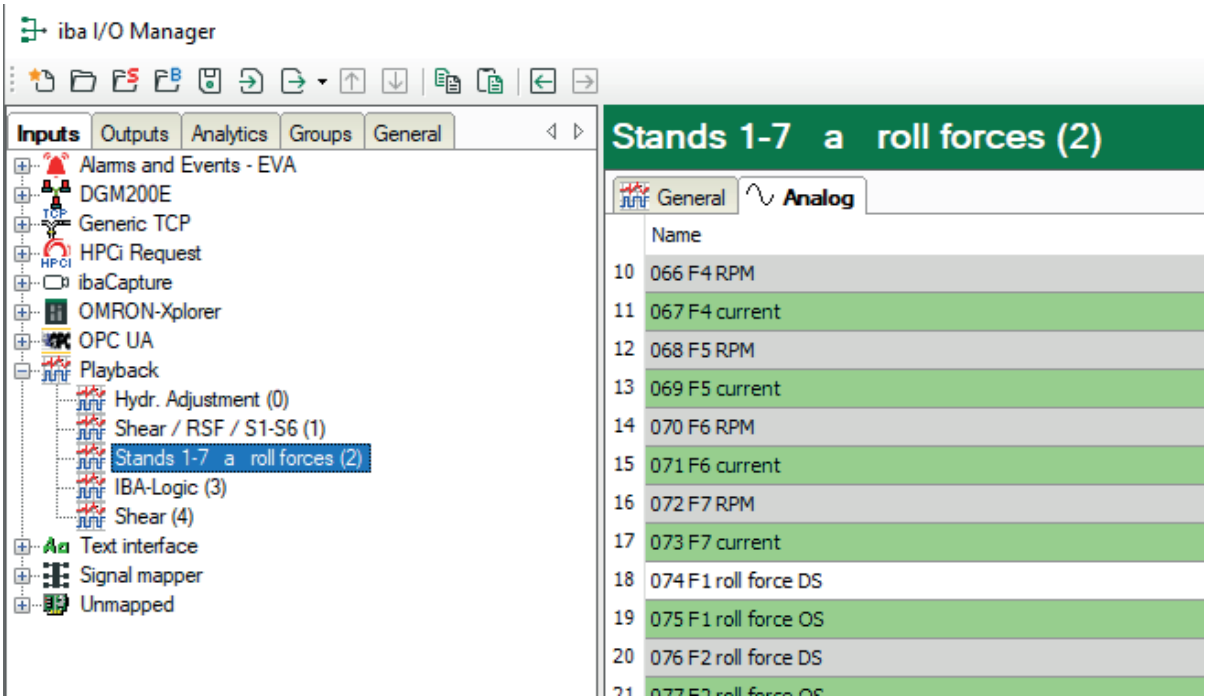
1. Click on the little question mark button in the row with your expression.



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.



4.6.12.2 Virtual – General tab

Basic settings

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

4.6.12.3 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 23.

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.

4.6.13 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 … <0.5 for FALSE or ≥0.5 for TRUE.

Note

Please note, that only functions, which support the retentive behavior, are allowed to be used in a virtual rententive module.

All other functions, including the basic arithmetic operations must not be used in a virtual rentive module and will cause errors.

4.6.13.1 Virtual retentive – General tab

Basic settings

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

4.6.13.2 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 23.

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.

4.6.14 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.

4.6.14.1 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.

4.6.14.2 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 23.

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.

4.6.15 Computation module

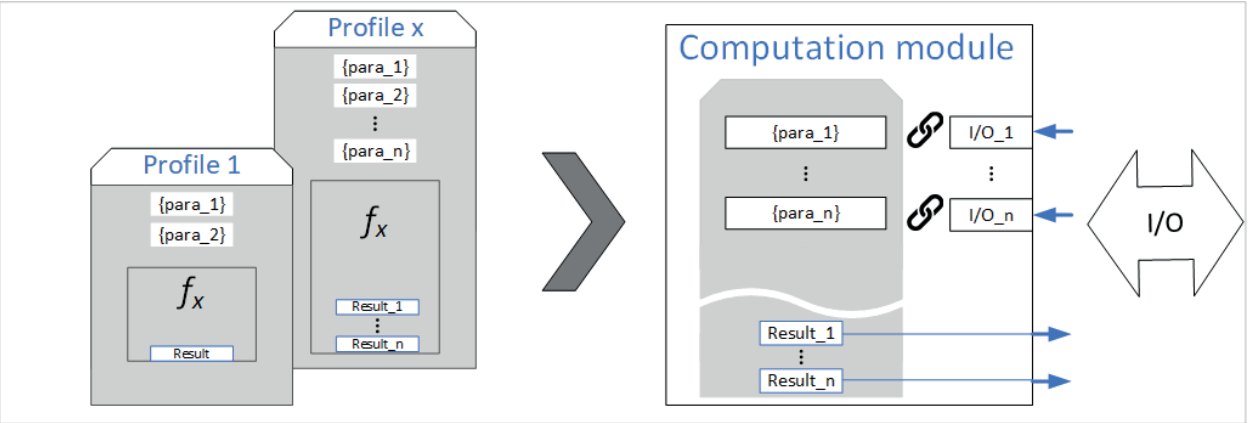
This module type is always available. No license is needed.

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 98*.

4.6.15.1 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:
   a) Define the placeholders.
   b) 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.

4.6.15.2 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.

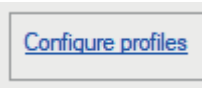
To do this, see ↗ *Add and configure profiles, page 92*.

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

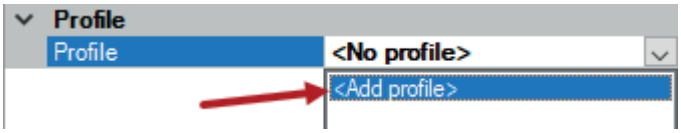
Assign the appropriate signal to the input parameters. If you click inside a field, the signal tree opens and you can select the appropriate signal.

4.6.15.3 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 **.computationProfile* file
	Export selected profile to a **.computationProfile* file

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

Then switch to the *Placeholder* tab.

4.6.15.3.1 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.

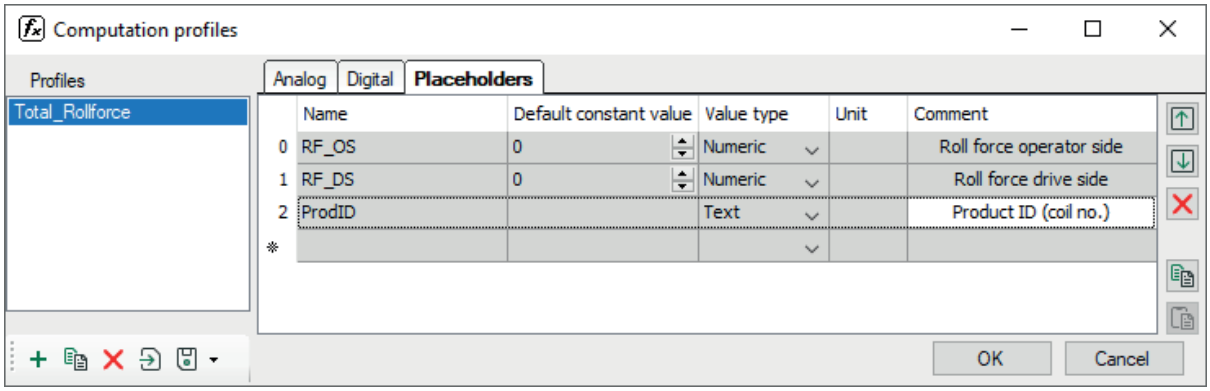
- Name: Name of the placeholder as it will subsequently be used in the calculation function.
- Default constant value: Enter a value here that the placeholder should have if no signal is linked or the signal is invalid. Note that this value must match the value type.
- Value type: Select the value type 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.



4.6.15.3.2 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.

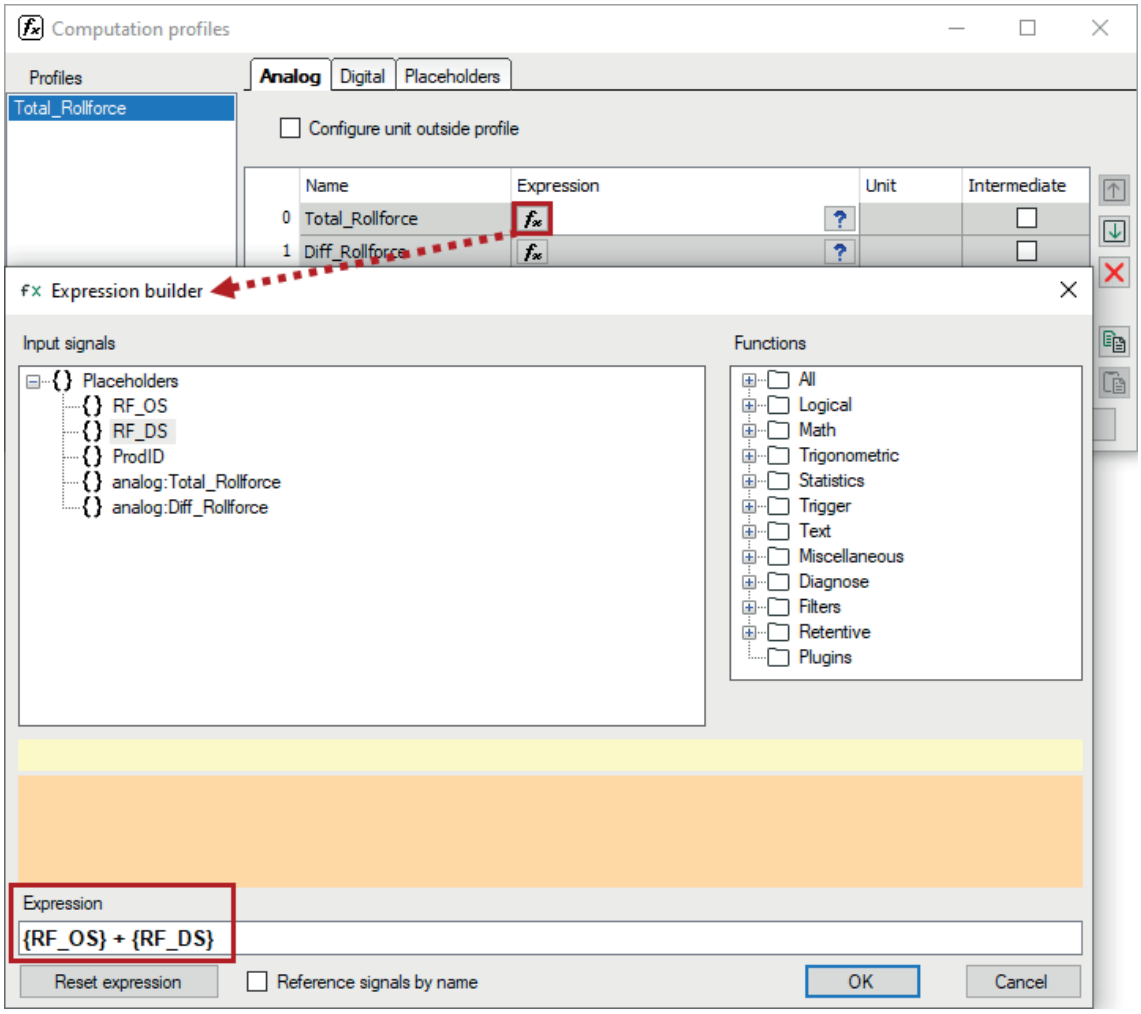
	Move selected rows up
	Move selected rows down
	Delete selected rows
	Copy all rows
	Paste from clipboard, starting from the selected row

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.

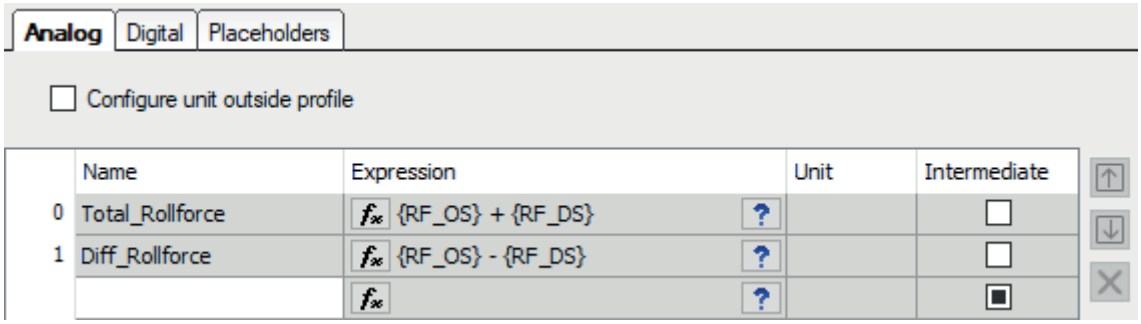
Example

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.



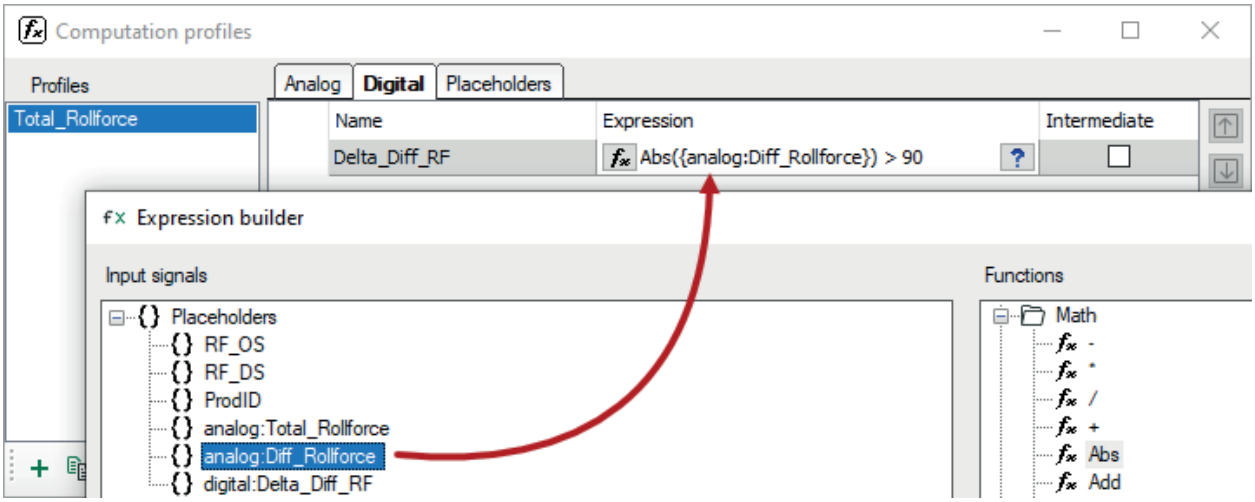
In the expression builder, only the defined placeholders and placeholders for the internal calculation results are available for the calculation. I/O signals cannot be used here.



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.

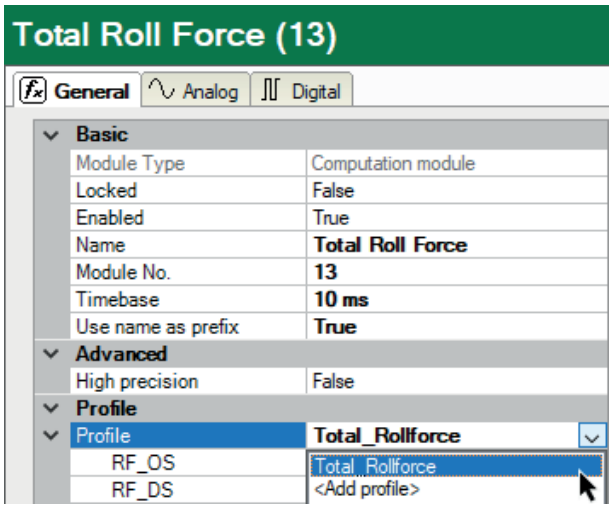


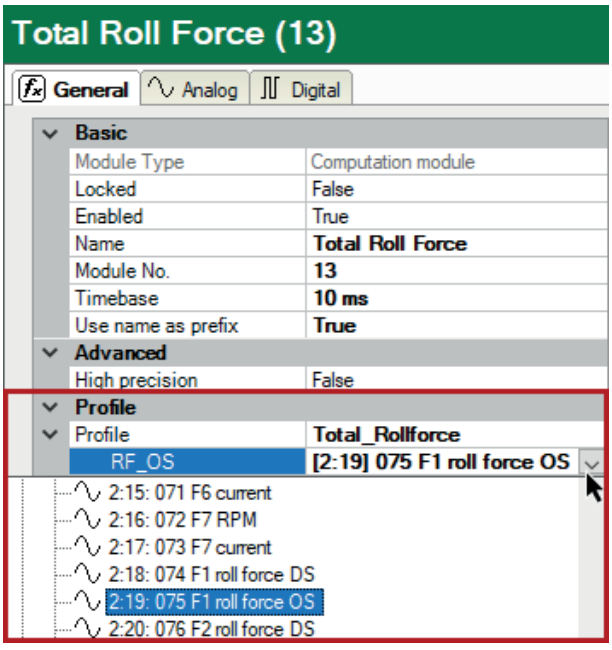
The limit value of 90 t is permanently entered here. If this value were variable or process-dependent, then a placeholder would also have to be defined for it, which would then be linked to a corresponding signal.

4.6.15.4 Computation module – assign signals

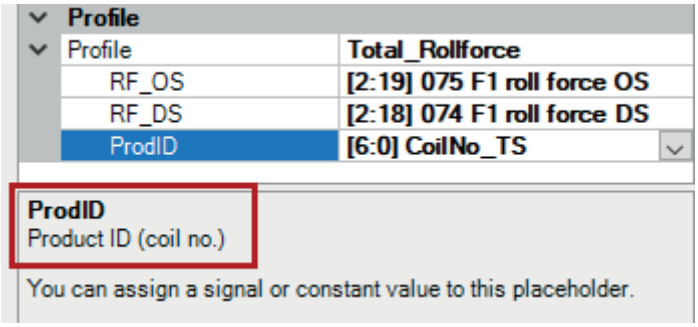
When the desired profile is ready and available, you can select it in the general module settings for the computation module and then assign the signals to the placeholders.

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

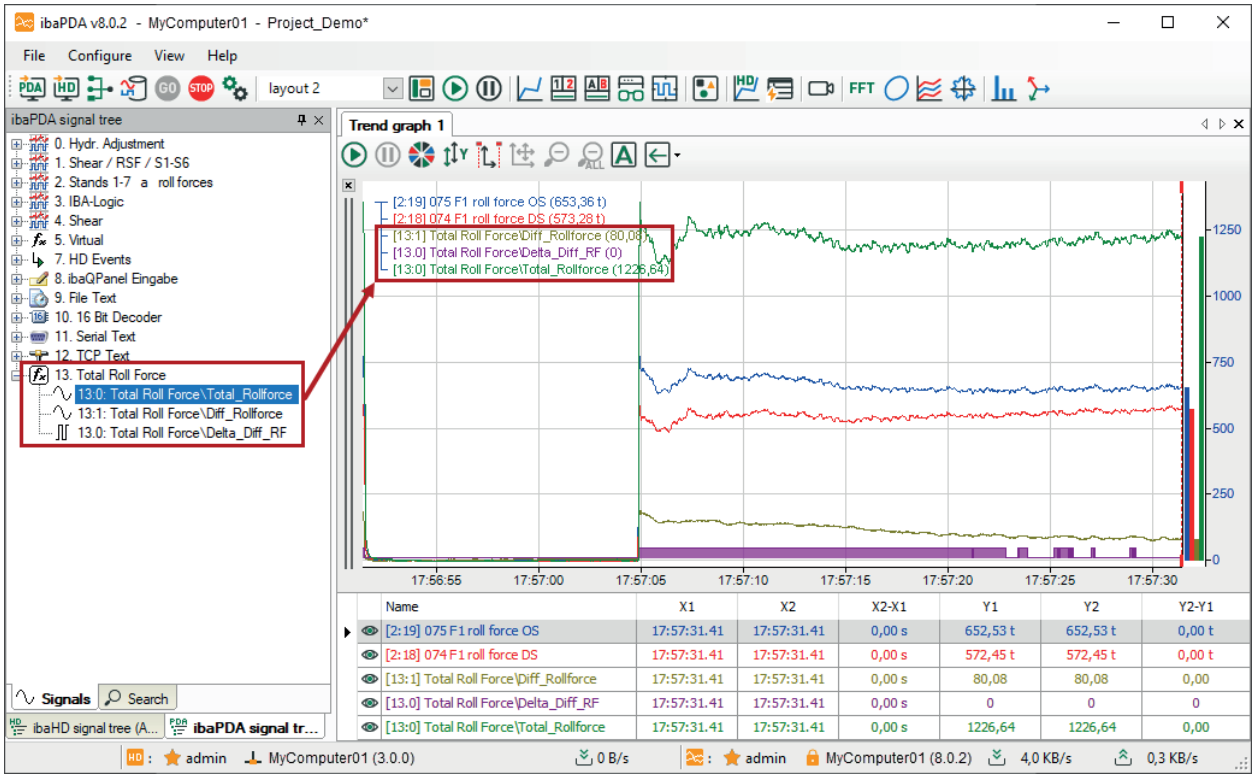




The comment from the placeholder definition is displayed here.



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



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.

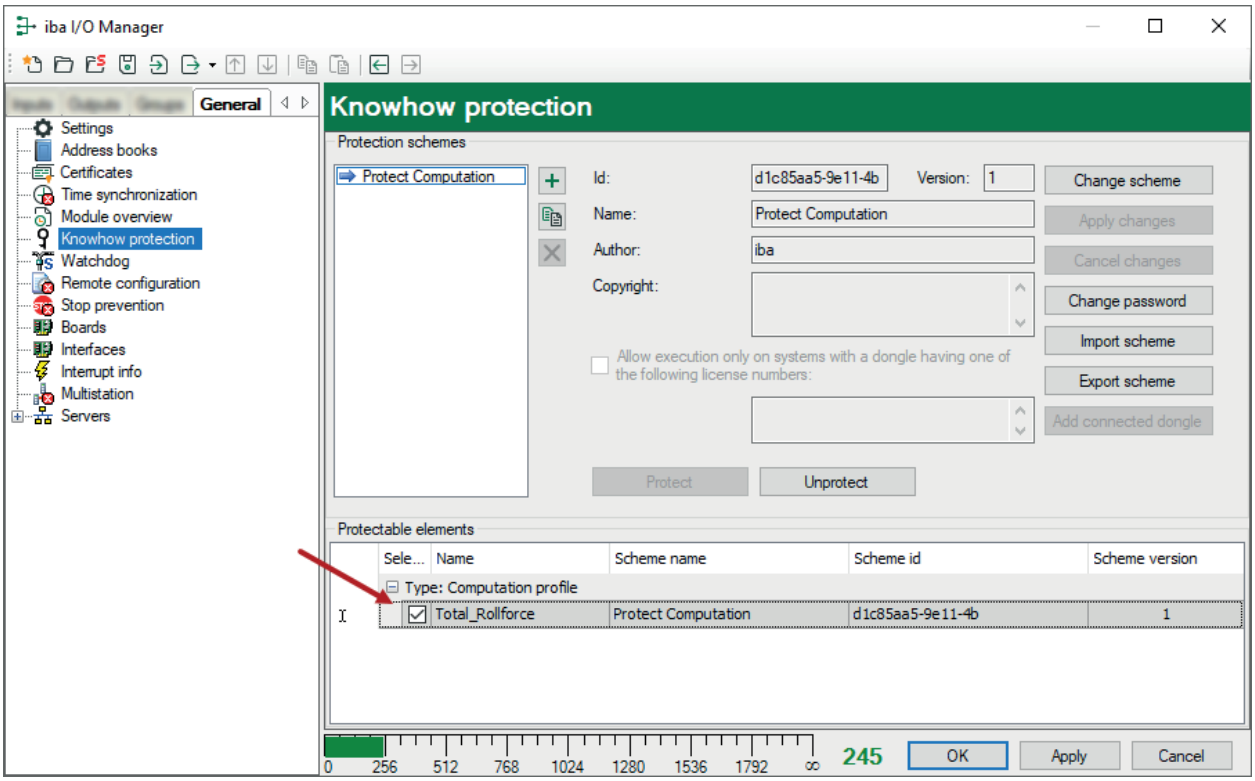
If the calculations change or more calculation variables are added, only the profile needs to be changed and all instances of the computation module will be automatically updated. The signal assignments must only be completed in each module if additional placeholders are added.

4.6.15.5 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:

- ibaInSpectra
- ibaInCycle
- Computation module
- Lookup table
- Process condition
- Parameter set

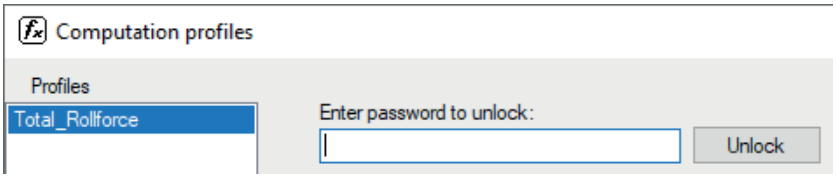
All defined profiles are listed in the know-how protection dialog. The following figure shows the example for a profile of a computation module.



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*.

4.6.16 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*.

4.6.17 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.

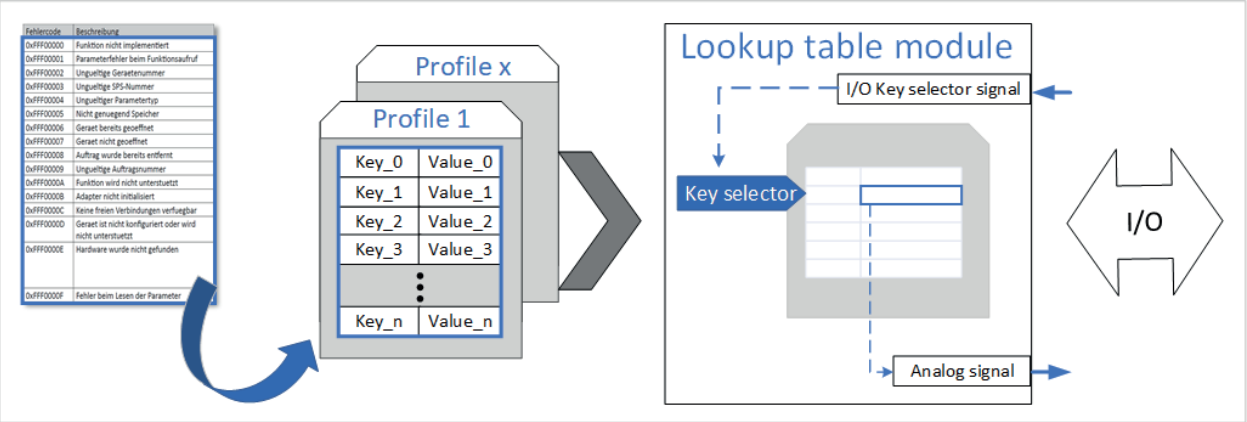
The table contents are transferred to a so-called *profile* and saved. The entry into the profile can be done line by line manually or via copy & paste from a table, e.g., in MS Excel.

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 98*.

4.6.17.1 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.

4.6.17.2 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 102*.

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.

4.6.17.3 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.

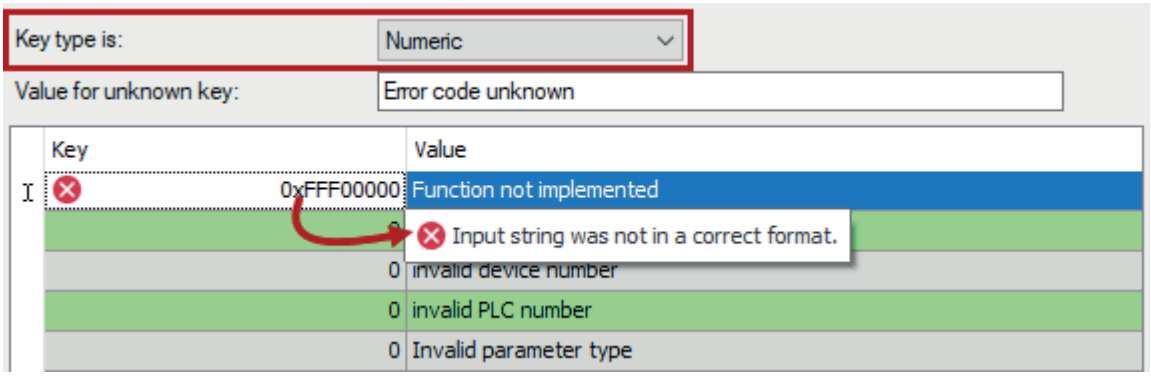
Below the left window of the dialog you will find a set of buttons with the following functions:

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

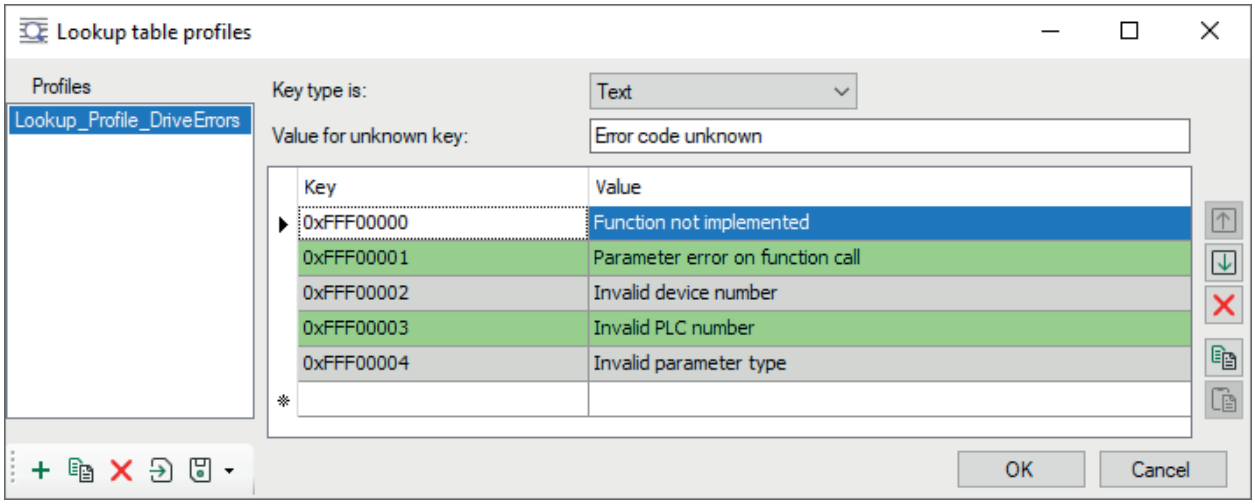
Make sure that the keys are always unique. If there are two or more identical keys in the table, only the result value of the first of these keys is output.

The buttons at the edge have the following functions:

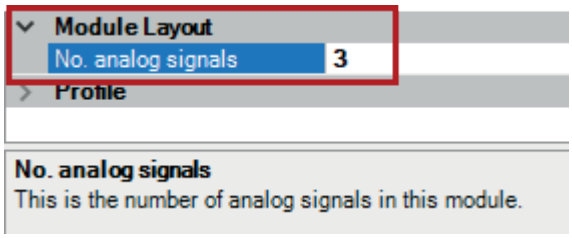
	Move selected rows up
	Move selected rows down
	Delete selected rows
	Copy all rows
	Paste from clipboard, starting from the selected row

Example

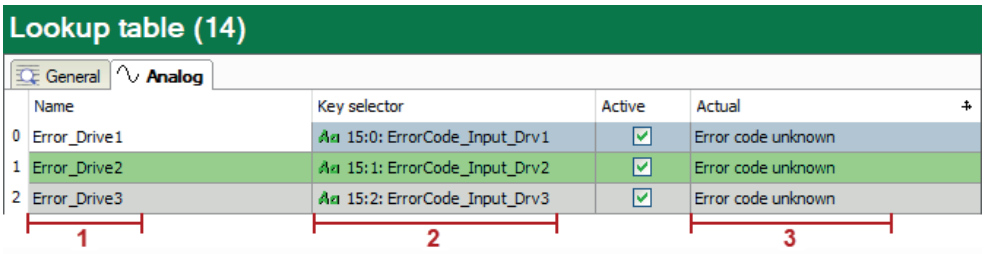
The following figure shows a profile with five error messages from drives. The error codes (keys) and values are entered as text.



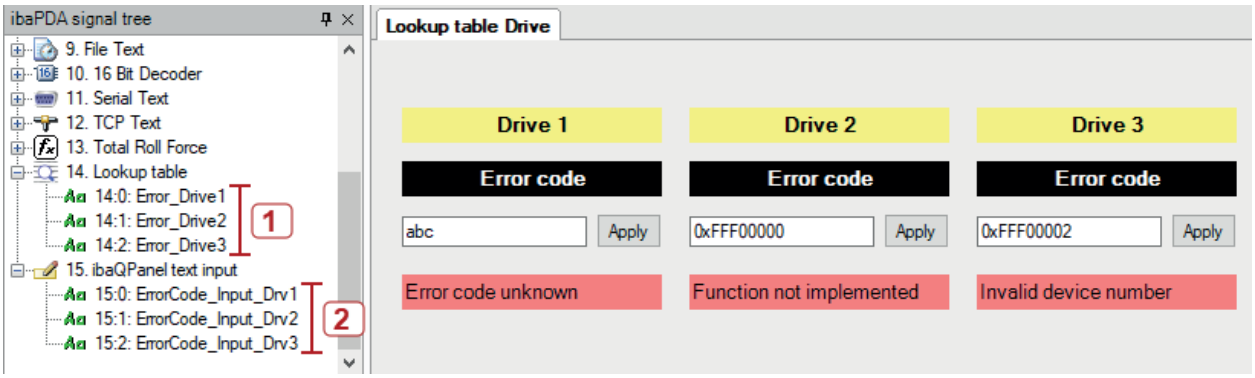
This lookup table is used for 3 drives. Therefore, the lookup table module is configured for 3 analog signals.



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



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.

4.6.18 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)

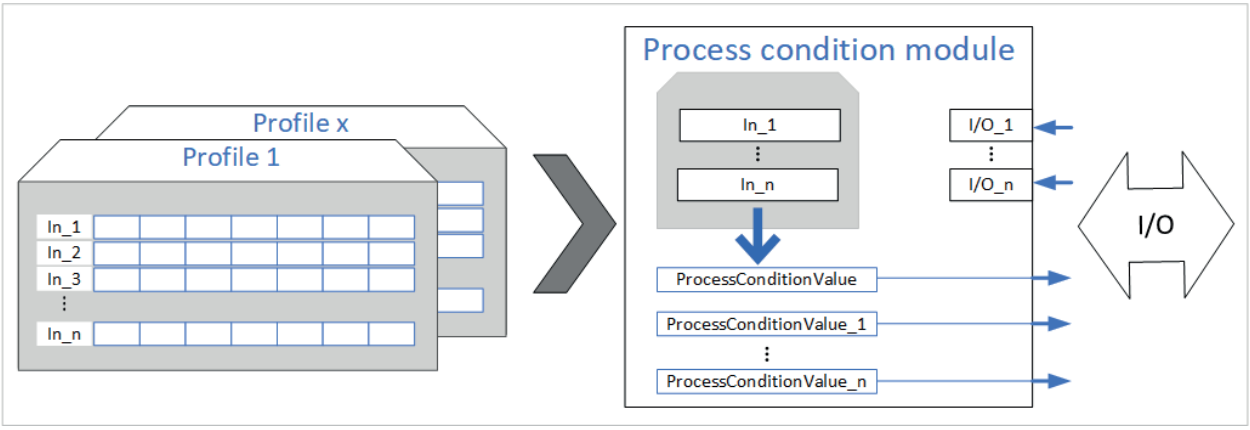
Each process condition is defined and stored in the form of a profile. When configuring a "Process condition" module, the appropriate profile is then loaded, which then allows the module to output the appropriate signals.

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 98*.

4.6.18.1 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.

4.6.18.2 Process condition – General tab

Basic settings

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

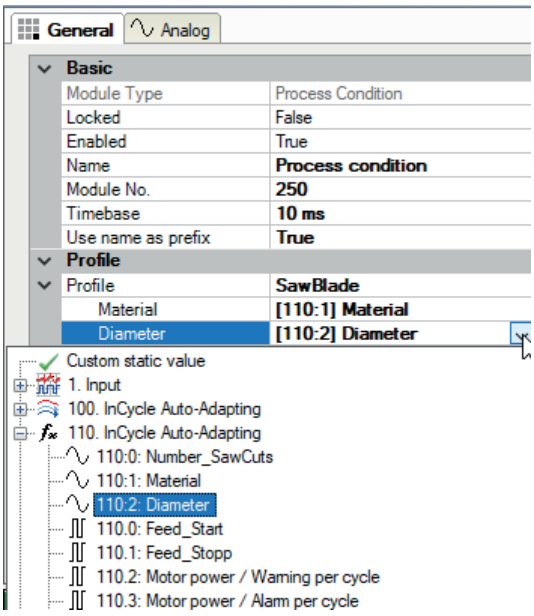
Profile

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 107*.

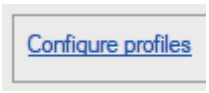
If you have created and selected a profile, then the parameters as defined in the profile will automatically appear here.

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



4.6.18.3 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.

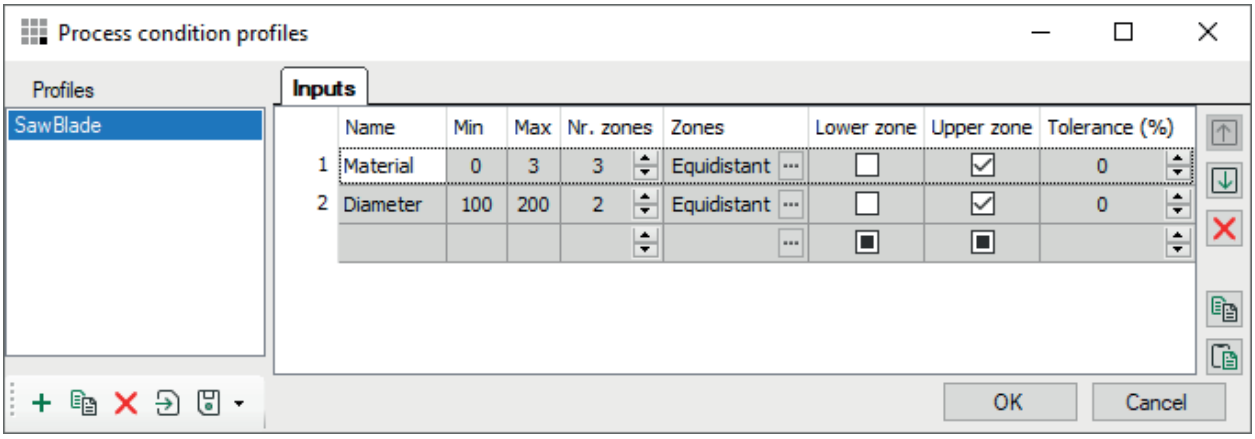
Below the left window of the dialog you will find a set of buttons with the following functions:

	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.

4.6.18.4 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.

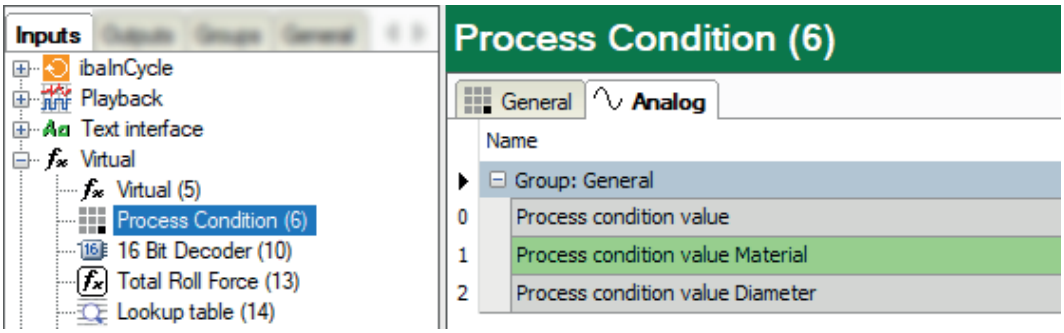
Tolerance (%)
Here, you can set a percentage tolerance value with respect to the zone boundaries. You can thus avoid frequent switching of the zones if, for example, the parameter value fluctuates around the zone limit value.

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.

4.6.18.5 Process condition – Analog tab

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



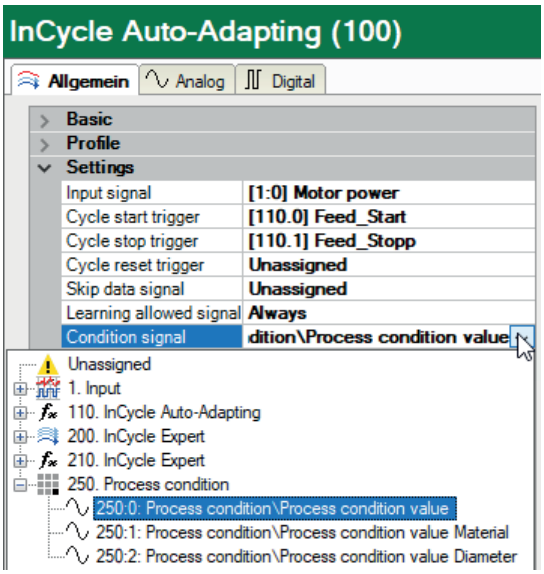
The first signal applies to the entire module, i.e., to the combination of all parameters included in the profile. In addition, there is a separate status signal for each parameter.

The values of these signals (integer) are determined internally in the module and do 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.

4.6.18.6 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*.

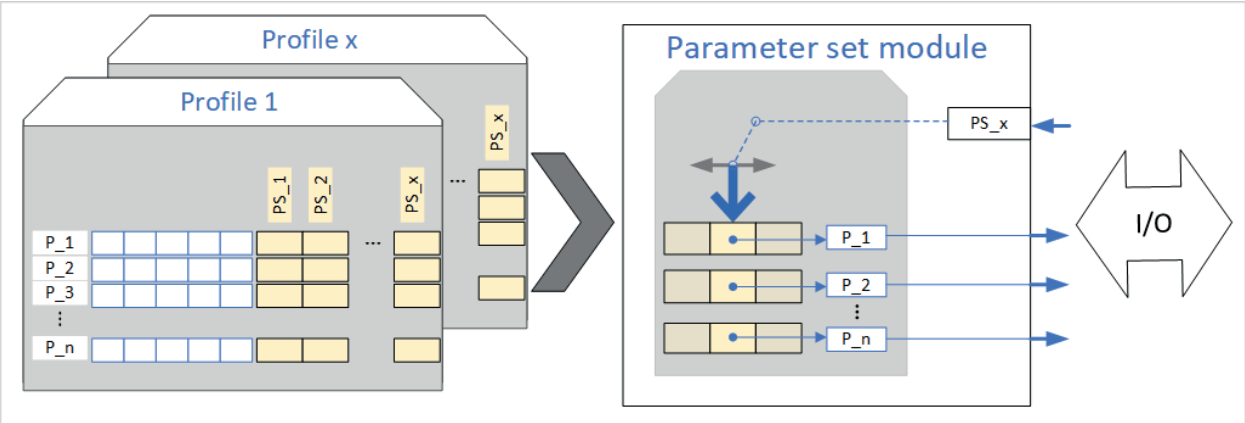
4.6.19 Parameter set

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 98*.

4.6.19.1 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.

4.6.19.2 Parameter set – General tab

Basic settings

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

Profile

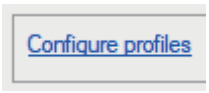
From the drop-down list, select the appropriate profile that describes the parameter set for this module. Without a profile, the module is invalid. If no profile is available yet, you must create one first.

To do this, see ↗ *Add profiles, page 113*.

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

4.6.19.3 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.

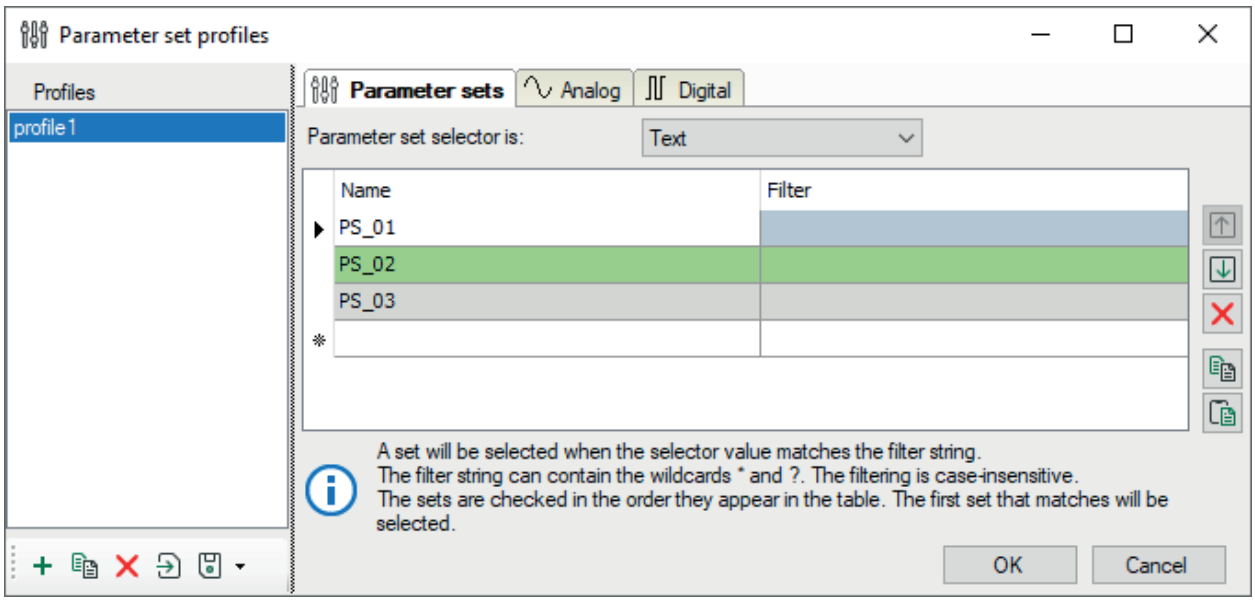
Below the left window of the dialog you will find a set of buttons with the following functions:

	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.

4.6.19.4 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.

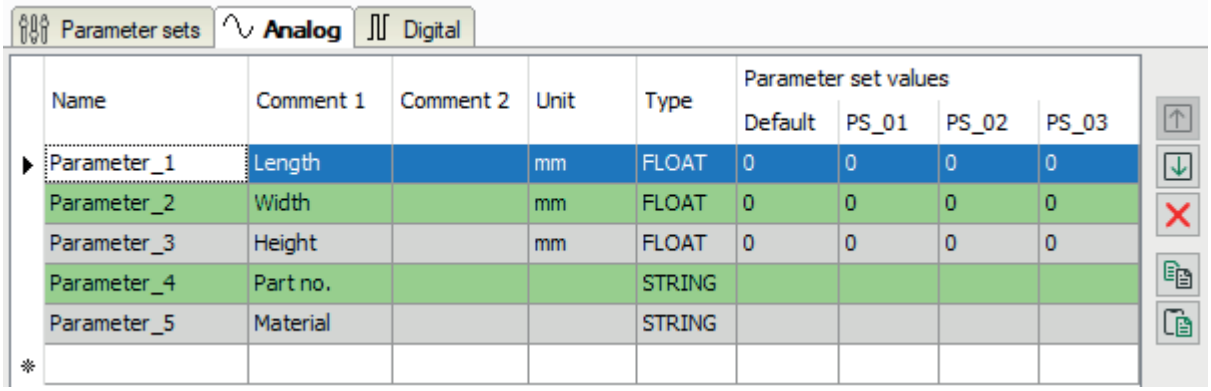
A set will be selected if the selection value is ≥Min and <Max. If Min and Max are equal, the set will be selected where the selection value is equal to Min. The sets are checked in the order in which they appear in the table. The first set that fits is selected.

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

4.6.19.5 Configure profiles – analog and digital parameters



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.

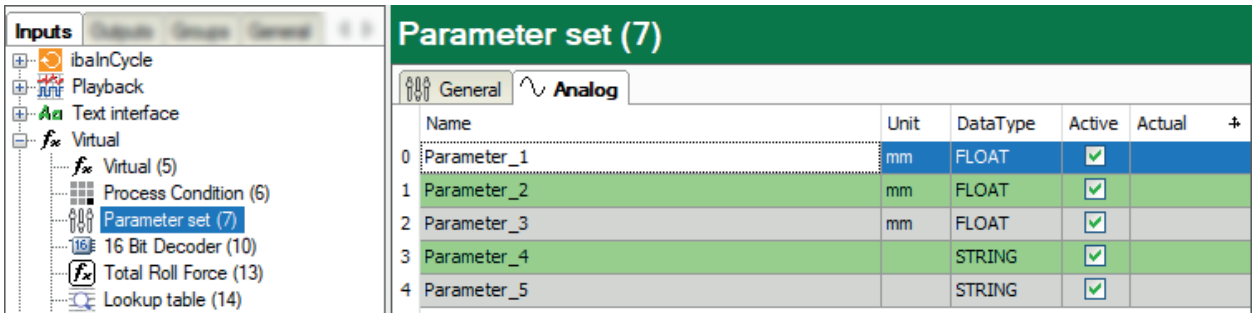
Each time you complete a line, the next free line is automatically created. If you have several parameters, enter them line by line. If you have many parameters, you can also prepare the table in another program (e.g., MS Excel) and then paste it via the Windows clipboard.

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

4.6.19.6 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.



4.6.20 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 0813 decoder* module, you can set these data fields as signals. You define how *ibaPDA* interprets the individual signals.

4.6.20.1 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

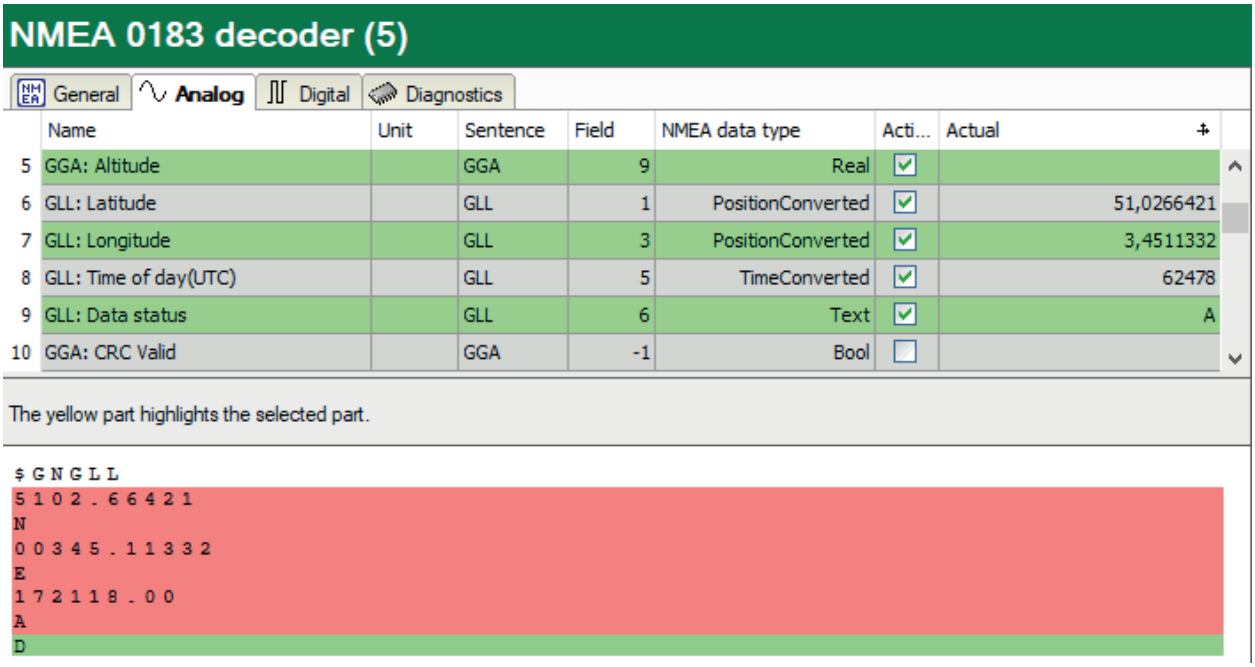
Select whether *ibaPDA* ignores NMEA records with invalid checksums (True) or processes all records (False). You can also select the validity of the checksum as a digital signal in the symbol browser.

Link "Select NMEA variables"

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

4.6.20.2 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*.



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 23.

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 *Field* column.
Red (2)	Red lines are assigned to multiple text signals in the *Field* 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.

4.6.20.3 NMEA 0183 decoder – Diagnostics tab

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

4.6.21 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.

4.6.21.1 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)

4.6.21.2 Vector calculation module - Analog tab

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

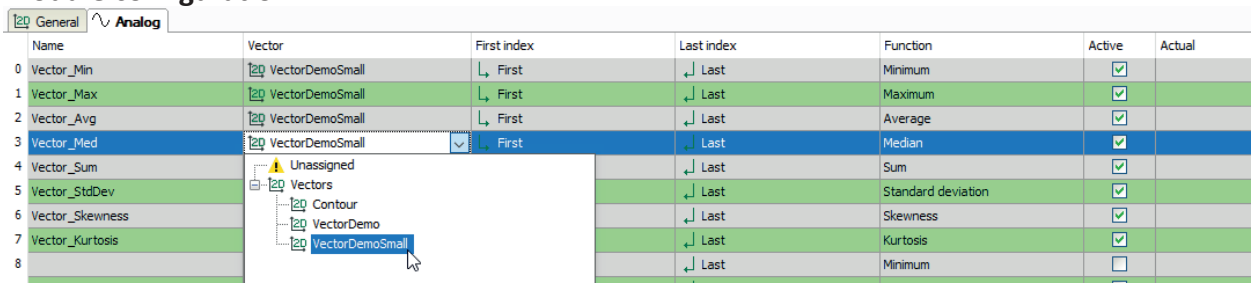
1. Enter a meaningful name for each signal in the *Name* column, which refers to the source vector and the calculated value type.
2. In the *Vector* column select the desired vector signal. Only vector signals are offered, which have been defined in the *Groups* section of the I/O Manager.
3. Define the range of vector elements supposed to be subject to calculation in the columns *First index* and *Last index*. There, you have different options:
   a) Select “First” and “Last”, if you want to include all vector elements in the calculation (default).
   b) 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.
   c) 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.

4.6.21.3 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.

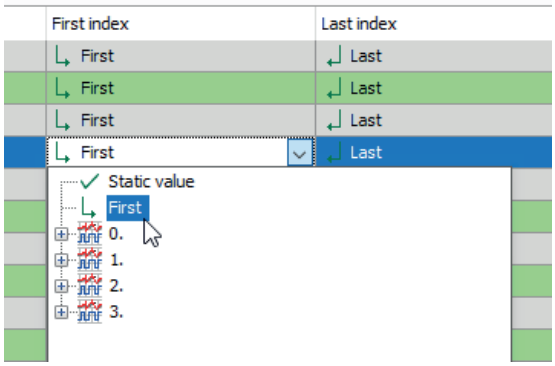
Module configuration



For this example one signal has been configured for each calculation function, e.g. Vector_Min for minimum, Vector_Avg for average.

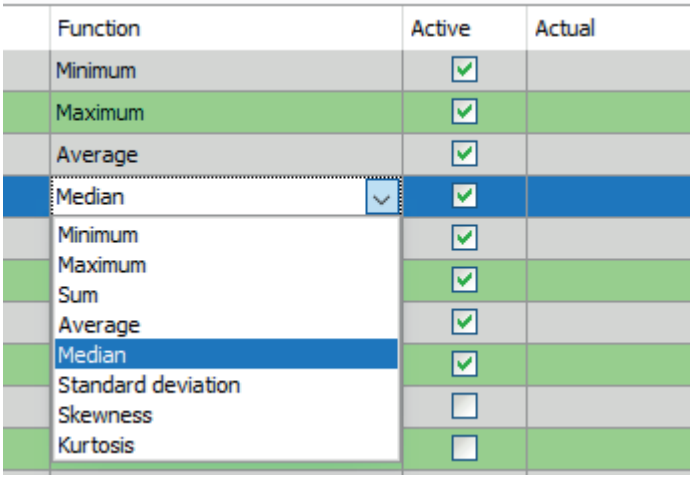
The vector *VectorDemoSmall* is selected in the *Vector* column for each signal.

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.



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.



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.

4.6.22 Phasor module

The phasor module calculates phase, RMS, effective frequency and ROCOF (Rate Of Change Of Frequency) of an input signal. The input signal can be a current or voltage measurement.

The input signal is checked against an ideal, internally generated reference signal of a given nominal frequency, e. g. 50 Hz.

The results give information about a phase shift and/or a deviating amplitude of the input signal and about the amount of the deviations.

Suitable applications of the module can be found in the field of electrical engineering and power grids. For example, you can monitor and compare the phase and amplitude of voltages and currents from different converters supplying the same grid (e. g. railway electrification).

The amount of deviation from the recommended value may be used as a basis for trigger conditions. The drifting of signals can be detected too.

4.6.22.1 Phasor module – General tab

Basic settings

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

Note

Please note, that the timebase (basic settings) must be at least ten times higher than the timebase of the input signal.

Example:

Timebase input signal = 1 ms --> timebase phasor module ≥ 10 ms

Input

Select the signal used for phasor analysis.

Phasor

Calculation timebase

Interval for the calculation of the phasor values

In order to store all calculations, the calculation timebase must be at least equal to the module timebase (basic settings).

Further limitations are subject to the properties of the input signal. For instance, the calculation timebase must be equal to half the period of the nominal frequency or higher.

Value samples:

- 50 Hz .... ≥ 10 ms
- 60 Hz .... ≥ 10 ms (because values <10 ms are not allowed)
- 16.7 Hz .... ≥ 30 ms

The higher the calculation timebase, the narrower is the allowed frequency deviation between input signal and nominal frequency.

A timebase of 250 ms allows a deviation of ±2 Hz from the nominal frequency

A timebase of 100 ms allows a deviation of ±5 Hz from the nominal frequency.

The allowed range of calculation timebase is between 10 ms and 250 ms.

Phase unit

Select here if the phase calculation results in degree (range [-180°...180°]) or radians (range [-π...π]).

Nominal frequency

Enter here the nominal frequency of the input signal, which is used as reference for the phasor analysis.

ibaPDA generates internally a sinosoidal signal with this frequency as reference signal for further calculation.

Value samples: 50 Hz, 60 Hz oder 16.7 Hz

4.6.22.2 Phasor module – Analog tab

In the *Analog* tab you'll find the results of the phasor analysis.

RMS
RMS value of the input signal

Phase angle
Phase shift in relation to 0° or 0 rad.

Frequency
Effective frequency of the input signal

ROCOF (Rate of Change of Frequency)
Rate of frequency change given in Hz/s

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)

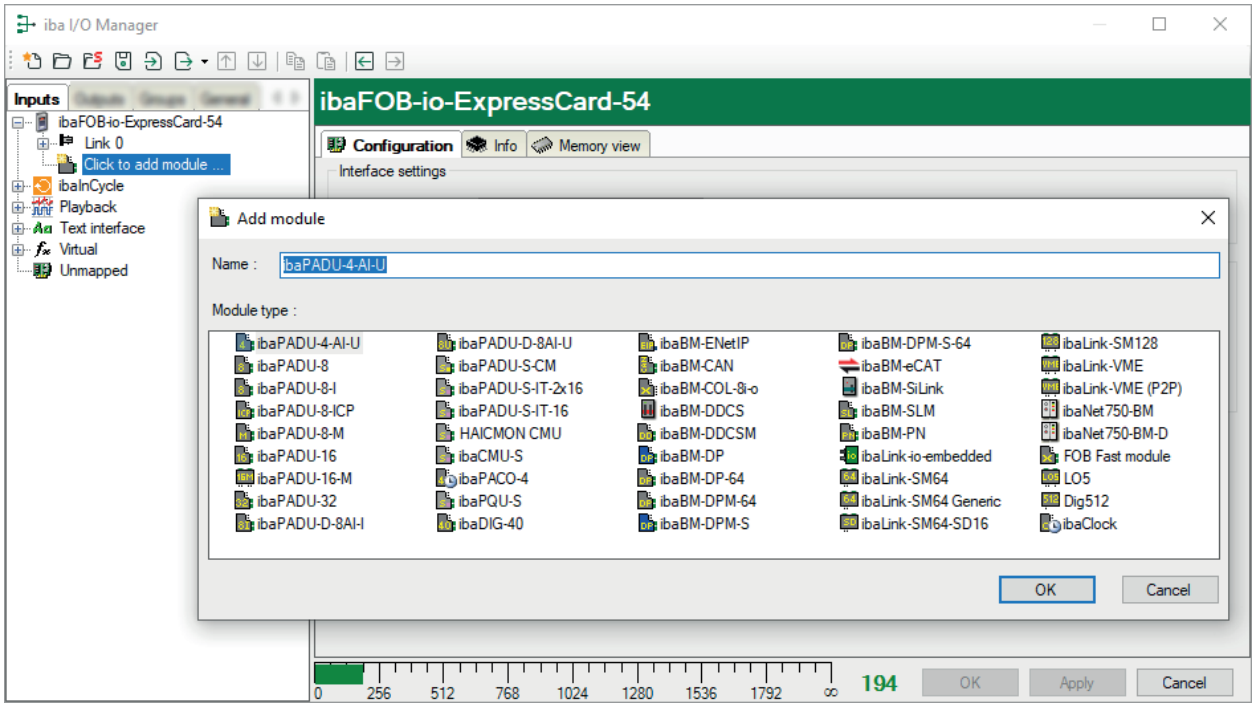
The interfaces of the ibaFOB cards are only visible in the I/O Manager of *ibaPDA* if one or more cards are installed in the computer. iba devices with an ibaNet connection (FO) can only be connected to *ibaPDA* via an ibaFOB card.

ibaNet-E is an Ethernet-based interface with a special protocol supported by compatible devices (e.g., ibaW-750).

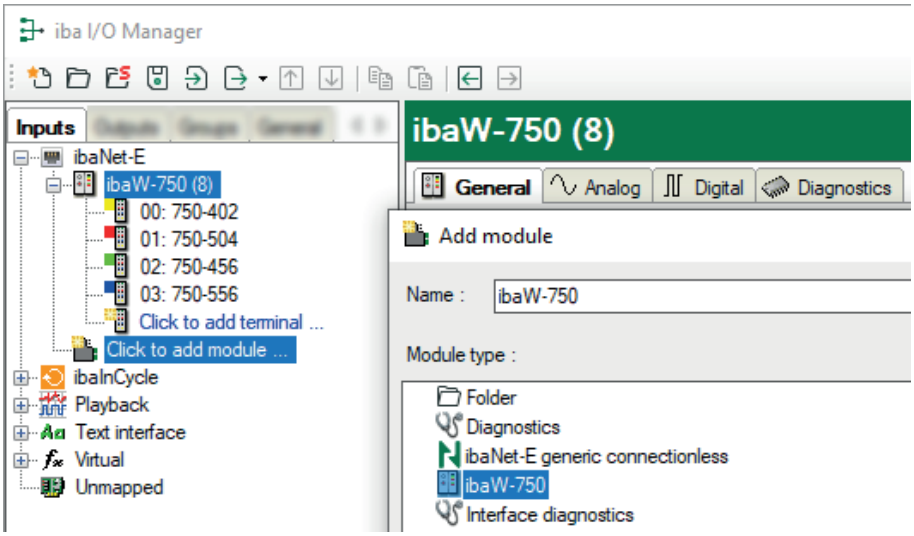
Name	Type	Connection to …	Comment	Link
ibaFOB-4io-D, -Dexp	PCI cards ibaFOB-io-D, -2i-D, -4i-D, -4o-D, -2io-D PCIe cards ibaFOB-io-Dexp, -2i-Dexp, 4i-Dexp, -2io-Dexp	ibaPADU, SM64, SM128, ibaNet750, ibaBM…, ibaPADU-S family ibaLink-VME	Supports all devices and FOB protocols incl. 32 Mbit flex	↗ *ibaFOB-io-, 2io-, 4io- etc., page 128*
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)	↗ *ibaFOB-io-ExpressCard, page 140*
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, ibaM-COM, ibaM-DAQ eLumina (GE Vernova), HiPAC (Danieli), Simatic S7 (Siemens)	The protocol can also be implemented on third-party devices (OEM)	↗ *ibaNet-E, page 141*

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 µs ... 1.4 ms	≥ 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
Topolgy	Ring	Line and/or star
Number of devices	≤ 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* (*FOB4i-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

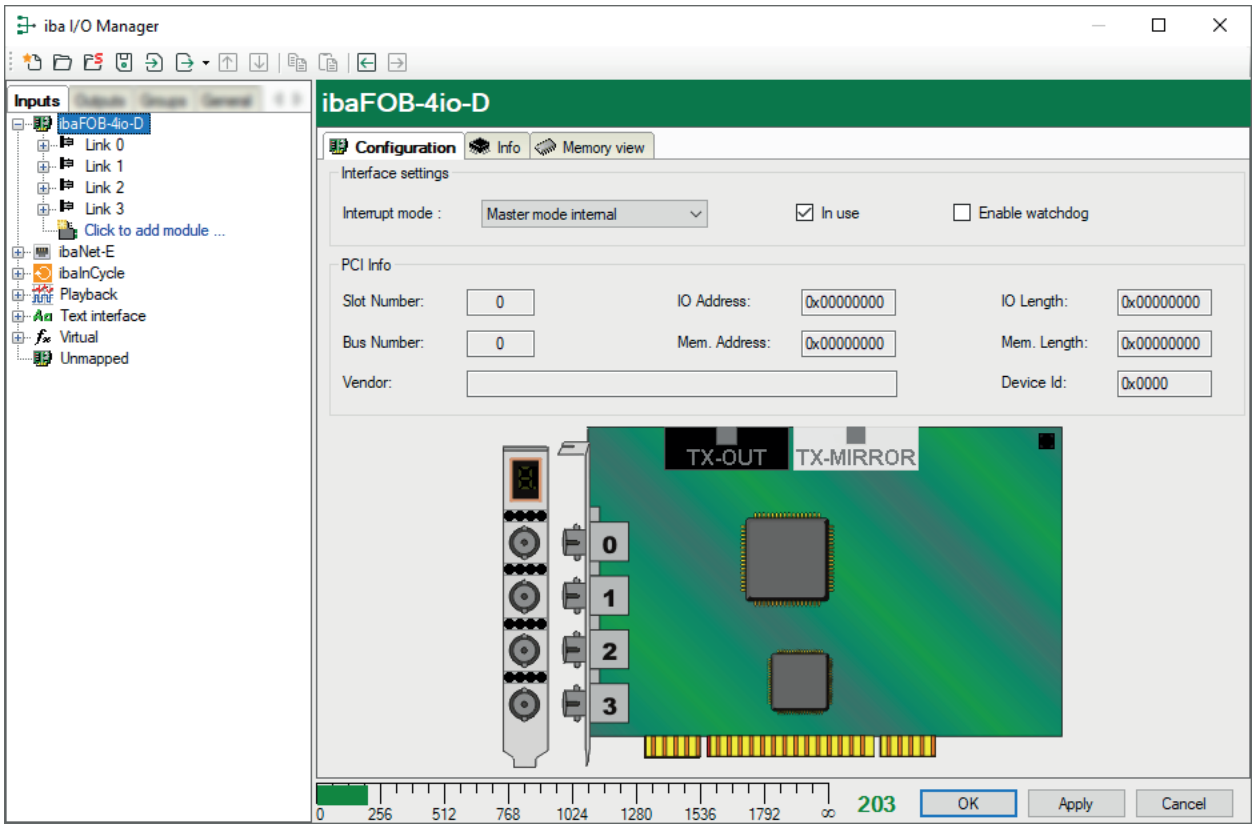
- Please note that the *ibaFOB-F*, *ibaFOB-S* and *ibaFOB-X* cards are only supported by 32-bit Windows operating systems (x86).
- Up to 8 *ibaFOB-4i-D* cards are supported in an ibaPDA computer.
- When using cards of the type *ibaFOB-D*, *ibaFOB-SDexp* or *ibaFOB-TDCexp*, only these cards can be configured 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

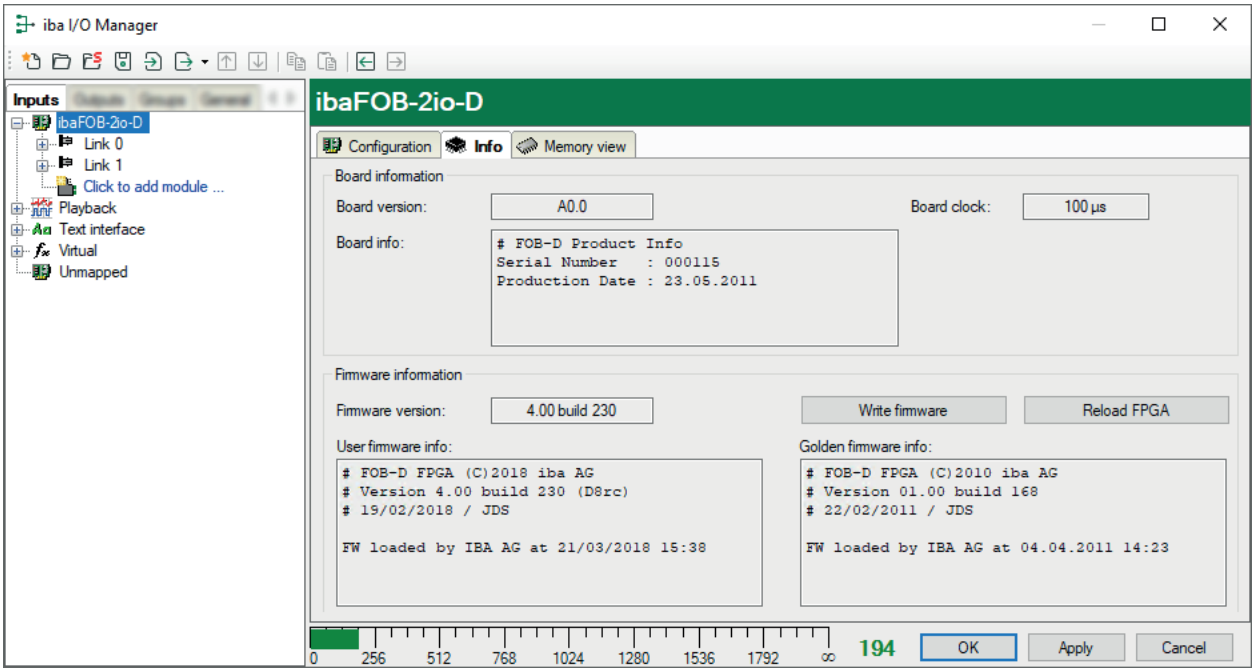
The *ibaFOB-4o* module can be connected to the "TX-OUT" or "TX-MIRROR" connector of an ibaFOB-...-D card and can thus fulfill different tasks.

- 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

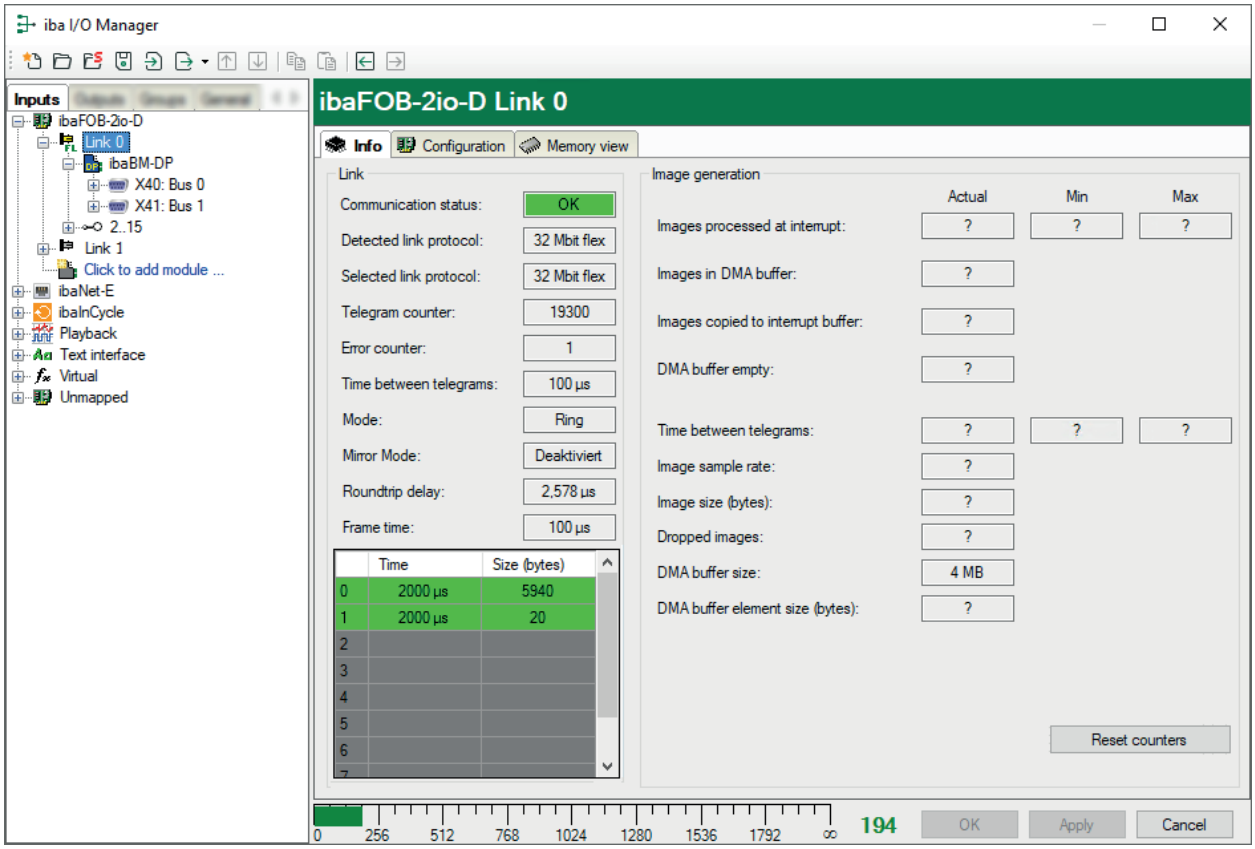
When the Link level is highlighted in the tree, you will find some information about communication and processors of the card.

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 i*baFOB-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 µs per device).

Due to the roundtrip delay the data of the connected devices might be captured asynchronously (up to one telegram cycle).

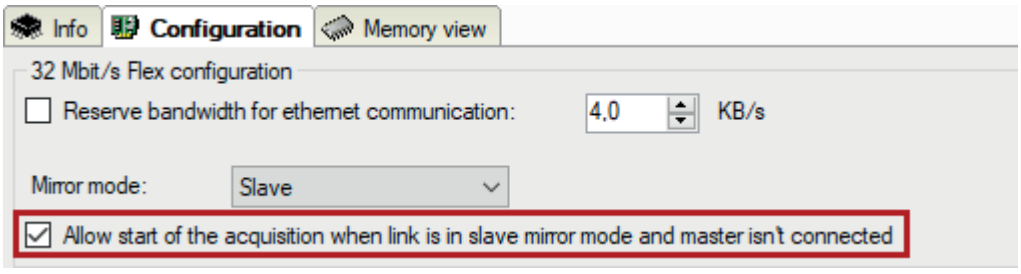
Frame time
Fixed cycle time the data frames are being sent. (Smallest set time base of the connected devices or 100 µs, if this time base is an integer multiple of 100 µs). The time base of all devices must be a multiple of the smallest time base.)

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 µs) 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.

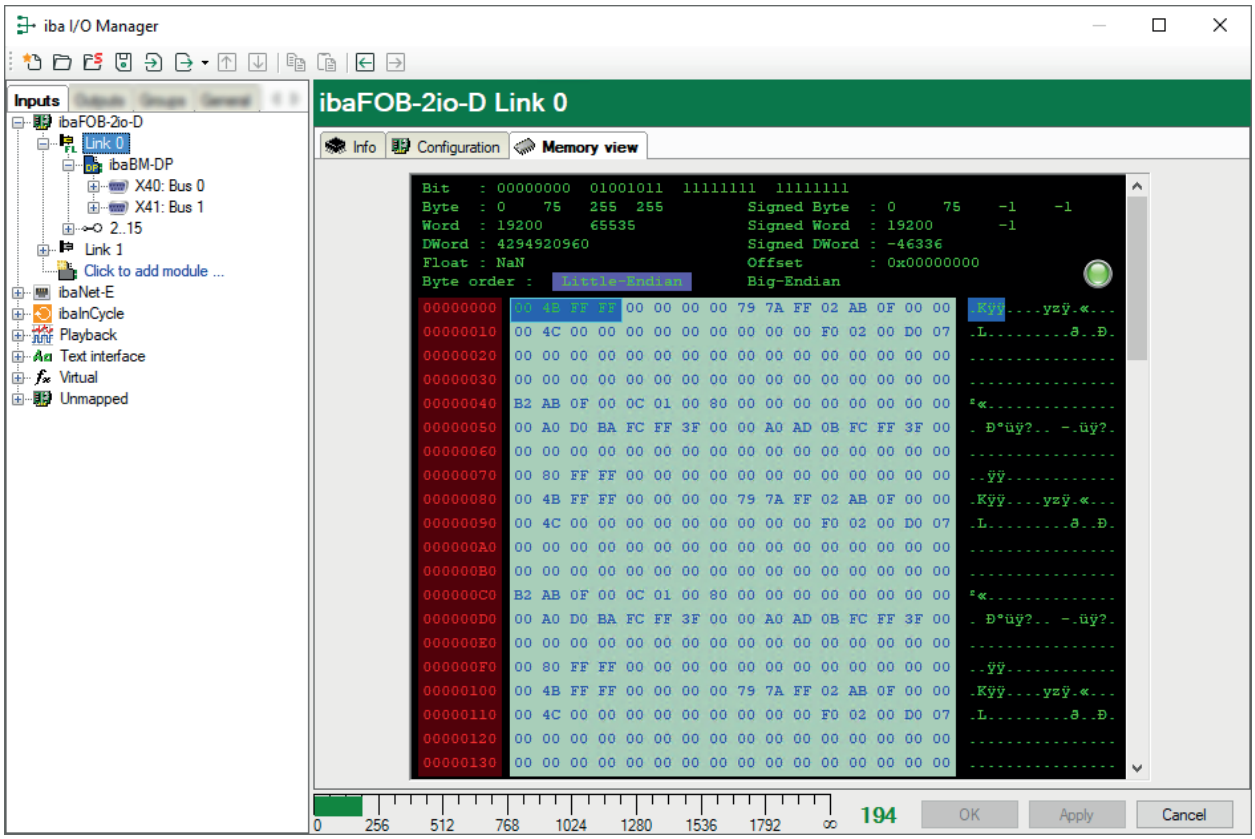
- Green: OK
- Orange: Band width for the Ethernet channel < 3 kB/s
- Red: Too much data configured.

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 (ibaNet-E generic connectionless) 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 are used for this interface. The sockets will be opened for communication only on the selected network adapters. In case a network adapter has multiple IP addresses configured, a socket will be opened for all of these IP addresses. At least one network adapter should be selected to get the interface configuration validated. If you select *None*, an error message will be displayed when validating the I/O configuration. By default, the option *All* is selected.

There is an option to stop the acquisition if a broken connection has been detected. Additionally, you can determine to set all signal values to zero either when a broken connection has been detected or when no data is available.

You can select the network adapters, which are dedicated for ibaNet-E communication in the *Discovery* tab and search for ibaNet-E compatible devices. All found active devices will be listed in a table.

The *Bandwidth* tab provides information about the network load. It shows the estimated total network load in send and receive direction as well as the network load per connected ibaNet-E device.

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.
- ibaM-COM
  This module is used to acquire data from input modules of the ibaMAQ system. For more information please refer to the manual of the *ibaM-COM* device.
- 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* 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.
- eLumina
  This module facilitates a connection to an eLumina CCU from GEVernova. The module is only available with an *ibaPDA-Interface-eLumina* license. One license allows up to 16 connections. For more information please refer to the manual of *ibaPDA-Interface-eLumina*.
- 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 16 connections

one-step-up-Interface-ibaNet-E (Art. no. 31.101006), extension license for 16 connections

These interface licenses are required if third-party devices will communicate with *ibaPDA* via ibaNet-E. The extension can be added up to 15 times, i.e., a maximum of 256 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-16 ibaPADU-32 ibaPADU-32 ibaPADU-8	ibaPADU-8 ibaPADU-16 ibaPADU-32 ibaPADU-32R ibaPADU-8AI-U	-S, -X, -D	8/16/32 AE ±10 V and 8/16/32 DE ±24 V Sampling rate 1 kHz	ibaPADU-8-16-32-32-R ibaPADU-8AI-U/-8AI-I
ibaPADU-8	ibaPADU-8-I ibaPADU-8-AI-I	-S, -X, -D	8 AE ±20 mA and 8 DE ±24 V acquisition rate 1 kHz	ibaPADU-8-I ibaPADU-8AI-U/-8AI-I
ibaPADU-8-M	ibaPADU-8-M	-S, -D	8 AE ±10 V and 8 DE ±24 V acquisition rate 25 kHz	ibaPADU-8-M
ibaPADU-8-ICP	ibaPADU-8-ICP	-S, -D	8 AE ICP sensor and 8 DE ±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 ±24 V acquisition rate 25 kHz	ibaPADU-16-M
ibaDig-40	ibaDIG-40	-S, -D	40 DE ±48 V acquisition rate 25 kHz	ibaDig-40
ibaPA-DU-D-8AI-U	ibaPA-DU-D-8AI-U	-D	8 AE ±2.5 V, ±10 V, ±24 V, ±60 V and 8 DE +/-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 ±20 mA, 0...20 mA, 4...20 mA and 8 DE ±24 V acquisition rate up to 40 kHz	ibaPADU-D-8AI-U/-8AI-I

Module name	Device	*ibaFOB*	Comment	Manual
ibaPADU-4-AI-U	ibaPA-DU-4-AI-U	-D	4 AE ±250 mV, ±500 mV, ±1 V, ±2.5 V, ±5 V, ±10 V, ±24 V acquisition rate up to 100 kHz	ibaPADU-4-AI-U
ibaPACO-4	ibaPACO-4	-S, -X, -D	4 counter inputs 8 DE ±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 23.

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
Central units			
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
Input modules			
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
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

Module name	Device	Comment	Manual
Output modules			
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
Input and output modules			
ibaMS4xADIO	ibaMS4xADIO	4 AE (±10 V oder ±20 mA) + 4 DE (24 V) + 4 AA (±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

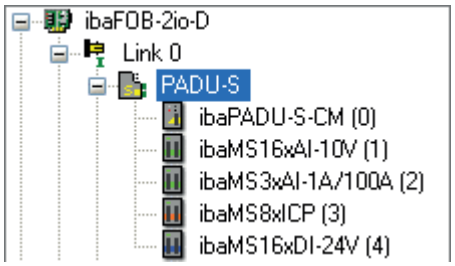
The structure of the devices and modules in the ibaPADU-S system is hierarchically structured.

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 DDCS Drive Bus; 32Mbit Flex	ibaBM-DDCS
ibaBM-DDCSM	ibaBM-DDCSM	-S, -X, -D/-Dexp	ABB DDCS 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	ibaBM-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
ibaBM-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		Module for monitoring	
Active slave decoder		as for active slaves, only for packed digital signals	
Sniffer decoder		as for sniffers, only for packed digital signals	
S7 Request		Module for request to S7 (optional access to symbols) [2)]	ibaPDA-Request-S7
S7 request (ibaCom-L2B-compatible)		Module for request to S7; for migration from old L2B connections [2]	
S7 request Dig512 (ibaCom-L2B-compatible)		Module for request to S7 (packed digital signals); for migration from old L2B connections [2]	
S7 request decoder		Module for request to S7 (packed digital signals) [2]	
FM458 Request		Module for request to S7-FM458 [2]	ibaPDA-Request-FM458/TDC
TDC request		Module for request to SIMATIC TDC [2]	

[2)] 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 [3)]	ibaPDA-Request-TwinCAT
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
S7 request decoder		Module for request to S7 (packed digital signals) (only available with license)	

[3)] 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.

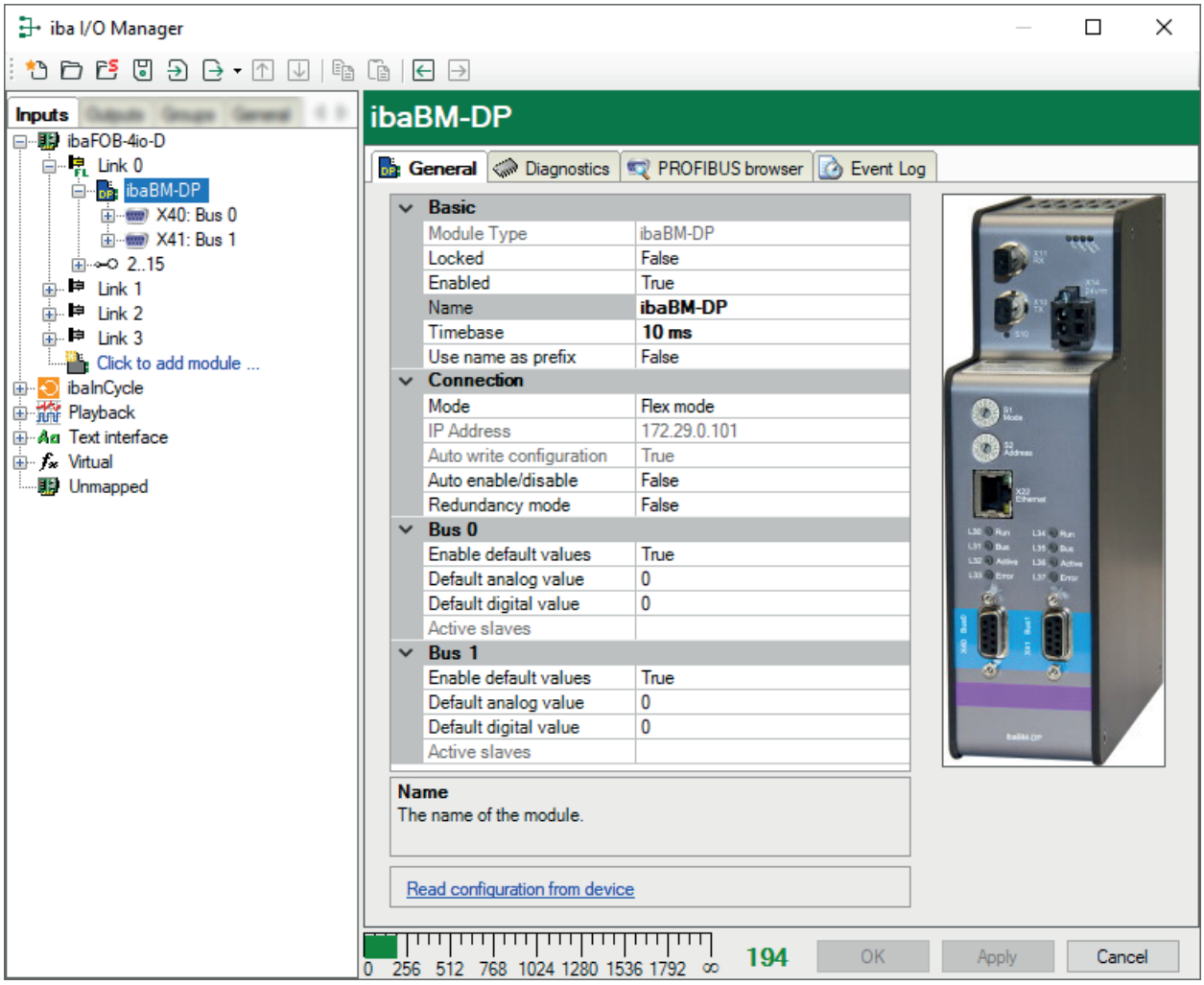
The "Active slave" submodule can also be used to output data via the Profibus. Devices and submodules are therefore also displayed in the *Outputs* area of the I/O Manager.

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



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

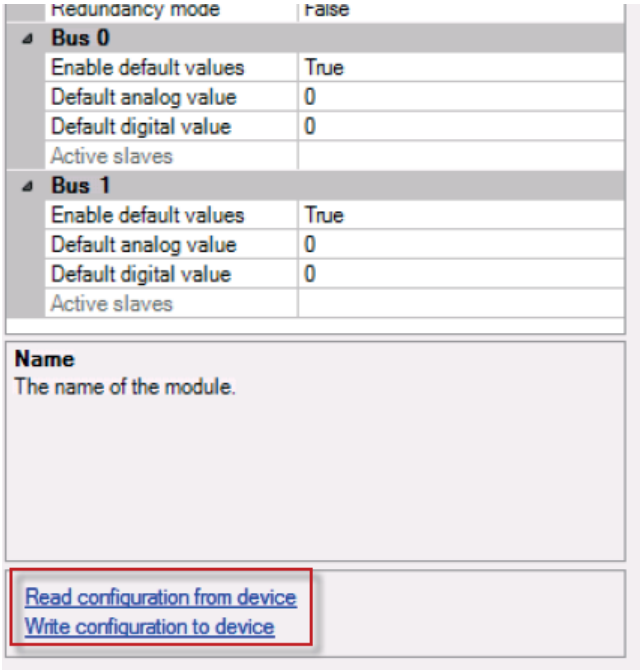
Should analog and digital access overlap and access the same addresses, the default analog value is overwritten by the default digital value at these points.

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.



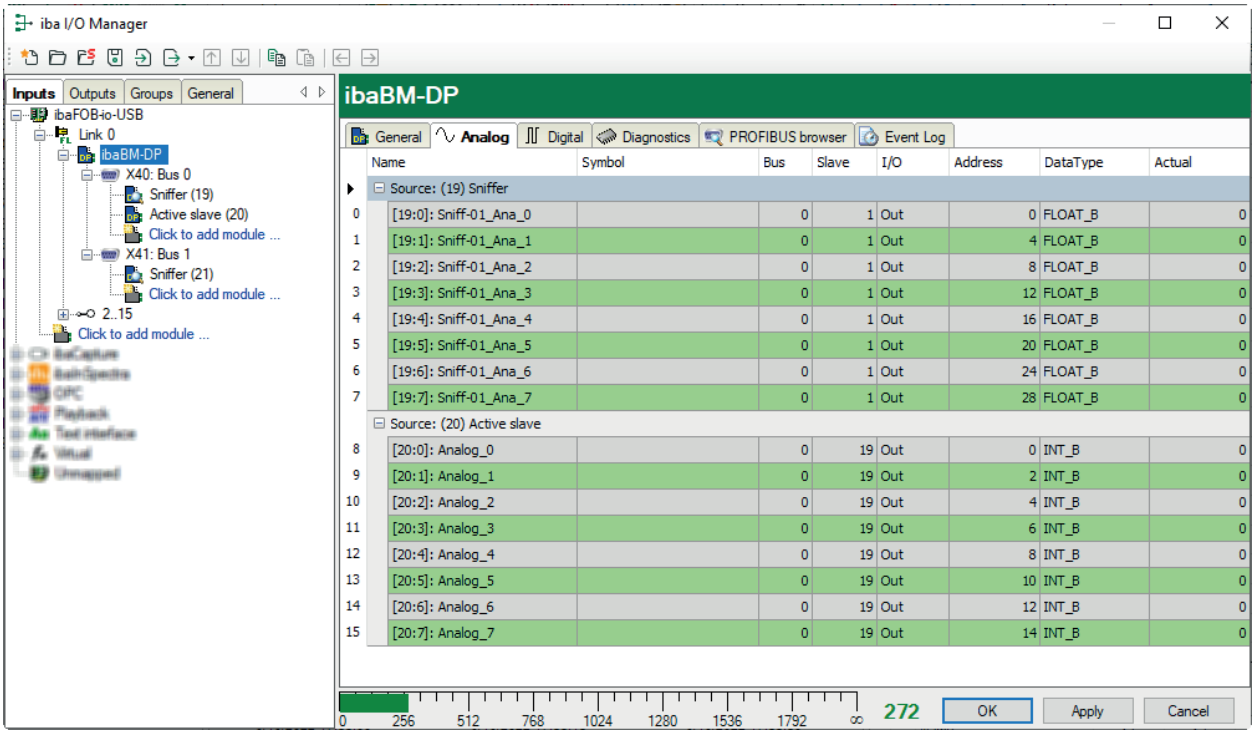
Notes

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.

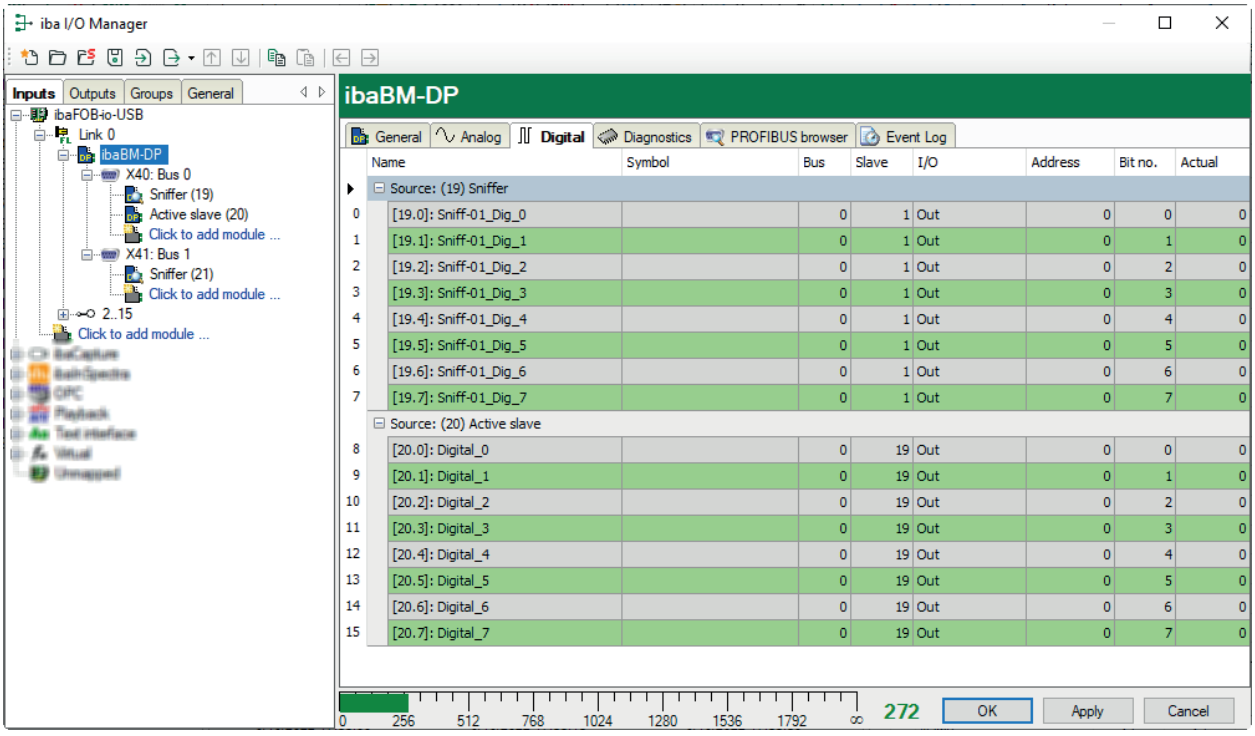
5.7.1.1.2 ibaBM-DP – Analog tab

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

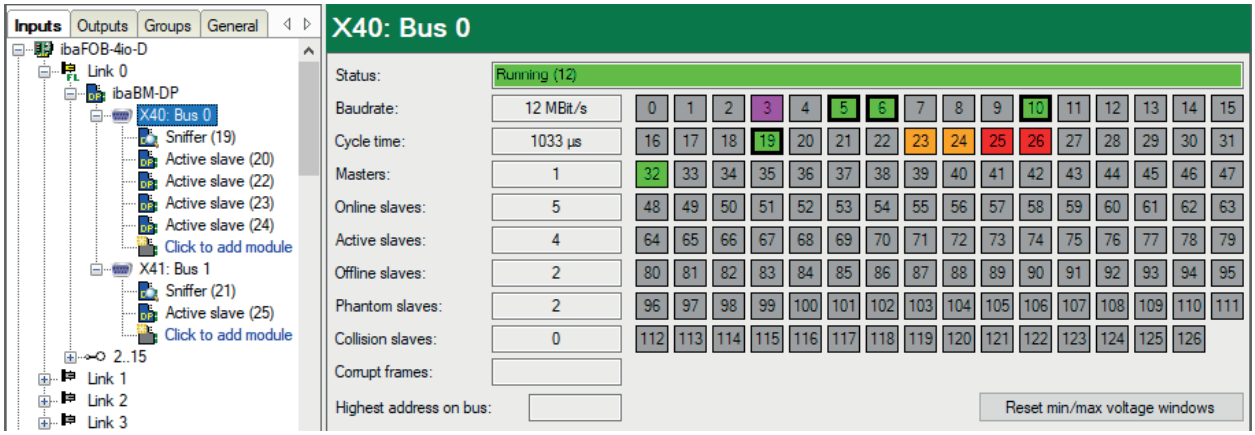
The detected masters and slaves as well as their respective status for the two PROFIBUS systems are displayed here.

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

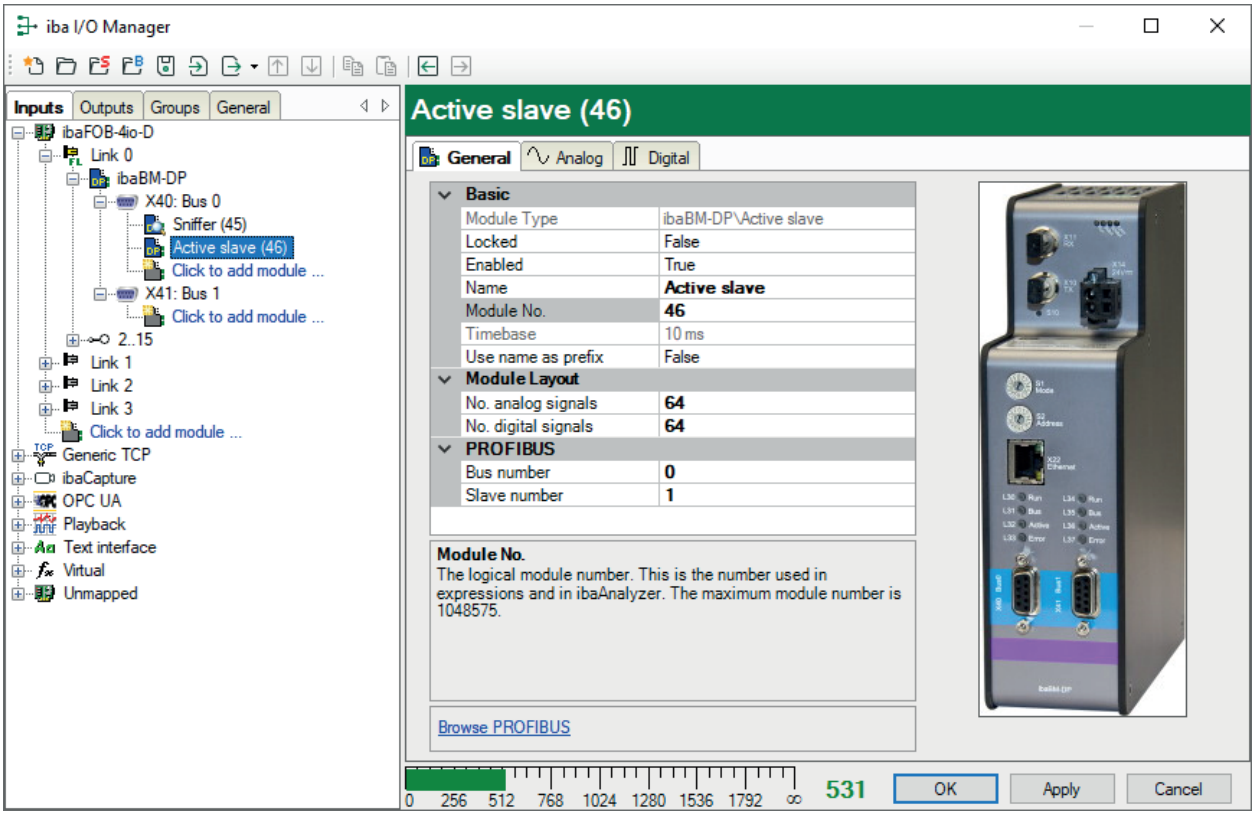
The status and diagnostics information for each connected PROFIBUS string is displayed here:



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



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.

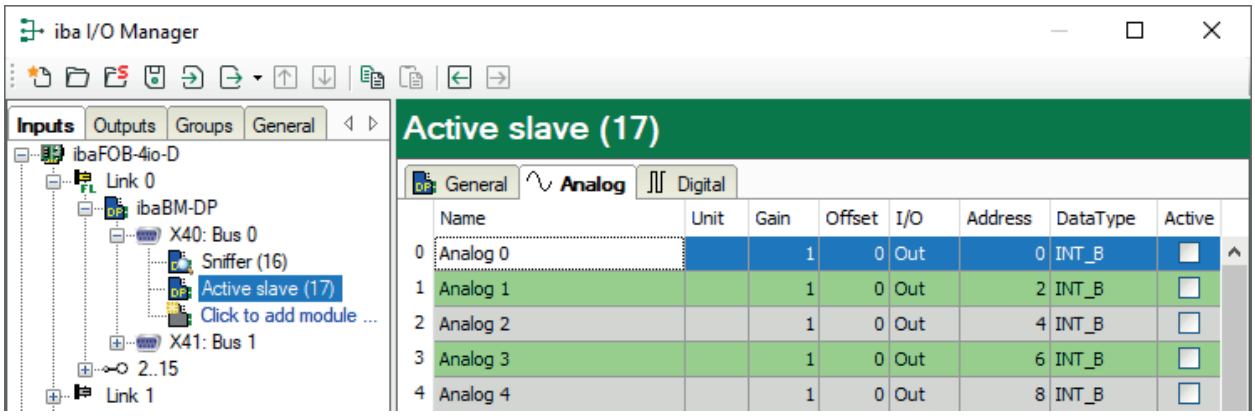
- Only connect the PROFIBUS cable after the "active slaves" have been properly configured so that it is ensured that there are no duplicate slave numbers.

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



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 23.

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.

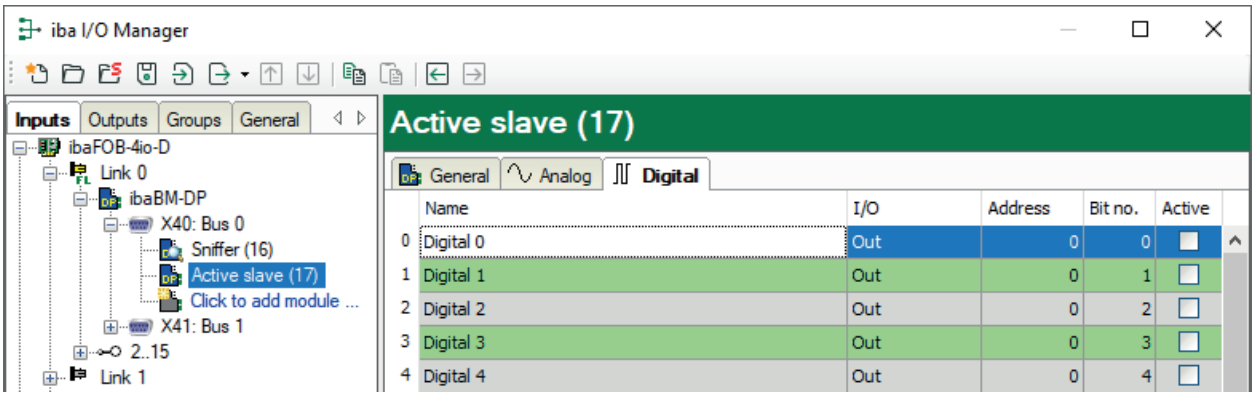
Data type
Data type of the signal. Available data types:

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 23.

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*.

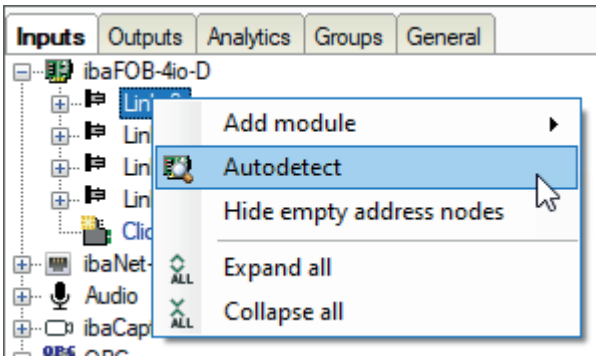
Information about the card's other applications and operating modes can be found in the card manual.

For more information see part 2, *Outputs*.

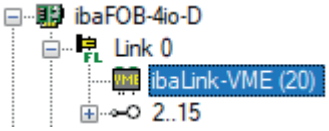
5.8.1.1 Add ibaLink VME module

In the 32Mbit Flex mode, the number of signals can be set flexibly in *ibaPDA*.

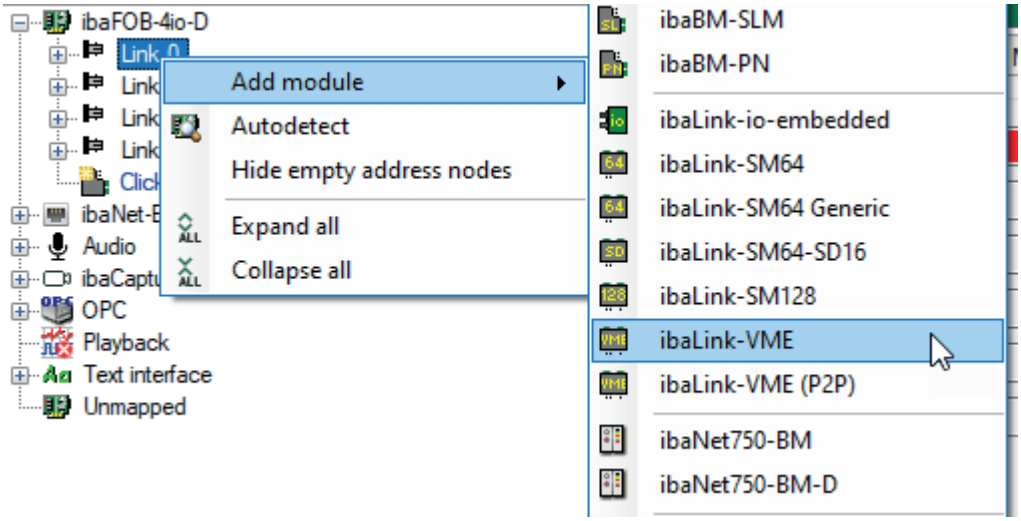
1. Start *ibaPDA* and open the I/O Manager.
2. In the signal tree (left), mark the link of the *ibaFOB-D* card to which *ibaLink-VME* is connected. Right-click the link to open a sub-menu. Select "Autodetect".



ibaPDA detects the component automatically and displays it in the signal tree.



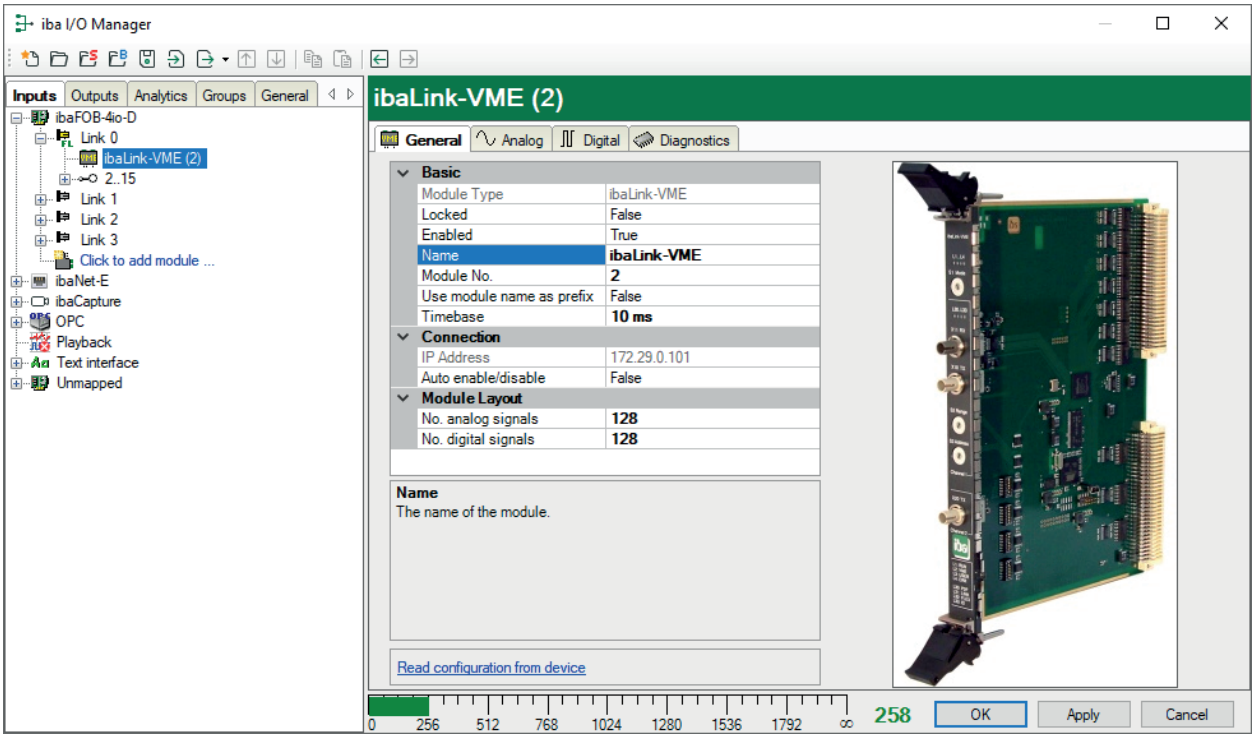
3. You can also add the component manually. Right-click on the link of the *ibaFOB-D* card to which the component should be connected and select "Add module" and "ibaLink-VME" from the list displayed.



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 µs (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.

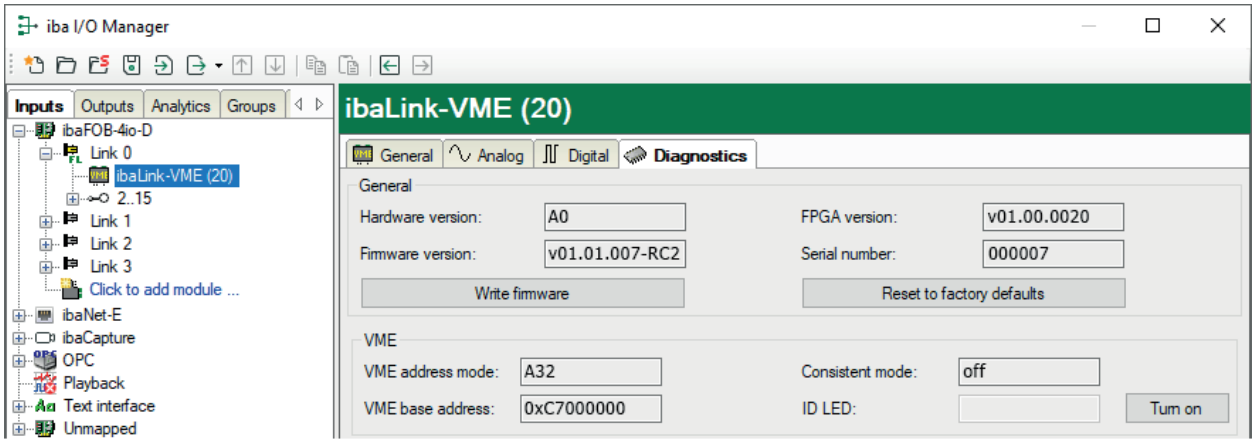
Read configuration from device
Reads the most recently stored configuration from the device.

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 23.

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 rate1 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.

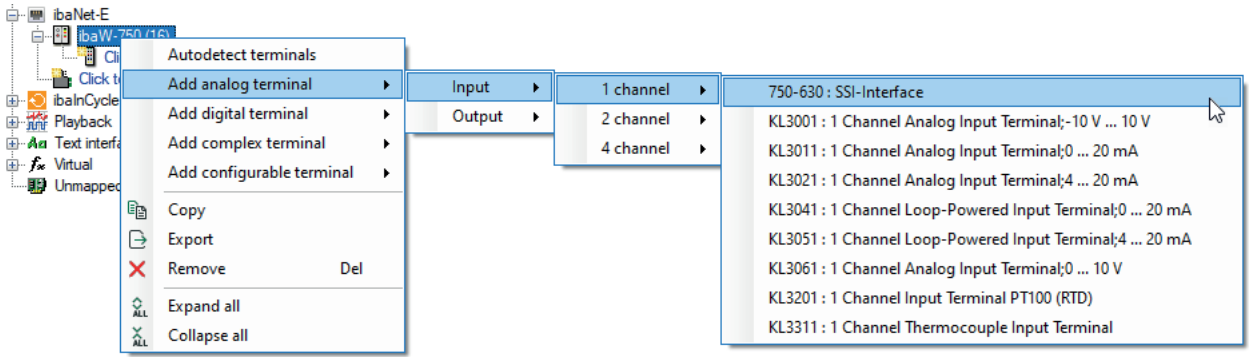
Adding connected devices

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.

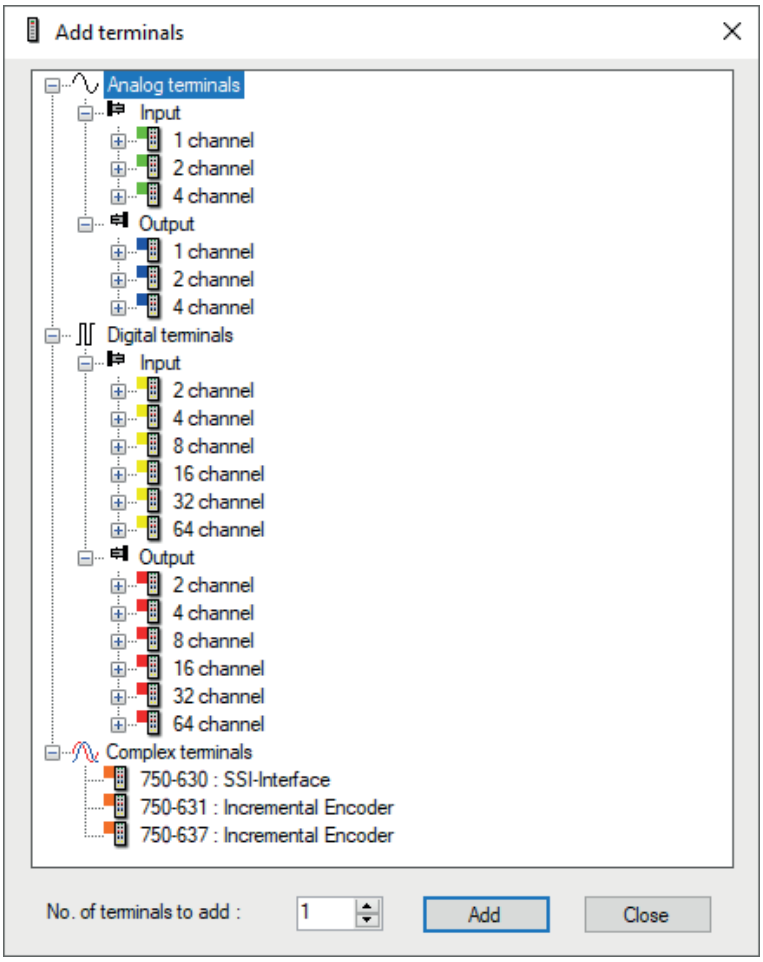


Via the Click to add terminal... command in the tree structure of the I/O Manager

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 23.

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 23.

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
AMEPA	NIC	AMEPA TSD measuring devices		↗ *AMEPA, page 176*
AN-X-DCSNet	NIC	RELIANCE DCS Network	via AN-X-DCSNet / Ethernet Gateway	↗ *AN-X-DCSNet, page 178*
CAN	NIC	CAN bus	via CAN-Ethernet gateway	↗ *CAN (Controller Area Network), page 179*
EGD	NIC	EGD network		↗ *EGD (Ethernet Global Data), page 181*
EtherNet/IP	NIC	Allen-Bradley, Ethernet-based	MicroLogix, ControlLogix, etc.	↗ *EtherNet/IP, page 182*
FBGS	NIC	FBGS interrogator	Temperature sensing	↗ *FBGS, page 184*
GCOM	NIC	GCOM multidrop bus	special modules for flatness available	↗ *GCOM, page 185*
Generic TCP	NIC	TCP/IP network		↗ *Generic TCP, page 186*
Generic UDP	NIC	UDP/IP network	optionally with HiPAC Request or TwinCAT Request	↗ *Generic UDP, page 188*
ibaLogic TCP	NIC	ibaLogic		↗ *ibaLogic TCP, page 189*
ibaNet-E	NIC	ibaW-750	Also 3rd party devices that support the ibaNet-E protocol.	↗ *ibaNet-E*, page 141
IEC 61850 Client	NIC	Compatible devices for protection and control technology	Station automation with MMS or GOOSE	↗ *IEC 61850 Client, page 190*
IEC 61850-9-2	NIC	Data sources with sampled values streams in accordance with IEC61850-9-2		↗ *IEC 61850-9-2, page 192*

Name	Type	Connection to …	Comment	Link
LAND Pyrometer	NIC	LAND SPOT+ pyrometer	Temperature measurement	↗ *LAND pyrometer, page 192*
LANDSCAN	NIC	Temperature scanner from LAND (Ametek)		↗ *LANDSCAN, page 193*
LMI Gocator	NIC	Laser scanners of the Gocator family	2D/3D profile acquisition	↗ *LMI-Gocator, page 194*
Micro-Epsilon	NIC	Laser scanners of the scanCONTROL family	2D/3D profile acquisition	↗ *Micro-Epsilon, page 195*
Modbus TCP Client	NIC	MODBUS over TCP, Modicon Quantum	ibaPDA is MODBUS client (master)	↗ *Modbus TCP client, page 197*
Modbus TCP Server	NIC	MODBUS over TCP, Modicon Quantum	ibaPDA is MODBUS server (slave)	↗ *Modbus TCP Server, page 198*
MQTT	NIC	MQTT broker	subscription only	↗ *MQTT interface, page 287*
OPC	NIC	OPC server (also redundant), OPC client	License-free standard interface OPC DA	↗ *OPC interface*, page 34
OPC UA	NIC	OPC UA server		↗ *OPC UA, page 196*
Raw Ethernet		Raw Ethernet Multicast		↗ *Raw Ethernet, page 200*
Raytek	NIC	Temperature scanner from Raytek (Fluke Process Instruments)		↗ *Raytek, page 201*
S7 TCP/UDP	NIC	SIMATIC S7		↗ *S7 TCP/UDP, page 201*
Sisteam TCP	NIC		Also SIMATIC TDC over CP5100	↗ *Sisteam TCP, page 203*
TDC TCP/UDP	NIC	SIMATIC TDC	via CP51M1	↗ *TDC TCP/UDP, page 204*
VIP TCP/UDP	NIC	ABB AC450 RMC, AC 800 M, AC 800 PEC	and Simatic S7	↗ *VIP TCP/UDP, page 206*

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 upt 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.

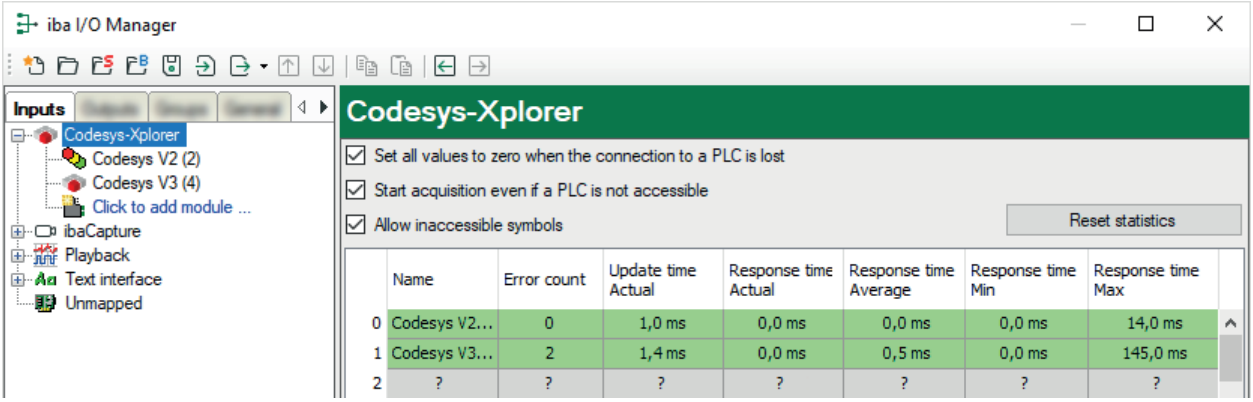
Typically, the target systems connected are identified in the first column (left) by its name or its IP address.

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.



6.3 AMEPA

Description

The bi-directional AMEPA interface is suitable for measured data acquisition of thermographic slag detection from AMEPA TSD measuring devices. Furthermore, you can send process data periodically to the connected AMEPA devices in order to optimize the measurement.

Interface configuration

The interface node AMEPA is available in the I/O Manager among the inputs and outputs. Depending on whether you select the *Inputs* or *Outputs* tab, the interface shows the information related to the messages received from or sent to the AMEPA devices by *ibaPDA*.

The interface has the following functions and configuration options:

Set all values to zero when the connection to a device is lost.
If enabled, all measured values of the AMEPA 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 the acquisition even if a device is not accessible
If this option is enabled, the acquisition will start even if the AMEPA device 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 device, *ibaPDA* will try to connect to the device at regular intervals.

<Open log file>
Messages related to the interface are written in a separate log file. To open the current log file click <Open log file>.

<Reset statistics>
Click this button to reset the calculated times and error counters in the table to 0.

Connection table
For each connection, the table shows the connection status, the current values for the update time (actual value, average, min. and max.) as well as the data size. In addition, there is an error counter for the individual connections during the acquisition.

Available modules

- AMEPA Generic
- AMEPA TSD

Product name

ibaPDA-Interface-AMEPA (art. no. 31.001019)

Other documentation

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

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

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.

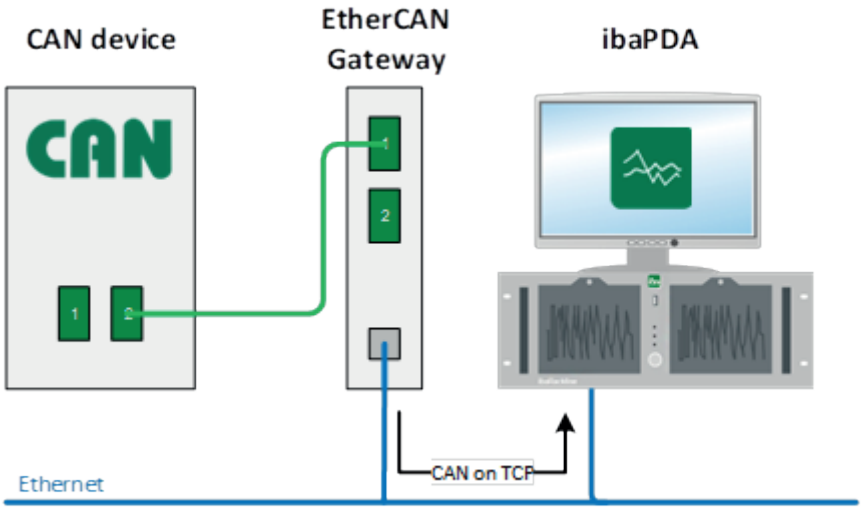
Configuration ID
This custom 32-bit configuration number determines the data structure.

6.5 CAN (Controller Area Network)

Description

The CAN interface is used to acquire data from a CAN bus as a data source. It was developed as a successor of the *ibaBM-CAN* device, which reached its end of life.

The connection between CAN bus and *ibaPDA* is established via a CAN-Ethernet gateway. On the *ibaPDA* side, simply the Ethernet network interface is required.



ibaPDA supports the following CAN-Ethernet gateways:

- CAN-Ethernet Gateway (2-channel) from iba (art. no. 19.000040)
- EtherCAN CI-ARM9/ MD-IBAGateway, Manufacturer: EMS Thomas Wuensche (art. no. 12-20-383-20)

  Ethernet/CAN interface with 32-bit microcontroller (ARM9 core) and two internal CAN controllers, preconfigured for iba AG

The device has two CAN bus connectors, which allow to communicate with two CAN bus networks on this device. The interface license permits up to two connections.

For a more convenient configuration of the measurement data, the interface provides an automatic detection of active CAN bus devices.

Interface configuration

This interface is only available in the *Inputs* tab of the I/O Manager, because outputs are not supported.

The interface has the following functions and configuration options:

Set all values to zero when the connection to a device is lost
If enabled, all measured values of the CAN 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 the acquisition even if a device is not accessible
If this option is enabled, the acquisition will start even if the CAN device 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 device, *ibaPDA* will try to connect to the device at regular intervals.

<Open log file>
Messages related to the interface are written in a separate log file. To open the current log file click <Open log file>.

<Reset statistics>
Click this button to reset the calculated times and error counters in the table to 0.

Connection table
The table shows the configured connections to the gateways. One row corresponds to one connection. For each connection there is one CAN module created in the I/O configuration.

The columns of the table provide information as follows:

- Name of the corresponding CAN module
- IP address or host name of the gateway
- Error counter for failed connection attempts and error messages from the CAN bus
- Counter for messages received
- Counter, which shows for how many different CAN IDs messages have been received.

The configuration of the connection has to be done in the settings of the CAN module.

Available modules

- CAN

Product name

ibaPDA-Interface-CAN (art.-no. 31.001027)

Other documentation

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

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

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*.

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

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*.

6.8 FBGS

The FBGS interface can be used to measure data from sensors, which are based on the FBG measuring principle and connected via a so called interrogator measuring device.

The abbreviation FBG means Fibre Bragg Gratings, a special opto-electronic measuring method with optical fibers by FBGS Technologies.

Out of the different applications which are possible with this measuring method *ibaPDA* supports only temperature sensing.

There are measuring devices (interrogators) available for 4, 8 and 15 channels. One channel corresponds to one optical fiber.

ibaPDA only supports interrogators with an Ethernet interface, such as

- FBG-scan 80x-EP
- FBG-scan 90x-EP

(x = 4, 8, 15)

Interface settings

Set all values to zero when the connection to an interrogator is lost.
If enabled, all measured values of the drive 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 the measurement even if an interrogator is not accessible
If this option is enabled, the acquisition will start even if an interrogator 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 an interrogator, *ibaPDA* will try to connect to the interrogator at regular intervals. If the connection attempt is successful, *ibaPDA* starts the acquisition of the data from the interrogator in question without restarting the general data acquisition.

<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 and error counters in the table to 0.

Connection table
For each connection, the table shows the connection status, the current values for the update time (actual value, average, min. and max.) as well as the data size. In addition, there is an error counter for the individual connections during the acquisition.

In addition, there is a column for the counter of the number of data sets (“spectra”) received from the FBGS-interrogator. One data set is a collection of peak wavelength and intensity pairs of all configured sensors of all configured channels (fibers) of the interrogator.

Available modules

- FBGS Interrogator (device module)
  - FBGS Raw data module
  - FBGS Temperature data module

Product name

ibaPDA-Interface-FBGS (Art. no. 31.001017)

One interrogator can be connected per interface license. Up to 64 licenses for a total of up to 64 devices can be activated. Regarding devices with 15 channels, this leads to a total of up to 960 channels. Each channel can contain one or more sensors.

Other documentation

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

6.9 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 sent 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*.

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

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*.

6.11 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.

The module type *Generic UDP* (unicast and multicast) allows to configure up to 6056 analog and 16384 digital signals.

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

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 correspondign products *ibaPDA-Request-HiPAC* and *ibaPDA-Request-TwinCAT*.

6.12 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 the respective license is valid. 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 are used for this interface. The sockets will be opened for communication only on the selected network adapters. In case a network adapter has multiple IP addresses configured, a socket will be opened for all of these IP addresses. At least one network adapter should be selected to get the interface configuration validated. If you select *None*, an error message will be displayed when validating the I/O configuration. By default, the option *All* is selected.

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 in 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*.

6.13 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 176*.

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.*

6.14 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*.

6.15 LAND pyrometer

The LAND pyrometer interface can be used to measure data from LAND/AMETEK pyrometers of the SPOT+ devices family.

You may use an unlimited number of pyrometer connections with one interface license.

The connection between the pyrometer and *ibaPDA* can be established via the computer’s standard network interfaces (Ethernet-TCP/IP).

One analog signal (temperature) is acquired from each pyrometer.

Interface configuration

Set all values to zero when the connection to a sensor is lost.
If enabled, all measured values of the LAND pyrometers 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 the measurement even if a sensor is not accessible
If this option is enabled, the acquisition will start even if a LAND pyrometer 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 device, *ibaPDA* will try to connect to the device at regular intervals.

Connection table
The table shows the read and error counters as well as the update times of the individual connections during data measurement. To reset the calculated times and counters to zero, simply click on the <Reset statistics> button.

Available modules

SPOT+

Product name

ibaPDA-Interface-LAND-SPOT+ Art. no. 31.001014)

Other documentation

A detailed description of this interface and its configuration can be found in the corresponding manual for the product *ibaPDA-Interface-LAND-SPOT+*.

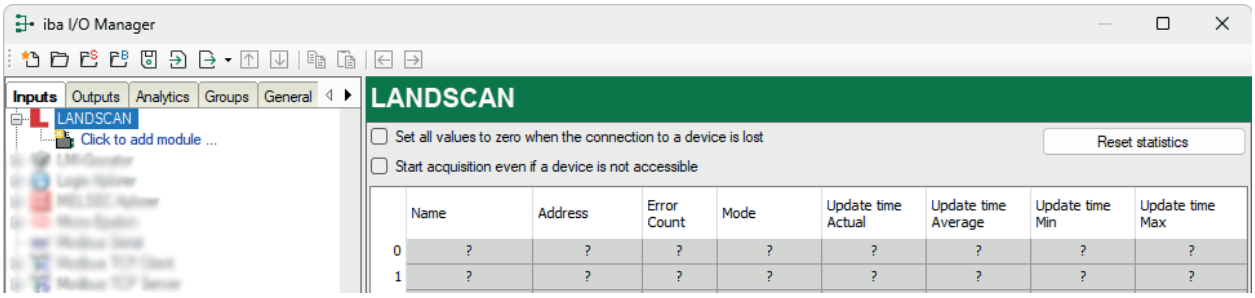
6.16 LANDSCAN

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:



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 statistics> 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.*

6.17 LMI-Gocator

Description
The LMI-Gocator interface is suitable for measurement data acquisition from Gocator® 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*.

6.18 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*.

6.19 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+*.

6.20 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 16 connections can be used per interface license. By purchasing up to 15 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-16 (Art. no. 31.001024)

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*.

6.21 Modbus TCP Server

Description

The Modbus TCP Server interface lets *ibaPDA* act as a server that expects a client connection. The PLC acts as client 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 are used for this interface. The sockets will be opened for communication only on the selected network adapters. In case a network adapter has multiple IP addresses configured, a socket will be opened for all of these IP addresses. At least one network adapter should be selected to get the interface configuration validated. If you select *None*, an error message will be displayed when validating the I/O configuration. By default, the option *All* is selected.

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 client
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*.

6.22 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*.

6.23 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*.

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

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 Request
- S7 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.*

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

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*.

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

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*.

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

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*.

6.28 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 173*

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	ibaPDA-Request-HiPAC
Codesys Request V2				
Codesys Request V3				
TwinCAT-Request			visible only with TwinCAT request license	ibaPDA-Request-TwinCAT
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	↗ *OPC interface*, 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		↗ *AB-Xplorer, page 211*
ABB-Xplorer	NIC	ABB AC800M, AC800PEC		↗ *ABB-Xplorer, page 212*
Bachmann-Xplorer	NIC	Bachmann M1		↗ *Bachmann-Xplorer, page 214*
B&R-Xplorer	NIC	B&R X20 and others		↗ *B&R-Xplorer, page 215*
Codesys-Xplorer	NIC	Control with CODESYS V2 or CODESYS V3	e.g., 3S CODESYS SP, ABB AC500, Danieli HiPAC and many others	↗ *Codesys-Xplorer, page 216*
Logix-Xplorer	NIC	Allen-Bradley ControlLogix, CompactLogix, MicroLogix		↗ *Logix-Xplorer, page 218*
Melsec-Xplorer	NIC	Mitsubishi MELSEC		↗ *MELSEC-Xplorer, page 219*
OMRON-Xplorer	NIC	OMRON PLC		↗ *OMRON-Xplorer, page 220*
S7-Xplorer	NIC, MPI/ Profibus	SIMATIC S7-200, -300, -400, -400H, -1200, -1500, LOGO! SIMATIC S5		↗ *S7-Xplorer, page 221*
Sigmatek-Xplorer	NIC	Sigmatek-CPUs		↗ *Sigmatek-Xplorer, page 223*
TwinCAT-Xplorer	NIC	Beckhoff PLC		↗ *TwinCAT-Xplorer, page 224*

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 a 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 176

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 176.

<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 176.

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 176.

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 176

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 176.

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 176.

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, an 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 176.

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. Interface adapters such as IBH Link S7++, can be used too. 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 (S5-LAN++, Process-Informatik or IBH Link S5++, IBHsoft-ec), you can also acquire measurement data from a SIMATIC S5 control 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 measured values of the PLC are set to zero as soon as the connection is lost. If this option is disabled, *ibaPDA* keeps the last valid measured value in memory when the connection is lost.

Start acquisition even if a PLC is not accessible
If this option is enabled, the acquisition starts even if the controller is not accessible. A warning is prompted in the validation dialog, not an error. If the system was started without a connection to the controller, *ibaPDA* periodically tries to connect to the PLC.

Allow inaccessible symbols
If this option is enabled, the acquisition starts even if symbols are not accessible. These inaccessible symbols are prompted as warnings in the validation dialog box, not as errors. This can only occur if the address book is not up-to-date.

If this option is disabled, measurement does not start when inaccessible symbols are present.

Enable S7-Xplorer outputs
Enable this option to use the output modules. With the S7 Xplorer outputs, you can 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 statistics>
Click this button to reset the calculated times and error counters in the table to 0.

Connection table
For information about the connection table, see ↗ *Connection table*, page 176.

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 (article 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-Sigmatek-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 176

Available modules

- Sigmatek-Xplorer

Product name

ibaPDA-Interface-Sigmatek-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 *ibaPDA*Symbol 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 176.

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		↗ *FOB-SD/-SDexp and FOB-TDC/-TD-Cexp, page 227*
ibaFOB-TDC ibaFOB-TDCexp	PCI card PCIe card	SIMATIC TDC		↗ *FOB-SD/-SDexp and FOB-TDC/-TD-Cexp, page 227*
Simadyn Request	Software	SIMADYN D	ibaFOB-SD required	↗ *TDC Request and Simadyn Request, page 229*
TDC Request	Software	SIMATIC TDC	ibaFOB-TDC required	↗ *TDC Request and Simadyn Request, page 229*
ibaFOB-PlusControl	PCI card	PlusControl	Special card for Siemens Energy	
CP1616 CP1626	Siemens PCI-/PCIe card	PROFINET		↗ *CP1616, page 230* ↗ *CP1626, page 231*
MMC request	Software	Simicro MMC216		↗ *MMC request, page 233*
S7-Xplorer	Software	SIMATIC S7	ibaPDA-PLC-Xplorer or ibaPDA-Interface-S7-Xplorer	↗ *S7-Xplorer*, page 221
SINAMICS-Xplorer	Software	SINAMICS frequency converters		↗ *SINAMICS-Xplorer, page 235*
SIMOTION-Xplorer	Software	SIMOTION frequency converters		↗ *SIMOTION-Xplorer, page 236*
SINUMERIK-Xplorer	Software	SINUMERIK CNC controls, NCK		↗ *SINUMERIK-Xplorer, page 238*

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 ibaFOB-SDexp	Request-SD-TDC	Request-SD
SIMATIC TDC	CP53				
	GDM		ibaFOB-TDC ibaFOB-TDC-exp		Request-TDC
	CP50	PROFIBUS	ibaBM-DP ibaBM-DPM-S ibaCom-L2B	Request-FM458-TDC	Request-FM458/TDC
S7-400	FM458				
S7-300	S7-CPU CP x43	PROFINET	ibaBM-PN	Request-S7	Request-S7
		UDP	ibaNet-E		
S7-1200			Ethernet	Request-S7	Request-S7
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 restarts.

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, incl. SINUMERIK One, 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*. NC data and SINAMICS servo symbols can be acquired in combination as well as symbols of multiple SINAMICS servo drives. Moreover, the SINUMERIK symbol browser supports user defined symbols ("GUD").

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	↗ *DTBox-Request, page 242*
GH180-Xplorer	Software license	Innomotics Perfect Harmony GH180 medium voltage drives	-	↗ *GH180-Xplorer, page 256*
Hitachi MicroSigma	Software license	Hitachi µΣNetwork 1000-Network	special hardware IKS-LM-SN1G or -SN100 required	↗ *Hitachi MicroSigma, page 257*
HPCi DGM200P	Software license	GE Energy Power Conversion HPCi, HPC LD2 (CC100, DGM 200)	DGM200P board (PCI) from GE Energy Power Conversion required	↗ *DGM200P, page 245*
HPCi DGM200E	Software license	GE Energy Power Conversion HPCi, HPC LD2 (CC100, DGM 200)	DGM200E adapter from GE Energy Power Conversion required	↗ *DGM200E, page 243*
HPCi Request	Software license	GE Energy Power Conversion HPCi	works with DGM200E, DGM200P, SM128V and Reflective Memory	↗ *HPCi Request, page 258*
ibaFOB-H	PCIe board	Hitachi MACH3 HVDC	via PS935	↗ *ibaFOB-H, page 261*
Modbus serial	S232 or similar	Modbus serial interface	Modbus Serial Slave or Modbus Serial Master	↗ *Modbus serial, page 262*
PC Link	DCS-NET interface card for PCLink/Automax e.g. Reliance 5136-RE2-PCI	PCLink network	-	↗ *PC Link, page 264*
Reflective Memory	VMIPCI 5565, 5576, 5579, 5588 etc.	VMIC counterparts, such as GE HPCi, SMS X-Pact	-	↗ *Reflective memory, page 270*
ScramNet+	CWC SC150/150e	SCRAMNet+ network	-	↗ *ScramNet+, page 271*
Toshiba ADMAP JAMI1	PCI card Toshiba ADMAP JAMI1	ADMAP-5M bus	-	↗ *Toshiba ADMAP JAMI1, page 276*

Name	Type	Connection to...	Comment	Link
XPact	Software license	SMS Siemag X-Pact	works with ibaLink-VME (SM128V) and Reflective Memory	↗ *X-Pact, page 277*
XPact Request	Software license	SMS Siemag X-Pact	works with ibaLink-VME (SM128V) and Reflective Memory	↗ *X-Pact Request, page 280*

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 Vernova Power Conversion. 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 Vernova Power Conversion (http://www.gevernova.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 Vernova Power Conversion. 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 Vernova Power Conversion (http://www.gevernova.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 DPM200P 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

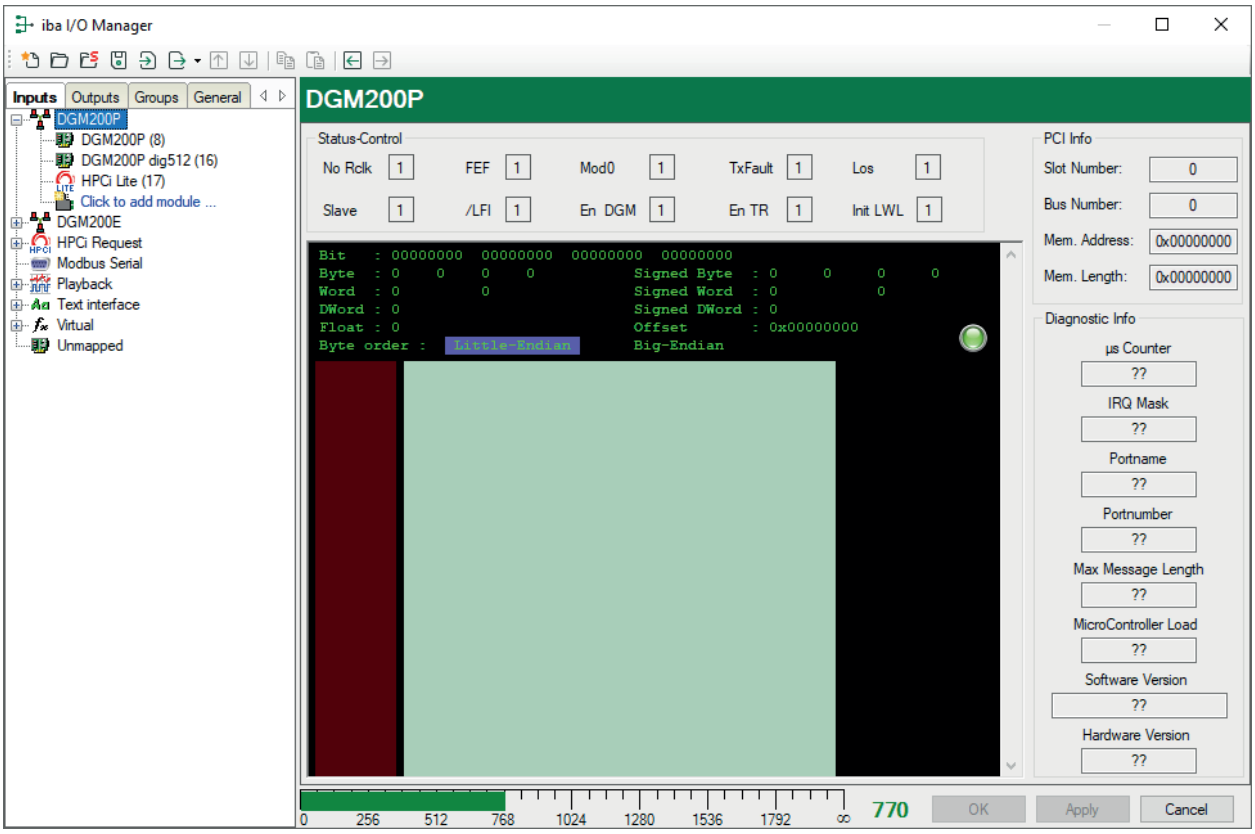
A maximum of up to 1024 modules are supported per interface.

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:

- ↗ *Module type HPCi Lite, page 252*
- ↗ *Module type DGM200P, page 248*
- ↗ *Module type DGM200P dig512, page 251*

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 Module type DGM200P

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 µs. 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 23.

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	$1.175 \cdot 10^{-38}$ …$3.403 \cdot 10^{38}$
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 23.

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 Module type DGM200P dig512

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 activated/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 Module type HPCi Lite

The *HPCi Lite* module type is used, in particular, for the acquisition of up to 1000 analog signals and 1000 digital signals from a GE HPCi system over DGM 200.

You can also use this module type in combination with an HPC ("Logidyn D2").

The licenses for the *DGM200P* and *DGM200E* interfaces include this module type. This allows you to easily select the signals to be measured using a 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 ("CC 100 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 CC 100 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 µs. 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 23.

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 23.

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.

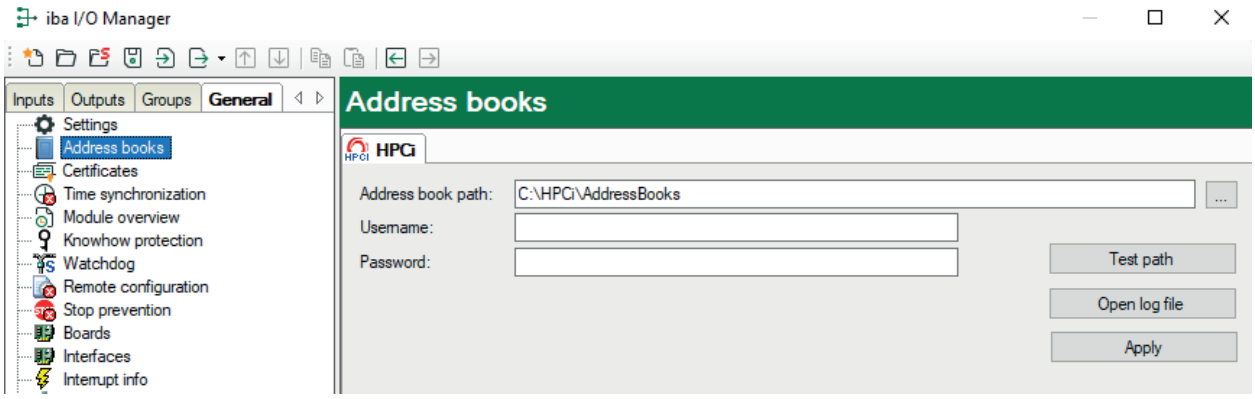
When selecting a signal from the signal browser, the HPCi symbol name is entered into the "Name" column. You may change the name manually afterwards.

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. Converteam). 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 GH180-Xplorer

The GH180-Xplorer interface can be used to measure data from Innomotics Perfect Harmony GH180 medium voltage drives.

You may use one connection to one drive per interface license. A total of 255 connections (= 255 drives) can be used with additional step-up licenses.

The communication between the drive and *ibaPDA* is established via the computer's standard network interfaces. The access is transparent for the drive controller. It is not necessary to configure or program the drive's side separately.

This is an "Xplorer" interface, i. e. the data is not sent actively by the drive controller but requested and read cyclically by *ibaPDA*.

Different module types are available for the signals to be measured. The standard module provides a pre-defined data set with analog and digital signals. Additionally, you have modules available for reading the SOP pages of the drive or for configuring your own measurement signals.

Interface settings

Set all values to zero when the connection to a drive is lost.
If enabled, all measured values of the drive 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 the measurement 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 prompted 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.

<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 and error counters in the table to 0.

Connection table

For each connection, the table shows the connection status, the current values for the update time and the response time (actual value, average, min. and max.) as well as the number of request commands. In addition, you will find an error counter here for the individual connections during the acquisition.

Available modules

- GH180 (one per drive)
  - Standard (pre-defined standard data set)
  - SOP
  - Custom

Product name

ibaPDA-Interface-GH180-Xplorer (Art. no. 31.001051)

The GH180-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-GH180-Xplorer*.

9.5 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 µΣNETWORK-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

- µΣ UDP data modules
- µΣ 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*.

9.6 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

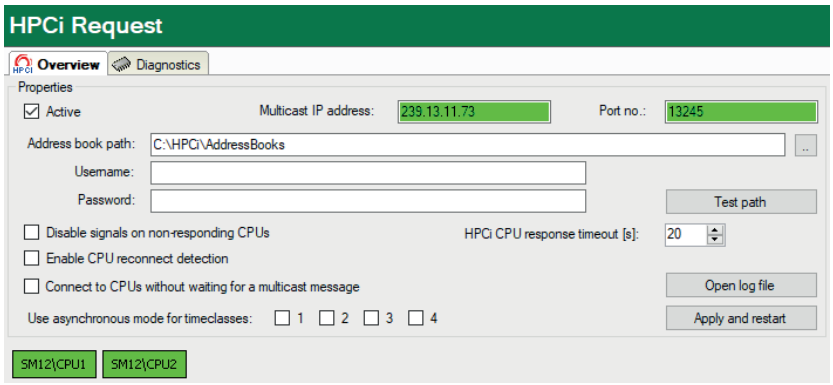
A special tool in the HPCi programming software, the address book generator, creates a reference file, which consists of system configuration and the available signals on the DGM200 and in the HPCi stations. After selection of the signals to be measured in the I/O Manager of *ibaPDA* (via browser or 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*.

9.6.1 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 µs 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

9.6.2 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.

9.7 ibaFOB-H

Description

The *ibaFOB-H* PC card is used for the connection from Hitachi Energy MACH3 HVDC systems to the *ibaPDA* data acquisition system. The card fits in a 1.0-x1-compatible PCI Express slot.

The connection supports a data transmission rate of 1 Gbit/s. Up to 400 analog or digital signals can be transmitted.

Updating the firmware can be done by *ibaPDA* without disassembling the card.

Interface configuration

As with the other ibaFOB cards, you should first select the card node in the interface tree in the I/O Manager.

You can set the interrupt mode in the *Configuration* tab, if needed.

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-H* 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 connection to the Hitachi Energy MACH3 HVDC system is established by clicking on the <Test connection> button in the *Connection* tab.

Available modules

- Hitachi Standard

Product name

ibaFOB-H (Art. no. 11.112610)

Other documentation

A detailed description of this interface and its configuration can be found in the corresponding manual for the card *ibaFOB-H*.

9.8 Modbus serial

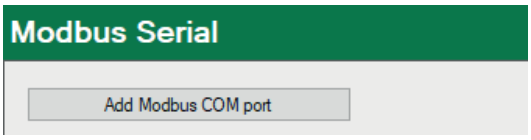
Description

This interface should be used when *ibaPDA* is connected to a Modbus network via a serial interface (COM port).

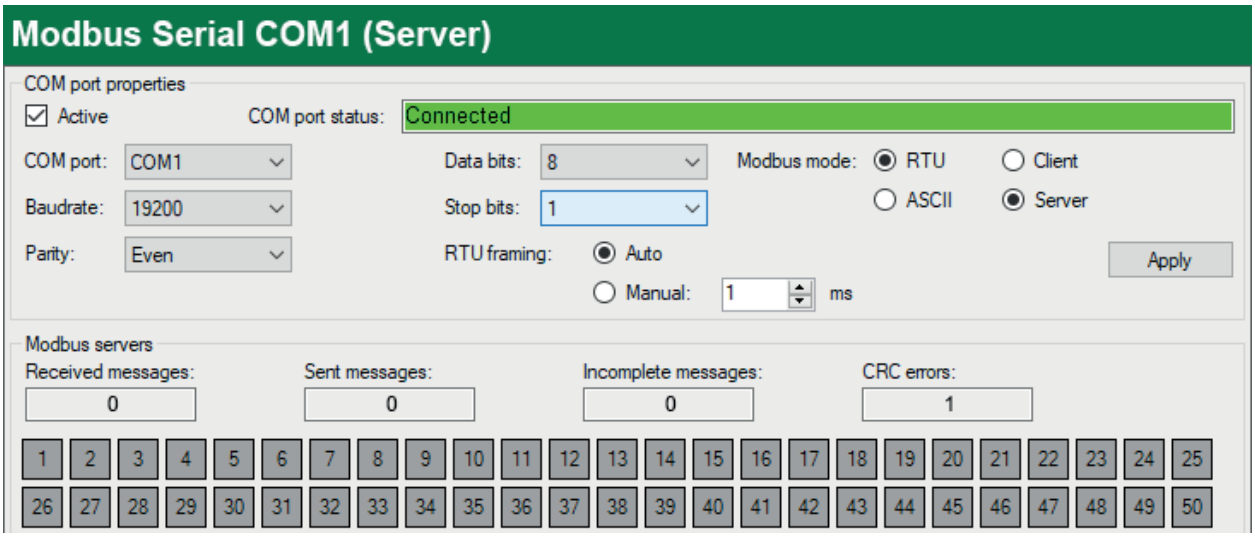
ibaPDA supports the Modbus modes master and slave as well as RTU and ASCII.

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.



- 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 *ibaPDA*.	The master is sending periodic requests to *ibaPDA*.
Disconnected (red)	The slave does not respond to periodic requests from *ibaPDA*.	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

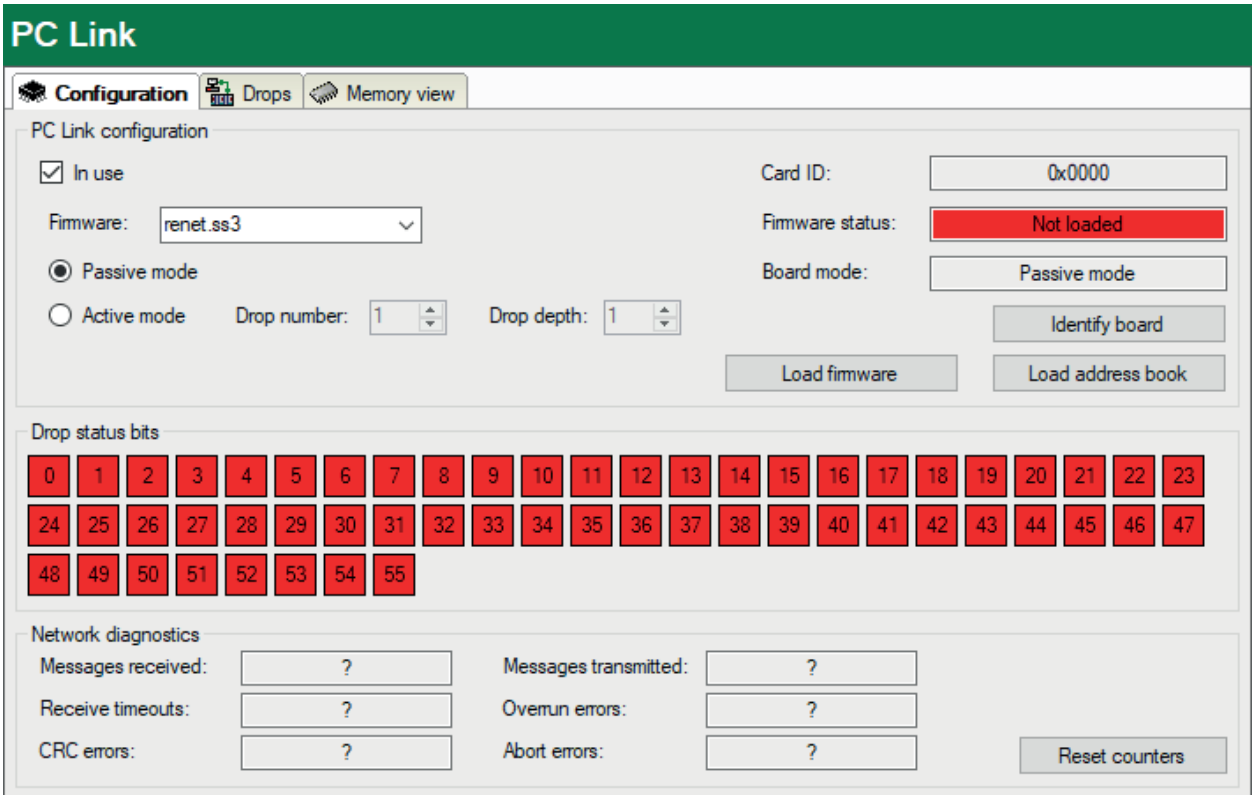
Product name

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*.

9.9 PC Link



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 266*
- ↗ *PC Link Dig512 module type, page 267*
- ↗ *PC Link Symbolic module, page 268*

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*.

9.9.1 PC Link module

The PC Link module type should be used if no address book of the DCS network is available.

9.9.1.1 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 µs. 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.

9.9.1.2 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 23.

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.

9.9.2 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.

9.9.2.1 PC Link Dig512 – General tab

Basic settings

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

9.9.2.2 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 23.

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.

9.9.3 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 264* 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.

9.9.3.1 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 µs. 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.

9.9.3.2 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 23.

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 saver.

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.

9.10 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 don 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.*.

9.11 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 272*
- ↗ *ScramNet dig512 module type, page 275*

9.11.1 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.

9.11.2 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.

9.11.2.1 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 µs. 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.

9.11.2.2 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 23.

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	±3.402823 · E+38 ... ±1.175495 · 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.

9.11.2.3 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 23.

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.

9.11.3 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.

9.11.3.1 ScramNet dig512 – General tab

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

9.11.3.2 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 23.

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.

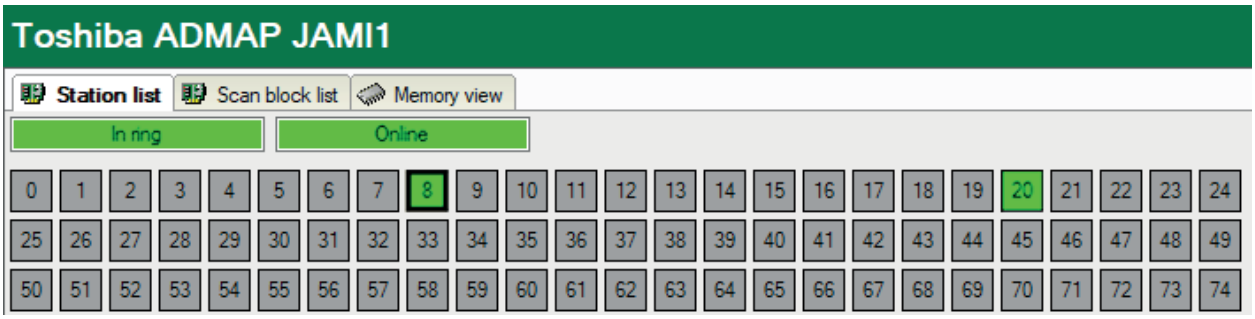
Address
In the address column (connector line), the byte offset in the range of every single connector (= integer package) may be specified by the user. The offset can be entered as hexadecimal or decimal values by selecting the appropriate option from the context menu. Usually, the default values need 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 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").

9.12 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*.

9.13 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 278*.

Product name

ibaPDA-Interface-Request-X-Pact (art. no. 31.001340)

Other documentation

On request.

9.13.1 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.

9.13.1.1 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 µs. 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.

9.13.1.2 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 23.

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.

9.14 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 282*.

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*.

9.14.1 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.

9.14.1.1 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).

9.14.1.2 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 23.

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) 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 *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 3.1.1 and MQTT 5 specifications are supported.

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)	*Inputs* tab	↗ *Audio, page 290*
ibaInCycle	Software license	internal, InCycle functions	*Analytics* tab	↗ *ibaInCycle, page 291*
ibaInSpectra	Software license	internal, InSpectra functions	*Analytics* tab	↗ *ibaInSpectra, page 292*
Snapshots	Software license	internal, Computation module, InSpectra Expert	*Analytics* tab	↗ *Snapshots, page 293*

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.

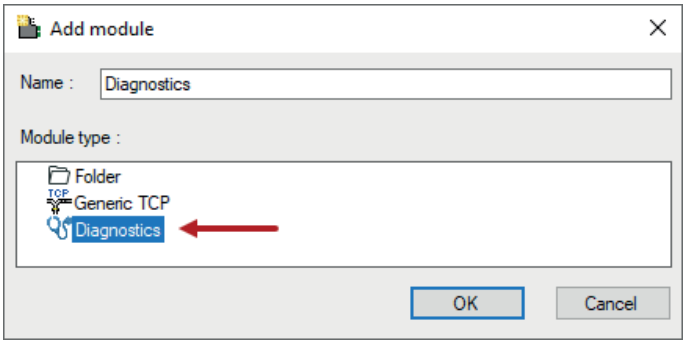
A diagnostic module is always assigned to a data acquisition module of the same interface and supplies its connection information. By using a diagnostic module, you can record and analyze the diagnostic information continuously in the *ibaPDA* system.

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:

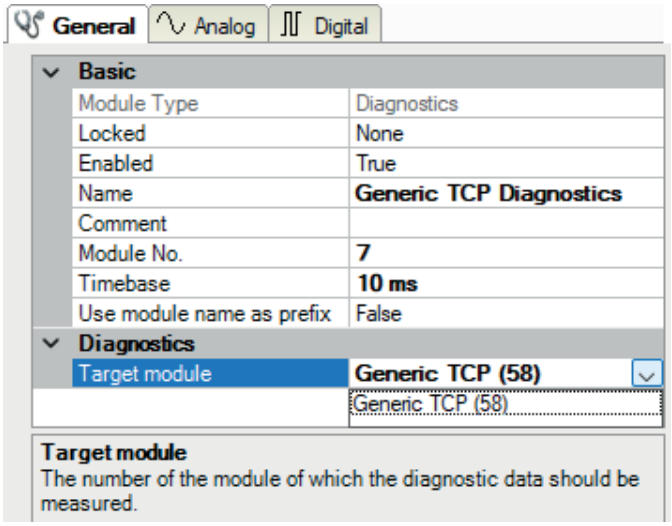
- A notification can be generated, whenever the error counter of a communication connection exceeds a certain value or the connection gets lost.
- In case of a disturbance, the current response times in the telegram traffic may be documented in an incident report.
- The connection status can be visualized 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

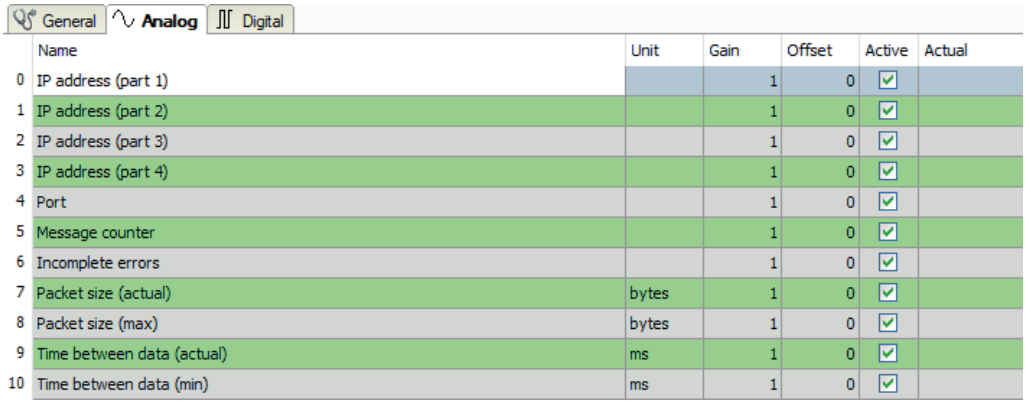
For a diagnostic module, you can make the following settings (example: Generic TCP):



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

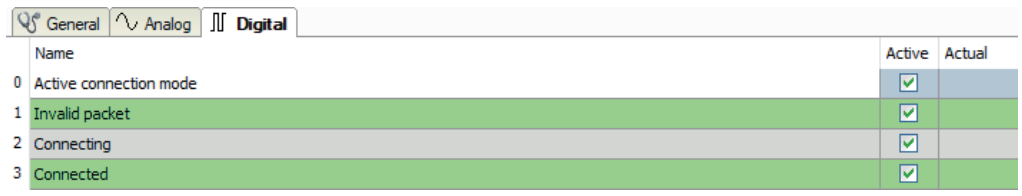
There is only one setting which is specific for the diagnostic module: the target module.

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.



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

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



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

Diagnostic signals

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

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

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

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

13 Support and contact

Support

Phone: +49 911 97282-14

Email: support@iba-ag.com

Note

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

Product Security Incident Response Team (PSIRT) at iba

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

Email: psirt@iba-ag.com

Contact

Headquarters

iba AG
Gebhardtstrasse 10–20
90762 Fuerth
Germany

Phone: +49 911 97282-0

Email: iba@iba-ag.com

Mailing address

iba AG
Postbox 1828
D-90708 Fuerth, Germany

Delivery address

iba AG
Gebhardtstrasse 10
90762 Fuerth, Germany

Regional and worldwide

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

www.iba-ag.com

ibaPDA

Expression Builder

Manual Part 4

Issue 8.14

Measurement Systems for Industry and Energy

www.iba-ag.com

Manufacturer

iba AG

Gebhardtstrasse 10-20

90762 Fuerth

Germany

Contacts

Main office	+49 911 97282-0
Support	+49 911 97282-14
Engineering	+49 911 97282-13
E-mail	iba@iba-ag.com
Web	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 2026, 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 and can be found in the iba help center docs.iba-ag.com.

Version	Date	Revision	Author	Version SW
8.14	06-2026	New: CharsToText; update retentive functions	rm	8.14.0

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

1 About this documentation 8

1.1 Target group and previous knowledge 8

1.2 Notations 8

1.3 Used symbols 9

1.4 Documentation structure 10

2 Expression builder (virtual signals) 11

2.1 Encryption of certain text arguments 12

2.2 Logical functions 12

2.2.1 Comparison functions >, >=, <, <=, <>, = ... 12

2.2.2 Boolean functions 12

2.2.3 Bitwise logical combinations 14

2.2.4 ExtendPulse 15

2.2.5 F_TRIG 16

2.2.6 FALSE 16

2.2.7 If 16

2.2.8 OneShot 17

2.2.9 R_TRIG 19

2.2.10 SetReset 20

2.2.11 Switch 22

2.2.12 TOF 23

2.2.13 Toggle 24

2.2.14 TON 25

2.2.15 TP 26

2.2.16 TRUE 26

2.3 Mathematical functions 27

2.3.1 Fundamental arithmetic operations +, -, *, / 27

2.3.2 Abs 28

2.3.3 Add 29

2.3.4 Ceiling 30

2.3.5 Diff 30

2.3.6 Eff 32

2.3.7 Exp 33
2.3.8 Floor 33
2.3.9 Int 34
2.3.10 Log 36
2.3.11 Log10 37
2.3.12 Mod 37
2.3.13 Pow 38
2.3.14 Round 39
2.3.15 Sqrt 40
2.3.16 Truncate 41
2.4 Trigonometric functions 42
2.5 Statistical functions 43
2.5.1 Avg 43
2.5.2 Avg2 44
2.5.3 AvgInTime 45
2.5.4 KurtosisInTime 46
2.5.5 MAvg 47
2.5.6 MAvgOnTrigger 49
2.5.7 Max 51
2.5.8 Max2 52
2.5.9 MaxInTime 53
2.5.10 Median2 55
2.5.11 MedianInTime 55
2.5.12 Min 57
2.5.13 Min2 58
2.5.14 MinInTime 59
2.5.15 MKurtosis 61
2.5.16 MMax 61
2.5.17 MMaxDynamic 63
2.5.18 MMedian 63
2.5.19 MMin 64
2.5.20 MMinDynamic 65

2.5.21 MSkewness 66
2.5.22 MStdDev 66
2.5.23 SkewnessInTime 67
2.5.24 StdDev 68
2.5.25 StdDev2 69
2.5.26 StddevInTime 69
2.6 Trigger functions 71
2.6.1 Periodic trigger 71
2.6.2 TriggerChangeRate 72
2.6.3 TriggerConstant 73
2.6.4 TriggerEdge 75
2.6.5 TriggerLevel 76
2.6.6 TriggerHarmonicLevel 78
2.7 Text functions 80
2.7.1 CharsToText 80
2.7.2 CharValue 81
2.7.3 CompareText 82
2.7.4 ConcatText 84
2.7.5 ConvertFromText 85
2.7.6 ConvertToText 88
2.7.7 CountText 90
2.7.8 DeleteText 91
2.7.9 FindText 91
2.7.10 InsertText 93
2.7.11 MidText 94
2.7.12 ReplaceText 95
2.7.13 TextLength 96
2.7.14 TrimText 97
2.8 Miscellaneous functions 98
2.8.1 ClientInfo 98
2.8.2 ClientInfoText 99
2.8.3 Count 99

2.8.4 CountUpDown 102
2.8.5 Delay 103
2.8.6 DelayLengthL 105
2.8.7 DelayLengthV 106
2.8.8 DWORD 107
2.8.9 ElapsedTime 108
2.8.10 ExecuteCommand 109
2.8.11 GenerateSignal 111
2.8.12 GenerateText 113
2.8.13 GetFloatBit 114
2.8.14 GetIntBit 116
2.8.15 GetSignalMetaData 117
2.8.16 GetSystemTime 118
2.8.17 GetSystemTimeAsText 120
2.8.18 GetWeekOfYear 122
2.8.19 LastChangeOf 123
2.8.20 LimitAlarm 123
2.8.21 ModuleSignalCount 125
2.8.22 PulseFreq 126
2.8.23 RestartAcquisition 126
2.8.24 SampleAndHold 127
2.8.25 SampleOnce 128
2.8.26 Sign 128
2.8.27 T 129
2.8.28 VarDelay 130
2.8.29 WindowAlarm 132
2.9 Diagnosis functions 134
2.9.1 CameraStatus 134
2.9.2 DataStoreInfo 135
2.9.3 DataStoreInfoDB, ...Influx, ...Kafka, ...MindSphere, ...MQTT 136
2.9.4 DataStoreInfoHD 137
2.9.5 DongleInfo 138

2.9.6 FobDLinkStatus 139
2.9.7 FobFastLinkStatus 139
2.9.8 FobFlexDeviceStatus 140
2.9.9 FobFLinkStatus 141
2.9.10 FobMLinkStatus 141
2.9.11 FobPlusControlLinkStatus 142
2.9.12 FobSDLinkStatus, FobSDexpLinkStatus 142
2.9.13 FobTDCLinkStatus, FobTDCexpLinkStatus 143
2.9.14 ICPSensorStatus 144
2.9.15 InterruptCycleTime 144
2.9.16 InterruptTime 145
2.9.17 IsProjectModified 146
2.9.18 LicenseInfo 147
2.9.19 MultiStationStatus 148
2.9.20 PerformanceCounter 149
2.9.21 Ping 152
2.9.22 TimeSinceLastSync 152
2.9.23 TimeSyncStatus 153
2.10 Filter functions 154
2.10.1 BP 154
2.10.2 HP 154
2.10.3 LP 155
2.10.4 EnvelopeSpectral 156
2.10.5 Preprocess 157
2.11 Retentive functions 158
2.12 Plugins 159
3 Support and contact 160

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 they are capable of assessing safety and recognizing possible consequences and risks on the basis of their specialist training, knowledge and experience and knowledge of the standard regulations.

1.2 Notations

In this manual, the following notations are used:

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

1.3 Used symbols

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

Danger!

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

Observe the specified measures.

Warning!

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

Observe the specified measures.

Caution!

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

Observe the specified measures.

Note

A note specifies special requirements or actions to be observed.

Tip

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

Other documentation

Reference to additional documentation or further reading.

1.4 Documentation structure

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.

Part 1	Introduction and installation	General notes, license policy and add-ons Installation and program start User interface, system architecture, client server User management, printing
Part 2	I/O Manager	Basic information about the I/O Manager, general settings Groups and vector signals, text signals, outputs, configuration files
Part 3	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.
Part 4	Expression builder	All functions for calculating virtual signals
Part 5	Data storage	Types of data recording, recording profiles, signal selection
Part 6	Data visualization	All display modes for live data, their operation and settings
Part 7	Appendix	Various additions, error listings, etc.

2 Expression builder (virtual signals)

With arithmetical and Boolean operations, you can create "virtual signals". These virtual signals can be recorded like regular input signals and/or used for creating complex trigger conditions. Using virtual signals, you can even calculate, e.g., sums, differences or check for limit violations and much more during measurement. Or you create reference signals or characteristics for further comparison in analysis.

Note

The datatype of the analog results of a computation in a virtual module is always Float, no matter what the datatype of the input signals is, which are used in the expression.

Note

In the expression builder and in all expressions and formulas the dot should be used as decimal seperator!

Note

A note about the notation of the function parameters (arguments) in tooltips and help texts:
If a value is assigned to an argument, this is the default value that is assumed if the parameter is omitted. You can only omit parameters which have a default value, e.g. *GenerateSignal('Type',* **'Amplitude=10'**, **'T1=1'**, **'T2=1'**).

In this case you may omit the arguments 2 (Amplitude), 3 (T1) and 4 (T2), if you agree to use the default values. *GenerateSignal (3,10,1,1)* means *GenerateSignal (3)*.

However, a notation like *GenerateSignal (3,,2,5)* is not permitted. If one argument is omitted all following (optional) arguments should be omitted as well. No gaps are allowed. But if following arguments are needed, the argument before must get a value or expression too. In this example the correct notation would be *GenerateSignal (3,10,2,5)*.

Note

If you want to use double quotes in a static text, then you should write two double quotes after each other.

2.1 Encryption of certain text arguments

When entering certain text arguments in expressions you can choose to encrypt the text in order to prevent readability. This mainly applies to arguments like username and password, e.g. as for the ExecuteCommand function.

When you start to type in an argument which is supported by the encryption feature, a pane opens showing the command "Encrypt text". If you want to encrypt the text, then just click on "Encrypt text". Another dialog will open with an entry field for the argument. Enter the argument and click on <OK>.

Instead of the readable text a generated string of characters with the prefix `encrypted_` will be entered as argument in the function.

2.2 Logical functions

2.2.1 Comparison functions >, >=, <, <=, <>, =

The comparative operations > (greater than), >= (greater than / equal to), < (smaller than), <= (smaller than / equal to), <> (unequal) and = (equal) enable comparisons of the values of two expressions (operands). The result of such an operation is the Boolean value TRUE or FALSE. Original signals, calculated expressions or constant values can be entered as operands. The result can be presented and evaluated as a new expression, such as a signal. So, binary signals can easily be generated and can then be used as conditions for other features.

Note.

If the crossing point of two charts is located between two measuring points, the result of the comparative operation of the last two measured values is retained until the next measuring point. I.e. that any change from TRUE to FALSE (or vice versa) is always entered in the grid of the measuring points. The line which connects two measuring points in the presentation of analog values is just a graphic approximation.

2.2.2 Boolean functions

```
e.g. ('Expression1') AND ('Expression2')
```

AND	Logical AND
OR	Logical OR
XOR	Logical exclusive OR
NOT	Logical NOT, negation

Description

The Boolean functions AND (logical AND), OR (logical OR), NOT (logical NOT, negation) and XOR (logical exclusive OR) can be used to connect binary expressions, such as digital signals. According to the rules of Boolean logic, the functions return the value TRUE or FALSE as their result. Digital signals, calculated (binary) expressions or the numerical values 0 or 1 can be entered as parameters.

The result can be presented and evaluated as a new expression, such as a signal. So, binary signals can easily be generated and can then be used as conditions for other features.

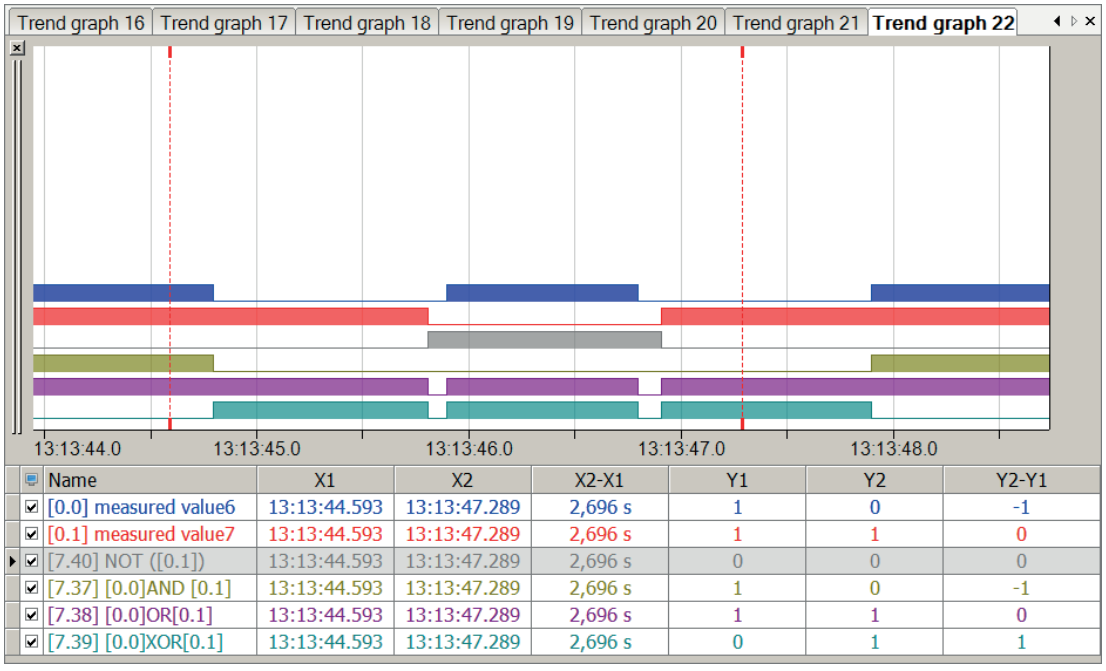
AND				OR				XOR				NOT	
A	B	f (A,B)		A	B	f (A,B)		A	B	f (A,B)		A	f (A)
0	0	0		0	0	0		0	0	0		0	1
1	0	0		1	0	1		1	0	1		1	0
0	1	0		0	1	1		0	1	1			
1	1	1		1	1	1		1	1	0			

Table 1: Logical functions, truth tables

Example

Graphical presentation of Boolean functions

Solution



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0.0] measured value6	13:13:44.593	13:13:47.289	2,696 s	1	0	-1
[0.1] measured value7	13:13:44.593	13:13:47.289	2,696 s	1	1	0
[7.40] NOT ([0.1])	13:13:44.593	13:13:47.289	2,696 s	0	0	0
[7.37] [0.0]AND [0.1]	13:13:44.593	13:13:47.289	2,696 s	1	0	-1
[7.38] [0.0]OR[0.1]	13:13:44.593	13:13:47.289	2,696 s	1	1	0
[7.39] [0.0]XOR[0.1]	13:13:44.593	13:13:47.289	2,696 s	0	1	1

2.2.3 Bitwise logical combinations

```
e.g. 'Expression1' bw_AND 'Expression2'
```

Overview

Operation	Function
bw_AND	Bitwise AND
bw_OR	Bitwise OR
bw_XOR	Bitwise XOR
bw_NOT	Bitwise NOT

Description

The result of this function is the bitwise combination of two analog values. Preferably, this function is to be applied to datatypes like BYTE, WORD, INTEGER etc.

With floating point values (FLOAT), only the integer part will be considered, because the value is converted internally into a 32 bit integer, not rounded.

Examples

By means of a combination of two decimal numbers the following table shows the result of the operations.

Bitwise AND (bw_AND)

```
[Value A] bw_AND [Wert B]
```

Value A = 25	0	0	0	1	1	0	0	1
Value B = 14	0	0	0	0	1	1	1	0
Result bw_AND (decimal = 8)	0	0	0	0	1	0	0	0

Bitwise OR (bw_OR)

```
[Value A] bw_OR [Wert B]
```

Value A = 25	0	0	0	1	1	0	0	1
Value B = 14	0	0	0	0	1	1	1	0
Result bw_OR (decimal = 31)	0	0	0	1	1	1	1	1

Bitwise XOR (bw_XOR)

```
[Value A] bw_XOR [Wert B]
```

Value A = 25	0	0	0	1	1	0	0	1
Value B = 14	0	0	0	0	1	1	1	0
Result bw_XOR (decimal = 23)	0	0	0	1	0	1	1	1

Bitwise NOT (bw_NOT)

Returns the complement of the bits of a value.

```
bw_NOT([Value A])
```

Value A = 25	0	0	0	1	1	0	0	1
Result bw_NOT (decimal = -26)	1	1	1	0	0	1	1	0

2.2.4 ExtendPulse

```
ExtendPulse('Input','Time*')
```

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

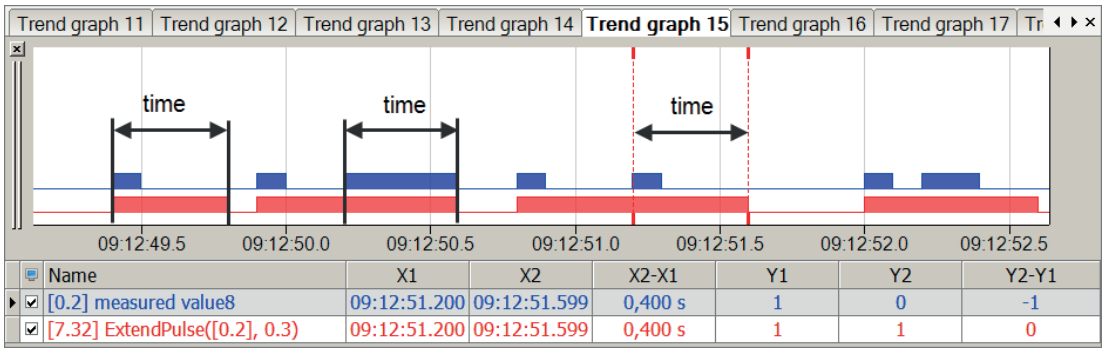
This function extends a pulse on 'Input' to the minimum length of 'Time' seconds. A further rising edge on 'Input' restarts the time.

Example

Extending the pulses of a measured value to a minimum length of 0.3.

Solution

In the figure below the blue bar shows the measured value and the red bar shows the extended pulses of the measured value.



Note

By switching the edges in the sampling grid, the graphic presentation and calculation of the minimum length can be one sample above the minimum length specified under 'Time'.

2.2.5 F_TRIG

```
F_TRIG('Expression')
```

Arguments

'Expression'	Digital input signal or expression

Description

This function returns the value TRUE for 1 sample, if the change from TRUE to FALSE has been detected on 'Expression'.

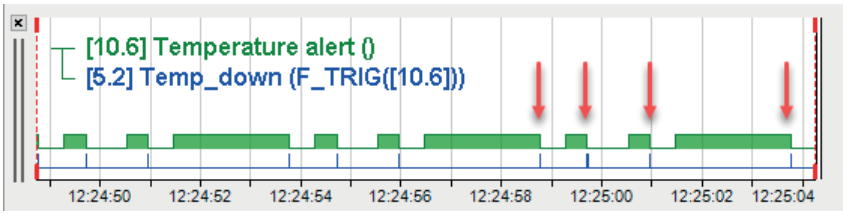
Specifically, falling edges can be detected and indicated by this function.

Example

Each time the signal of a temperature alert turns off, a signal shall be generated.

Solution

Generate a digital signal *Temp_down* by using the function F_TRIG, which only turns TRUE for a short time, if there is a falling edge on the temperature alert signal *Temperature_alert*.



2.2.6 FALSE

```
FALSE()
```

Description

Returns the logical expression, FALSE or zero (0).

2.2.7 If

```
If('Condition','Expression1','Expression2')
```

Arguments

'Condition'	Condition as an operation with the Boolean results, TRUE or FALSE	
'Expression1'	Operation is performed if 'Condition' is TRUE	
'Expression2'	Operation is performed if 'Condition' is FALSE	

Description

The If-function can be used for a conditioned execution of further calculations. Depending on the Boolean result of a 'Condition', which can itself be an operation, the operation 'Expression1' will be executed if the result is TRUE and the operation 'Expression2' if the result is FALSE.

In conclusion, different process-controlled calculations can be performed. You can use this function of course in an interlaced form to realize further branches. Text signals are supported.

Tip

If only one value is entered for 'Condition', as a condition it will be checked whether the value is greater than (TRUE) or less than (FALSE) 0.5.

Note

Instead of "If" function you may use the "Switch" function. Moreover, the "Switch" function offers the advantage to configure more than two cases.

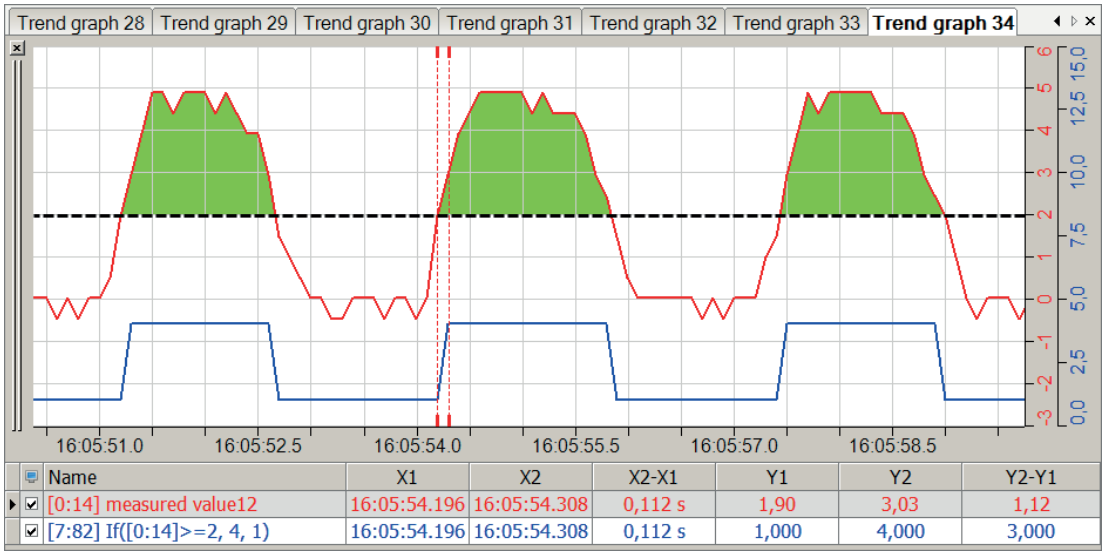
Example

Detecting when a measurement signal is over a set limit

Solution

The limit is formulated in 'Condition' as a condition with logical operands. If 'Condition' is TRUE, the value of 4 is returned, if FALSE then the value of 1.

In the figure below the condition is met above the dotted line (marked in green). Below the dotted line the condition is not met.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:14] measured value12	16:05:54.196	16:05:54.308	0,112 s	1,90	3,03	1,12
[7:82] If([0:14]>=2, 4, 1)	16:05:54.196	16:05:54.308	0,112 s	1,000	4,000	3,000

2.2.8 OneShot

```
OneShot('Expression')
```

Description

This function returns the result TRUE, if the current measured value of 'Expression' is not equal to the previous one. It returns the result FALSE, if the current measured value does equal the previous one. The function supports text signals.

Example 1

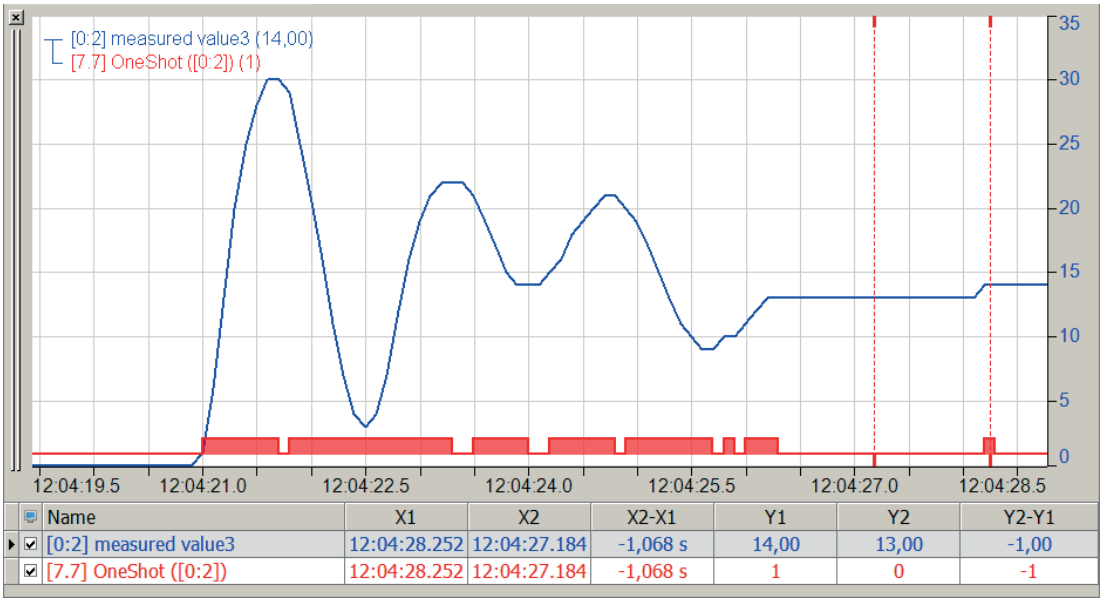
Detecting value changes

Task description

For a real signal waveform, value changes should be displayed.

Solution

In the figure below the blue curve shows the original signal and the red bar shows the area with value changes of the signal.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:2] measured value3	12:04:28.252	12:04:27.184	-1,068 s	14,00	13,00	-1,00
[7.7] OneShot ([0:2])	12:04:28.252	12:04:27.184	-1,068 s	1	0	-1

Example 2

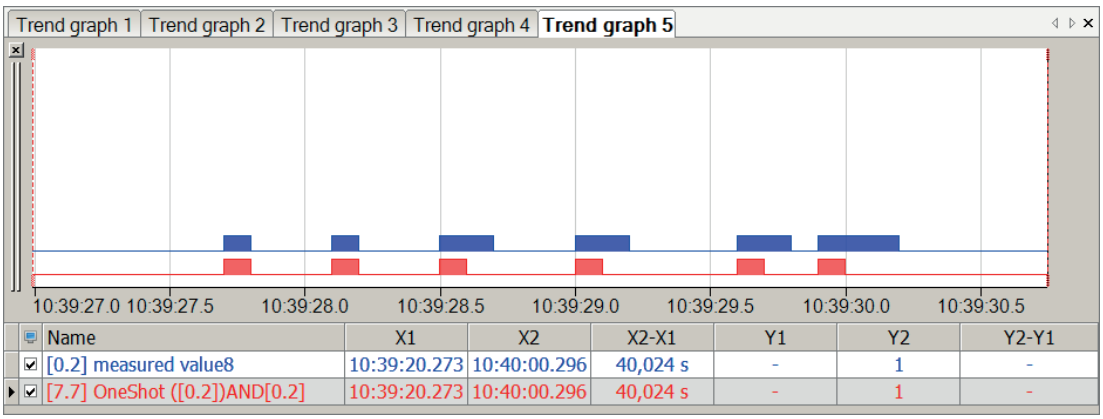
Detecting edges

Task description

The positive edges of a digital signal should be formed.

Solution

In the figure below the blue bar shows the original signal and the red bar shows the positive edges.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0.2] measured value8	10:39:20.273	10:40:00.296	40,024 s	-	1	-
[7.7] OneShot ([0.2])AND[0.2]	10:39:20.273	10:40:00.296	40,024 s	-	1	-

2.2.9 R_TRIG

```
R_TRIG('Expression')
```

Arguments

'Expression'	Digital input signal or expression

Description

This function returns the value TRUE for 1 sample, if the change from FALSE to TRUE has been detected on ‘Expression’.

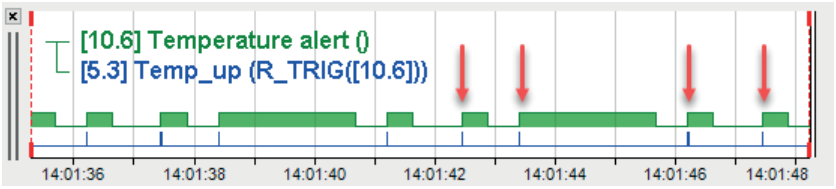
Specifically, rising edges can be detected and indicated by this function.

Example

Each time the signal of a temperature alert turns on, a signal shall be generated.

Solution

Generate a digital signal *Temp_up* by using the function R_TRIG, which only turns TRUE for a short time, if there is a rising edge on the temperature alert signal *Temperature_alert*.



2.2.10 SetReset

```
SetReset('Set','Reset','SetDominant=1*')
```

Arguments

'Set'	Positive edge sets function to TRUE	
'Reset'	Positive edge sets function to FALSE	
'Setdominant*'	Optional parameter (default = 1), which controls which input argument is dominant if both arguments simultaneously receive a positive edge.	
	'Setdominant' = 1	Set takes precedence over Reset
	'Setdominant' = 0	Reset takes precedence over Set

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function can be used to control a digital result (TRUE/FALSE) with the help of positive edges (transition from 0 to 1) of the arguments ‘Set’ and ‘Reset’.

A positive edge of ‘Set’ returns a static TRUE as result. A positive edge of ‘Reset’ resets the result to FALSE. The argument *'SetDominant'* is optional and determines the dominance of ‘Set’ or ‘Reset’.

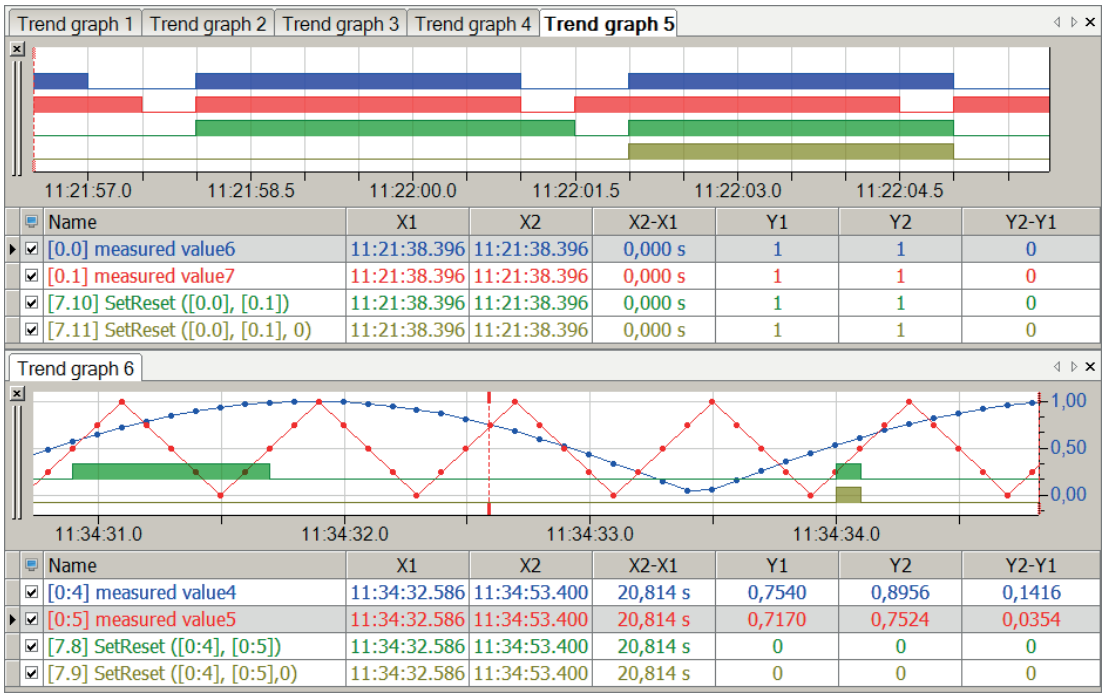
Tip

For an analog signal, exceeding the value 0.5 corresponds to a positive edge.

Example

This function can be used to enable conditional calculations with a signal and to disable them with another signal (e.g. combined with IF function).

The figure below shows the function SetReset with digital and analog signals. For simultaneous edges, the parameter determined under 'SetDominant' is set. For staggered edges, the 'Set-Dominant' parameter has no effect.



2.2.11 Switch

```
Switch('Selector', 'Case1', 'Value1', 'Case2', 'Value2',..., 'Default')
```

Arguments

'Selector'	Expression, which is to be checked against different conditions
'CaseN'	Expression, which is to be compared to ‘Selector’
'ValueN'	Result, if ‘Selector’ and ‘CaseN’ match
'Default'	Result, if none of the ‘CaseN’ matches ‘Selector’

Description

This command compares a ‘Selector’ expression to any number of ‘CaseN’, referring to the SQL statement CASE. At least 3 arguments are required. In case of an even number of arguments the last one will be considered automatically as ‘Default’, which will be taken if no ‘CaseN’ expression matches the ‘Selector’ expression.

If ‘Selector’ matches ‘CaseN’ the respective ‘ValueN’ will be returned as a result. If more than one ‘CaseN’ match ‘Selector’, the first ‘CaseN’ expression will be taken automatically.

The following signals are permitted to be used as ‘Selector’:

- A numerical constant
- A text constant
- An equidistant or non-equidistant sampled channel
- A text channel

Basically, the types of the expressions to be compared must match, otherwise the respective case will not be selected.

2.2.12 TOF

```
TOF('IN','PT*')
```

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

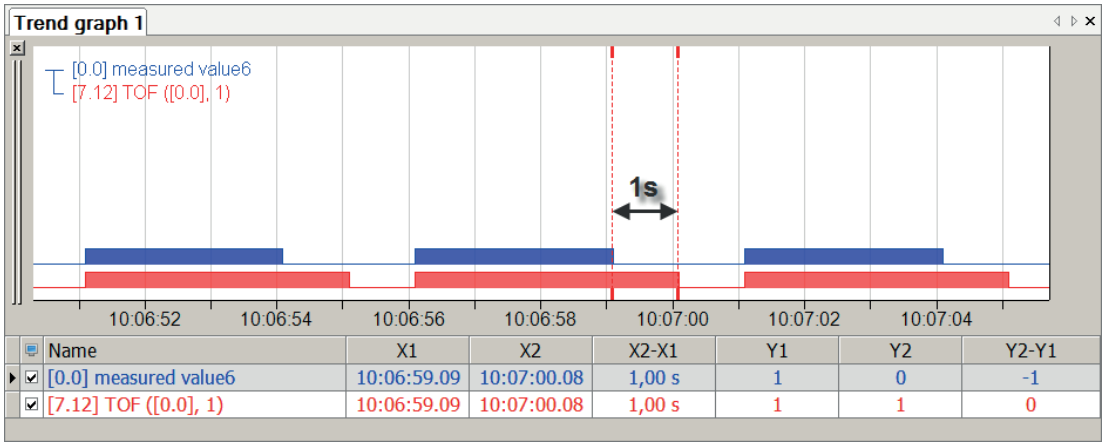
Off Delay Timer. The output is switched off 'PT' seconds after switching off the 'IN' input.

Example

Delaying switching off by one second

Solution

In the figure below the blue bar shows the measured value and the red bar shows the output value with output delayed by one second.



2.2.13 Toggle

```
Toggle('Trigger', 'Initial=0')
```

Arguments

'Trigger'	Trigger signal (rising edge), which toggles the output
'Initial=0'	Value at start of acquisition If this argument is omitted, the start value is 0.

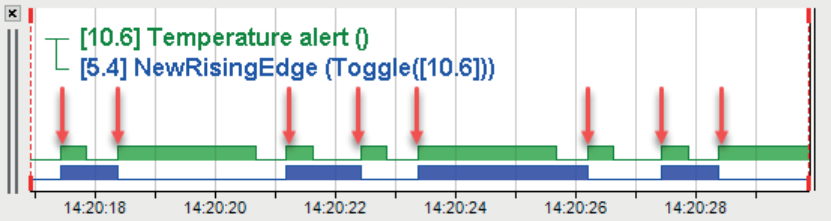
Description

This function toggles the output with each rising edge on ‘Trigger’. The last parameter ‘Initial’ is optional and determines the value at start of acquisition.

Example

The rising edges of a temperature alert signal *Temperature alert* should switch a digital signal *NewRisingEdge* on and off.

Solution



2.2.14 TON

```
TON('IN','PT*')
```

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

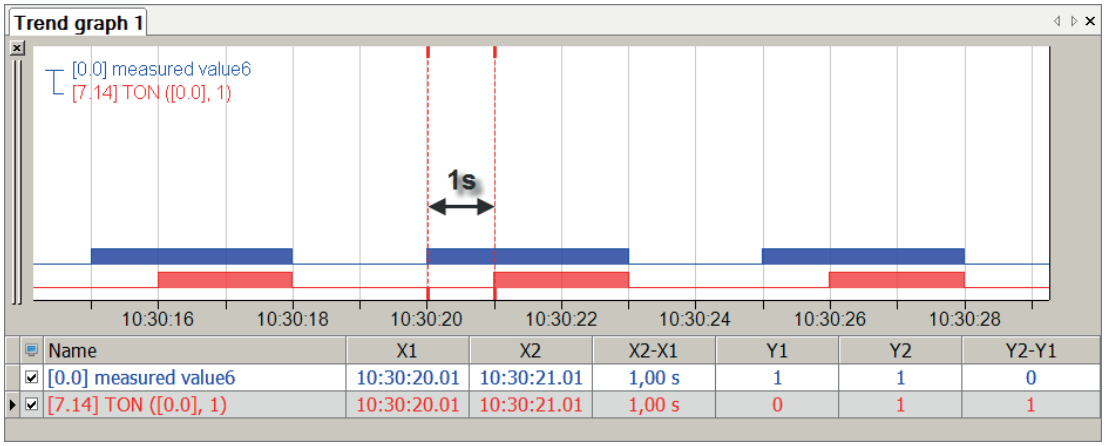
On Delay Timer. The output is switched on 'PT' seconds after switching on the 'IN' input.

Example

Delaying switching on by one second

Solution

In the figure below the blue bar shows the measured value and the red bar shows the input value with input delayed by one second.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0.0] measured value6	10:30:20.01	10:30:21.01	1,00 s	1	1	0
[7.14] TON ([0.0], 1)	10:30:20.01	10:30:21.01	1,00 s	0	1	1

2.2.15 TP

```
TP('in','pt*')
```

Parameters ending with * are only evaluated once at the start of the acquisition..

Description

Pulse Timer. The output is switched on for 'PT' seconds after rising edge at the 'IN' input.

Tip.

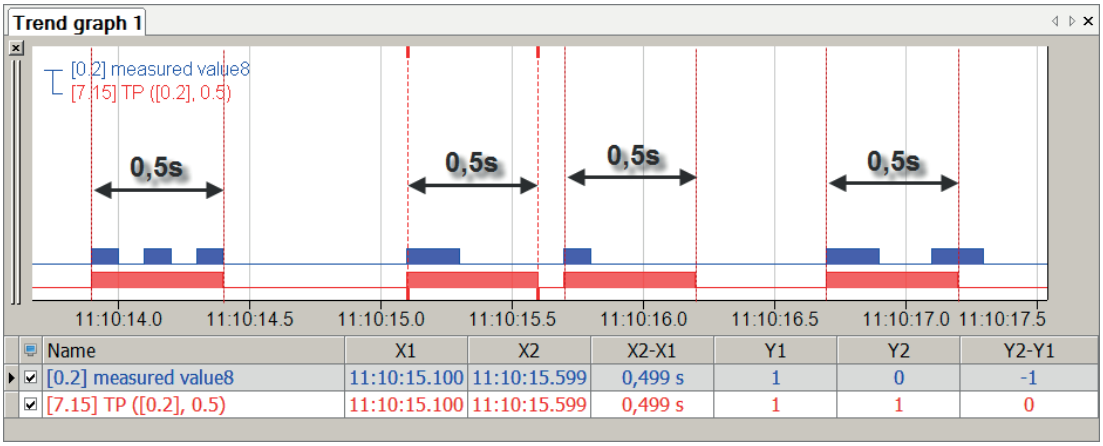
Another rising edge during the output pulse does not extend the output pulse and does not restart the pulse.

Example

Generating a 0.5 second pulse from a random signal

Solution

In the figure below the blue bar shows the measured value and the red bar shows the output value in 0.5 second pulses.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0.2] measured value8	11:10:15.100	11:10:15.599	0,499 s	1	0	-1
[7.15] TP ([0.2], 0.5)	11:10:15.100	11:10:15.599	0,499 s	1	1	0

2.2.16 TRUE

```
TRUE()
```

Description

Returns the logical expression, TRUE or 1.

2.3 Mathematical functions

2.3.1 Fundamental arithmetic operations +, -, *, /

```
('Expression1')+ ('Expression2')
```

Description

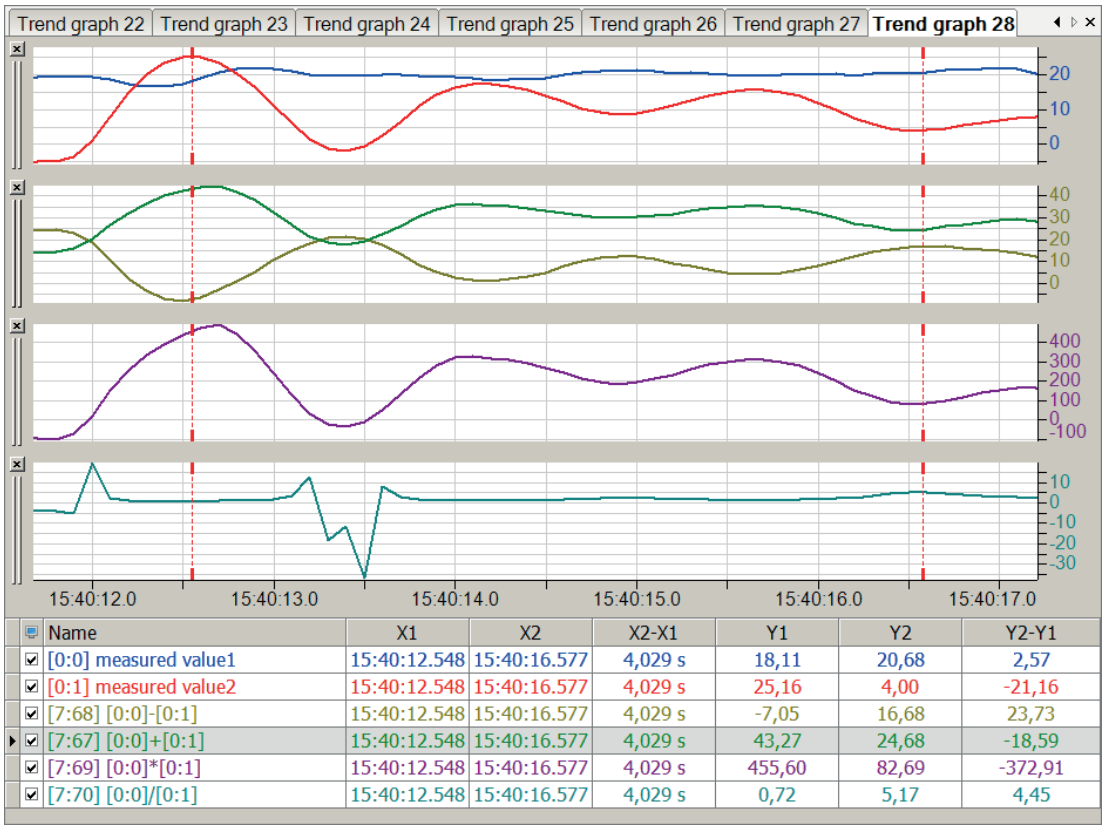
All signals and expressions can be processed by fundamental arithmetic operations (addition, subtraction, multiplication and division). If digital signals or expressions are used as operands in fundamental arithmetic operations, the program translates the TRUE value as 1.0 and FALSE as 0.0. The result of a fundamental arithmetic operation is always an analog expression.

Example

Graphical presentation of fundamental arithmetic operations

Solution

The figure below shows the resulting charts for the different fundamental arithmetic operations.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:0] measured value1	15:40:12.548	15:40:16.577	4,029 s	18,11	20,68	2,57
[0:1] measured value2	15:40:12.548	15:40:16.577	4,029 s	25,16	4,00	-21,16
[7:68] [0:0]-[0:1]	15:40:12.548	15:40:16.577	4,029 s	-7,05	16,68	23,73
[7:67] [0:0]+[0:1]	15:40:12.548	15:40:16.577	4,029 s	43,27	24,68	-18,59
[7:69] [0:0]*[0:1]	15:40:12.548	15:40:16.577	4,029 s	455,60	82,69	-372,91
[7:70] [0:0]/[0:1]	15:40:12.548	15:40:16.577	4,029 s	0,72	5,17	4,45

Blue	Measured value 1	Red	Measured value 2
Green	Addition	Yellow	Subtraction
Purple	Multiplication	Light blue	Division

2.3.2 Abs

```
Abs('Expression')
```

Description

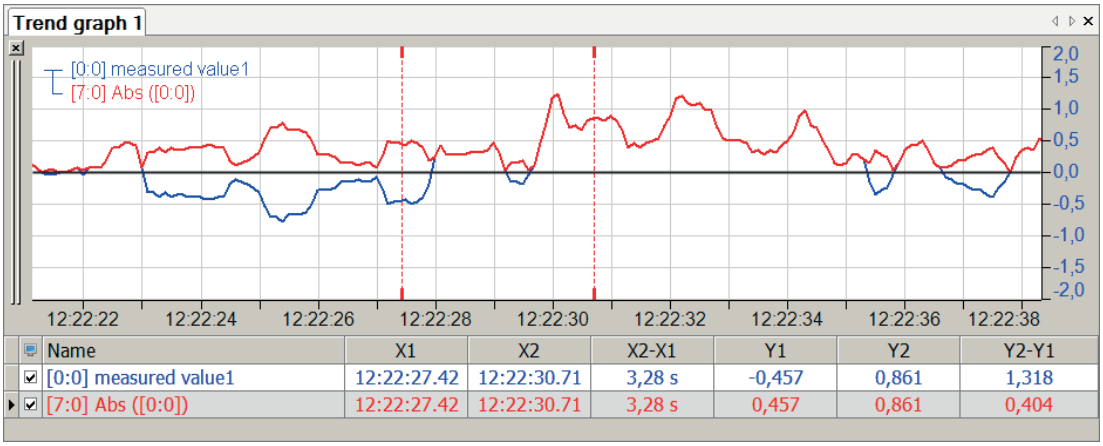
The absolute function returns the absolute value (= |value|) of 'Expression.'

Example

Only the absolute value of the value from a measured signal is relevant

Solution

In the figure below the blue curve shows the original signal and the red curve is the absolute value of the signal.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:0] measured value1	12:22:27.42	12:22:30.71	3,28 s	-0,457	0,861	1,318
[7:0] Abs ([0:0])	12:22:27.42	12:22:30.71	3,28 s	0,457	0,861	0,404

Tip

Interpolated values in the case of a change of sign between two samples may differ in value.

2.3.3 Add

```
Add('Expression', 'Enable', 'Reset=0')
```

Arguments

'Expression'	Analog input signal or expression	
'Enable'	Digital input signal or expression; *'Enable'* = 1 enables the addition	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation:	
	' *Reset*' = 0	Perform calculation
	' *Reset'* = 1	Stop calculation and set result to 0

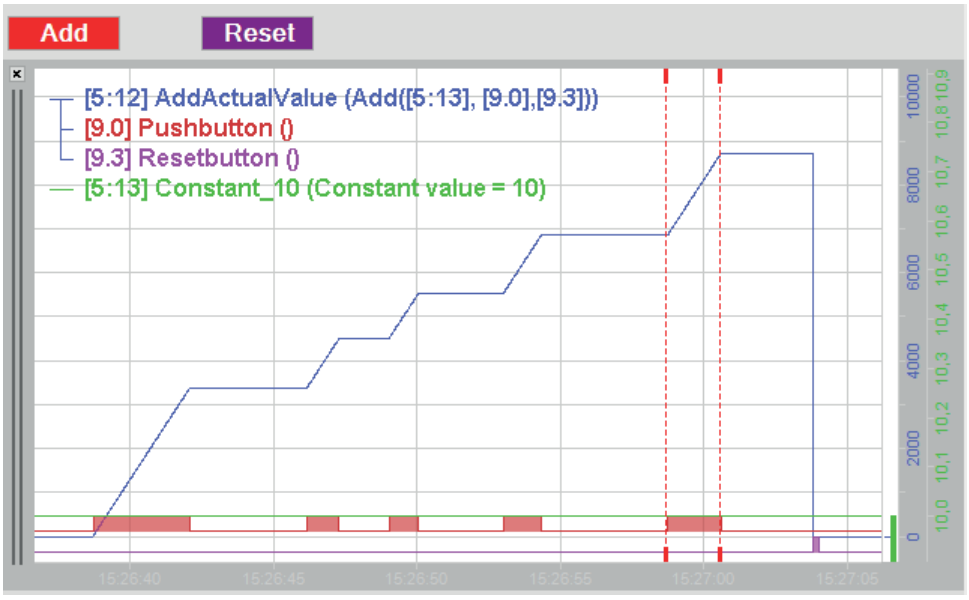
Description

This function adds the value of 'Expression' and its own actual value every cycle if 'Enable' = TRUE or 1. If 'Enable' = FALSE or 0, the last actual value will be held. 'Reset' = TRUE or 1 resets the actual value to 0.

Example

Each time a pushbutton is pressed, an analog value should be increased by 10 every cycle. With a reset-pushbutton the value should be reset to 0.

Solution



2.3.4 Ceiling

```
Ceiling('Expression')
```

Description

This function returns the smallest integer value that is greater than or equal to 'Expression'.

Example

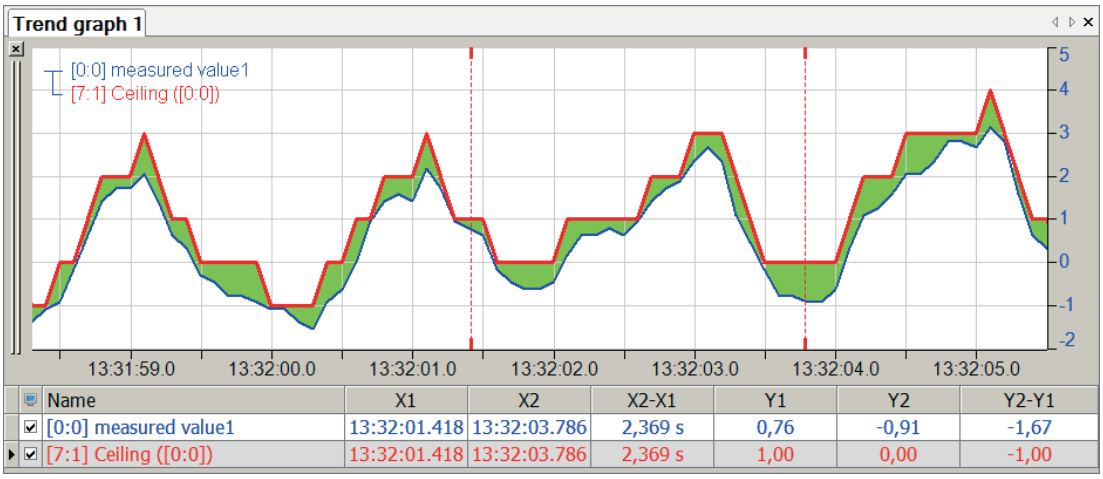
Rounding up values of a signal waveform to whole numbers.

Task description

From a real signal waveform, the measured value should be rounded up to the nearest integer value for each sample.

Solution

In the figure below the blue curve shows the original signal and the red curve shows the signal trend with rounded-up values.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:0] measured value1	13:32:01.418	13:32:03.786	2,369 s	0,76	-0,91	-1,67
[7:1] Ceiling ([0:0])	13:32:01.418	13:32:03.786	2,369 s	1,00	0,00	-1,00

2.3.5 Diff

```
Diff('Expression')
```

Description

This function returns the differential dx/dt of 'Expression'.

Example

If 'Expression' is a length measuring signal, this diff-function can be used to determine a speed curve.

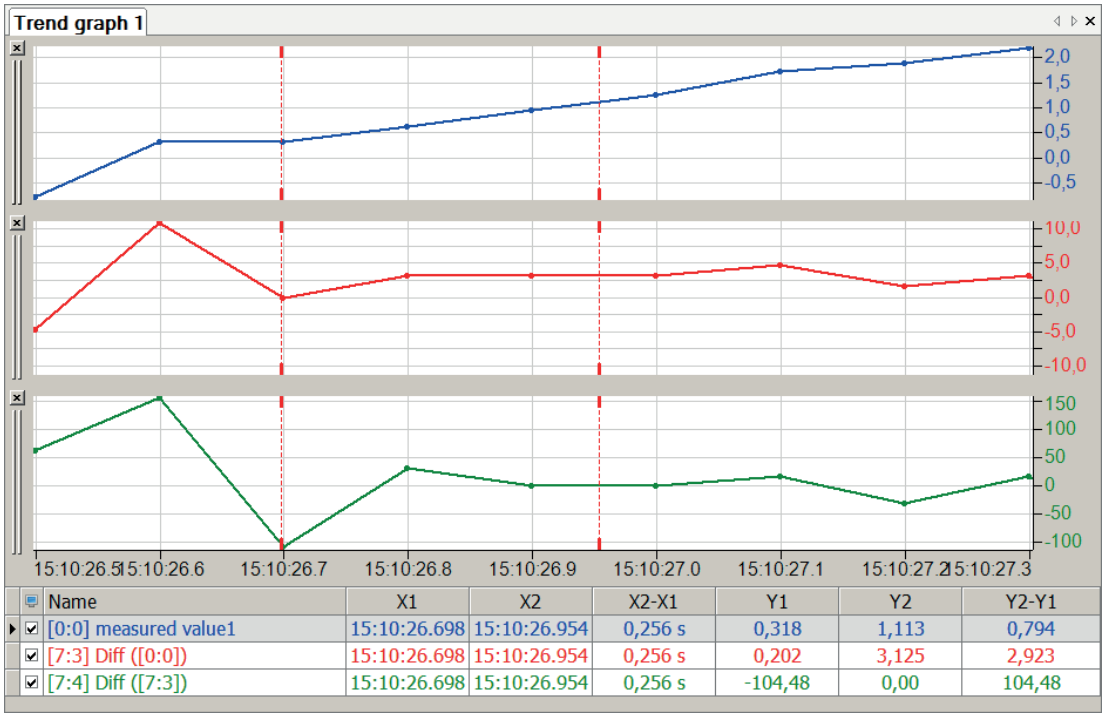
Task description

The speed curve should be determined from a length measuring signal and, from this, the acceleration curve.

Solution

Through repeated (iterative) performance of the diff-function using the values of the first as 'Expression' we obtain the curve of the acceleration.

In the figure below the blue curve shows the covered distance ("length"), the red curve shows the calculated course of speed and the green curve shows the calculated course of acceleration.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:0] measured value1	15:10:26.698	15:10:26.954	0,256 s	0,318	1,113	0,794
[7:3] Diff ([0:0])	15:10:26.698	15:10:26.954	0,256 s	0,202	3,125	2,923
[7:4] Diff ([7:3])	15:10:26.698	15:10:26.954	0,256 s	-104,48	0,00	104,48

Tip

If only the course of acceleration is of interest and speed is not relevant, this can also be determined directly by recursively using the

Diff(Diff([0:0])) function.

2.3.6 Eff

```
Eff('Expression','Frequency')
```

Description

This function calculates the effective value of ‘Expression’ with a fundamental frequency of ‘Frequency’.

The following formula is used to calculate the effective value:

$$E_{eff} = \sqrt{\frac{1}{N}\sum_{n=1}^{N} e^2(n)}$$

e(n): Value of sample n of signal e ('Expression')
N: Number of samples per period

Note

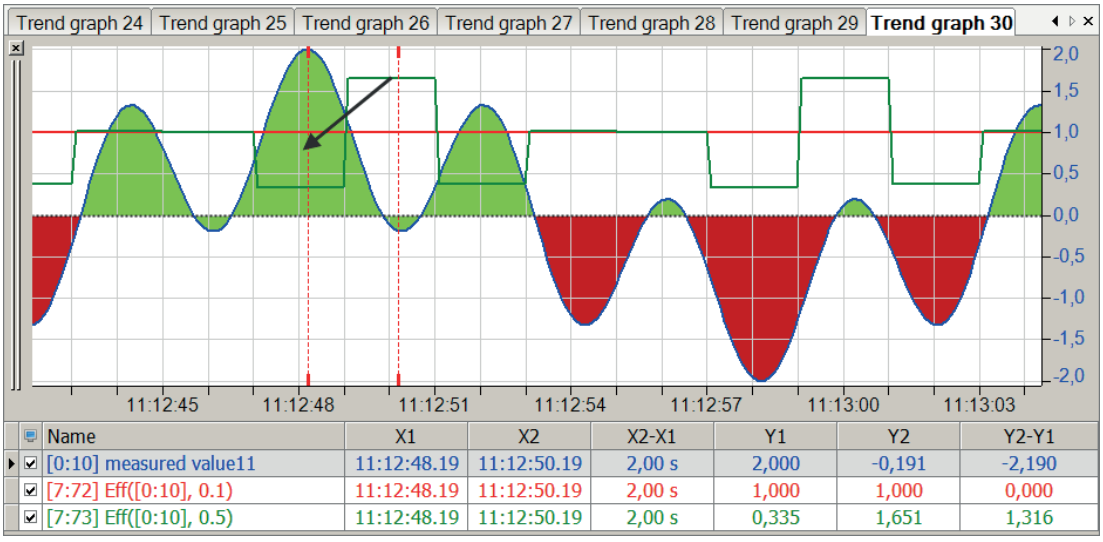
The result of the effective value function is always displayed in the subsequent cycle of the fundamental frequency.

Example

For an alternating current course with a frequency of 0.1 kHz, which is overlaid by a second AC voltage with 0.5 kHz, the effective value of the voltage should be determined for both frequencies.

Solution

In the figure below the blue curve shows the AC current signal and the green curve shows the effective voltage with calculation frequency 0.5 kHz. The red straight line shows the effective voltage with calculation frequency 0.1 kHz.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:10] measured value11	11:12:48.19	11:12:50.19	2,00 s	2,000	-0,191	-2,190
[7:72] Eff([0:10], 0.1)	11:12:48.19	11:12:50.19	2,00 s	1,000	1,000	0,000
[7:73] Eff([0:10], 0.5)	11:12:48.19	11:12:50.19	2,00 s	0,335	1,651	1,316

2.3.7 Exp

```
Exp('Expression')
```

Description

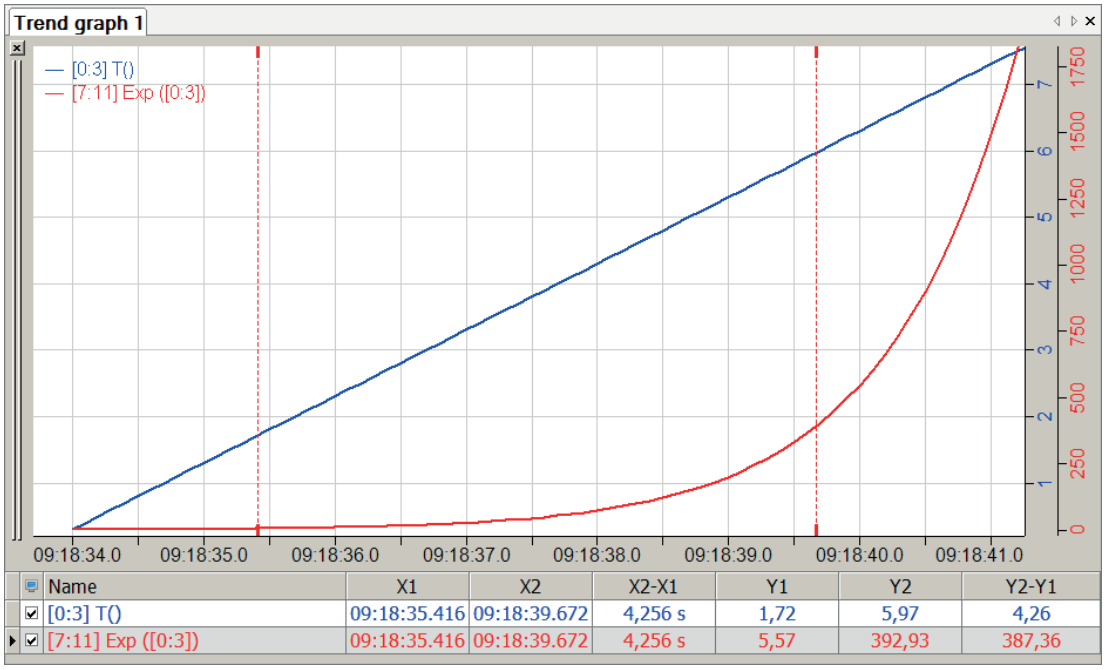
This function calculates the result of (e) 'Expression'

Example

Graphical presentation of the exp-function

Solution

In the figure below the blue curve shows the time function and the red curve shows the exponential function of time.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:3] T()	09:18:35.416	09:18:39.672	4,256 s	1,72	5,97	4,26
[7:11] Exp ([0:3])	09:18:35.416	09:18:39.672	4,256 s	5,57	392,93	387,36

2.3.8 Floor

```
Floor('Expression')
```

Description

This function returns the largest integer value that is less than or equal to 'Expression'.

Example

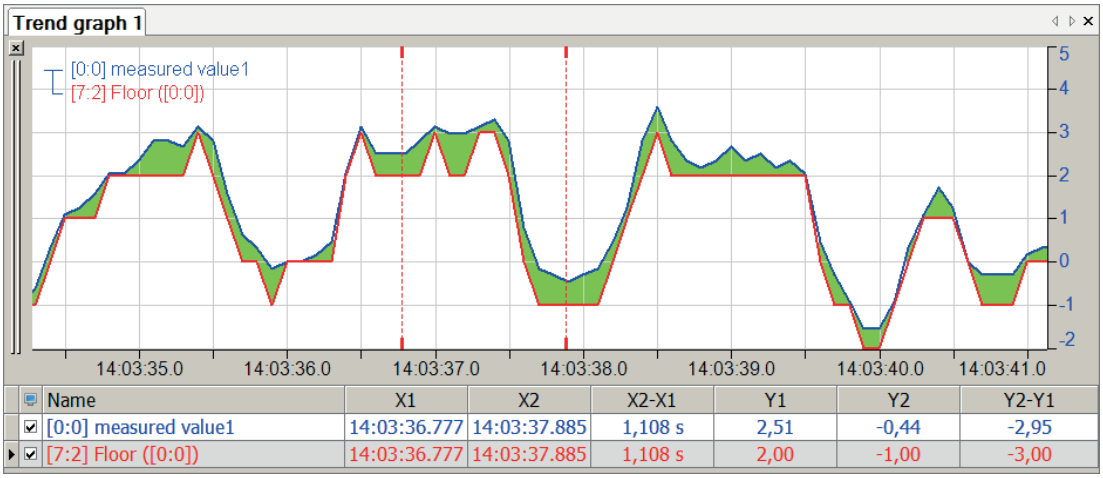
Rounding down values of a signal waveform to whole numbers.

Task description

From a real signal waveform, the measured value should be rounded down to the nearest integer value for each sample.

Solution

In the figure below the blue curve shows the original signal and the red curve shows the signal trend with rounded-down values.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:0] measured value1	14:03:36.777	14:03:37.885	1,108 s	2,51	-0,44	-2,95
[7:2] Floor ([0:0])	14:03:36.777	14:03:37.885	1,108 s	2,00	-1,00	-3,00

2.3.9 Int

```
Int('Expression','Reset')
```

Arguments

'Expression'	Measured value	
'Reset'	Optional digital parameter, which can be used to reset the integral or suppress the integration process. 'Reset' can be an expression as well.	
	'Reset' > 0	Integral is reset.
	'Reset' = 0	Integration released (default)

Description

This function returns the integral (y * dt) of 'Expression' as its result. The 'Reset' parameter can be used for resetting the integral to zero or suppressing the integration process, e.g. to integrate the same signal for periodical occurrences or reversing processes a number of times. 'Reset' can be an expression as well.

Examples:

Int([0:0])	No reset happens ('Reset' omitted)
Int([0:0],If(Mod(T(),20)=0,TRUE(),FALSE()))	The integral is reset every 20 seconds.
Int([0:0], [3.1])	e.g. with [3.1] = If([0:0]>10, 1, 0) The integral is reset as soon as the expression [3.1] returns TRUE, i.e. if the expression [0:0] exceeds the limit value 10.

Example 1

Tip

This function can be used in a virtual retentive module. Its result values can thus be obtained via stopping and restarting the measurement.

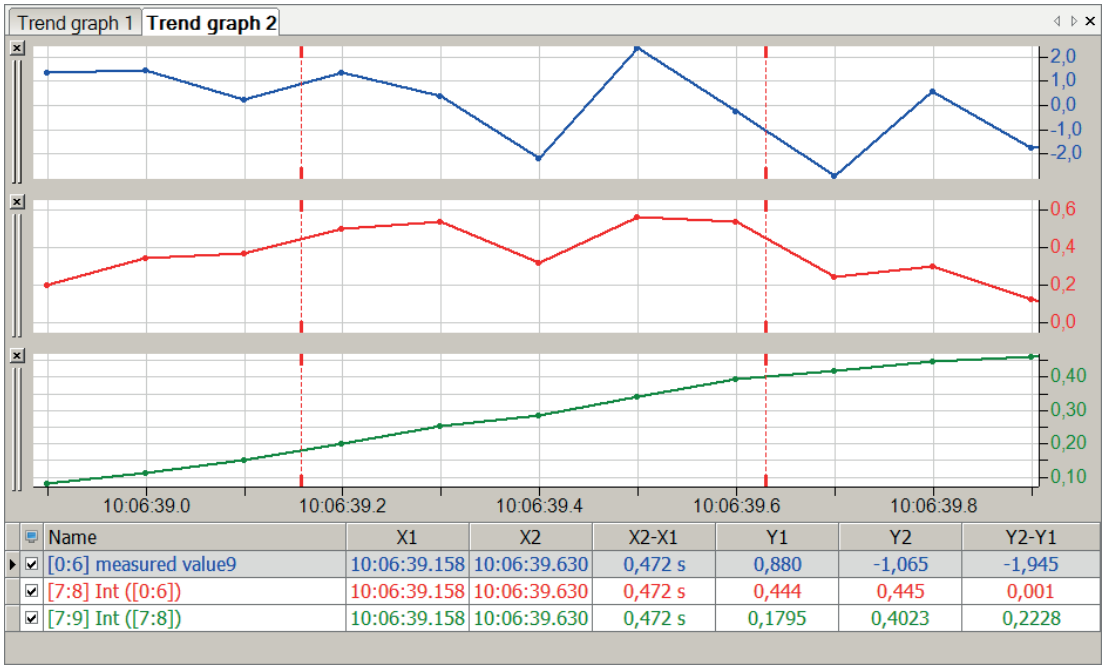
If 'Expression' is an acceleration signal, the distance covered can be determined by iteratively performing the Int-function.

Task description

The speed and distance covered should be determined with the help of an acceleration sensor.

Solution

In the figure below the blue curve shows the measured acceleration, the red curve shows the calculated course of speed and the green curve shows the calculated distance covered.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:6] measured value9	10:06:39.158	10:06:39.630	0,472 s	0,880	-1,065	-1,945
[7:8] Int ([0:6])	10:06:39.158	10:06:39.630	0,472 s	0,444	0,445	0,001
[7:9] Int ([7:8])	10:06:39.158	10:06:39.630	0,472 s	0,1795	0,4023	0,2228

Tip

If only the distance covered is of interest, this can also be determined by recursively using the `Int(Int([0:6]))` function.

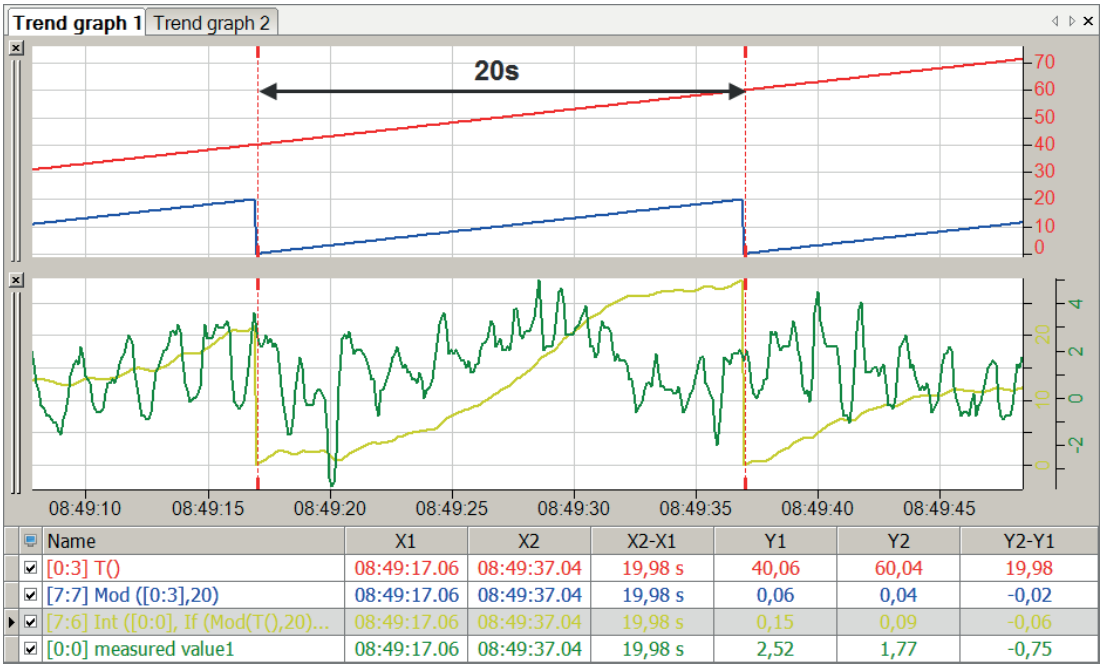
Example 2

The integral should be restarted at an interval of 20 seconds.

Task description

The modulo function can be used to reset the integral since every 20 seconds T() mod 20 = 0 applies.

Solution



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:3] T()	08:49:17.06	08:49:37.04	19,98 s	40,06	60,04	19,98
[7:7] Mod ([0:3],20)	08:49:17.06	08:49:37.04	19,98 s	0,06	0,04	-0,02
[7:6] Int ([0:0], If (Mod(T(),20)...	08:49:17.06	08:49:37.04	19,98 s	0,15	0,09	-0,06
[0:0] measured value1	08:49:17.06	08:49:37.04	19,98 s	2,52	1,77	-0,75

Red	Time function	Blue	Modulo20 of the time function
Green	Measured value	Yellow	Integral of the measured value with 20 seconds reset

2.3.10 Log

```
Log('Expression')
```

Description

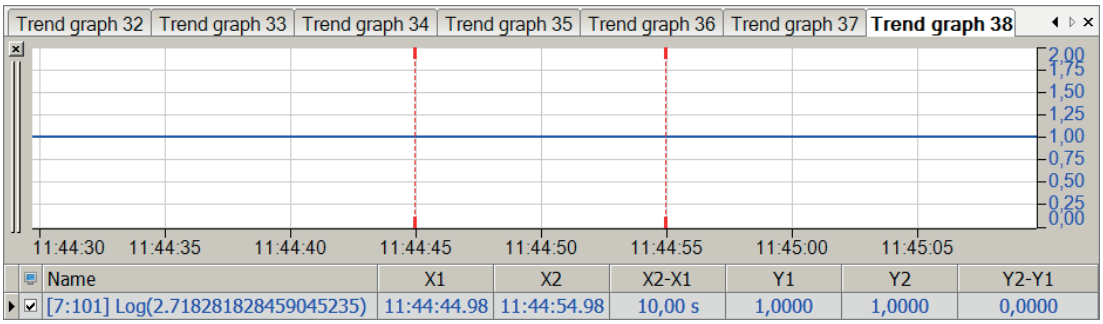
This function returns the natural logarithm (ln x) of 'Expression' as its result.

Example

Calculating a known natural logarithm.

Solution

Logarithm of Euler's number e must constantly give 1: ln e = 1



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[7:101] Log(2.718281828459045235)	11:44:44.98	11:44:54.98	10,00 s	1,0000	1,0000	0,0000

Tip

Although negative values for 'Expression' do not produce an error message, they do not produce a result either.

2.3.11 Log10

```
Log10('Expression')
```

Description

This function returns the decadic logarithm (lg x) of 'Expression' as its result.

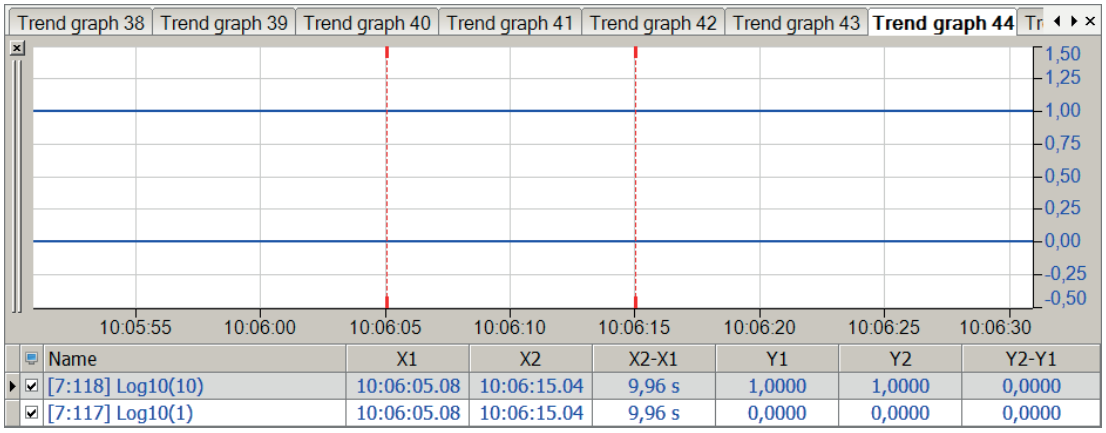
Example

Calculating known decadic logarithms.

Solution

lg 1 = Log10(1) = 0

lg 10 = Log10(10) = 1



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[7:118] Log10(10)	10:06:05.08	10:06:15.04	9,96 s	1,0000	1,0000	0,0000
[7:117] Log10(1)	10:06:05.08	10:06:15.04	9,96 s	0,0000	0,0000	0,0000

Tip

Although negative values for 'Expression' do not produce an error message, they do not produce a result either.

2.3.12 Mod

```
Mod('Expression1','Expression2')
```

Description

This function returns the modulo of 'Expression1' and 'Expression2' as its result. Internally, the function uses the fmod C-function, which permits the use of floating-point values for 'Expression1' and 'Expression2'.

Modulo r is the remainder of the division Expression1 / Expression2 so that the following relationship applies in the opposite direction:

Expression1 = Expression2 * i + r, where i is an integer number (integer).

Modulo r always has the same sign as 'Expression1' and the absolute value of r is always smaller than the absolute value of 'Expression2'.

If Expression1' < 'Expression2', the function returns the value of 'Expression1' as its result. Mathematically speaking, the modulo can also be described as "Expression1 modulo Expression2".

Examples: Mod(7, 3) = 1; seven divided by three equals two, remainder 1.

Mod(20.0, 10.5) = 9.5

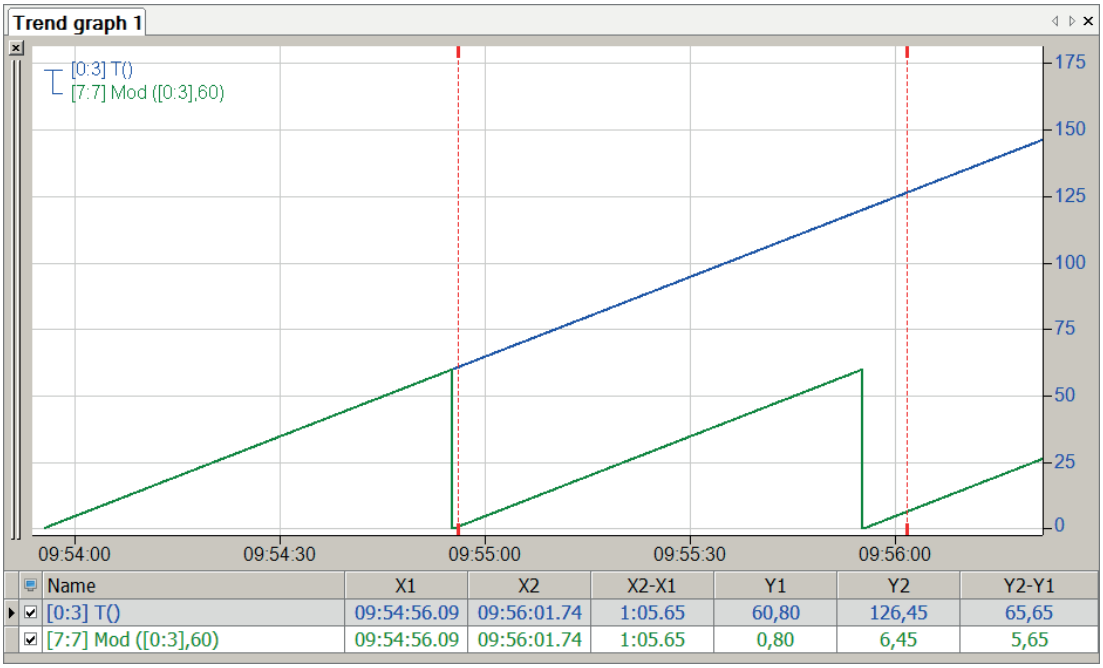
Example

Only the seconds from the time function are relevant; hours and minutes should be cut off

Solution

With the help of Modulo60 of the time function, only the seconds are kept as the remainder.

In the figure below the blue curve shows the time function and the green curve shows Modulo60 of the time function.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:3] T()	09:54:56.09	09:56:01.74	1:05.65	60,80	126,45	65,65
[7:7] Mod ([0:3],60)	09:54:56.09	09:56:01.74	1:05.65	0,80	6,45	5,65

2.3.13 Pow

```
Pow('Expression1','Expression2')
```

Arguments

'Expression1'	Basis
'Expression2'	Exponent

Description

This function increases 'Expression1' (basis) to the power of 'Expression2' (exponent): ('Expression1')'Expression2'

Example

Calculating some important powers

Solution

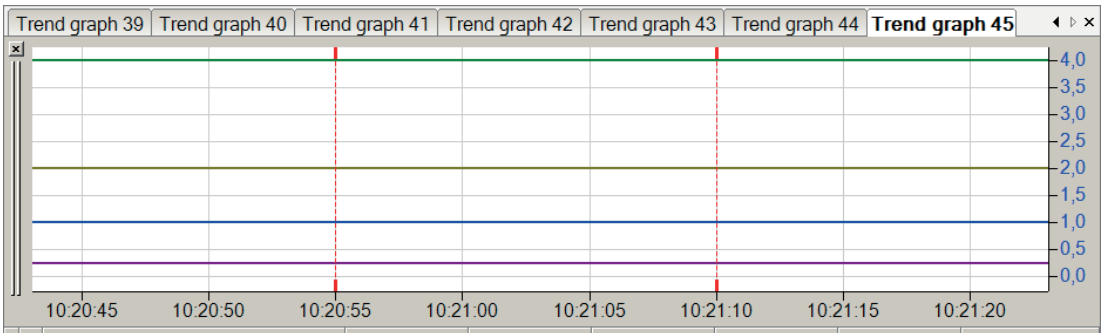
(2)0 = Pow(2, 0) = 1

(2)-2 = Pow(2, -2) = 0.25

(-2)2 = Pow(-2, 2) = 4

(10)lg 2 = Pow(10, lg 2) = 2

(0)-1 = Pow(0, -1) = +∞ (infinity)



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[7:119] Pow(2, 0)	10:20:55.00	10:21:10.03	15,03 s	1,000	1,000	0,000
[7:120] Pow(2, -2)	10:20:55.00	10:21:10.03	15,03 s	0,250	0,250	0,000
[7:121] Pow(-2, 2)	10:20:55.00	10:21:10.03	15,03 s	4,000	4,000	0,000
[7:122] Pow(10, Log10(2))	10:20:55.00	10:21:10.03	15,03 s	2,000	2,000	0,000
[7:123] Pow(0, -1)	10:20:55.00	10:21:10.03	15,03 s	+unendlich	+unendlich	-

Tip

The exponentiation of 0 with -1, which is equivalent to a division by 0, does not return an error message but the limit value +infinity.

2.3.14 Round

```
Round('Expression')
```

Description

This function rounds ‘Expression’ up or down to the nearest whole number (integer).

Example

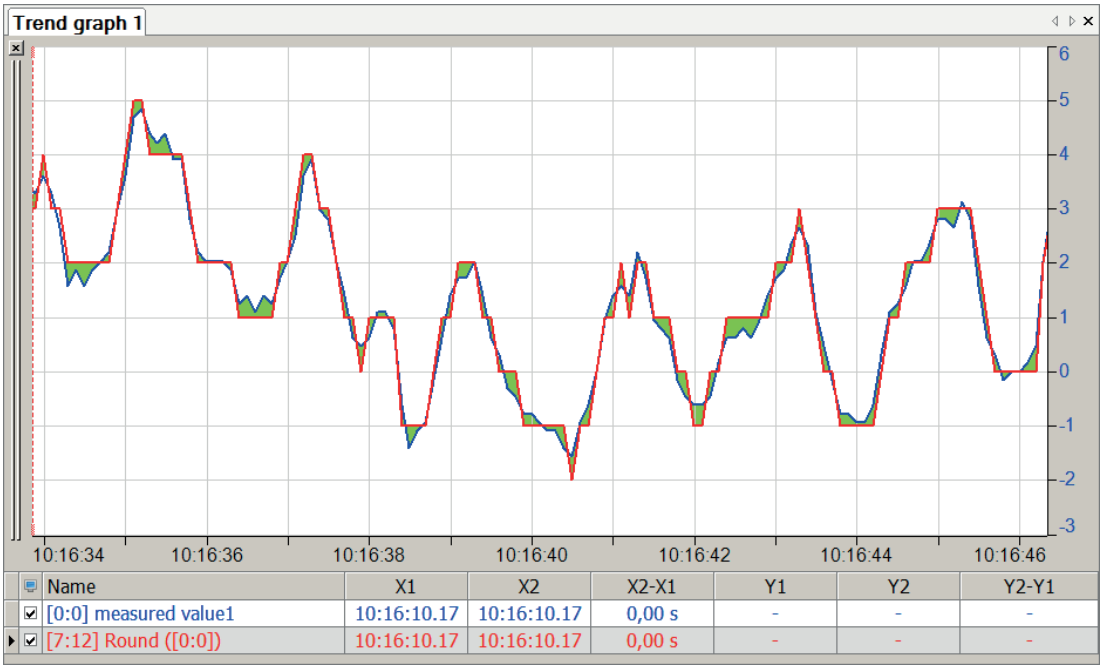
Rounding values of a signal waveform to whole numbers

Task description

From a real signal waveform, the measured value should be rounded to the nearest integer value for each sample.

Solution

In the figure below the blue curve shows the original signal and the red curve shows the signal trend with rounded values.



2.3.15 Sqrt

```
Sqrt('Expression')
```

Description

This function returns the square root of 'Expression' as its result.

Example

Calculating some known square roots

Solution

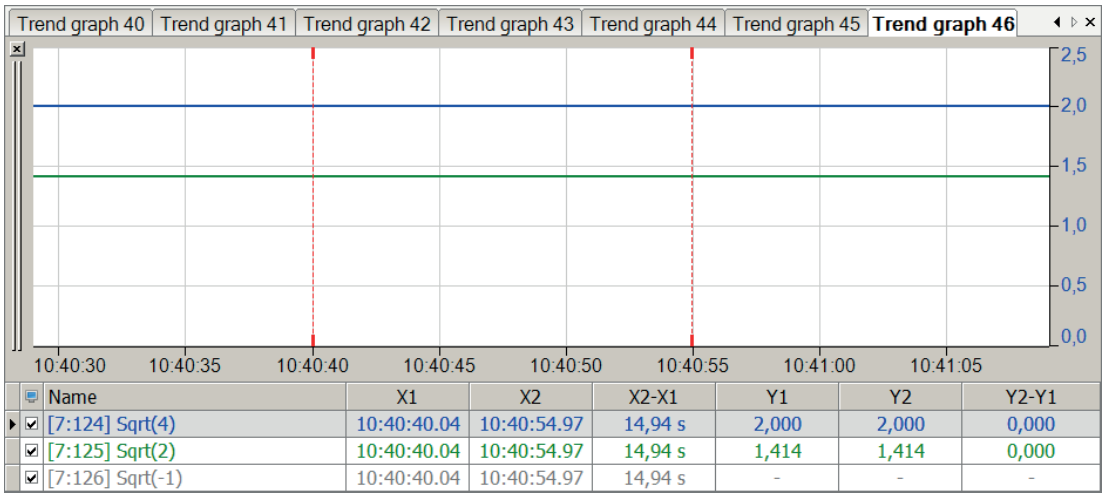
√(4) = Sqrt(4) = 2

√(2) = Sqrt(2) = 1.41421356...

√(-1) = Sqrt(-1) = i (complex calculation required)

Tip

Although negative values for 'Expression' do not produce an error message, they do not produce a result either.



2.3.16 Truncate

```
Truncate('Expression')
```

Description

The truncate function truncates the decimal places of a floating point value. Negative numbers are thus rounded up to the nearest integer value, positive numbers rounded down.

Example

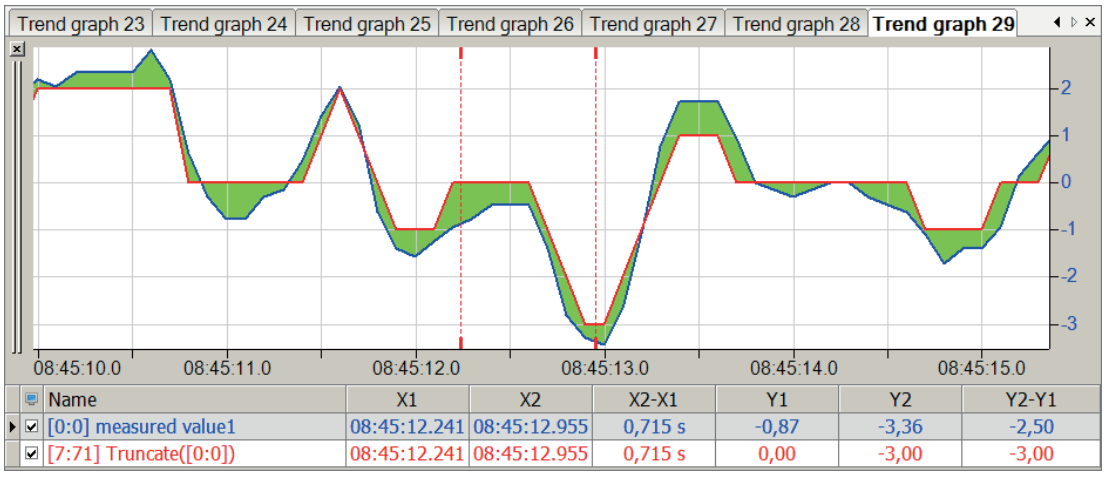
Truncating the decimal places of the values of a signal waveform

Task description

From a real signal waveform, the measured value should be displayed without decimal places for each sample.

Solution

In the figure below the blue curve shows the original signal and the red curve shows the signal trend with truncated decimal places.



2.4 Trigonometric functions

```
Function('Expression')
```

Description

The standard functions and the corresponding inverse functions are available for the most varied kinds of calculations in which trigonometric functions are needed, for example, the calculation of power in AC-systems.

Function	Description
Sin('Expression')	This function returns the result as the sine of 'Expression' in rad.
Cos('Expression')	This function returns the result as the cosine of 'Expression' in rad.
Tan('Expression')	This function returns the result as the tangent of 'Expression' in rad.
Asin('Expression')	This function returns the result as the arcsine of 'Expression' in rad.
Acos('Expression')	This function returns the result as the arccosine of 'Expression' in rad.
Atan('Expression')	This function returns the result as the arctangent of 'Expression' in rad.

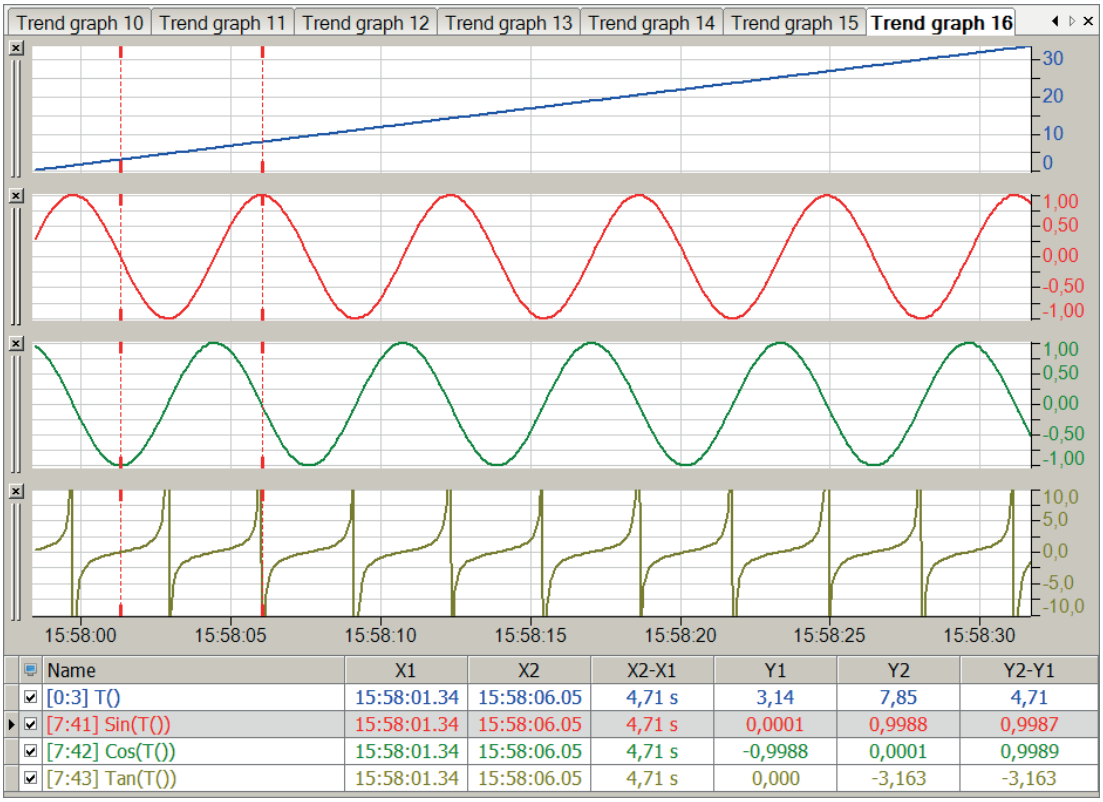
Tip

In order to generate signales, the GenerateSignal function can be used.

Example

Presentation of trigonometric functions

Solution



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:3] T()	15:58:01.34	15:58:06.05	4,71 s	3,14	7,85	4,71
[7:41] Sin(T())	15:58:01.34	15:58:06.05	4,71 s	0,0001	0,9988	0,9987
[7:42] Cos(T())	15:58:01.34	15:58:06.05	4,71 s	-0,9988	0,0001	0,9989
[7:43] Tan(T())	15:58:01.34	15:58:06.05	4,71 s	0,000	-3,163	-3,163

Blue	Time function T() as the basis for trigonometric functions	Red	Sine function of time
Green	Cosine function of time	Yellow	tangent function of time

```
Pi()
```

Description

The number Pi () is stored as a constant (p = 3.1415927…) in the system for various kinds of calculations. Use this function to insert the number pi into your calculation.

2.5 Statistical functions

2.5.1 Avg

```
Avg('Expression', 'Reset=0')
```

Arguments

'Expression'	Measured value for the average is formed	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation:	
	'Reset' = 0	Perform calculation
	'Reset' = 1	Stop calculation and set result to the instantaneous "Expression" value

Description

As a result, this function returns the average value of "Expression" since the measuring started or the last reset. The signal is written to the RAM of the computer. The arithmetic mean is continuously calculated. If 'Reset' = 1 (TRUE), then the result equals the actual value of "Expression."

Note

The result of the Avg-function is only displayed in the subsequent interval.

2.5.2 Avg2

```
Avg2('Expression1', 'Expression2', …)
```

Arguments

'Expression'	Measured value, for which the average is formed

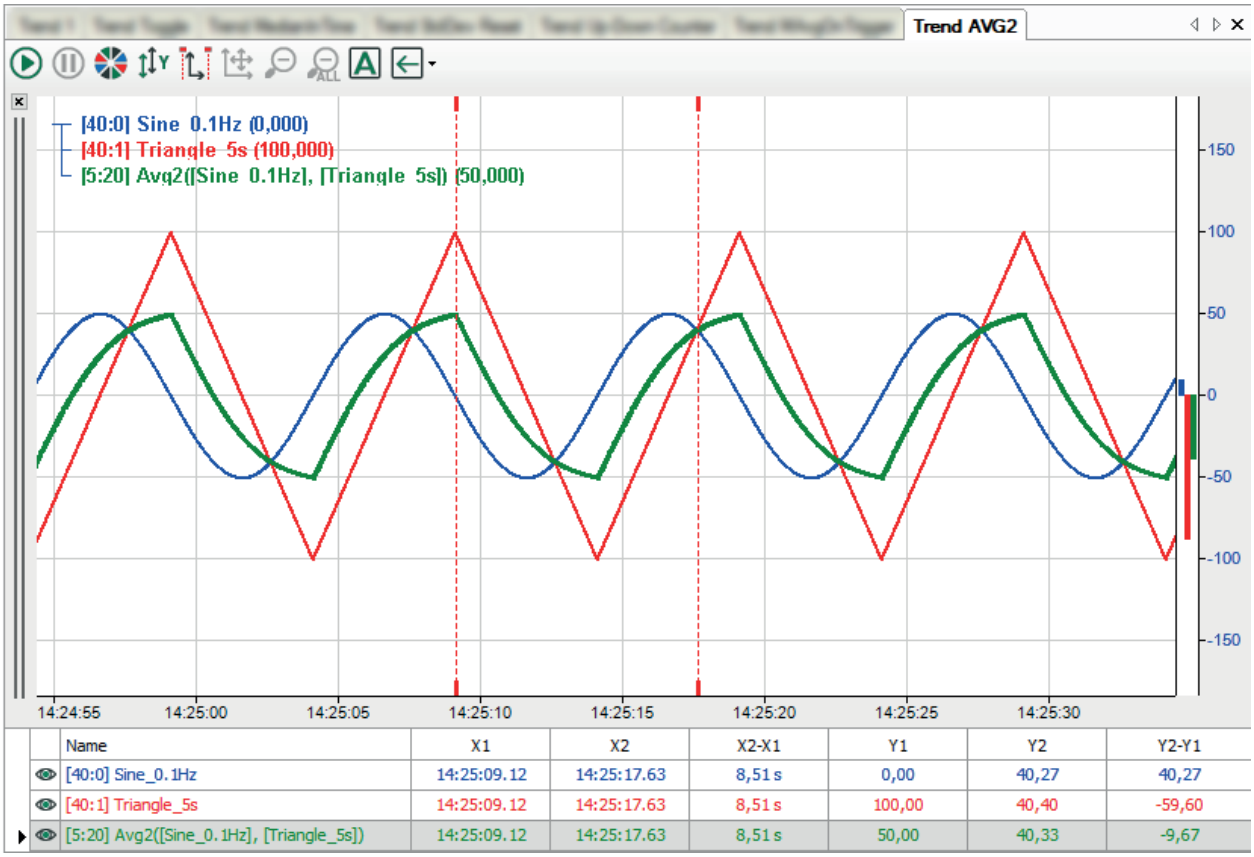
Description

This function returns the average value of all arguments 'Expression1', 'Expression2', 'Expression3' etc. The arithmetic average will be formed from the actual values of all arguments in every acquisition cycle. Up to 1000 arguments are allowed.

Example

Evaluate the avarage value from two signals, a sine- and a sawtooth-shaped signal.

Solution



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[40:0] Sine_0.1Hz	14:25:09.12	14:25:17.63	8,51 s	0,00	40,27	40,27
[40:1] Triangle_5s	14:25:09.12	14:25:17.63	8,51 s	100,00	40,40	-59,60
[5:20] Avg2([Sine_0.1Hz], [Triangle_5s])	14:25:09.12	14:25:17.63	8,51 s	50,00	40,33	-9,67

2.5.3 AvgInTime

```
AvgInTime('Expression','Interval',' Reset=0')
```

Arguments

'Expression'	Measurement value for which the average is formed	
'Interval'	Specification of the length of the interval in seconds	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation:	
	'Reset' = 0	Perform calculation
	'Reset' = 1	Stop calculation and set result to 0
	'Reset' = 2	Stop calculation and keep result
	'Reset' = 3	Calculate now and then stop calculation

Description

This function returns the average value per time segment of the 'interval' length of 'Expression' as its result. The signal is written to the RAM of the computer. After an interval has passed, the arithmetic average value over this interval is evaluated.

Tip

The result of the AvgInTime function is displayed only in the subsequent interval.

Example

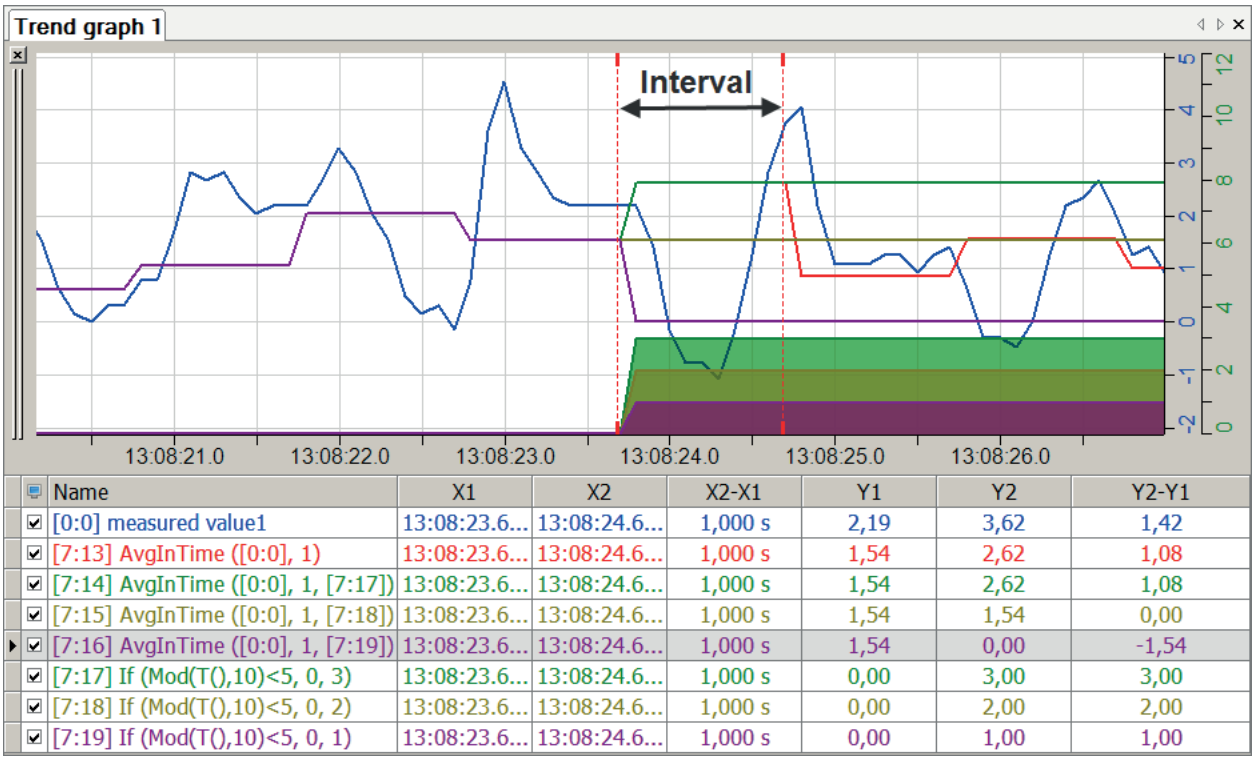
Effects of the 'Reset' parameter

Task description

The 'Reset' parameter should switch from 0 to 1, 2, or 3 in 5-second intervals in order to make the effects of the parameter visible.

Solution

Three if-queries relay, via a modulo function of the time, the value 0 and the values 1, 2, 3 in intervals of five seconds on the 'Reset' parameter of the AvgInTime function.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:0] measured value1	13:08:23.6...	13:08:24.6...	1,000 s	2,19	3,62	1,42
[7:13] AvgInTime ([0:0], 1)	13:08:23.6...	13:08:24.6...	1,000 s	1,54	2,62	1,08
[7:14] AvgInTime ([0:0], 1, [7:17])	13:08:23.6...	13:08:24.6...	1,000 s	1,54	2,62	1,08
[7:15] AvgInTime ([0:0], 1, [7:18])	13:08:23.6...	13:08:24.6...	1,000 s	1,54	1,54	0,00
[7:16] AvgInTime ([0:0], 1, [7:19])	13:08:23.6...	13:08:24.6...	1,000 s	1,54	0,00	-1,54
[7:17] If (Mod(T(),10)<5, 0, 3)	13:08:23.6...	13:08:24.6...	1,000 s	0,00	3,00	3,00
[7:18] If (Mod(T(),10)<5, 0, 2)	13:08:23.6...	13:08:24.6...	1,000 s	0,00	2,00	2,00
[7:19] If (Mod(T(),10)<5, 0, 1)	13:08:23.6...	13:08:24.6...	1,000 s	0,00	1,00	1,00

Red	Continuous calculation without 'Reset' indication	Green	('Reset' = 3) value is calculated before switching
Yellow	('Reset' = 2) value remains constant during switching	Purple	('Reset' = 1) value = 0 during switching

2.5.4 KurtosisInTime

```
KurtosisInTime('Expression','Interval',' Reset=0')
```

Arguments

'Expression'	Measured value, for which the kurtosis is formed	
'Interval'	Specification of the length of the interval in seconds, over which the kurtosis should be calculated.	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation	
	'Reset'=0	Perform calculation
	'Reset'=1	Stop calculation and set result to 0
	'Reset'=2	Stop calculation and keep result
	'Reset'=3	Calculate now and then stop calculation

Description

The calculation of the kurtosis is used, e.g. for the evaluation and analysis of oscillations. It serves to determine the number of outliers within an oscillation signal.

In mathematical terms, the kurtosis is a measure for the relative "flatness" of a distribution (compared to the normal distribution which has a kurtosis of zero). A positive kurtosis indicates

a tapering distribution (a leptokurtic distribution), whereas a negative kurtosis indicates a flat distribution (platykurtic distribution).

This statistical method is particularly suitable for analyzing random or stochastic signals, e.g. in terms of condition-based maintenance (Condition Monitoring) when analyzing oscillations. For characterizing the signal curve, methods of probability density or frequency are used. It is assumed that a noise signal with a Gaussian amplitude distribution can be measured in machines in good order after filtering out, e.g., rotational frequency oscillation components. In the event of damage, individual pulse signals interfere with this signal, altering the distribution function. An evaluation of the system's condition can be carried out through the formation of suitable statistical values, such as the crest factor or the kurtosis factor.

If regularly measured, these methods offer an overview of the machine status. However, the disadvantage is that after they have increased, the characteristic values decrease again. The reason for this is that the number of pulse signals increases with progressive damage. Whereas in turn this has influence on the effective value but barely no effect on the peak value.

Modifications of the time signal caused by shock pulses induce a change in the resulting distribution function. Distinctively discrete damage can cause an increase of the kurtosis factor. Its absolute value thus allows statements on a damage.

2.5.5 MAvg

```
MAvg('Expression','WindowInterval',' UpdateInterval=timebase',' Reset=0')
```

Arguments

'Expression'	Measured value, for which the average value is formed	
'WindowInterval'	Specification in seconds of the length of the interval over which the average is formed; must be a multiple of 'UpdateInterval'.	
'UpdateInterval'	Optional parameter (default = time base); specifies in which cycle the calculation is performed.	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation.	
	'Reset'=0	Perform calculation
	'Reset'=1	Stop calculation and set result to 0
	'Reset'=2	Stop calculation and keep result

Description

This function returns its result as the floating arithmetic mean value of 'Expression' calculated over a ' WindowInterval' in seconds. The calculation is performed in the *'UpdateInterval'* cycle. *'UpdateInterval'* is an optional parameter and is expressed in seconds. If *'UpdateInterval'* is not specified then it is set equal to the time base (default), i.e. as small as possible. The calculation is then carried out progressively by one sample in each case. The calculation can be carried out in larger intervals if a (multiple) value of the time base is entered for 'UpdateInterval'.

'WindowInterval' determines the period for which the mean is calculated each time. 'WindowInterval' must be a multiple of *'UpdateInterval'*. Otherwise, 'WindowInterval' is automatically changed to the first multiple of *'UpdateInterval'* greater than or equal to 'WindowInterval.'

Tip

Using these functions, signals and expressions that are not time-based, i.e. which have the basis length, frequency or 1/length, can also be processed. Instead of seconds, the X-axis range should then be entered in m, Hz or 1/m corresponding to the base.

Example

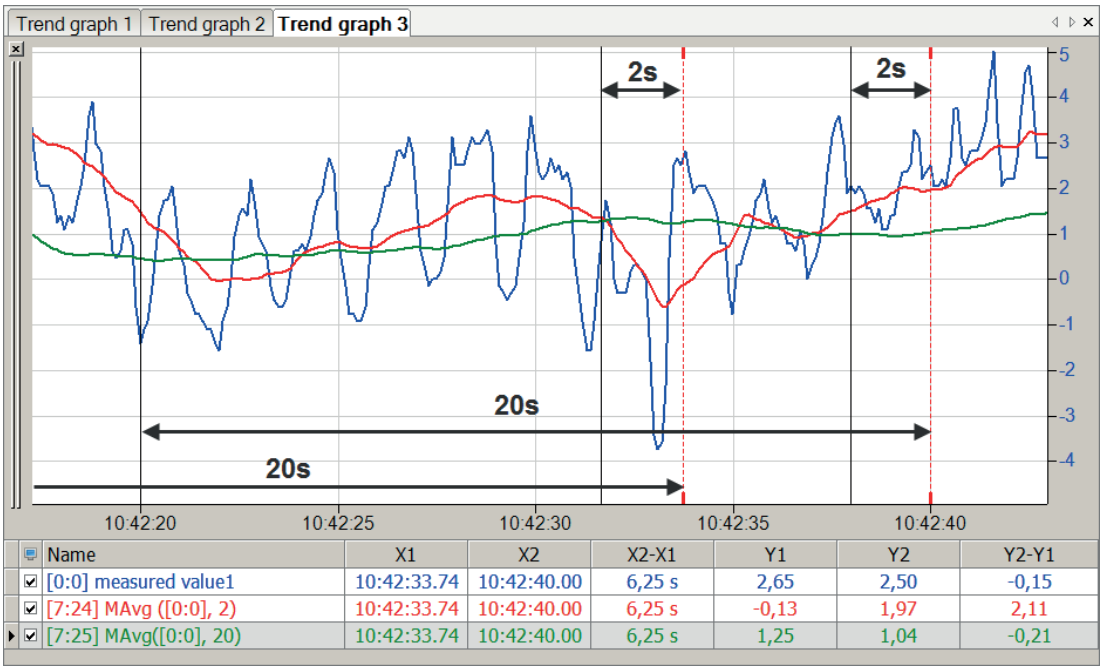
The mean values of the previous 2 and 20 seconds are relevant for a measured value.

Task description

The MAvg function is used once with an interval time of 2 seconds and once with an interval time of 20 seconds.

Solution

In the figure below the blue curve shows the measured value, the red curve shows the average over an interval of 2 seconds, and the green curve shows the average over an interval of 20 seconds



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:0] measured value1	10:42:33.74	10:42:40.00	6,25 s	2,65	2,50	-0,15
[7:24] MAvg ([0:0], 2)	10:42:33.74	10:42:40.00	6,25 s	-0,13	1,97	2,11
[7:25] MAvg([0:0], 20)	10:42:33.74	10:42:40.00	6,25 s	1,25	1,04	-0,21

2.5.6 MAvgOnTrigger

```
MAvgOnTrigger('Expression', 'Trigger', 'NumberOfValues*', 'Reset=0')
```

Parameters ending with * are only evaluated once at the start of the acquisition..

Arguments

'Expression'	Measured value, for which the average is formed	
'Trigger'	Digital signal or expression as a trigger for the execution of the command	
'Number-OfVlaues*'	Number of samples which are used for the calculation of the average; Parameter will be evaluated at start of acquisition. Therefore, rather use a fix value instead of an expression.	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation.	
	'Reset' =0	Perform calculation
	'Reset' =1	Stop calculation, delete all buffered data and set result to 0
	'Reset' =2	Stop calculation and keep result

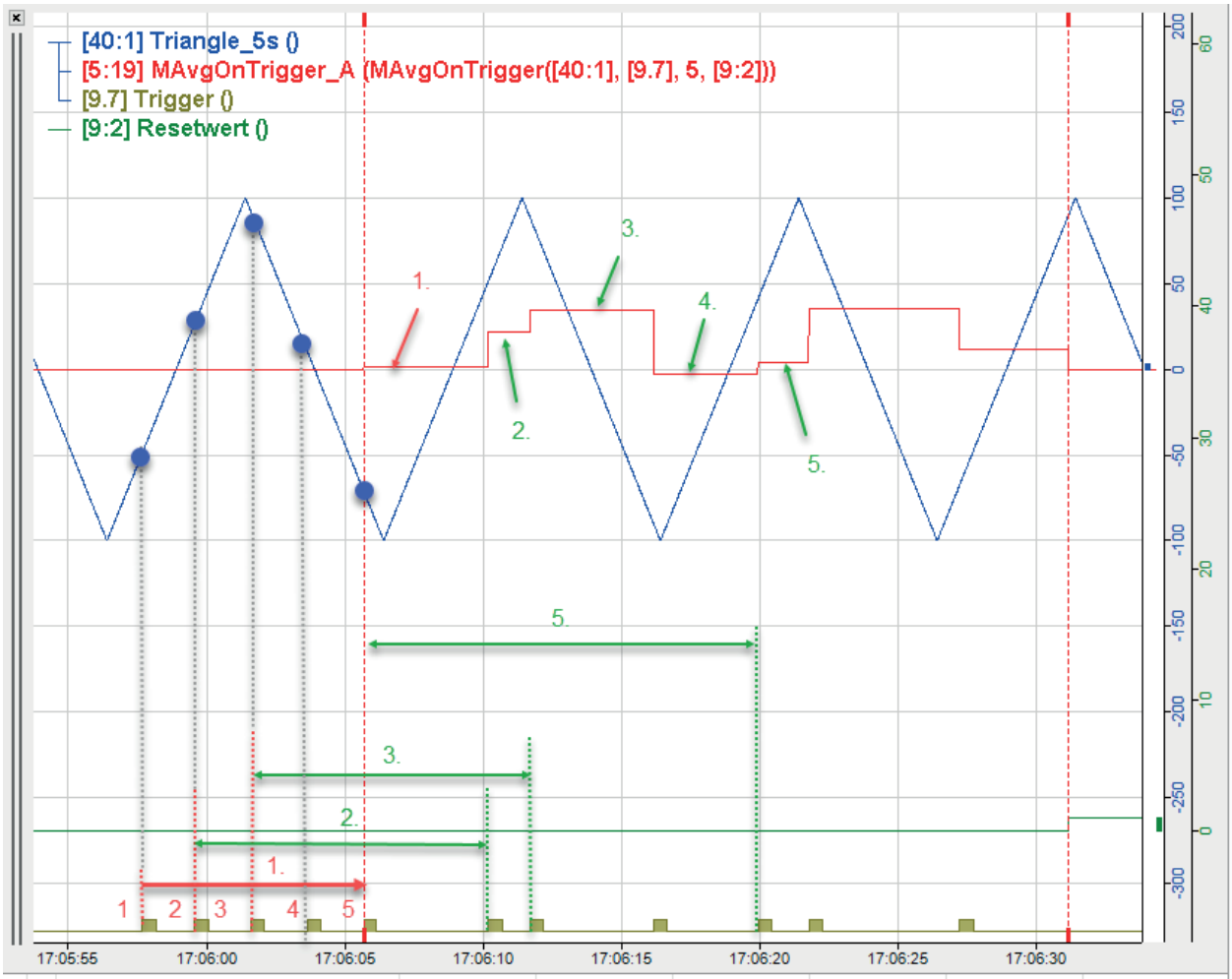
Description

This function returns the moving arithmetic average of 'Expression', which is evaluated on every rising edge of 'Trigger'. Only as many values as specified by 'NumberOfValues' are taken into account for the calculation. After start of acquisition or after a reset with 'Reset' = 1, 'Trigger' has to be fired 'NumberOfValues' times before the first average value can be calculated. With each other trigger the average for the last 'NumberOfValues' triggered samples is calculated.

This makes it possible to calculate moving averages controlled by events. The calculation then can be performed e. g. per revolution of a machine or per produced part instead only linear over time.

Example

Apply MAvgOnTrigger on a triangular signal



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[40:1] Triangle_5s	17:06:05.72	17:06:31.16	25,44 s	-72,00	89,64	161,63
[5:19] MAvgOnTrigger_A	17:06:05.72	17:06:31.16	25,44 s	1,60	0,00	-1,60
[9.7] Trigger	17:06:05.72	17:06:31.16	25,44 s	1	0	-1
[9:2] Resetwert	17:06:05.72	17:06:31.16	25,44 s	0,00	1,00	1,00

For explanatory reasons only 'NumberOfValues' = 5 has been specified in the example.

Not until the 5th trigger after start of acquisition a first average value (red 1.) is calculated from the 5 selected samples (blue dots). The distance between the trigger events does not matter. The small and even distance between the first 5 triggers is due to graphical reasons.

With each other trigger the new actual and the last 4 samples are taken into account for the calculation, returning the respective average values (green 2., 3., 4., 5. etc.)

2.5.7 Max

```
Max('Expression',' Reset=0')
```

Arguments

'Expression'	Signal, for which the maximum value should be determined	
'Reset'	Optional digital parameter that can be used to reset the maximum, e.g., in order to ignore the leveling-off processes of the measuring signal during the start-up phase. 'Reset' can be an expression as well.	
	'Reset' = TRUE	Stop calculation and set result to the instantaneous "Expression" value
	'Reset' = 0	Perform calculation; most recently detected maximum is displayed. (Default)

Description

This function returns the maximum value of 'Expression' as its result. It is displayed as a constant value (horizontal line) in the signal strip. Each value is compared to the previous one. If the new value is higher than the previous one, the higher value will be incorporated into the curve. If the new value is equal or lower than the previous one, the previous value will be included. With the digital 'Reset' signal, the maximum value calculation can be stopped and the result can be reset to the current value of the input signal. Without the reset signal there is no way to reset the display unless the measurement is stopped and restarted. 'Reset' can also be formulated as an expression.

Examples:

Max([0:0])	No reset takes place.
Max([0:0],If(Mod(T(),20)=0,TRUE(),FALSE()))	The maximum value is reset every 20 seconds.
Max([0:0],[3.1])	e.g. with [3.1] = If([0:0]<10, 1, 0) The maximum value is reset as soon as the expression [3.1] returns TRUE, i.e., if the expression [0:0] falls below the limit value of 10.

Example

Tip

This function can be used in a virtual retentive module. Its result values can thus be obtained despite stopping and restarting the measurement.

The maximum should be determined for a signal. The start phase should be ignored in the calculation.

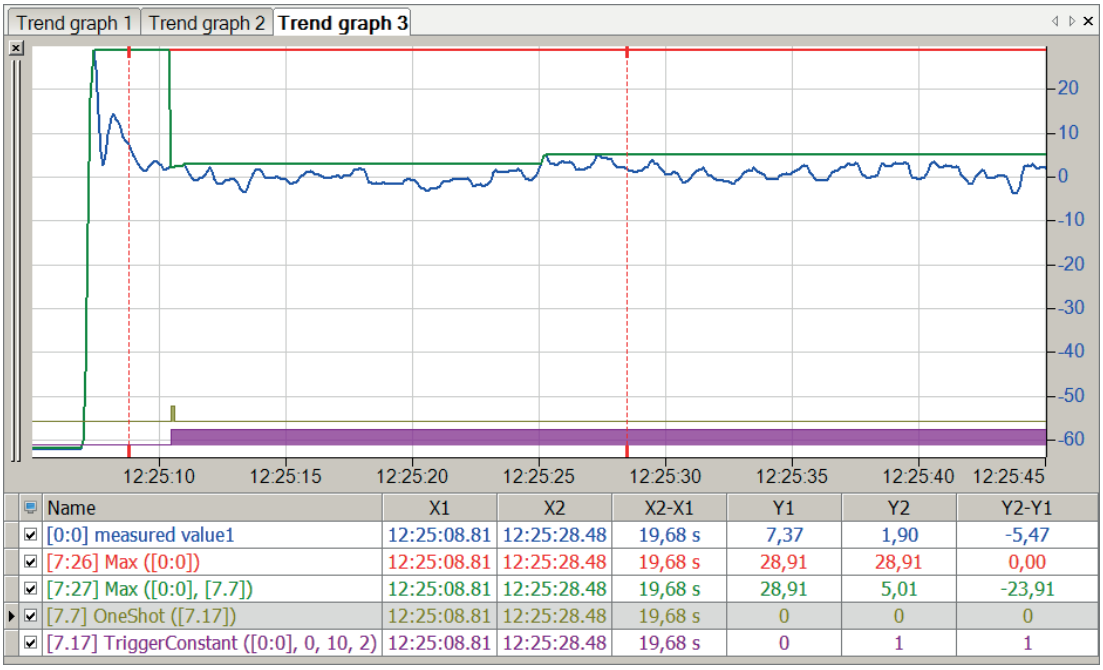
Task description

In order to remove fluctuations during the start phase, the maximum value is reset using the 'reset' function. This can be done using the TriggerConstant function, which waits for the leveling off of the signal. The setting of the 'reset' parameter is done via the edge detection ('One-Shot') of the trigger.

Note

The 'Reset' parameter should not be permanently set to TRUE because the maximum value would then be reset permanently and thus correspond to the signal.

Solution



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:0] measured value1	12:25:08.81	12:25:28.48	19,68 s	7,37	1,90	-5,47
[7:26] Max ([0:0])	12:25:08.81	12:25:28.48	19,68 s	28,91	28,91	0,00
[7:27] Max ([0:0], [7.7])	12:25:08.81	12:25:28.48	19,68 s	28,91	5,01	-23,91
[7.7] OneShot ([7.17])	12:25:08.81	12:25:28.48	19,68 s	0	0	0
[7.17] TriggerConstant ([0:0], 0, 10, 2)	12:25:08.81	12:25:28.48	19,68 s	0	1	1

Blue	Measured value	Red	Falsified maximum value without 'Reset'
Green	Maximum value with reset after start phase	Yellow	One-shot function for triggering the 'Reset'
Purple	Trigger after the start phase		

2.5.8 Max2

```
Max2('Expression1','Expression2', …)
```

Description

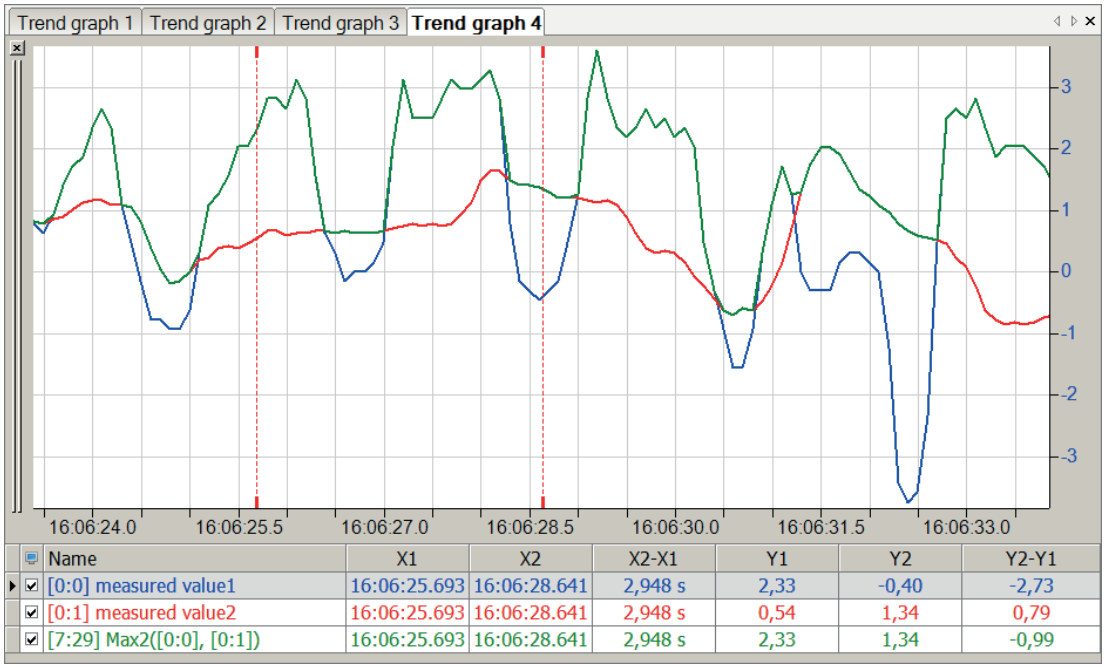
This function returns the maximum of all expressions, 'Expression1', 'Expression2', etc. as its result. The expressions or signals are compared measured value by measured value, with the largest value in each case being presented as the result. Up to 1000 arguments are permitted.

Example

Only the larger value of the two measured values is relevant.

Solution

In the figure below the blue curve shows the measured value 1, the red curve shows the measured value 2, and the green cure shows the course of the maximum values.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:0] measured value1	16:06:25.693	16:06:28.641	2,948 s	2,33	-0,40	-2,73
[0:1] measured value2	16:06:25.693	16:06:28.641	2,948 s	0,54	1,34	0,79
[7:29] Max2([0:0], [0:1])	16:06:25.693	16:06:28.641	2,948 s	2,33	1,34	-0,99

2.5.9 MaxInTime

```
MaxInTime('Expression','Interval',' Reset=0')
```

Arguments

'Expression'	Measured value, for which the maximum is formed	
'Interval'	Specification of the interval length in seconds, over which the maximum should be calculated.	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation	
	'Reset' =0	Perform calculation
	'Reset'=1	Stop calculation and set result to 0
	'Reset'=2	Stop calculation and keep result
	'Reset'=3	Calculate now and then stop calculation

Description

This function returns the maximum value of 'Expression' within each interval of the 'interval' length (in seconds) as its result.

Note

The result of the MaxInTime function is displayed only in the subsequent interval.

Example

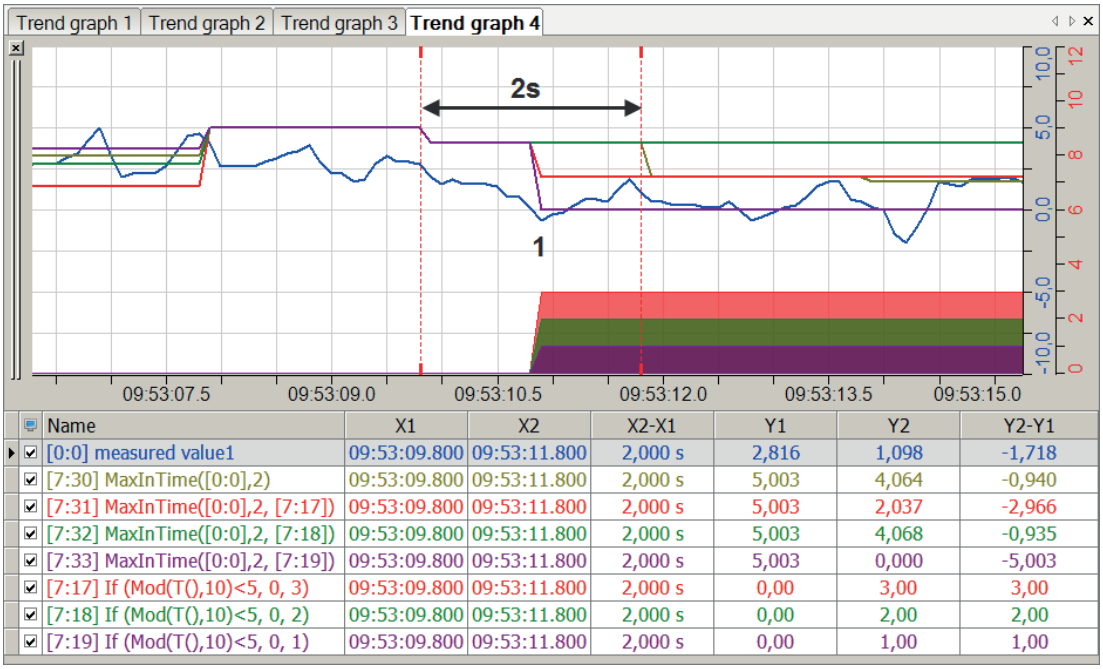
Effects of the 'Reset' parameter

Task description

The 'Reset' parameter should switch from 0 to 1, 2, or 3 in 5-second intervals in order to make the effects of the parameter visible.

Solution

Three If-queries relay, via a modulo function of the time, the value 0 and the values 1, 2, 3 in intervals of 5 seconds on the 'Reset' parameter of the MaxInTime function.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:0] measured value1	09:53:09.800	09:53:11.800	2,000 s	2,816	1,098	-1,718
[7:30] MaxInTime([0:0],2)	09:53:09.800	09:53:11.800	2,000 s	5,003	4,064	-0,940
[7:31] MaxInTime([0:0],2, [7:17])	09:53:09.800	09:53:11.800	2,000 s	5,003	2,037	-2,966
[7:32] MaxInTime([0:0],2, [7:18])	09:53:09.800	09:53:11.800	2,000 s	5,003	4,068	-0,935
[7:33] MaxInTime([0:0],2, [7:19])	09:53:09.800	09:53:11.800	2,000 s	5,003	0,000	-5,003
[7:17] If (Mod(T(),10)<5, 0, 3)	09:53:09.800	09:53:11.800	2,000 s	0,00	3,00	3,00
[7:18] If (Mod(T(),10)<5, 0, 2)	09:53:09.800	09:53:11.800	2,000 s	0,00	2,00	2,00
[7:19] If (Mod(T(),10)<5, 0, 1)	09:53:09.800	09:53:11.800	2,000 s	0,00	1,00	1,00

Blue	Measured value	Yellow	Continuous calculation without 'Reset' application
Red	('Reset' = 3) value is calculated before interruption, even within an interval (1)	Green	('Reset'= 2) value remains constant during interruption
Purple	('Reset' = 1) value=0 during interruption, even within an interval (1)		

2.5.10 Median2

```
Median2('Expression1', 'Expression2', …)
```

Arguments

'Expression'	Measured value(s), for which the median value is formed

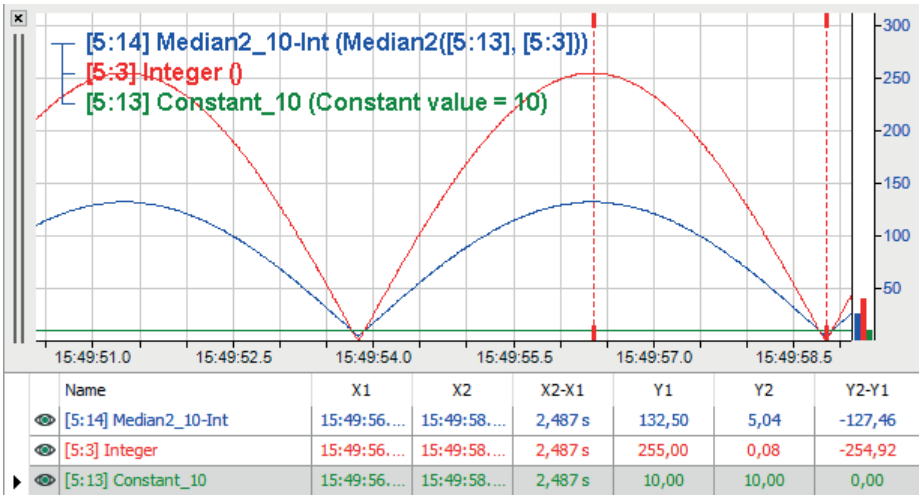
Description

This function returns the median value of all arguments Every cycle, the actual values of all arguments are added and the sum is divided by two.

Example

The median of a constant and a periodic value should be formed.

Solution



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[5:14] Median2_10-Int	15:49:56....	15:49:58....	2,487 s	132,50	5,04	-127,46
[5:3] Integer	15:49:56....	15:49:58....	2,487 s	255,00	0,08	-254,92
[5:13] Constant_10	15:49:56....	15:49:58....	2,487 s	10,00	10,00	0,00

The example shows a median of 132.5 ((10 + 255)/2) at the peak of the curve.

2.5.11 MedianInTime

```
MedianInTime('Expression', 'Interval', ' Reset=0')
```

Arguments

'Expression'	Measured value, for which the median is formed	
'Interval'	Specification of the length of the interval in seconds	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation:	
	'Reset' = 0	Perform calculation
	'Reset' = 1	Stop calculation and set result to 0
	'Reset' = 2	Stop calculation, delete all buffered data and keep result
	'Reset' = 3	Calculate now and then stop calculation

Description

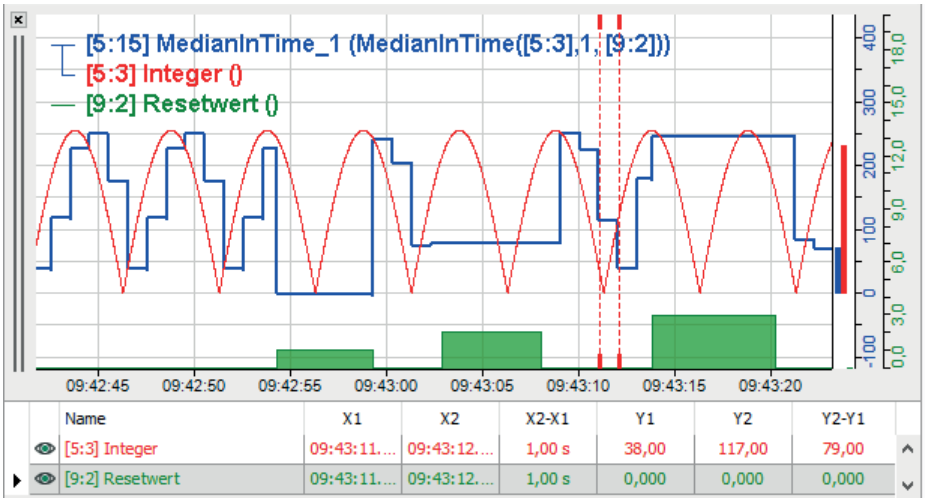
This function returns the median value per time segment of the 'interval' length of 'Expression' as its result. The signal is written to the RAM of the computer. After an interval has passed, the median value over this interval is evaluated.

Tip

The result of the MedianInTime function is displayed only in the subsequent interval.

Example

The following figure shows the calculation of the MedianInTime (blue) of a signal (red) and the impact of the parameter ‘Reset’ (green).



2.5.12 Min

```
Min('Expression',' Reset=0')
```

Arguments

'Expression'	Signal, for which the minimum value should be determined	
'Reset'	Optional digital parameter that can be used to reset the minimum, e.g., in order to ignore the leveling-off processes of the measuring signal during the start-up phase. ‘Reset’ can be an expression as well.	
	'Reset' = TRUE	Stop calculation and set result to the instantaneous "Expression" value
	'Reset' = 0	Perform calculation; most recently detected minimum is displayed. (Default)

Description

This function returns the minimum value of the 'Expression' signal as its result. It is displayed as a constant value (horizontal line) in the signal strip. Each value is compared to the previous one. If the new value is lower than the previous one, the lower value will be incorporated into the curve. If the new value is equal or higher than the previous one, the previous value will be included. With the digital 'Reset' signal, the minimum value calculation can be stopped and the result can be reset to the current value of the input signal. Without the Reset signal there is no way to reset the display unless the measurement is stopped and restarted. . 'Reset' can also be formulated as an expression.

Examples:

Min([0:0])	No reset takes place.
Min([0:0],If(Mod(T(),20)=0,TRUE(),FALSE()))	The minimum value is reset every 20 seconds.
Min([0:0],[3.1])	e.g. with [3.1] = If([0:0]<10, 1, 0) The minimum value is reset as soon as the expression [3.1] returns TRUE, i.e. if the expression [0:0] falls below the limit value 10.

Example

Tip

This function can be used in a virtual retentive module. Its result values can thus be obtained via stopping and restarting the measurement.

The minimum should be determined for a signal. The start phase should be ignored in the calculation.

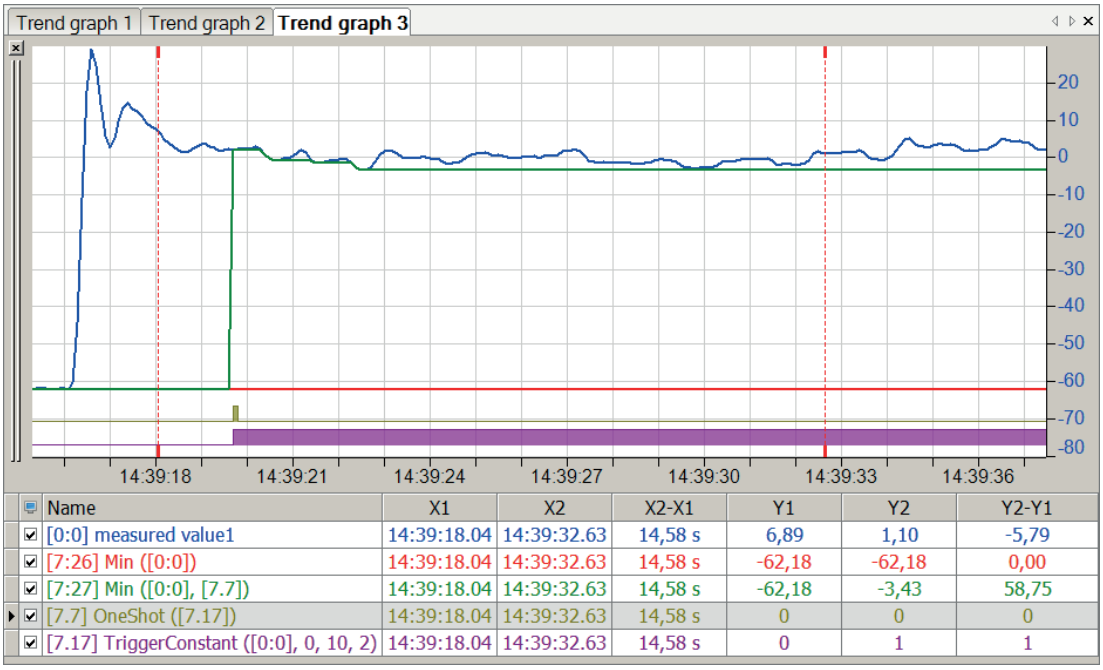
Task description

In order to remove fluctuations during the start phase, the minimum value is reset using the 'reset' function. This can be done via the TriggerConstant function, which waits for the leveling off of the signal. The setting of the 'reset' parameter is done via the edge detection ('OneShot') of the trigger.

Note

The 'reset' parameter should not be permanently set to TRUE because the minimum value would then be reset permanently and thus correspond to the signal.

Solution



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:0] measured value1	14:39:18.04	14:39:32.63	14,58 s	6,89	1,10	-5,79
[7:26] Min ([0:0])	14:39:18.04	14:39:32.63	14,58 s	-62,18	-62,18	0,00
[7:27] Min ([0:0], [7.7])	14:39:18.04	14:39:32.63	14,58 s	-62,18	-3,43	58,75
[7.7] OneShot ([7.17])	14:39:18.04	14:39:32.63	14,58 s	0	0	0
[7.17] TriggerConstant ([0:0], 0, 10, 2)	14:39:18.04	14:39:32.63	14,58 s	0	1	1

Blue	Measured value	Red	falsified minimum value without 'Reset'
Green	Minimum value with reset after start phase	Yellow	One-shot function for triggering the 'Reset'
Purple	Trigger after the start phase		

2.5.13 Min2

```
Min2('Expression1','Expression2')
```

Description

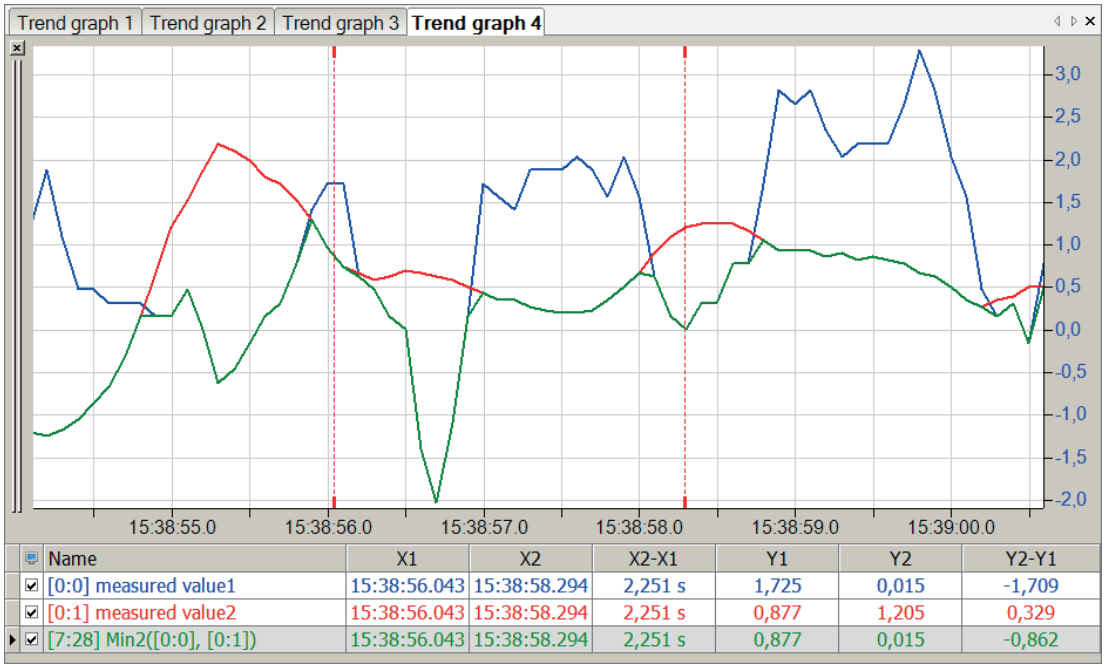
This function returns the minimum of all expressions, 'Expression1', 'Expression2', etc. as its result. The expressions or signals are compared measured value by measured value, with the largest value in each case being presented as the result. Up to 1000 arguments are permitted.

Example

Only the smaller value of the two measured values is relevant.

Solution

In the figure below the blue curve shows the measured value 1, the red curve shows the measured value 2, and the green cure shows the course of the minimum values.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:0] measured value1	15:38:56.043	15:38:58.294	2,251 s	1,725	0,015	-1,709
[0:1] measured value2	15:38:56.043	15:38:58.294	2,251 s	0,877	1,205	0,329
[7:28] Min2([0:0], [0:1])	15:38:56.043	15:38:58.294	2,251 s	0,877	0,015	-0,862

2.5.14 MinInTime

```
MinInTime('Expression','Interval',' Reset=0')
```

Arguments

'Expression'	Measured value, for which the minimum is formed	
'Interval'	Specification of the length of the interval in seconds, over which the minimum should be calculated.	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation	
	'Reset'=0	Perform calculation
	'Reset'=1	Stop calculation and set result to 0
	'Reset'=2	Stop calculation and keep result
	'Reset'=3	Calculate now and then stop calculation

Description

This function returns the minimum value of 'Expression' within each interval of the 'Interval' length (in seconds) as its result.

Note

The result of the MinInTime function is displayed only in the subsequent interval.

Example

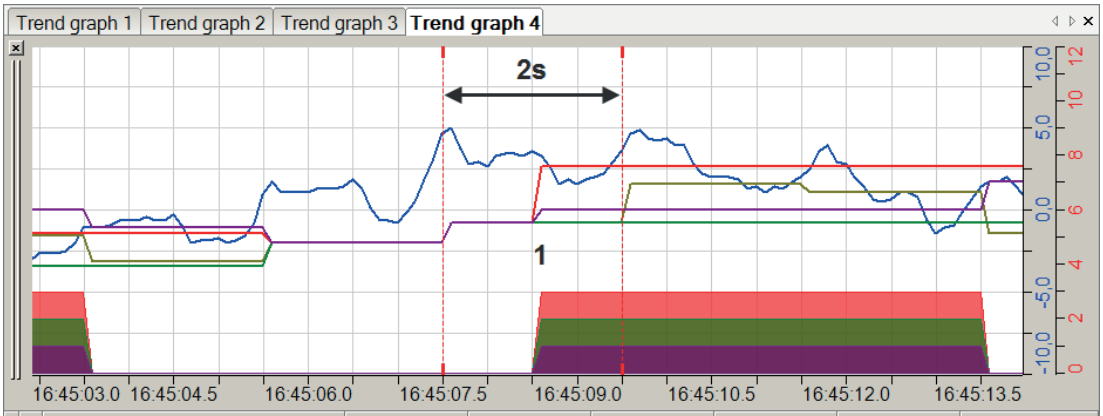
Effects of the 'Reset' parameter

Task description

The 'Reset' parameter should switch from 0 to 1, 2, or 3 in 5-second intervals in order to make the effects of the parameter visible.

Solution

Three If-queries relay, via a modulo function of the time, the value 0 and the values 1, 2, 3 in intervals of 5 seconds on the 'Reset' parameter of the MinInTime function.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:0] measured value1	16:45:07.500	16:45:09.500	2,000 s	4,695	3,602	-1,093
[7:30] MinInTime([0:0],2)	16:45:07.500	16:45:09.500	2,000 s	-2,018	-0,770	1,248
[7:31] MinInTime([0:0],2, [7:17])	16:45:07.500	16:45:09.500	2,000 s	-2,018	2,662	4,680
[7:32] MinInTime([0:0],2, [7:18])	16:45:07.500	16:45:09.500	2,000 s	-2,018	-0,775	1,243
[7:33] MinInTime([0:0],2, [7:19])	16:45:07.500	16:45:09.500	2,000 s	-2,018	0,000	2,018
[7:17] If (Mod(T(),10)<5, 0, 3)	16:45:07.500	16:45:09.500	2,000 s	0,00	3,00	3,00
[7:18] If (Mod(T(),10)<5, 0, 2)	16:45:07.500	16:45:09.500	2,000 s	0,00	2,00	2,00
[7:19] If (Mod(T(),10)<5, 0, 1)	16:45:07.500	16:45:09.500	2,000 s	0,00	1,00	1,00

Blue	Measured value	Yellow	Continuous calculation without 'Reset' application
Red	('Reset' = 3) value is calculated before interruption, even within an interval (1)	Green	('Reset'= 2) value remains constant during interruption
Purple	('Reset' = 1) value=0 during interruption, even within an interval (1)		

2.5.15 MKurtosis

```
MKurtosis('Expression','WindowInterval', 'UpdateInterval=timebase', 'Reset=0')
```

Arguments

'Expression'	Measured value, for which the kurtosis is formed	
'WindowInterval'	Specification in seconds of the length of the interval over which the kurtosis is formed; must be a multiple of 'UpdateInterval'.	
'UpdateInterval'	Optional parameter (default = time base); specifies in which cycle the calculation is performed.	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation.	
	'Reset'=0	Perform calculation
	'Reset'=1	Stop calculation and set result to 0
	'Reset'=2	Stop calculation and keep result

Description

Returns the kurtosis value of 'Expression' every *'UpdateInterval'* seconds based on a floating window of 'WindowInterval' seconds.

The *'UpdateInterval'* parameter is optional. If not specified then it is set equal to the time base of the function (i.e. as small as possible).

'WindowInterval' must be a multiple of *'UpdateInterval'*. Otherwise, 'WindowInterval' is automatically changed to the first multiple of *'UpdateInterval'* greater than or equal to 'WindowInterval.'

For more information about Kurtosis, see Chapter ↗ *KurtosisInTime, page 46*

2.5.16 MMax

```
MMax('Expression','WindowInterval', 'UpdateInterval=timebase', 'Reset=0')
```

Arguments

'Expression'	Measured value, for which the maximum is formed	
'WindowInterval'	Specification in seconds of the interval length over which the maximum should be calculated; must be a multiple of 'UpdateInterval'.	
'UpdateInterval'	Optional parameter (default = time base); specifies in which cycle the calculation is performed.	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation	
	'Reset'=0	Perform calculation
	'Reset'=1	Stop calculation, reset and set result to 0
	'Reset'=2	Stop calculation and keep result

Description

This function returns the maximum of 'Expression' every 'UpdateInterval' seconds based on a floating window of 'WindowInterval' seconds.

The *'UpdateInterval'* parameter is optional. If not specified then it is set equal to the time base of the function (i.e. as small as possible).

'WindowInterval' must be a multiple of *'UpdateInterval'*. Otherwise, 'WindowInterval' is automatically changed to the first multiple of *'UpdateInterval'* greater than or equal to 'WindowInterval.'

2.5.17 MMaxDynamic

```
MMaxDynamic('Expression','WindowInterval', 'MaxWindowInterval*')
```

Arguments

'Expression'	Measured value, for which the maximum is formed
'WindowInterval'	Specification in seconds of the interval length over which the maximum should be calculated
'MaxWindowInterval*'	Limit for 'WindowInterval'; the size of 'WindowInterval' will be limited to this value, given in seconds.

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns the maximum of 'Expression' over a floating window of 'WindowInterval' seconds. 'WindowInterval' can be a signal or expression and may change with every sample. The limit value of 'WindowInterval' is 'MaxWindowInterval' seconds.

2.5.18 MMedian

```
MMedian('Expression', 'WindowInterval', 'UpdateInterval=timebase', ' Reset=0')
```

Arguments

'Expression'	Measured value, for which the median is formed	
'WindowInterval'	Specification in seconds of the interval length over which the median should be calculated; must be a multiple of 'UpdateInterval'.	
'UpdateInterval'	Optional parameter (default = time base); specifies in which cycle the calculation is performed.	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation:	
	'Reset' = 0	Perform calculation
	'Reset' = 1	Stop calculation, delete all buffered data and set result to 0
	'Reset' = 2	Stop calculation, delete all buffered data and keep result

Description

This function returns the median of 'Expression' every 'UpdateInterval' seconds based on a floating window of 'WindowInterval' seconds.

The 'UpdateInterval' parameter is optional. If not specified then it is set equal to the time base of the function (i.e. as small as possible).

'WindowInterval' must be a multiple of *'UpdateInterval'*. Otherwise, 'WindowInterval' is automatically changed to the first multiple of *'UpdateInterval'* greater than or equal to 'WindowInterval.'

'WindowInterval' and 'UpdateInterval' can be determined by expressions. Changes of these interval values are applied not until a reset.

Tip

The result of the MMedian function is displayed only in the subsequent interval.

2.5.19 MMin

```
MMin('Expression','WindowInterval', 'UpdateInterval=timebase', 'Reset=0')
```

Arguments

'Expression'	Measured value, for which the minimum is formed	
'WindowInterval'	Specification in seconds of the interval length over which the minimum should be calculated; must be a multiple of 'UpdateInterval'.	
'UpdateInterval'	Optional parameter (default = time base); specifies in which cycle the calculation is performed.	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation	
	'Reset'=0	Perform calculation
	'Reset'=1	Stop calculation, reset and set result to 0
	'Reset'=2	Stop calculation and keep result

Description

This function returns the minimum of 'Expression' every *'UpdateInterval'* seconds based on a floating window of 'WindowInterval' seconds.

The *'UpdateInterval'* parameter is optional. If not specified then it is set equal to the time base of the function (i.e. as small as possible).

'WindowInterval' must be a multiple of *'UpdateInterval'*. Otherwise, 'WindowInterval' is automatically changed to the first multiple of *'UpdateInterval'* greater than or equal to 'WindowInterval.'

2.5.20 MMinDynamic

```
MMinDynamic('Expression','WindowInterval', 'MaxWindowInterval*')
```

Arguments

'Expression'	Measured value, for which the minimum is formed
'WindowInterval'	Specification in seconds of the interval length over which the minimum should be calculated
'MaxWindowInterval'	Limit for 'WindowInterval'; the size of 'WindowInterval' will be limited to this value, given in seconds.

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns the minimum of 'Expression' over a floating window of 'WindowInterval' seconds. 'WindowInterval' can be a signal or expression and may change with every sample. The limit value of 'WindowInterval' is 'MaxWindowInterval' seconds.

2.5.21 MSkewness

```
MSkewness('Expression','WindowInterval', 'UpdateInterval=timebase', 'Reset=0')
```

Arguments

'Expression'	Measured value, for which the skewness is formed	
'WindowInterval'	Specification in seconds of the interval length over which the skewness should be calculated; must be a multiple of 'UpdateInterval'.	
'UpdateInterval'	Optional parameter (default = time base); specifies in which cycle the calculation is performed.	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation.	
	'Reset'=0	Perform calculation
	'Reset'=1	Stop calculation and set result to 0
	'Reset'=2	Stop calculation and keep result

Description

Returns the skewness value of 'Expression' every *'UpdateInterval'* seconds based on a floating window of 'WindowInterval' seconds.

The *'UpdateInterval'* parameter is optional. If not specified then it is set equal to the time base of the function (i.e. as small as possible).

'WindowInterval' must be a multiple of *'UpdateInterval'*. Otherwise, 'WindowInterval' is automatically changed to the first multiple of *'UpdateInterval'* greater than or equal to 'WindowInterval.'

For more information about skewness, see Chapter ↗ *SkewnessInTime, page 67*

2.5.22 MStdDev

```
MStdDev('Expression', 'WindowInterval', 'UpdateInterval=timebase', ' Reset=0')
```

Arguments

'Expression'	Measured value, for which the standard deviation is formed	
'WindowInterval'	Specification in seconds of the interval length over which the median should be calculated; must be a multiple of 'UpdateInterval'.	
'UpdateInterval'	Optional parameter (default = time base); specifies in which cycle the calculation is performed.	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation:	
	'Reset' = 0	Perform calculation
	'Reset' = 1	Stop calculation, delete all buffered data and set result to 0
	'Reset' = 2	Stop calculation, delete all buffered data and keep result

Description

This function returns the the standard deviation 'Expression' every 'UpdateInterval' seconds based on a floating window of 'WindowInterval' seconds.

The 'UpdateInterval' parameter is optional. If not specified then it is set equal to the time base of the function (i.e. as small as possible).

'WindowInterval' must be a multiple of 'UpdateInterval'. Otherwise, 'WindowInterval' is automatically changed to the first multiple of *'UpdateInterval'* greater than or equal to 'WindowInterval.'

'WindowInterval' and 'UpdateInterval' can be determined by expressions. Changes of these interval values are applied not until a reset.

Tip

The result of the MStdDev function is displayed only in the subsequent interval.

2.5.23 SkewnessInTime

```
SkewnessInTime('Expression','Interval','Reset=0')
```

Arguments

'Expression'	Measured value, for which the skewness is formed	
'Interval'	Specification of the interval length in seconds, over which the skewness should be calculated.	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation	
	'Reset' = 0	Perform calculation
	'Reset' = 1	Stop calculation and set result to 0
	'Reset' = 2	Stop calculation and keep result
	'Reset' = 3	Calculate now and then stop calculation

Description

Like the kurtosis factor, the skewness factor can be used for evaluating and analyzing oscillations. The skewness factor can be used if the symmetrical properties of an oscillation signal are to be checked (e.g. acceleration signal).

With this function, the selected expression is divided into equal-duration intervals of size 'Interval'. For these intervals, the skewness is subsequently calculated.

In mathematical terms, this is the evaluation of the skewness of a distribution function. A distribution is called right-skewed (or positively-skewed) when the majority of the distribution is concentrated on the left side. A distribution is called left-skewed (or negatively-skewed) when the majority of the distribution is concentrated on the right. The skewness level is defined by the third order of the central moment of the distribution.

The procedure to calculate the skewness is similar to that of the KurtosisInTime function.

2.5.24 StdDev

```
StdDev('Expression', 'Reset=0')
```

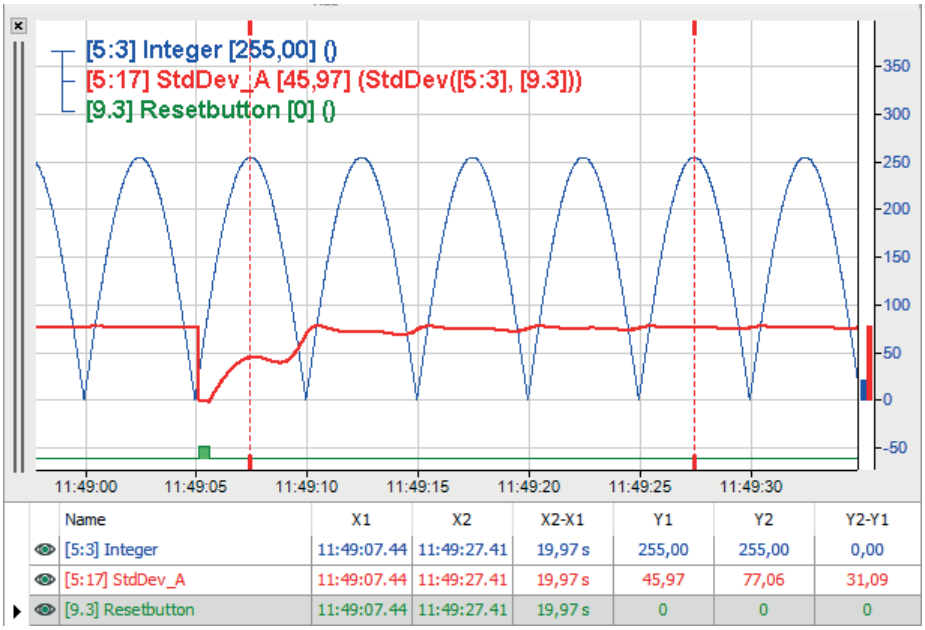
Arguments

'Expression'	Signal, for which the standard deviation should be determined	
'Reset'	Optional parameter (default = 0) to stop and restart the	
	'Reset' = 0	Perform calculation
	'Reset' = 1	Stop calculation and set result to 0

Description

This function returns the standard deviation of 'Expression' as its result. If 'Reset' is TRUE the result is reset to 0.

Example



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[5:3] Integer	11:49:07.44	11:49:27.41	19,97 s	255,00	255,00	0,00
[5:17] StdDev_A	11:49:07.44	11:49:27.41	19,97 s	45,97	77,06	31,09
[9.3] Resetbutton	11:49:07.44	11:49:27.41	19,97 s	0	0	0

2.5.25 StdDev2

```
StdDev2('Expression1', 'Expression2', …)
```

Arguments

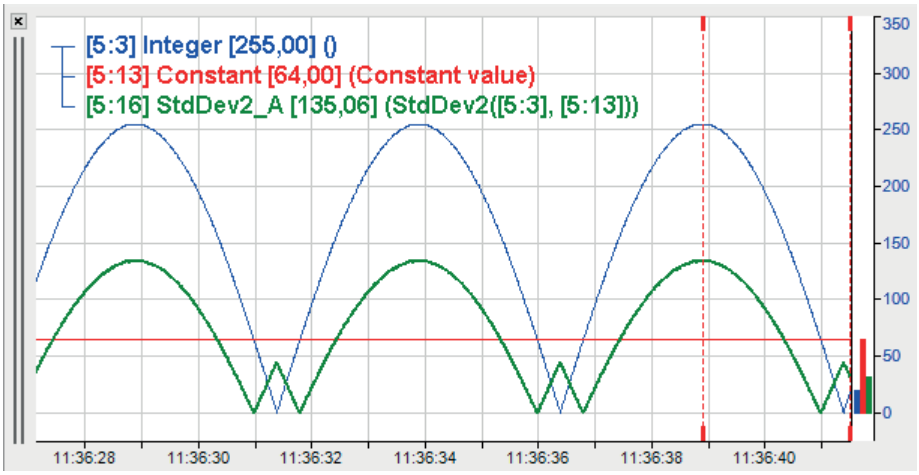
'Expression'	Measured value(s), for which the standard deviation is formed

Description

This function returns the standard deviation of all arguments

Example

Standard deviation of a constant and a variable value



2.5.26 StddevInTime

```
StddevInTime('Expression','Interval',' Reset=0')
```

Arguments

'Expression'	Measured value, for which the standard deviation is formed	
'Interval'	Specification of the length of the interval in seconds, over which the standard deviation should be calculated.	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation	
	'Reset' = 0	Perform calculation
	'Reset' = 1	Stop calculation and set result to 0
	'Reset' = 2	Stop calculation and keep result
	'Reset' = 3	Calculate now and then stop calculation

Description

This function returns the standard deviation of 'Expression' over each time interval of the length 'Interval' as its result. The calculation can be stopped using the optional parameter, 'Reset.'

The standard deviation is calculated by the following formula:

$$s_x = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}}$$

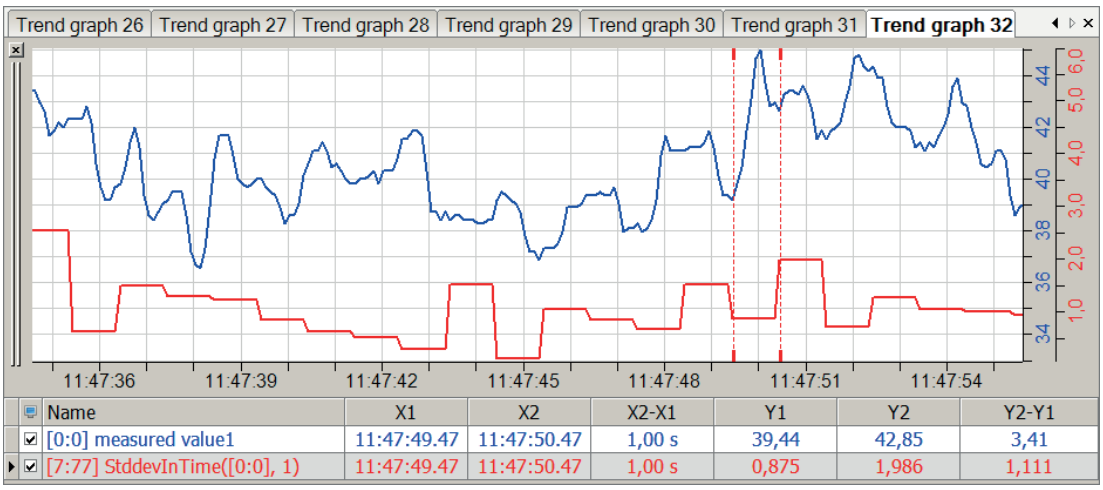
s_x = standard deviation
$\bar{x}$ = average value
n = number of samples

Example

For a signal waveform, the standard deviation should be determined in time intervals of one second. A reset is not necessary.

Solution

In the figure below the blue curve shows the measured values and the red curve shows the standard deviation in the intervals of length of one second.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:0] measured value1	11:47:49.47	11:47:50.47	1,00 s	39,44	42,85	3,41
[7:77] StddevInTime([0:0], 1)	11:47:49.47	11:47:50.47	1,00 s	0,875	1,986	1,111

Note

The result of the StdDevInTime function is always displayed in the subsequent interval.

2.6 Trigger functions

2.6.1 Periodic trigger

```
PeriodicTrigger('Interval*','StartTime*','UseSystemTime*')
```

Arguments

'Interval*'	Interval in minutes, in which the value TRUE is returned	
'StartTime*'	Specification in minutes; produces the start time ('StartTime' modulo 'Interval') after using the modulo function	
'UseSystemTime*'	Determines whether the system time or internal high-resolution timer is used.	
	'UseSystemTime = 0	No system time but internal timer is used.
	'UseSystemTime' > 0	System time is used.

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns the value TRUE every ‘Interval’ minutes, starting at ‘StartTime’ modulo 'Interval' minutes. The ‘UseSystemTime’ flag determines whether the system time or the internal high-resolution timer is to be used.

Example

Displaying a trigger every 10 seconds

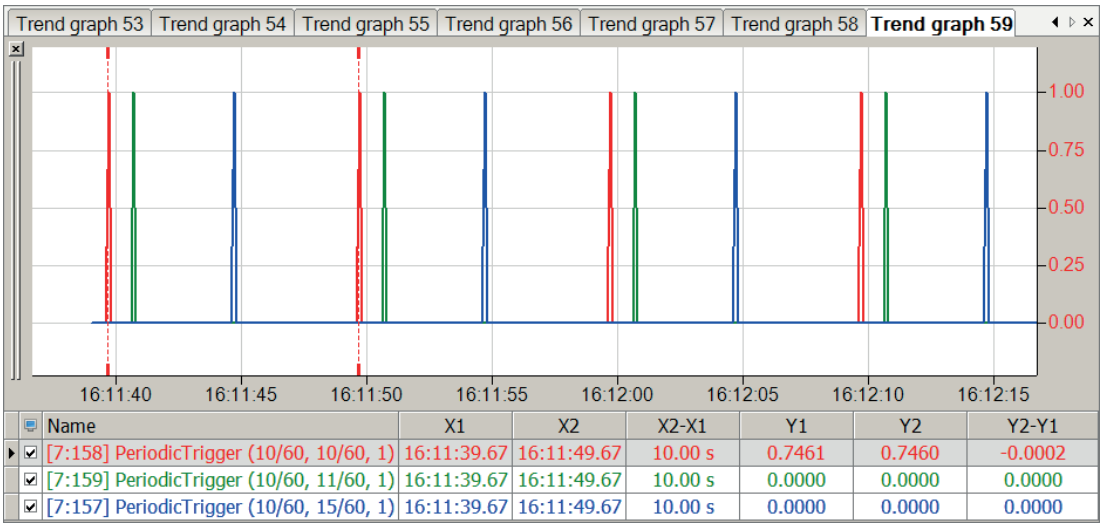
Solution

Display of the following functions:

- PeriodicTrigger(10/60, 10/60, 1)
- PeriodicTrigger(10/60, 11/60, 1)
- PeriodicTrigger(10/60, 15/60, 1)

with 10/60 min (corresponds to 10 s) for 'Interval,' again 10/60 min or 11/60 min (11 s) and 15/60 min (15 s) for 'StartTime' and 1 (system time) for 'UseSystemTime.'

As the following illustration shows, the triggers every 10 s differ in their start time. According to the calculation method, the trigger starts the interval of the red chart at time 0 s, the green chart at 1 s and the blue chart at 5 s after the acquisition.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[7:158] PeriodicTrigger (10/60, 10/60, 1)	16:11:39.67	16:11:49.67	10.00 s	0.7461	0.7460	-0.0002
[7:159] PeriodicTrigger (10/60, 11/60, 1)	16:11:39.67	16:11:49.67	10.00 s	0.0000	0.0000	0.0000
[7:157] PeriodicTrigger (10/60, 15/60, 1)	16:11:39.67	16:11:49.67	10.00 s	0.0000	0.0000	0.0000

Note

Negative values for 'StartTime' are invalid and result in an error message.

2.6.2 TriggerChangeRate

```
TriggerChangeRate('Expression','DeltaY*','DeltaT*','DeadTime*')
```

Arguments

'Expression'	Measured value
'DeltaY*'	Required value distance for releasing the trigger
'DeltaT*'	Time interval considered; limit value = 1,000,000 × module timebase
'DeadTime*'	Time specification in seconds, for which the trigger condition must be fulfilled until release. If no delay is desired, the value 0 must be entered.

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

The function returns TRUE as long as the change of the measured value 'Expression' (dy) within the interval 'DeltaT' is greater than 'DeltaY'.

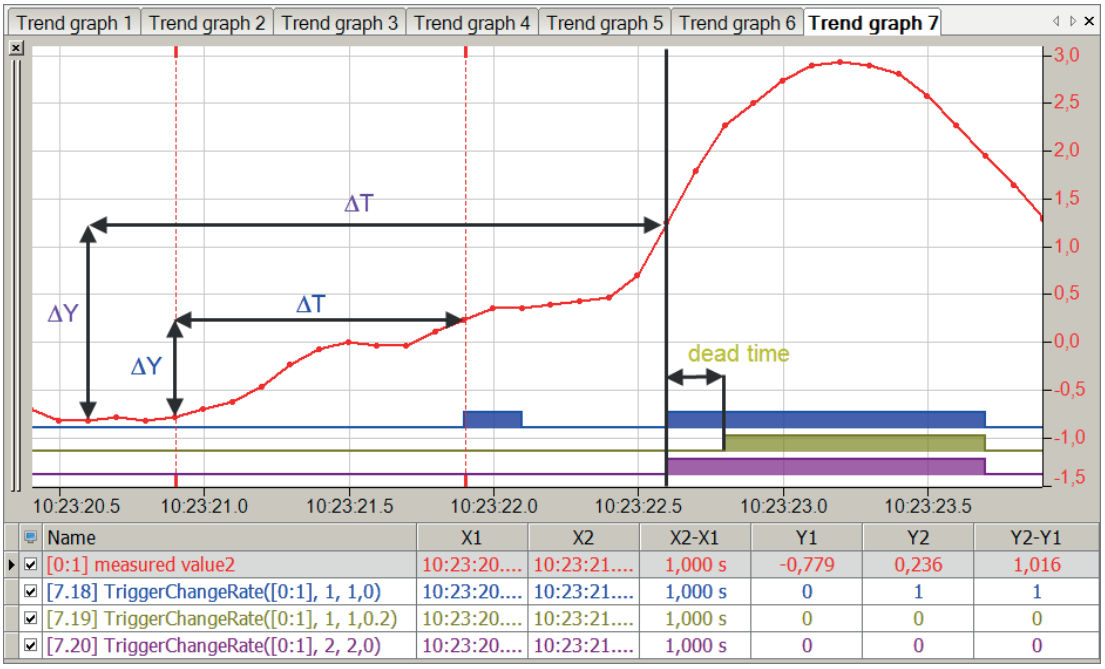
Note

The function compares the distance of the values with the time difference between DeltaT and DeltaY; the slope of the curve can also vary between the values. The time difference to be considered DeltaT is limited to 1-millionfold the module timebase.

Example

Differences in trigger initiation and impact of the 'DeadTime'

Solution



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:1] measured value2	10:23:20....	10:23:21....	1,000 s	-0,779	0,236	1,016
[7.18] TriggerChangeRate([0:1], 1, 1,0)	10:23:20....	10:23:21....	1,000 s	0	1	1
[7.19] TriggerChangeRate([0:1], 1, 1,0.2)	10:23:20....	10:23:21....	1,000 s	0	0	0
[7.20] TriggerChangeRate([0:1], 2, 2,0)	10:23:20....	10:23:21....	1,000 s	0	0	0

Red	Measured value	Blue	Trigger release with ΔY=1 and ΔT=1
Yellow	Trigger release with ΔY=1 and ΔT=1	Purple	Trigger release with ΔY=2 and ΔT=2

2.6.3 TriggerConstant

```
TriggerConstant('Expression','Level*','Epsilon*','DeadTime*')
```

Arguments

'Expression'	Measured value
'Level*'	Value specifying the center line of the area in which the trigger should release
'Epsilon*'	Value specifying the distance of both area boundaries for the center line
'DeadTime*'	Time specification in seconds, for which the trigger condition must be fulfilled until release. If no delay is desired, the value 0 must be entered.

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

The function returns TRUE as long as 'Expression' remains within the area ['Level' - 'Epsilon', 'Level' + 'Epsilon'] for at least the duration of the 'DeadTime'.

Note

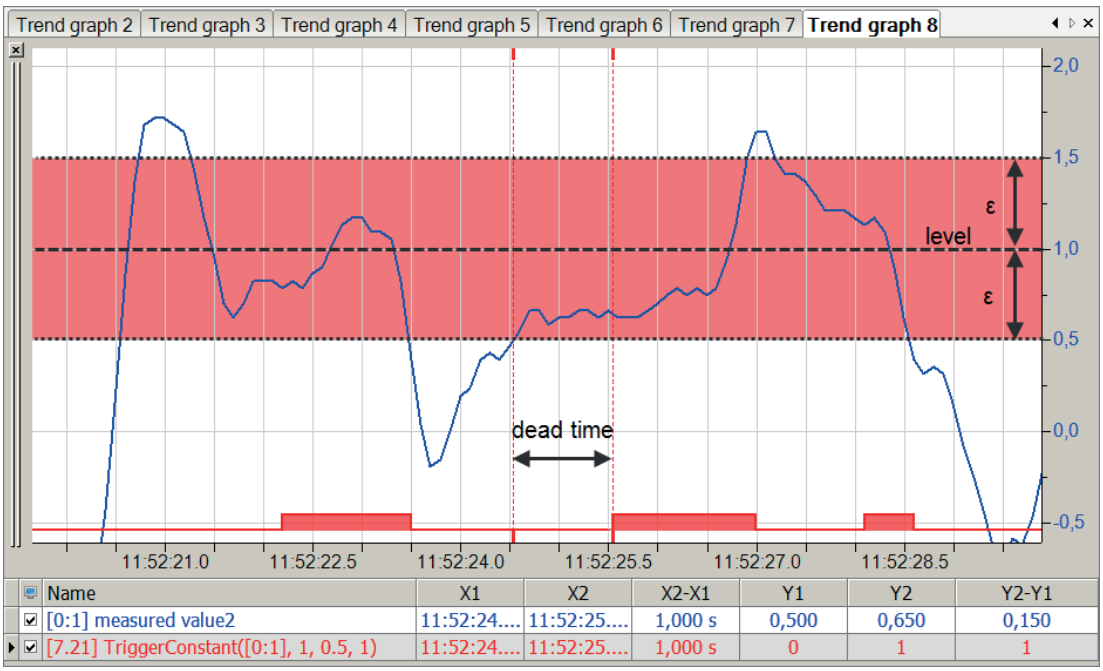
The trigger signal is emitted (true) for the entire duration for which the level is exceeded or undershot.

Example

The trigger should be released if the measured value is longer than one second in a range between 0.5 and 1.5.

Solution

In the figure below the blue curve shows the original signal and the red bar shows the released trigger.



2.6.4 TriggerEdge

```
TriggerEdge('Expression','Level*','EdgeType*','DeadTime*')
```

Arguments

'Expression'	Measured value	
'Level*'	Specification of the level value	
'EdgeType*'	Specification of whether rising, falling or both edges, i.e. crossings of the level value are counted	
	'EdgeType' < 0	only falling edges (crossing in the negative direction)
	'EdgeType' > 0	only rising edges (crossing in the positive direction)
	'EdgeType' = 0	falling and rising edges
'DeadTime*'	Time specified in seconds for which the measured value must exceed or fall below the level value in order to release the trigger.	

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

Triggers if 'Expression' exceeds or falls below 'Level' and stays on the same 'Level' side for at least DeadTime' seconds. If 'Expression' is a digital signal, 'Level' is fixed at 0.5. 'EdgeType' determines which edges or crossing are counted:

Note

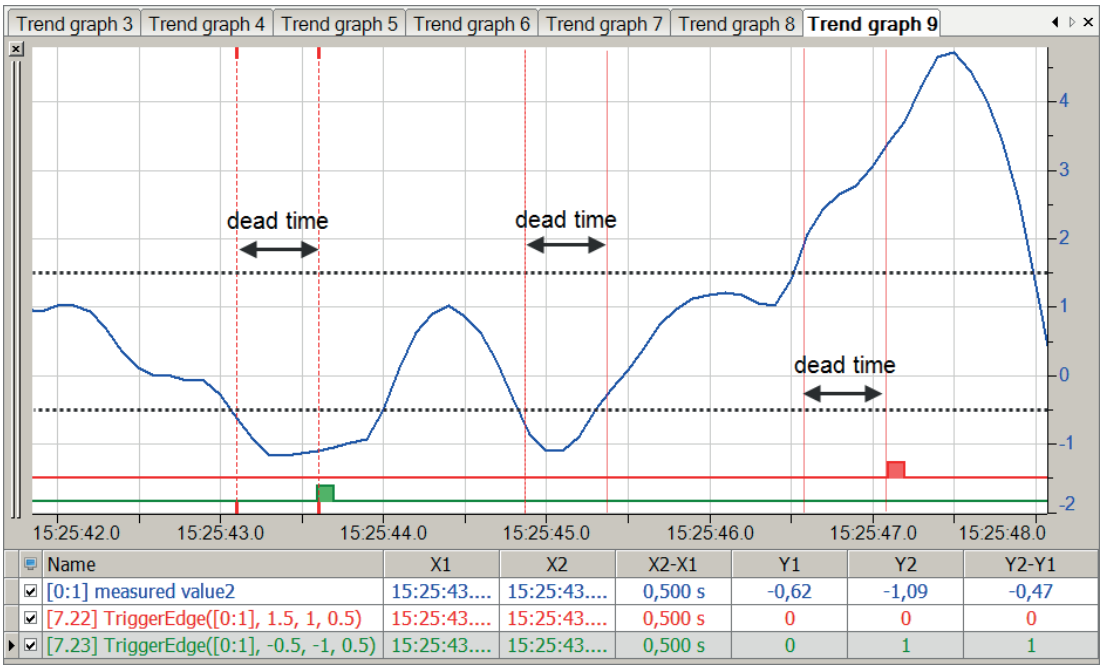
Unlike the TriggerLevel and TriggerConstant functions, only a pulse is emitted when the level value is crossed.

Example

A trigger should be released in each case where the measured value is above 1.5 or below -0.5 for 0.5 seconds.

Solution

In the figure below the blue curve shows the measured value and the red bar shows the trigger if measured value is longer than 0.5 seconds above 1.5. The green bar shows the trigger if measured value is longer than 0.5 seconds below -0.5.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:1] measured value2	15:25:43....	15:25:43....	0,500 s	-0,62	-1,09	-0,47
[7.22] TriggerEdge([0:1], 1.5, 1, 0.5)	15:25:43....	15:25:43....	0,500 s	0	0	0
[7.23] TriggerEdge([0:1], -0.5, -1, 0.5)	15:25:43....	15:25:43....	0,500 s	0	1	1

2.6.5 TriggerLevel

```
TriggerLevel('Expression','Level*','LevelType*','DeadTime*')
```

Arguments

'Expression'	Measured value	
'Level*'	Specification of the level value	
'LevelType*'	Specification of which side of 'Level' is considered	
	'LevelType' =0	below level
	'LevelType' = 1	above level
'DeadTime*'	Time specified in seconds for which the measured value must remain on the side considered of the level value in order to initiate the trigger.	

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

Triggers when 'Expression' remains above or below 'Level' for at least 'DeadTime' seconds. 'LevelType' determines which 'level' side is monitored.

Note

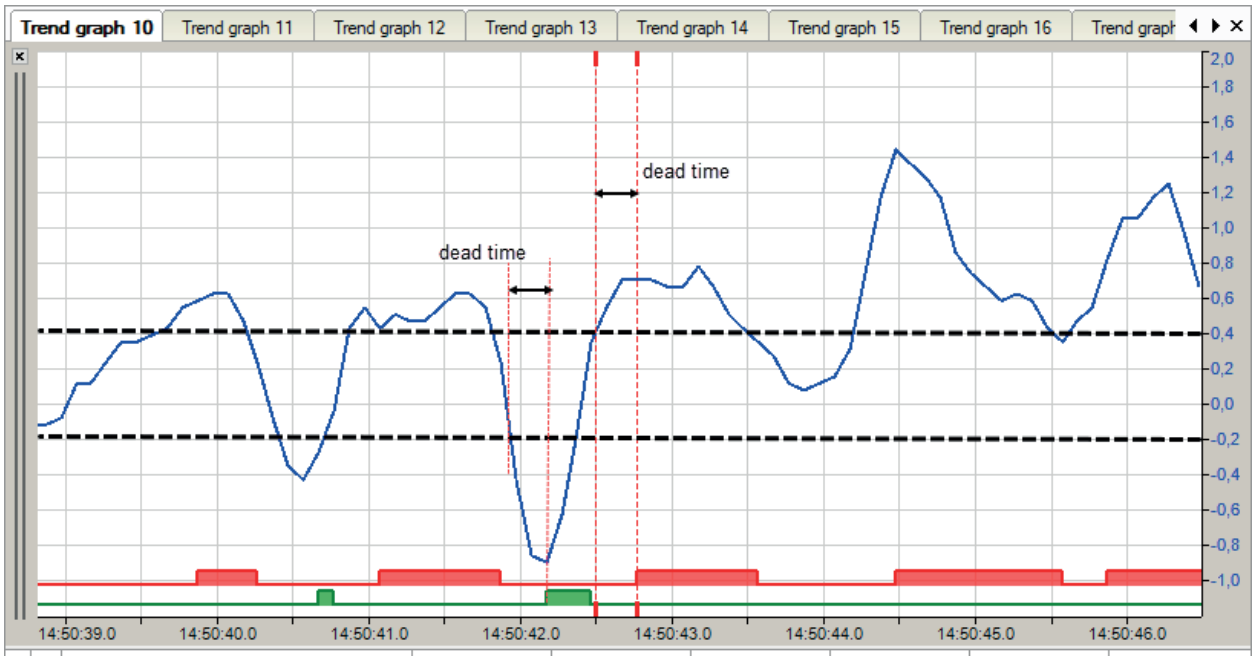
Like for the TriggerConstant function, the trigger signal remains true until the value crosses 'Level' on the other side.

Example

A trigger should be released in each case where the measured value is above 0.8 and below -0.2 for 0.4 seconds. A trigger pulse should be emitted periodically for the entire duration of the exceeding or undershooting.

Solution

In the figure below the blue curve shows the measured value and the red bar shows the trigger. The trigger is true, beginning at 0.4 seconds after the measured value rose above 0.4 and it returns to false when the value dropped below 0.4. The green bar shows the trigger when the measured value falls below -0.2. The trigger is true 0.4 seconds after the value dropped below -0.2, and it is false agawn hen the value is >-0.2.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:1] measured value2	14:50:42.500	14:50:42.770	0,270 s	0,400	0,705	0,304
[7.24] TriggerLevel([0:1], 0.4, 1, 0.2)	14:50:42.500	14:50:42.770	0,270 s	0	0	0
[7.29] TriggerLevel([0:1], - 0.1, 0, 0.2)	14:50:42.500	14:50:42.770	0,270 s	0	0	0

2.6.6 TriggerHarmonicLevel

```
TriggerHarmonicLevel('Expression',"'LimitProfile*'",'Harmonic*')
```

Arguments

'Expression'	Measuring signal
"'LimitProfile*'"	Name of the limit profile defined in the PQU
'Harmonic*'	Order (harmonic), the limit value of which is to be monitored, values 0 to 50

Parameters ending with * are only evaluated once at the start of the acquisition.

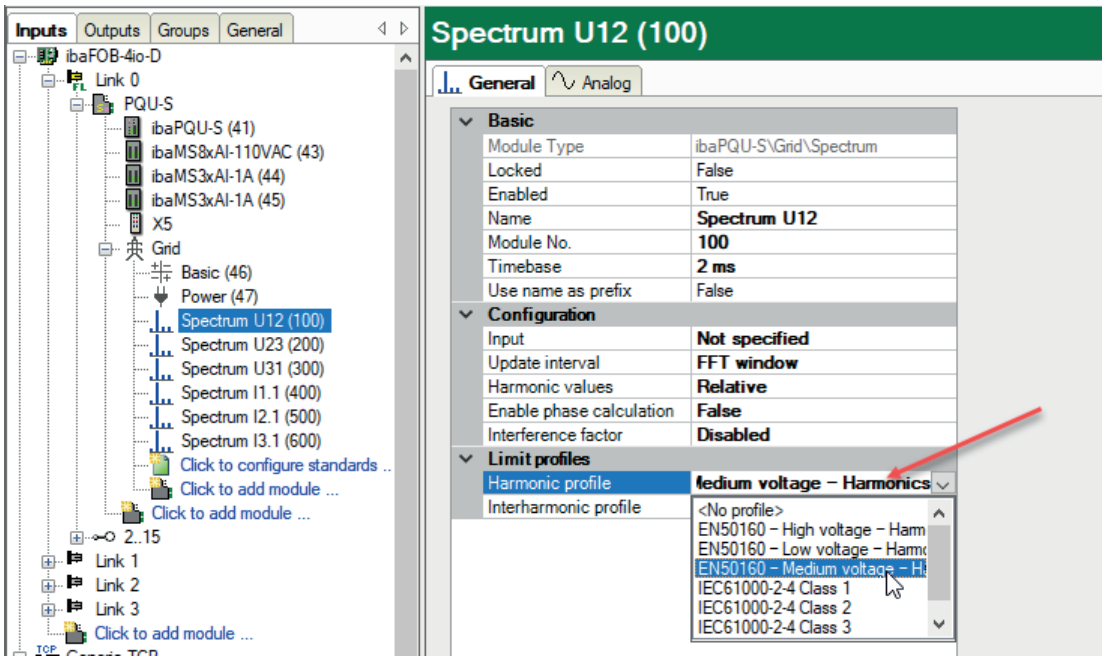
Description

Trigger fires (result true) when the signal value is above the harmonic limit defined in the "LimitProfile" profile of PQU. The "Harmonic" parameter determines which limit value is used from the limit profile.

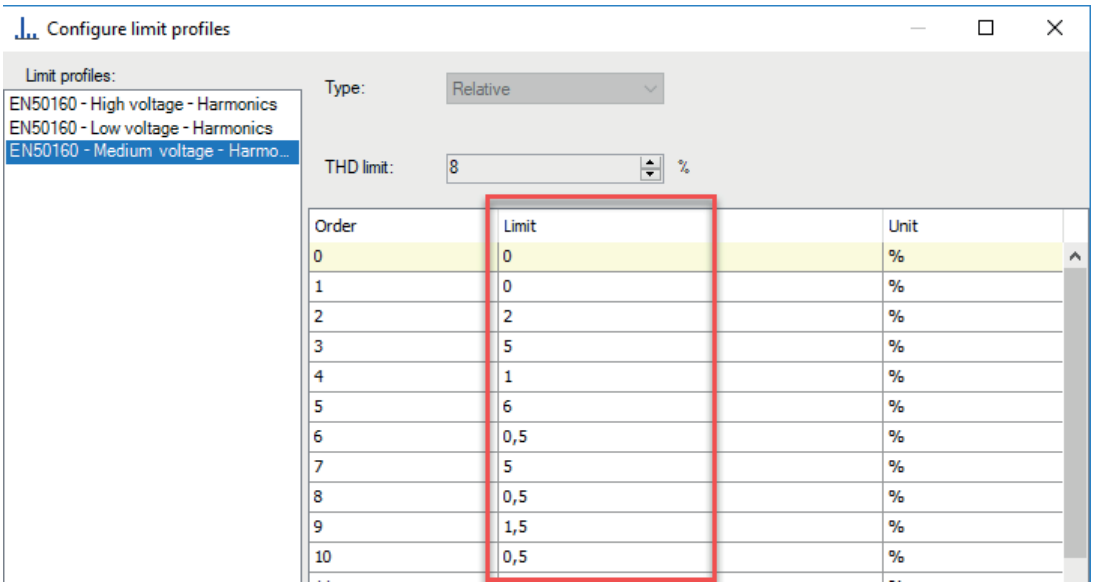
Example

When measured on medium voltage switchgear, the U12 conductor voltage should be monitored for its harmonic content. A trigger signal should be triggered if the limit value specified in EN50160 for the third harmonic in a 10 minute interval is exceeded.

The limit profile is configured in the I/O Manager in the spectrum modules. The profile name is "EN50160 – Medium voltage – Harmonics".



The limit values can be viewed with *Configure profiles*. In our example, the limit for the third harmonic is 5 %.



Solution

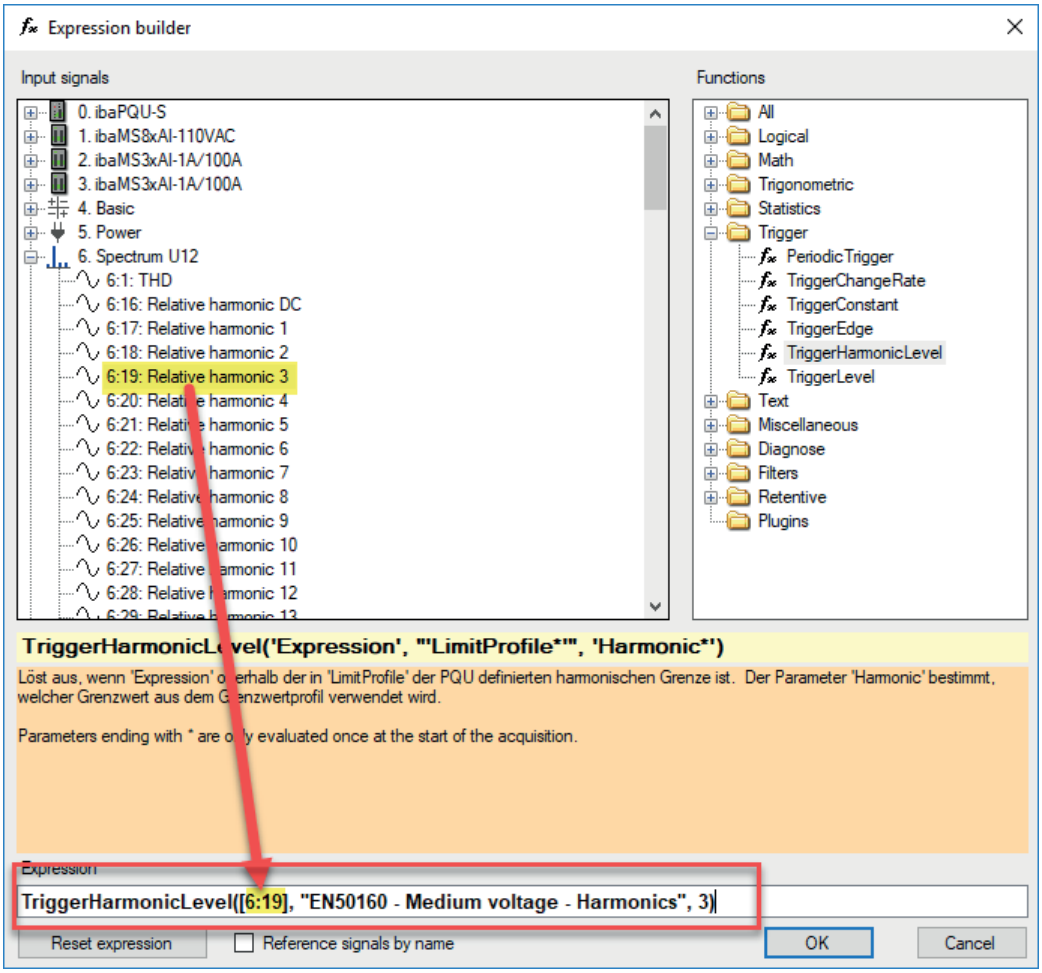


Fig. 1: Configuration of the TriggerHarmonicLevel expression

The trigger fires if the 10 min value of the third harmonic of U12 is above 5 %.

2.7 Text functions

2.7.1 CharsToText

```
CharsToText ('Encoding'*, 'Value1', 'Value2', ...)
```

Arguments

'Encoding*'	Character encoding, specification of the name or code page ID	
	'Encoding*' = "ASCII" 'Encoding*' = 1252	ASCII character encoding as plain text (in quotation marks!) or Code page ID for ASCII
	'Encoding*' = "UTF-8" 'Encoding*' = 65001	UTF-8 character encoding as plain text (in quotation marks!) or Codepage ID for UTF-8
	'Encoding*' = "UTF-16" 'Encoding*' = 1200	UTF-16 character encoding as plain text (in quotation marks!) or Codepage ID for UTF-16
'Value1'... 'Value*n*'	The decimal character code for each character according to the selected character encoding; Entries as numerical values or as analog signals	

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function assembles text from individual character codes and outputs it (text signal). The individual character codes 'Value1' through 'Value*n*' are passed to the function as numeric values based on a configurable encoding. The encodings ASCII, UTF-8 and UTF-16 as well as other valid codepage ids are supported. If a code page is not supported by the system, this will be indicated in the expression builder.

Each 'value' must correspond to a numeric value with a byte size that matches the minimum byte size of the encoding.

In ASCII and UTF-8, for example, each "value" corresponds to a single byte. In UTF-16, each 'value' corresponds to a 2-byte integer.

The number of characters to be concatenated in a string is limited to 999. Decimal codes are preferred, but hexadecimal codes are also supported (prefixed with 0x).

Generating readable text from sources which can only process numerical values is made easier by this function.

It makes sense to use analog signals to transmit the decimal character codes so that the characters can be changed during data acquisition.

Examples

For clarity, the character codes are entered here as numerical values. In the application, an analog signal can be assigned to each numeric code, which is used, for example, to transmit the code from the PLC. A combination of analog signals and hard-coded numerical codes is also possible)

Expression	Result	Comment
CharsToText("ASCII", 105, 98, 97, 32, 65, 71)	iba AG	ASCII encoding, decimal
CharsToText(1252,105,98,97,32,65,71)	iba AG	Code page 1252, decimal
CharsToText("ASCII", 65, 66, 67)	ABC	ASCII encoding, decimal
CharsToText("UTF-8", 65, 66, 67)	ABC	UTF-8 encoding, decimal
CharsToText("UTF-8", 0xE2, 0x82, 0xAC)	€	Euro symbol, UTF-8 encoding, hexadecimal
CharsToText("UTF-8", 226, 130, 172)	€	Euro symbol, UTF-8 encoding, decimal
CharsToText("UTF-16", 8364)	€	Euro symbol, UTF-16 encoding, decimal
CharsToText(869, 213, 226)	Ωθ	Code page 869 (Greek alphabet), decimal
CharsToText(869, 0xD5, 0xE2)	Ωθ	Code page 869 (Greek alphabet), hexadecimal

2.7.2 CharValue

```
CharValue('Text', 'CharNumber=0')
```

Arguments

'Text'	Text signal	
'CharNumber'	Position of the character in the text, default = 0	

Description

This function returns the ASCII decimal value of the character at the 'CharNumber' position in a text. The default provides the first character (position = 0).

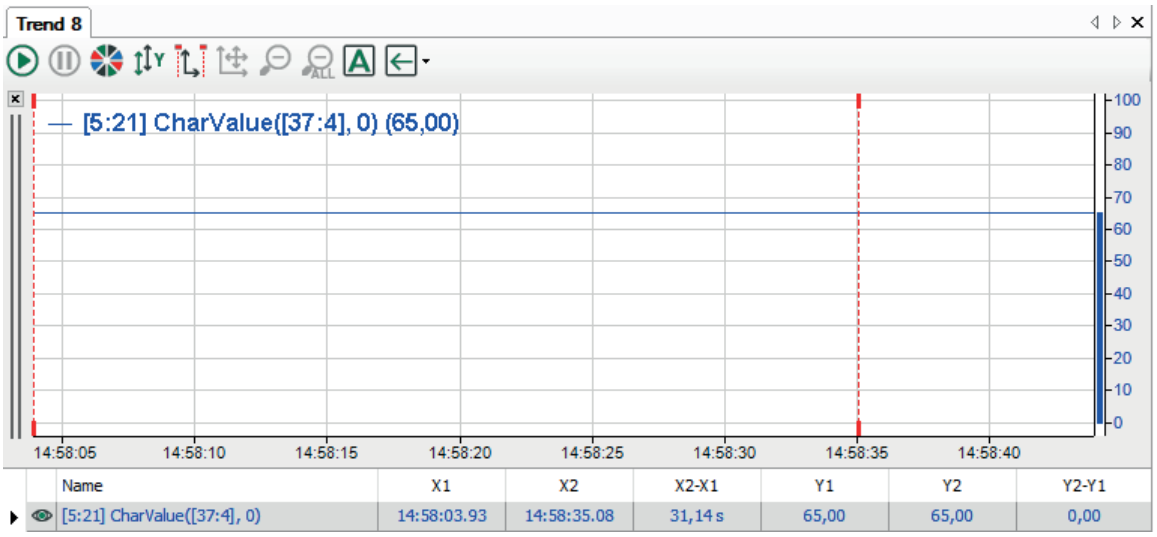
Example: If 'text' has the value "A_text for test", then the result of CharValue('Text', 0) is the ASCII value 65 for the uppercase A.

Example

Presenting the example referred to in the description

Solution

Query of the ASCII decimal value of the first character in the text "A_Text for test". A has the value 65 in the ASCII table.



Tip

See ASCII table under http://www.ascii-code.com/.

2.7.3 CompareText

```
CompareText("'Text1'", "'Text2'", ' CaseSensitive=1*')
```

Comment:
The original name of this function was TextCompare. It was later renamed as CompareText. For compatibility reasons, the TextCompare function can still be used. The arguments are identical.

Arguments

'Text1'	Specification of the first comparison text	
'Text2'	Specification of the second comparison text	
'CaseSensitive'*	Optional parameter (default = 1) for determining whether a case sensitive comparison (consideration of upper and lower case) should be made	
	'CaseSensitive' ≠ 0	Take upper and lower case into account
	'CaseSensitive' = 0	Ignore upper and lower case

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function compares 2 texts alphabetically. This means that the characters of the 2 strings are compared individually starting with the left character. The function uses the current culture setting (Windows) to obtain culture-specific information such as spelling and alphabetical order. The comparison does not consider multi-digit values, word meaning or the length of the string. Blanks are taken into account in the comparison. With the optional parameter 'CaseSensitive=1', it can be determined whether upper and lower case should be considered (=1 or not specified), or not (<>1).

Dynamic text can be used by selecting a text signal. A text signal is set in square brackets []. It is also possible to enter constant text by just typing the text between the double quotes.

The function returns an analog value as the result.

Results

-1	if the characters of the first text appear in the alphabet before the characters of the second text.
0	if the two texts are equal
1	if the characters of the first text appear in the alphabet after the characters of the second text.

Example

The following table shows a couple of examples:

Text1	Text2	Result		Comment
		CompareText ("Text1","Text2",0)	CompareText ("Text1","-Text2",1)	
1234 abcd	1234 abcd	0	0	1 = 2
1234 abcd	1234 bcde	-1	-1	Text1 < Text2 “a” comes before “b”
1234 Abcd	1234 abcd	0	1	Text1 = Text2 (not case sensitive) Text1 > Text2 (case sensitive) “A” comes after “a”
12340 abcd	1234 ␣ abcd	1	1	Text1 > Text2 "0" comes after " ␣"
1234 0abcd	1234 abcd	-1	-1	Text1 < Text2 “0” comes before “a”
12034 abcd	1234 abcd	-1	-1	Text1 < Text2 “0” comes before “3”
1234 abcd	1y34 abcd	-1	-1	Text1 < Text2 “2” comes before “y”
1z34 abcd	1Y34 abcd	1	1	Text1 > Text2 “z” comes after “Y”

2.7.4 ConcatText

```
ConcatText('Text1', Text2', …)
```

Arguments

'Text*n*'	Text signal

Description

This function returns the concatenation of 'Text1', 'Text2' etc. as its result.

If you want to use double quotes in static text, then write two double quotes after each other.

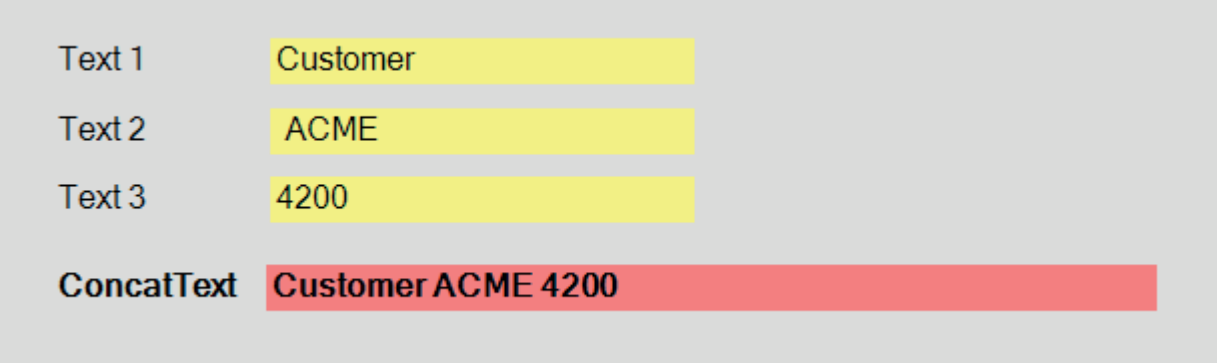
Example

The values of tree text signals should be concatenated.

Text signals are [37:0], [37:1] und [37:2].

Solution

```
ConcatText([37:0],[37:1],[37:2])
```



2.7.5 ConvertFromText

```
ConvertFromText('Expression', 'DecimalPoint*=0', 'Begin=0', 'End'=-1 (end of
text)', 'Base*=10', 'Bitsize*=32')
```

Arguments

'Expression'	Name of the text signal	
'DecimalSeparator*'	Decimal point	
	DecimalSeparator = 0 DecimalSeparator = 1	Period Comma
'Begin'	Index of the first character of the desired text, default = 0; if <0, it will be internally converted to 0.	
'End'	Index of the first character of the desired text, default = -1 (end of the text)	
'Base*'	Expected numerical base of the number in the text	
	'Base=2' 'Base=8' 'Base=10' (default) 'Base=16'	binary octal decimal hexadecimal
'Bitsize*'	Number of bits, the parsed number should have	Typ. values: 32 (default), 64, 16, 8

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

The function parses a floating-point number from the text and delivers the numeric value as an decimal number. 0 is output if there is no number in the first position of the range defined by the 'Begin' and 'End' arguments. Exception: if there are only spaces present up to the first numeric character. Otherwise the text is read up to the first non-numerical position or maximum to the 'end'. Leading zeros before a number must not to be interrupted by spaces or non-numerical characters.

By the argument 'DecimalSeparator' you determine the decimal point. This is only relevant, if the number in the text is already a floating-point number ('Base=10'). 'DecimalSeparator' is ignored, if the argument 'Base' is not 10.

The 'Base' argument determines the numerical base expected in the text input. Possible values are 2 (binary), 8 (octal), 10 (decimal) and 16 (hexadecimal). While parsing the text, the function interprets the number in the text as a number with the numerical base 'Base' and converts it to the corresponding decimal number. If the number in the text is a decimal number you can omit the 'Base' argument. If, for instance, the number in the text is a hexadecimal number, then enter 16 for 'Base'. The result for the number FF in the text will be 255.

The “Bitsize” argument is only relevant if the 'Base' argument is set to something different than 10. It determines how many bits the parsed number should have. This is necessary for correctly parsing numbers that are represented in two’s complement. For example, parsing the number

0xFFFF results in -1 when using 'Bitsize=16', but it results in 65535 when a larger Bitsize (e.g. 32) is used.

ConvertFromText with a 'Base' value different than 10 internally always works on 64-bit signed integers, so signal values larger than $2^{63} - 1$ are not correctly converted. Also note, that 'Bitsize' is limited to 64 and consequently only 64 bit signed integers can be parsed.

Example 1: Converting a decimal number

Reading in a defined text

Solution

The function is: ConvertFrom Text('Text', 0,22,-1)

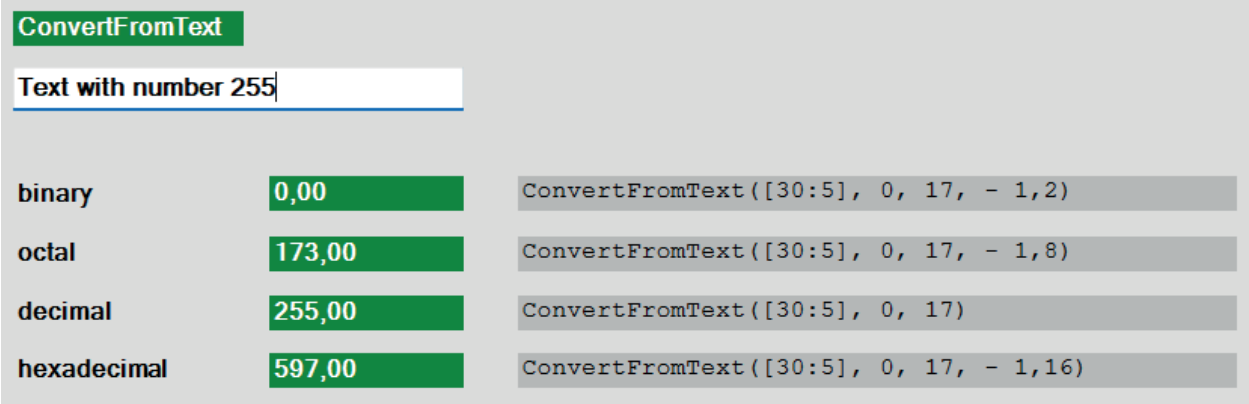
The content of the text signal is: `Voltage target value: 6.9 V`

The voltage value starts with position 22. -1 is used as the end index so that values with more preceding and/or following commas can also be acquired.

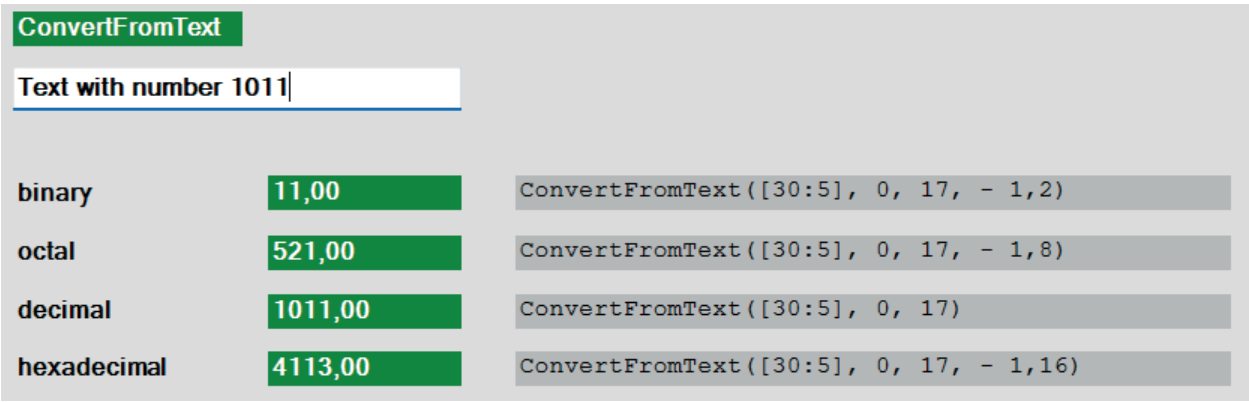
Example 2: Converting with numerical base

Parsing and converting a number from a text, taking into consideration the numerical base

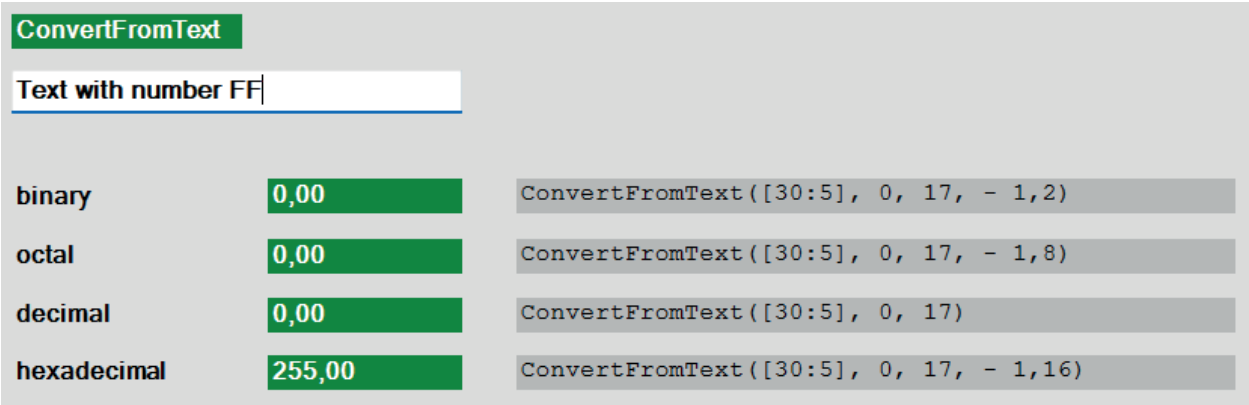
The text line in the following examples contains a number beginning at position 17, which is converted by the function *ConvertFromText,* considering different numerical bases. The right column shows the syntax of the function. The resulting values (highlighted in green) are always decimal numbers.



The binary conversion has no result, because the numeric characters 2 and 5 do not exist in the binary system.



The binary conversion shows a result, because the numerical characters 0 and 1 do exist in the binary system.



The conversion for binary, octal and decimal systems shows no result because the character F does not exist in these numerical systems.

2.7.6 ConvertToText

```
ConvertToText('Expression','IntegerDigits*=1','FractionalDigits*='',Plus-
Sign*=2','DecimalSeparator*=0','Base*=10')
```

Arguments

'Expression'	Expression (floating point value), which should be converted into text	
'IntegerDigits*'	Minimum number of integer digits	
'FractionalDigits*'	Number of decimal digits	
'PlusSign*'	Representation of positive values (plus sign)	
	'PlusSign' = 0	Space
	'PlusSign' = 1	"+"
	'PlusSign' = 2	Nothing
	'PlusSign' = 3	Prefix for numbers with other bases than 10: 0x for hexadecimal numbers 0b for binary numbers 0o for octal numbers
'DecimalSeparator*'	Decimal point	
	'DecimalSeparator*'= 0	Dot
	'DecimalSeparator*'= 1	Comma
'Base*'	Numerical base of the resulting number	
	'Base=2'	binary
	'Base=8'	octal
	'Base=10' (default)	decimal
	'Base=16'	hexadecimal

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns a floating-point value as text. You can specify the minimum number of integer digits with the argument 'IntegerDigits'. By means of the argument 'FractionalDigits' you specify the number of decimal digits. If 'FractionalDigits' is smaller than zero, then only non-zero digits will be printed. If 'FractionalDigits' is larger than zero, then zeros are printed. The parameter 'PlusSign' determines the printing of the plus sign.

The 'Base' argument determines the numerical base which should be used for the representation of the converted number in the text. You can choose between the values 2 (binary), 8 (octal), 10 (decimal, default) or 16 (hexadecimal).

If the 'Base' argument has a different value than 10, it has an effect on other arguments:

- The 'IntegerDigits' argument determines the exact number of digits instead of the minimal number of digits like for decimal values.
- The 'FractionalDigits' and 'DecimalSeparator' arguments are ignored entirely. The behavior here is to round the number and therefore remove the fractional part.

You may use the 'PlusSign' argument to add a prefix to the resulting number, corresponding to its numerical base, e.g. "0x" for hexadecimal values.

Note

When converting negative numbers to text, two's complement representation is used. This uses 64 bits of precision, so the resulting strings become very large. To limit the length of the string, the 'IntegerDigits' argument is used. It is important to note, that excess digits are simply cut off.

Example 1: Conversion of a floating-point value to text

Examples for the floating-point value [FloatValue] = 42.471130 with different argument settings

The 'Base' argument is not mentioned because the base is 10, which is the default value.

'Expression'	'Expression'	'IntegerDigits'	'FractionalDigits'	'PlusSign'	'DecimalPoint'
	[FloatValue] Value: 42.471130	1	4	2	1
Formula	`ConvertToText([FloatValue],1,4,2,1)`				
Result	42.4711				

'Expression'	'Expression'	'IntegerDigits'	'FractionalDigits'	'PlusSign'	'DecimalPoint'
	[FloatValue] Value: 42.471130	1	6	1	1
Formula	`ConvertToText([FloatValue],1,6,1,1)`				
Result	+42.471130				

'Expression'	'Expression'	'IntegerDigits'	'FractionalDigits'	'PlusSign'	'DecimalPoint'
	[FloatValue] Value: 42.471130	1	-6	0	1
Formula	`ConvertToText([FloatValue],1,-6,0,1)`				
Result	42.47113				

Example 2 Conversion of decimal values to text with consideration of the numerical base

The 'PlusSign' argument is set to 3 in order to indicate the numerical base of the resulting number by a prefix.

Base 2 (binary)

Formula	`ConvertToText([DecimalValue],8,-2,3,0,2)`					
[DecimalValue]	'IntegerDigits'	'FractionalDigits'	'PlusSign'	'DecimalSeparator'	'Base'	Result
9.00	8	-2	3	1	2	**0b00001001**
16.00	8	-2	3	1	2	**0b00010000**
255	8	-2	3	1	2	**0b11111111**

Base 8 (octal)

Formula	`ConvertToText([DecimalValue],6,-2,3,0,8)`					
[DecimalValue]	'IntegerDigits'	'FractionalDigits'	'PlusSign'	'DecimalSeparator'	'Base'	Result
9.00	6	-2	3	1	8	**0o000011**
16.00	6	-2	3	1	8	**0o000020**
255	6	-2	3	1	8	**0o000377**

Base 16 (hexadecimal)

Formula	`ConvertToText([DecimalValue],4,-2,3,0,16)`					
[DecimalValue]	'IntegerDigits'	'FractionalDigits'	'PlusSign'	'DecimalSeparator'	'Base'	Result
9.00	4	-2	3	1	16	**0x0009**
16.00	4	-2	3	1	16	**0x0010**
25.00	4	-2	3	1	16	**0x00FF**

2.7.7 CountText

```
CounText('Text', 'CountOnlyDifferent*=0', 'Reset=0')
```

Arguments

'Text'	Name of the text signal	
'CountOnlyDifferent*'	Optional parameter (default = 0)	
	'CountOnlyDifferent' <> 0	New text is only counted if it is different from the preceding text.
	'CountOnlyDifferent' = 0	Each new text is counted.
'Reset'	Optional parameter that can be used to reset the counter reading. 'Reset' can be an expression as well.	
	'Reset' > 0	Counter is reset.
	'Reset' = 0	Counting released (default)

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function counts the reception or the change of a text signal and returns the counter reading.

2.7.8 DeleteText

```
DeleteText('Text','StartPos','Lenght=100000')
```

Arguments

'Text'	Text signal or "text" where characters should be deleted
'StartPos'	Analog signal or number; specifies the start position for the deletion in the text. Counting starts at zero (0 = 1st character).
'Length'	Analog signal or number; specifies the number of characters to be deleted. Default=100000

Description

This function deletes 'Length' characters from a text beginning with the character at position 'StartPos'. If 'Length' is not specified, all characters from 'StartPos' until the end will be deleted.

Example

Starting at position 4, four characters should be deleted from a text.

Arguments	'Text'	'StartPos'	'Length'
	[TextSignal] Value: ABCDE1234XYZ	4	4
Formula	`DeleteText([TextSignal],4,4)`		
Result	ABCD4XYZ		

2.7.9 FindText

```
FindText("'Text1'", "'Text2'", ' CaseSensitive=1*')
```

Arguments

'Text1'	Specification of the first comparison text	
'Text2'	Specification of the second comparison text	
'CaseSensitive'*	Optional parameter (default = 1) for determining whether a case sensitive comparison (consideration of upper and lower case) should be made	
	'CaseSensitive' <> 0	Take upper and lower case into account
	'CaseSensitive' = 0	Ignore upper and lower case

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function checks whether 'Text2' is contained in 'Text1' and returns the index as a result (position in 'Text1, first character = index 0) where 'Text2' was found for the first time. Blanks are considered in the search. With the optional parameter 'CaseSensitive=1', it can be determined whether upper and lower case should be considered (=1 or not specified), or not (<>1).

Dynamic text can be used by selecting a text signal. A text signal is set in square brackets []. It is also possible to enter constant text by just typing the text between the double quotes.

The function returns an analog value as the result.

Results

-1	if the second text is not included in the first text
0....n	Position where 'Text2' is in 'Text1' the first time

Example

The following table shows a couple of examples:

Text1	Text2	Result		Comment
		FindText ("Text1","Text2",0)	FindText ("Text1","Text2",1)	
The Sun is shining.	-1	4	4	
The moon is shining	-1	-1	-1	
The sun is shining	-1	4	-1	

2.7.10 InsertText

```
InsertText('Text1','Text2','Pos')
```

Arguments

'Text1'	Text signal or "text" where another text should be inserted	
'Text2'	Text signal or "text" whose text should be inserted into 'Text1'	
'Pos'	Analog signal or number; specifies the position where the text should be inserted.	
	'Pos' ≤ 0	'Text2' is prepended to 'Text1'.
	'Pos' ≥ length of 'Text1'	'Text2' is appended to 'Text1'.

Description

This function inserts a 'Text2' into a 'Text1' at position 'Pos'.

Example

Arguments	'Text1'	'Text2'	'Pos'
	[TextSignal_1] Value: 123_XYZ	[TextSignal2] Value: abcd_	4
Formula	`InsertText([TextSignal1],[TextSignal2],4)`		
Result	123_abcd_XYZ		

2.7.11 MidText

```
MidText('Text','StartPos','Lenght=100000')
```

Arguments

'Text'	Text signal or "text" from where characters should be returned
'StartPos'	Analog signal or number; specifies the start position for the reading in the text. Counting starts at zero (0 = 1st character).
'Length'	Analog signal or number; specifies the number of characters to be read, Default=100000

Description

This function reads and returns 'Length' characters from a text beginning with the character at position 'StartPos'. If 'Length' is not specified, all characters from 'StartPos' until the end will be read.

Example

A standardized order number (text) contains always at the 6th position a four-digit customer code. This customer code should be read out.

Arguments	'Text'	'StartPos'	'Length'
	[TextSignal_Order] Value: 4711_ACME_1234XYZ	5	4
Formula	`MidText([TextSignal_Order],5,4)`		
Result	ACME		

2.7.12 ReplaceText

```
ReplaceText('Text','SearchText','ReplaceText')
```

Arguments

'Text'	Text signal or "text" with content to be replaced
'SearchText'	Text signal or "text" with the text to be replaced
'ReplaceText'	Text signal or "text" as replacement

Description

This function replaces all occurrences of 'SearchText' within 'Text' with 'ReplaceText'. All arguments can be text signals or static text. If you use static text in the formula, put it in double quotes. By using text signals, you can vary dynamically both the text to be replaced ('SearchText') and the replacement text ('ReplaceText'). Even the basic text ('Text'), which contains parts to be replaced, can be a text signal.

If you want to use double quotes in static text, then write two double quotes after each other.

Examples

The following table shows several simplyfied applications.

Arguments	'Text'	'SearchText'	'ReplaceText'
	[TextSignal] Value: AAA BBB CCC	"BBB"	[ReplaceTextSignal] Wert: XXX
Formula	`ReplaceText([TextSignal],"BBB",[ReplaceTextSignal])`		
Result	AAA XXX CCC		

Arguments	'Text'	'SearchText'	'ReplaceText'
	[TextSignal] Value: AAA BBB CCC	[SearchTextSignal] Value: BB CC	"XXX"
Formula	`ReplaceText([TextSignal],[SearchTextSignal],"XXX")`		
Result	AAA BXXXC		

Arguments	'Text'	'SearchText'	'ReplaceText'
	"Hello Name"	"Name"	[ReplaceTextSignal] Value: John
Formula	`ReplaceText("Hello Name","Name",[ReplaceTextSignal])`		
Result	Hello John		

2.7.13 TextLength

```
TextLength('Text')
```

Arguments

'Text'	Text signal or "text" whose length should be returned in number of characters

Description

This function returns the total number of characters of a text.

Example

Arguments	'Text'
	"How many characters are in this text?"
Formula	`TextLength("How many characters are in this text?")`
Result	37

2.7.14 TrimText

```
TrimText('Text', 'TrimOption=0')
```

Arguments

'Text'	Text signal to be processed	
'TrimOption'	Optional argument, specifying which space characters should be removed	
	'TrimOption' = 0	Default; remove leading and trailing space characters.
	'TrimOption' = 1	Remove only leading space characters.
	'TrimOption' = 2	Remove only trailing space characters.
	'TrimOption' = 3	Remove all space characters, including the internal space characters.

Description

This function removes space characters from a text. By using the argument 'TrimOption' you can control, which space characters are concerned.

Example

In different steps, the space characters should be removed from the following text of a text signal 'TextToTrim'. The pipe characters mark begin and end of the text including space, but they are not part of the text.

| Lorem ipsum dolor sit amet |

Solution

`TrimText('TextToTrim')`	|Lorem ipsum dolor sit amet|
`TrimText('TextToTrim',1)`	|Lorem ipsum dolor sit amet |
`TrimText('TextToTrim',2)`	| Lorem ipsum dolor sit amet|
`TrimText('TextToTrim',3)`	|Loremipsumdolorsitamet|

2.8 Miscellaneous functions

2.8.1 ClientInfo

```
ClientInfo('"ClientAddress*"','InfoType*')
```

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns information about the client, which is specified by 'ClientAddress'. The parameter 'ClientAddress' can be the IP address or the name of the client computer (case insensitive). This information may be used for controlling other functions or for display and diagnostic purposes.

Specify the information type ('InfoType') you want to receive.

The following information types are supported:

Information types	Possible results
0	0 = No connection to server 1 = Client and server are connected
1	Number of signals requested by the client
2	0 = Client does not use a client license 1 = Client uses a client license
3	0 = Client does not use a QPanel license 1 = Client uses a QPanel license

Table 2: Information types and possible results of the ClientInfo function

2.8.2 ClientInfoText

```
ClientInfoText('"ClientAddress*"','InfoType*')
```

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns textual information about the client, which is specified by 'ClientAddress'. The parameter 'ClientAddress' can be the IP address or the name of the client computer (case insensitive). This information may be used for controlling other functions or for display and diagnostic purposes.

Specify the information type ('InfoType') you want to receive.

The following information types are supported:

Information types	Possible results
0	Name of the client computer
1	IP address of the client computer
2	Windows user name of the client
3	Name of the logged in ibaPDA user
4	Connection starttime of the client
5	ibaPDA Version of the client

Table 3: Information types and possible results of the ClientInfoText function

2.8.3 Count

```
Count('Expression','Level*','Hysteresis*','EdgeType*', 'Reset=0')
```

Arguments

'Expression'	Measured value	
'Level*'	Specification of the level value	
'Hysteresis'	Specification of a hysteresis band	
'EdgeType*'	Indication of whether rising, falling or rising and falling edges should be counted	
	'EdgeType' <0	only falling edges (leaving out hysteresis band in the negative direction)
	'EdgeType' >0	only rising edges (leaving out hysteresis band in the positive direction)
	'EdgeType' = 0	falling and rising edges
'Reset'	Optional digital parameter that can be used to reset the counter. ‘Reset’ can be an expression as well. An analog signal signal can be used for reset too, but it's value must be exactly = 1 (integer).	
	'Reset' = 1 or True	Counter is reset; analog reset signal: value has to be integer
	'Reset' ≠ 1	Counter value is retained / continues to count (default)

Parameters ending with * are only evaluated once at the start of the acquisition.

Note

The 'Reset' condition must not be related to the count function itself.

Description

The function counts the crossings of 'Expression' through 'Level'.

The 'Hysteresis' parameter can be used to define a tolerance band which is above and below 'Level' by equal amounts. Only complete crossings through the tolerance band are counted.

The 'EdgeType' parameter determines which kind of edges are counted.

The 'Reset' parameter is used to reset the counter value to 0. 'Reset' can also be formulated as an expression.

Examples:

Count([0:0],10,1,1)	No reset happens ('Reset' omitted)
Count([0:0],10,1,1,If(Mod(T(),20)=0,TRUE()m,FALSE()))	The counter is reset in a time interval of 20 seconds.
Count([0:0], [3.1])	e.g. with [3.1] = If([0:0]<1, 1, 0) The counter is reset as soon as the expression [3.1] returns TRUE, i.e. if the expression [0:0] falls below the limit value 1.

Tip

The COUNT function can also be used for binary signals. For this purpose, choose 0.5 for Level and, for example, 0.1 for Hysteresis. This then means that all changes from FALSE to TRUE and vice versa are detected and counted.

Example

Tip

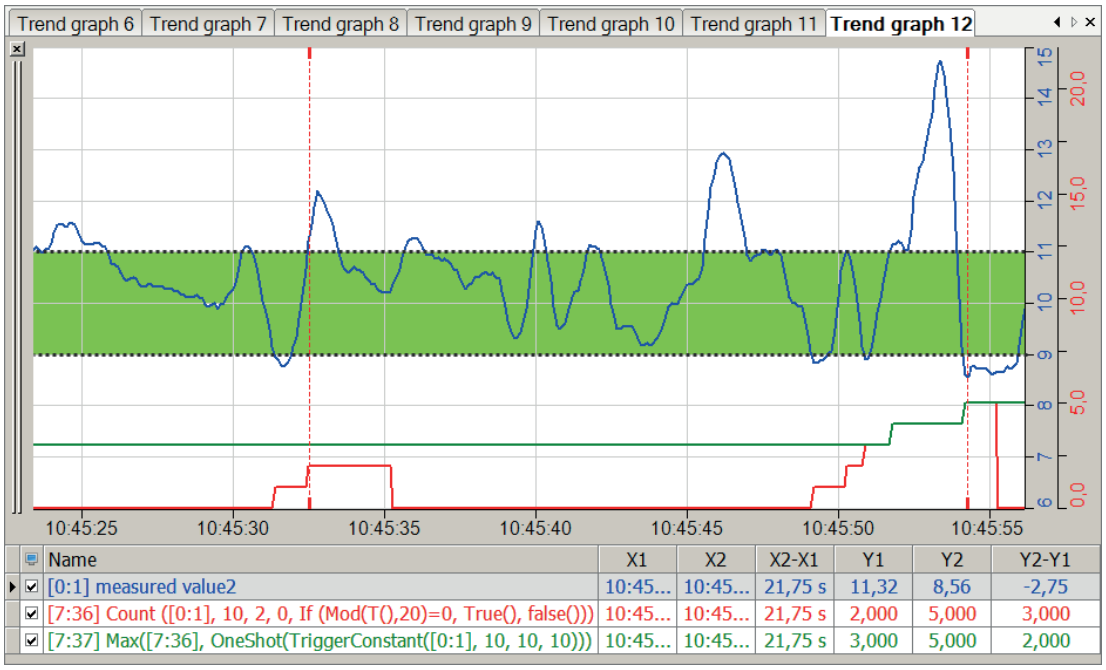
This function can be used in a virtual retentive module. Its result values can thus be obtained despite stopping and restarting the measurement.

The number of deviations of the measuring signal from the value of 10 in a 20 second interval in both directions should be counted, but only after complete crossings through the hysteresis band of width 2. The maximum value of each interval without the start phase should be saved.

Task description

The 'Reset' parameter of the count function consists of an If query of the modulo function of time. The maximum value is reset once via the OneShot function as soon as the measured value has leveled off at the beginning of the measurement.

Solution



Blue	Measured value	Green area	Hysteresis band
Red	Count function	Green signal	Maximum value of the intervals of the count function

Tip

When specifying a hysteresis band of 2 around 'Level' 10, the crossings in the increasing direction are first counted for 'Expression'>11 and in the descending direction for 'Expression'<9.

2.8.4 CountUpDown

```
CountUpDown('Up','Down','Reset','UpperLimit=none','LowerLimit=none','ResetOnLim-
it=0')
```

Arguments

'Up'	Digital signal whose rising edge (FALSE --> TRUE) increases the counter by 1	
'Down'	Digital signal whose falling edge (TRUE --> FALSE) decreases the counter by 1	
'Reset'	Optional parameter (default = 0) to stop, reset and restart the calculation	
	'Reset' = 0	Perform calculation
	'Reset' = 1	Stop calculation and set result to 0
'UpperLimit'	Optional parameter (default = 0); upper limit for counter value	
'LowerLimit'	Optional parameter (default = 0); lower limit for counter value	
'ResetOnLimit'	Optional parameter (default = 0) for resetting the counter value when reaching a limit	
	'ResetOnLim-it'= 0	Counter value stays on limit value after reaching a limit
	'ResetOnLim-it' = 1	Counter value is reset to 0 after reaching a limit

Description

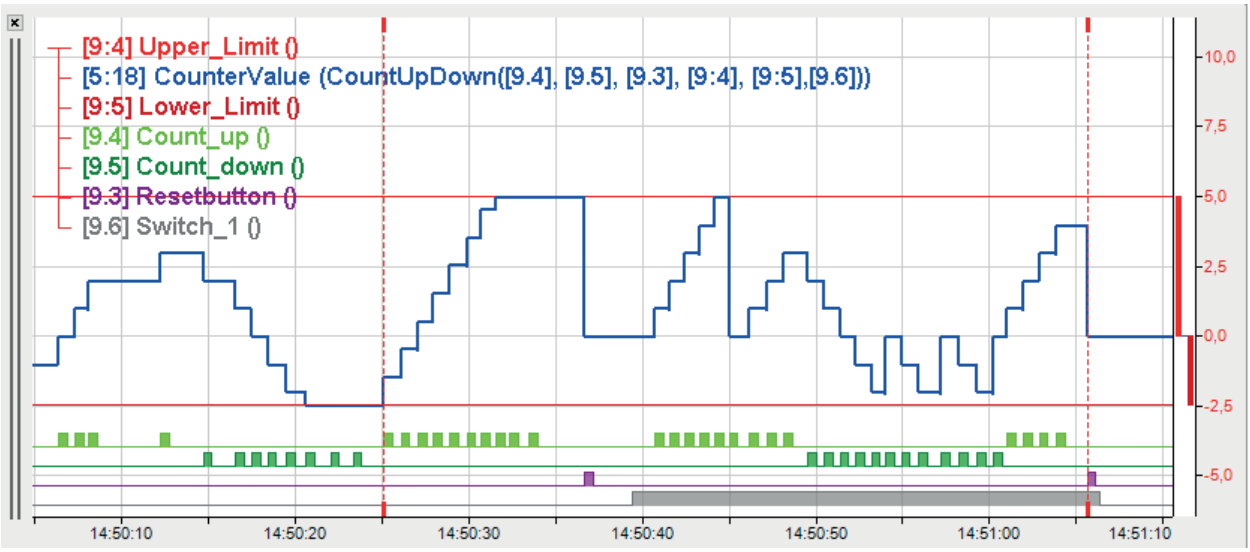
This function generates a counter value which is increased by 1 with every rising edge on 'Up' and decreased by 1 with every rising edge on 'Down'. Counting is not limited but can be reset to zero with 'Reset' = TRUE or 1.

'UpperLimit' and 'LowerLimit' are optional and can be specified either by a fix value or an expression. If limit arguments are specified and the counter reaches a limit, it does not continue counting but stays on the limit value until either a counter pulse in the other direction occurs or 'Reset' is set to TRUE.

If the optional parameter 'ResetOnLimit' is TRUE or 1, then the counter will be reset to zero as soon as it reaches or exceeds a limit.

Resetting to zero only works if the lower limit is <0 and the upper limit is >0.

Example



2.8.5 Delay

```
Delay('Expression','NumberSamples*')
```

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns a delayed copy of the ‘Expression’ signal. The delay is specified in number of measurements ('NumberSamples'). The result is a signal curve with the values of the original signal for ‘NumberSamples’ before the current time.

To avoid a memory overflow, ‘NumberSamples’ is limited to 10,000.

Tip

The time base of the function is relevant for the number of measurements for different time bases of the measured value and function.

Example

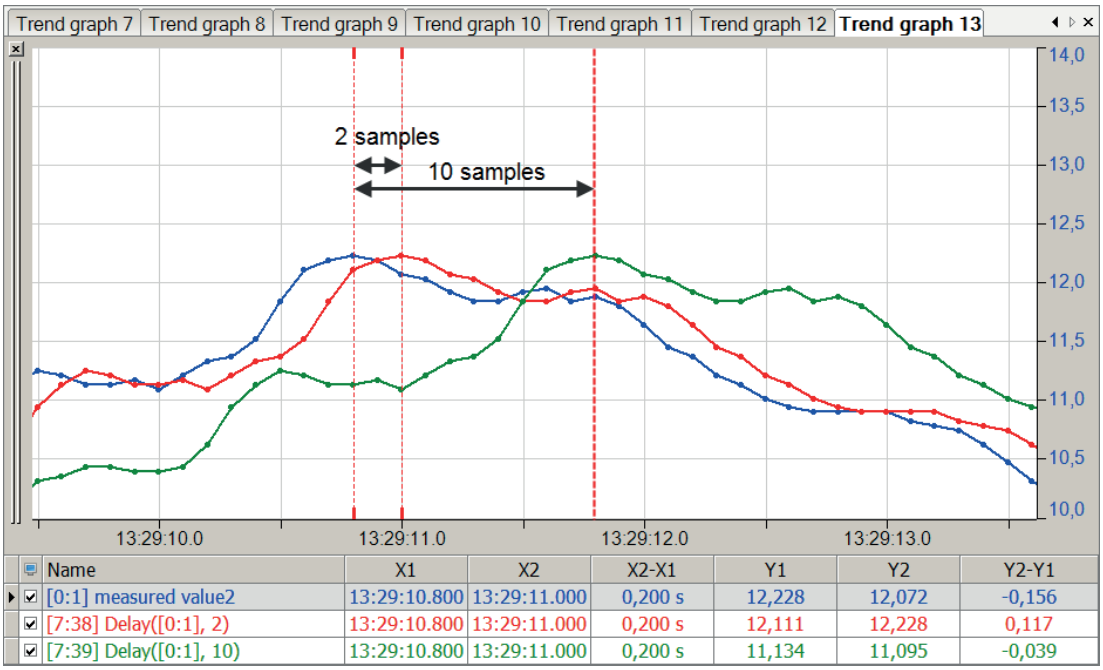
A measured value should be displayed delayed by 2 measured values and by 10 measured values.

Task description

For the chosen time base of 100 ms, a delay of two measured values corresponds to a time delay of 0.2 seconds and a delay of 10 measured values corresponds to 1 second.

Solution

In the figure below the blue curve shows the original signal and the red curve shows the delayed signal by 2 measured values (= 0.2 seconds). The green curve shows the delayed signal by 10 measured values (= 1 second).



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:1] measured value2	13:29:10.800	13:29:11.000	0,200 s	12,228	12,072	-0,156
[7:38] Delay([0:1], 2)	13:29:10.800	13:29:11.000	0,200 s	12,111	12,228	0,117
[7:39] Delay([0:1], 10)	13:29:10.800	13:29:11.000	0,200 s	11,134	11,095	-0,039

2.8.6 DelayLengthL

```
DelayLengthL('Expression','Length','MaxLengthDelta','DelayInMeter','Resolution*',
'Filter=0*')
```

Arguments

'Expression'	input signal	
'Length'	Length signal (in m)	
'MaxLengthDelta'	Upper limit to take into account the changes of the length signal	
'DelayInMeter'	Delay (in m)	
'Resolution*'	Resolution: Length basis of the result (in m)	
'Filter'*	Optional parameter (default = 0) to set the filter for time-length conversion	
	'Filter' = 1	Minimum filter
	'Filter' = 2	Maximum filter
	'Filter' = 0 and others	No filter

Parameters ending with * are only evaluated once at the start of the acquisition..

Description

This function uses the 'Length' length signal (in m) to create a length-based version of 'Expression' that is delayed over 'DelayInMeter' meters. Changes in the length signal that exceed the 'MaxLengthDelta' are ignored. The resolution is the length base of the result (in m).

'Filter' determines what filter is used during the time-to-length conversion.

Mainly for quality data monitoring in conjunction with ibaQDR, measured signals from distant sources can be aligned with regard to the product length. The distance between the signal sources determines the ‘DelayInMeter’ value.

2.8.7 DelayLengthV

```
DelayLengthV('Expression','Speed','DelayInMeter','Resolution*', 'Filter=0*')
```

Arguments

'Expression'	input signal	
'Speed'	Speed signal (in m/s)	
'DelayInMeter'	Delay (in m)	
'Resolution*'	Resolution: Length basis of the result (in m)	
'Filter'*	Optional parameter (default = 0) to set the filter for time-length conversion	
	'Filter' = 1	Minimum filter
	'Filter' = 2	Maximum filter
	'Filter' = 0 and others	No filter

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function uses the 'Speed' speed signal (in m/s) to create a length-based version of 'Expression' that is delayed over 'DelayInMeter' meters. The resolution is the length base of the result (in m).

'Filter' determines what filter is used during the time-to-length conversion.

2.8.8 DWORD

```
DWORD('Low','High')
```

Arguments

'Low'	16 bit integer: Word	
'High'	16 bit integer: Word	

Description

This function returns the 32 bit integer DWORD consisting of the int16 WORDS ‘Low’ and ‘High.’

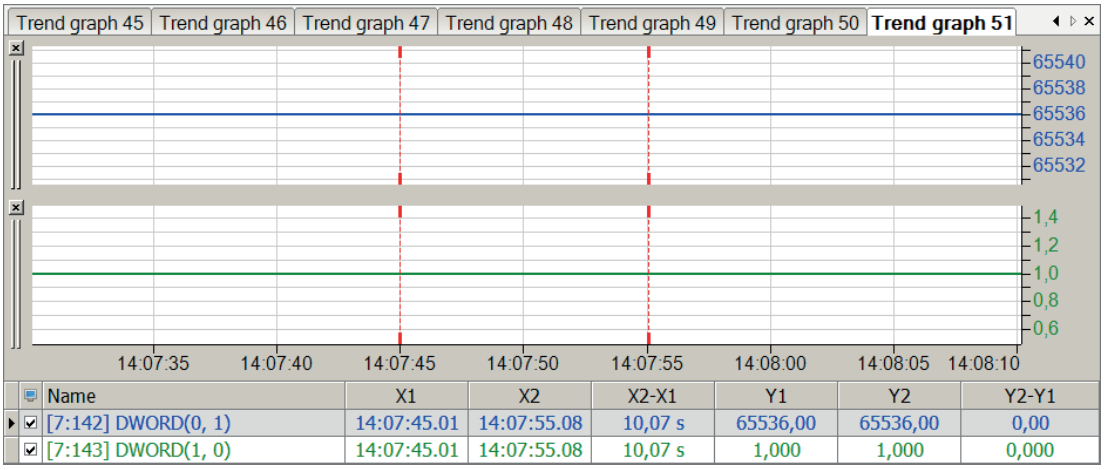
Example

Calculating two DWORDS from a simple 0 and 1 combination of 'Low' and 'High'.

	from int16 WORDS compound DWORD			
DWORD('Low','High')	int16 'High'	int16 'Low'	Conv.	Dec.
DWORD(0, 1)	0000 0000 0000 0001	0000 0000 0000 0000	216	65536
DWORD(1, 0)	0000 0000 0000 0000	0000 0000 0000 0001	20	1

Solution

The figure below shows the calculation of two simple DWORDS.



2.8.9 ElapsedTime

```
ElapsedTime('Start','Stop')
```

Description

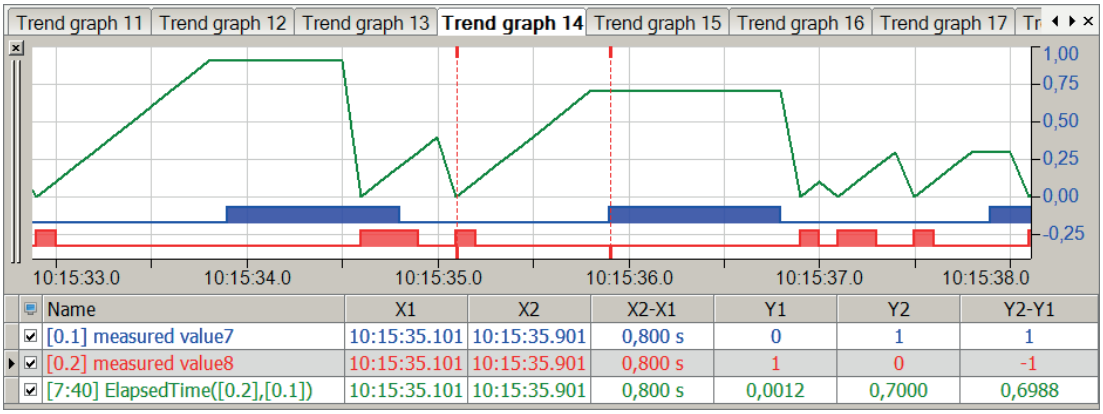
This function returns the time since the last rising edge for 'Start'. The time is stopped at the rising edge for 'Stop'.

Example

Measurement of the time gap between the last rising edge of a measured value up to the next rising edge of a different measured value.

Solution

In the figure below the blue bar shows the 'Stop' signal and the red bar shows the 'Start' signal. The green curve shows the time that elapsed between the rising edges of 'Start' to 'Stop'.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0.1] measured value7	10:15:35.101	10:15:35.901	0,800 s	0	1	1
[0.2] measured value8	10:15:35.101	10:15:35.901	0,800 s	1	0	-1
[7:40] ElapsedTime([0.2],[0.1])	10:15:35.101	10:15:35.901	0,800 s	0,0012	0,7000	0,6988

Note

By switching the edges in the sampling grid, the graphic presentation and calculation of the elapsed time may differ by one sample.

2.8.10 ExecuteCommand

```
ExecuteCommand('Trigger',"'Command'","'Arguments'","'UserName'","'Password'",'Tim-
eout=0')
```

Arguments

'Trigger'	Binary signal or expression as a trigger for the execution of the command
"'Command'"	Full and absolute path to the command (.exe, .bat etc.) that should be executed
"'Arguments'"	Arguments/parameters required to execute the command
"'UserName'"	Username of an account which has the permission to execute "'Command'".
"'Password'"	The password, related to the user name
'Timeout'	Optional parameter; if specified and >0, the execution will be aborted as soon as 'Timeout' [s] has expired.

Description

This function executes the "'Command'" "'Arguments'" command line for a rising edge of 'Trigger'. The return value is 1 (TRUE) while the invoked process is running. Thus, this function is to used as expression for digital signals. Other rising edges are ignored as long as the process is running.

The full, absolute path for the executable file must be specified for "'Command'", relative to the *ibaPDA* server.

If ‘Timeout’ is not specified, the command execution is not aborted and can run endlessly. If a value larger than zero is specified for ‘Timeout’, the execution of the command is aborted if it is not finished by ‘Timeout’ seconds.

The specification of username ("'UserName'") and password ("'Password'") is required for using this function. For security reasons the executables and programs called by this function must run under a dedicated user account. The account specified by "'UserName'" and "'Password'" must have the permission to execute the specified command.

When entering argument, username and password you can use the text encryption feature to obfuscate this information. The text encryption feature is offered as soon as you start to enter text between quotation marks ("…").

Note

If you use this function to call programs which generally have a user interface (GUI), then the GUI is not visible. Because the started program runs in the services session without an associated desktop, it is listed in the task manager, but it’s GUI is missing.

Example 1

Computer shutdown

Task description

Depending on a signal from the system control due to switching to an uninterrupted power supply (USP) during a power outage, the *ibaPDA* computer should be shut down.

Solution

The "shutdown" command from the Windows Command Shell is used to shutdown the computer. A batch file is used because multiple commands, such as stopping the *ibaPDA* service, are required.

```
ExecuteCommand([4.14], "D:\Schulung\ibaPDA_ExecuteCommand\Shutdown_PC.bat","","MyUsername","MyPassword")
```

- `[4.17]`: Digital trigger signal to execute the command
- `D:\training\ibaPDA_ExecuteCommand\Shutdown_PC.bat`: Program path for the batch file
- `MyUsername` and `MyPassword`: Specifications for user account

Contents of the `Shutdown_PC.bat` batch file:

```
1  SETLOCAL
2  sc stop ibaPDAService
3  shutdown /f /s /t 10
4  ENDLOCAL
```

Example 2

Starting a batch file, which creates a text file with up-to-date content.

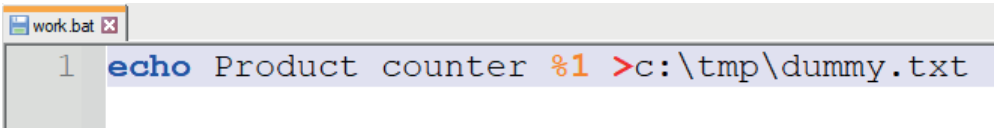
Task description

On a rising edge of a digital signal an actual counter value should be written

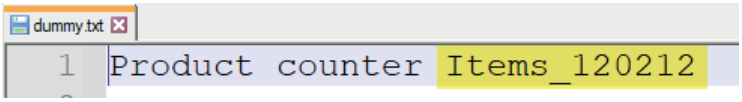
Solution

```
ExecuteCommand([3.17],"c:\tmp\work.bat", GenerateText("Items_%1"), "MyUserName","MyPassword",0)
```

- `[3.17]`: Digital trigger signal to execute the command
- `c:\tmp\work.bat`: Program path for the batch file to be executed `work.bat`; specifying a batch file is like using a command.
  Contents of the batch file (example):



- `GenerateText("Items_%1")`: Argument for the batch file; the result of this expression (generating a string with counter value) is put into the text file `dummy.txt` as parameter %1 of the batch file.
  Contents of the text file `dummy.txt` (example, dynamic content highlighted in yellow):



- `MyUserName, MyPassword`: Login credentials of the user account
- `0`: Value for timeout, i.e. the execution is not aborted.

2.8.11 GenerateSignal

```
GenerateSignal('Type', 'Amplitude=10','T1=1','T2=1')
```

Arguments

'Type'	Specification of the signal type	
	'Type' = 0	Sine function, with A = amplitude and T1 = period
	'Type' = 1	Cosine function, with A = amplitude and T1 = period
	'Type' = 2	Triangle function, with A = amplitude, T1 = rising edge time, T2 = falling edge time
	'Type' = 3	Block function, with A = amplitude, T1 = duration upper level, T2 = duration lower level
	'Type' = 4	Random signal with A = maximum amplitude
'Amplitude'	Optional parameter, specifying the amplitude; default = 10	
'T1'	Optional parameter, time specification 1 (except for type 4) given in s; default = 1	
'T2'	Optional parameter, time specification 2 (only for type 2 and type 3) given in s; default = 1	

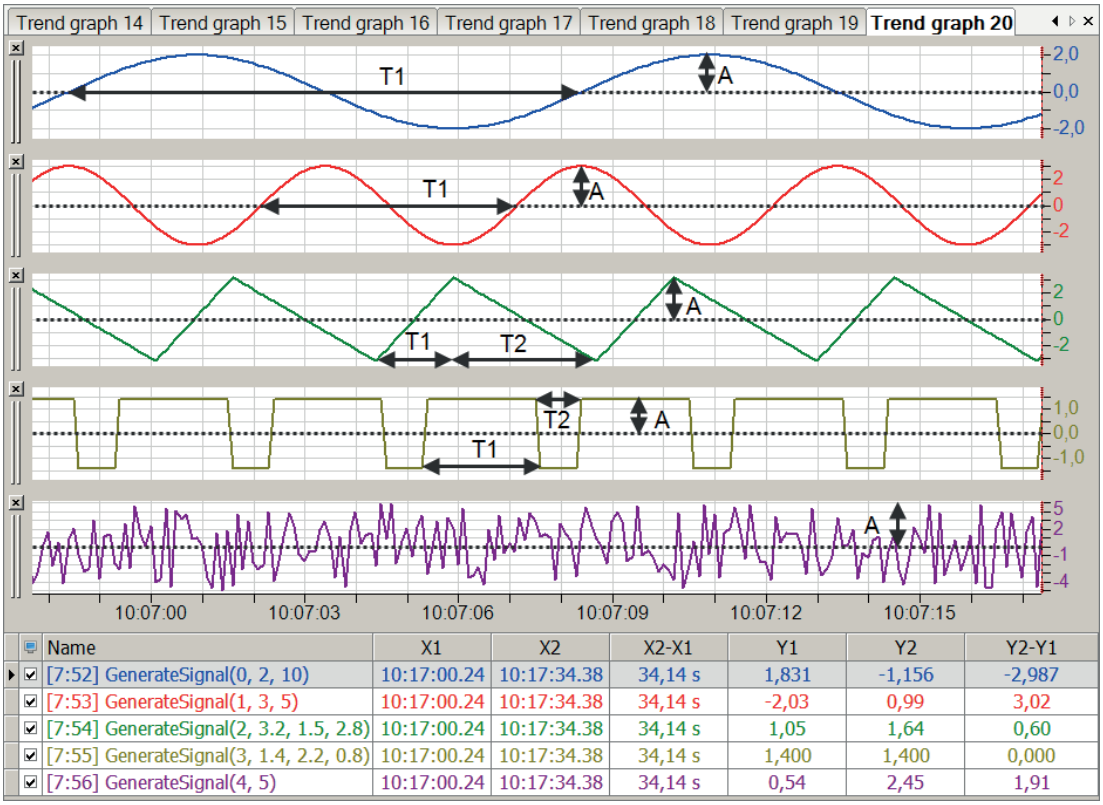
Description

This function generates a test signal with the 'Amplitude' amplitude and periods 'T1' and 'T2.'
The following signal types can be created:

Example

Several examples of generated signals

Solution



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[7:52] GenerateSignal(0, 2, 10)	10:17:00.24	10:17:34.38	34,14 s	1,831	-1,156	-2,987
[7:53] GenerateSignal(1, 3, 5)	10:17:00.24	10:17:34.38	34,14 s	-2,03	0,99	3,02
[7:54] GenerateSignal(2, 3.2, 1.5, 2.8)	10:17:00.24	10:17:34.38	34,14 s	1,05	1,64	0,60
[7:55] GenerateSignal(3, 1.4, 2.2, 0.8)	10:17:00.24	10:17:34.38	34,14 s	1,400	1,400	0,000
[7:56] GenerateSignal(4, 5)	10:17:00.24	10:17:34.38	34,14 s	0,54	2,45	1,91

Blue	Sine function type 0	Red	Cosine function type 1
Green	Triangle function type 2	Yellow	Block function type 3
Purple	Random function type 4		

2.8.12 GenerateText

```
GenerateText('Text*')
```

Parameters ending with * are only evaluated once at the start of the acquisition..

Arguments

'Text'	Text which should be generated as content of the text signal. The text has to be put between quotation marks (".."). The text will be evaluated only once at start of acquisition.	
	%n	Parameter, which can be inserted in the text and will be replaced in the generated text by an automatic counter value. The counter counts up by 1 with every nth cycle, based on the timebase of the *ibaPDA* system, e.g. %1 - with each cycle %2 - with each second cycle %10 - with each tenth cycle %100 - with each 100th cycle. Cycle

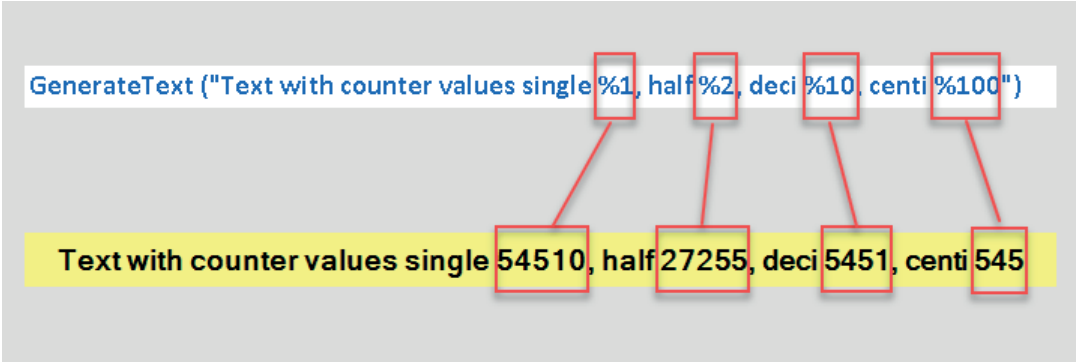
Description

This function generates a text signal. It can only be used in the signal table *Analog* of a suitable module (e.g. virtual module or output module)

The generated text signal can be used like other text signals, e.g. in a text creator module, a text digital display or a QPanel label.

You may use placeholders %1, %2,... in the text. The placeholder %n will be replaced by the counter value which increased every n samples.

Example



The upper line shows the complete function call. The lower line shows the result in a QPanel label.

2.8.13 GetFloatBit

```
GetFloatBit('Expression','BitNo')
```

Arguments

'Expression'	Any number
'BitNo'	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).

Description

This function interprets 'Expression' as the bitmask of a float value and returns as its result the Boolean value of the 'BitNumber' bit of 'Expression'. Valid bit number sequence: 0 (LSB) to 31 (MSB).

This function was created especially for the case that 32 bits are packed in a float data for transmission or recording. The GetBitFloat function only takes the value of the specified 'BitNo' bit regardless of whether it is part of the mantissa or exponent. Unlike the GetBit function, there is no rounding of values.

Example

A value entered in 'Expression' is converted to the floating-point format IEEE 754 and the digits bits which are set to TRUE accordingly, are retrieved.

Solution

Floating-point numbers according to the IEEE 754 format can generally be written as follows: (sign)*mantissa*10^Exponent

The number 0.15625 reads as a floating-point number according to IEEE 754 as follows:

Sign (1 bit)	Exponent (8 bit)	Mantisse (23 bit)
0	0111 1100	0100 0000 0000 0000 0000 000

In this case, the LSB (least significant bit) is at the extreme right point.

Tip

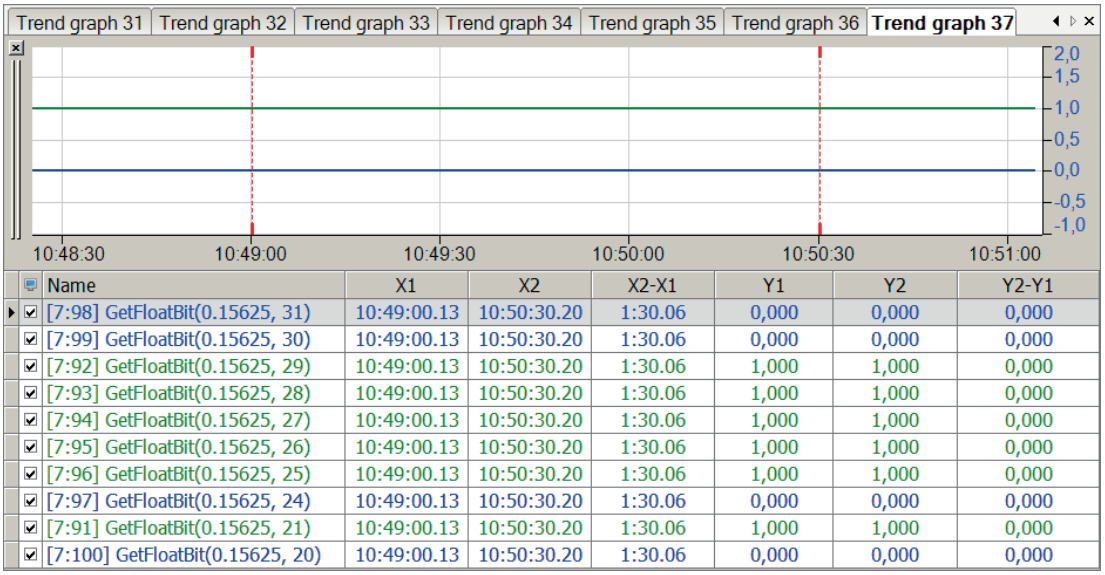
Floating-point numbers according to the IEEE 754 format are written in the manner shown. The following points for the individual components should be noted:

Sign: 0 stands for positive; 1 for negative

Exponent: Since with the conversion to a floating-point number, the exponent value is shifted by 127, 127 must be subtracted from the displayed exponent in the conversion into a decimal number.

Mantissa: Because the mantissa has a 1 before the decimal point by default, this digit is no longer represented. All mantissa digits are therefore decimal digits and must take into account the conversion with negative exponents to base 2.

The figure below shows the retrieval of the valence of selected bits of the float value of 0.15625.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[7:98] GetFloatBit(0.15625, 31)	10:49:00.13	10:50:30.20	1:30.06	0,000	0,000	0,000
[7:99] GetFloatBit(0.15625, 30)	10:49:00.13	10:50:30.20	1:30.06	0,000	0,000	0,000
[7:92] GetFloatBit(0.15625, 29)	10:49:00.13	10:50:30.20	1:30.06	1,000	1,000	0,000
[7:93] GetFloatBit(0.15625, 28)	10:49:00.13	10:50:30.20	1:30.06	1,000	1,000	0,000
[7:94] GetFloatBit(0.15625, 27)	10:49:00.13	10:50:30.20	1:30.06	1,000	1,000	0,000
[7:95] GetFloatBit(0.15625, 26)	10:49:00.13	10:50:30.20	1:30.06	1,000	1,000	0,000
[7:96] GetFloatBit(0.15625, 25)	10:49:00.13	10:50:30.20	1:30.06	1,000	1,000	0,000
[7:97] GetFloatBit(0.15625, 24)	10:49:00.13	10:50:30.20	1:30.06	0,000	0,000	0,000
[7:91] GetFloatBit(0.15625, 21)	10:49:00.13	10:50:30.20	1:30.06	1,000	1,000	0,000
[7:100] GetFloatBit(0.15625, 20)	10:49:00.13	10:50:30.20	1:30.06	0,000	0,000	0,000

2.8.14 GetIntBit

```
GetIntBit('Expression','BitNo')
```

Description

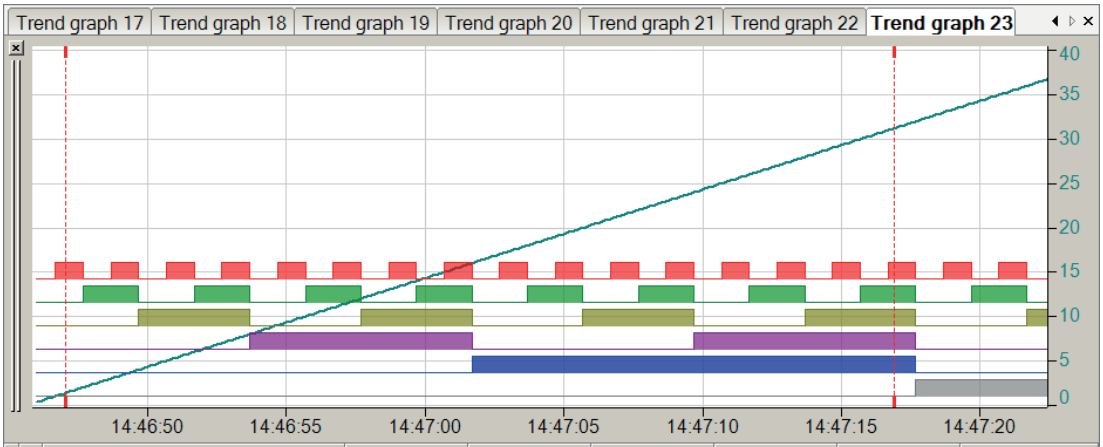
This function returns the Boolean value of the 'BitNo' bit of 'Expression' as its result after rounding the 'Expression' to the nearest integer value. The rounding limit is in each case the next 0.5 increment. (2.49 → 2; 2.50 → 3).
Valid bit number sequence: 0 (LSB) to 31 (MSB)

Example

Representation of time in Boolean values

Solution

The figure below shows the GetIntBit function of the first six bits of the time function, T().



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:3] T()	14:46:47.06	14:47:16.92	29,86 s	1,36	31,22	29,86
[7.41] GetIntBit(T(), 0)	14:46:47.06	14:47:16.92	29,86 s	1	1	0
[7.42] GetIntBit(T(), 1)	14:46:47.06	14:47:16.92	29,86 s	0	1	1
[7.43] GetIntBit(T(), 2)	14:46:47.06	14:47:16.92	29,86 s	0	1	1
[7.44] GetIntBit(T(), 3)	14:46:47.06	14:47:16.92	29,86 s	0	1	1
[7.45] GetIntBit(T(), 4)	14:46:47.06	14:47:16.92	29,86 s	0	1	1
[7.46] GetIntBit(T(), 5)	14:46:47.06	14:47:16.92	29,86 s	0	0	0

2.8.15 GetSignalMetaData

```
GetSignalMetaData('Signal*"','InfoType*')
```

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns metadata information of a signal. The parameter 'Signal' can be supplied as signal number ([m:n] resp. [m.n]) or as signal name ([SignalName]).

Specify the information type ('InfoType') you want to receive.

The following information types are supported:

Information types	Possible results
0	Signal name
1	Signal unit
2	Comment 1
3	Comment 2

Table 4: Information types and possible results of the GetSignalMetaData function

2.8.16 GetSystemTime

```
GetSystemTime('Part*')
```

Arguments

'Part*'	Part to be displayed of the system time specification	
	'Part' = 0	milliseconds
	'Part' = 1	seconds
	'Part' = 2	minutes
	'Part' = 3	hours
	'Part' = 4	day of the month
	'Part' = 5	month
	'Part' = 6	year
	'Part' = 7	day of the year
	'Part' = 8	day of week (Monday=1, Sunday=7)
	'Part' = 9	system time in Unix time format Requires the "High precision" setting of the module to be enabled
	'Part' = 10	UTC offset in seconds

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns the part specified under 'Part' of the system time.

Tip

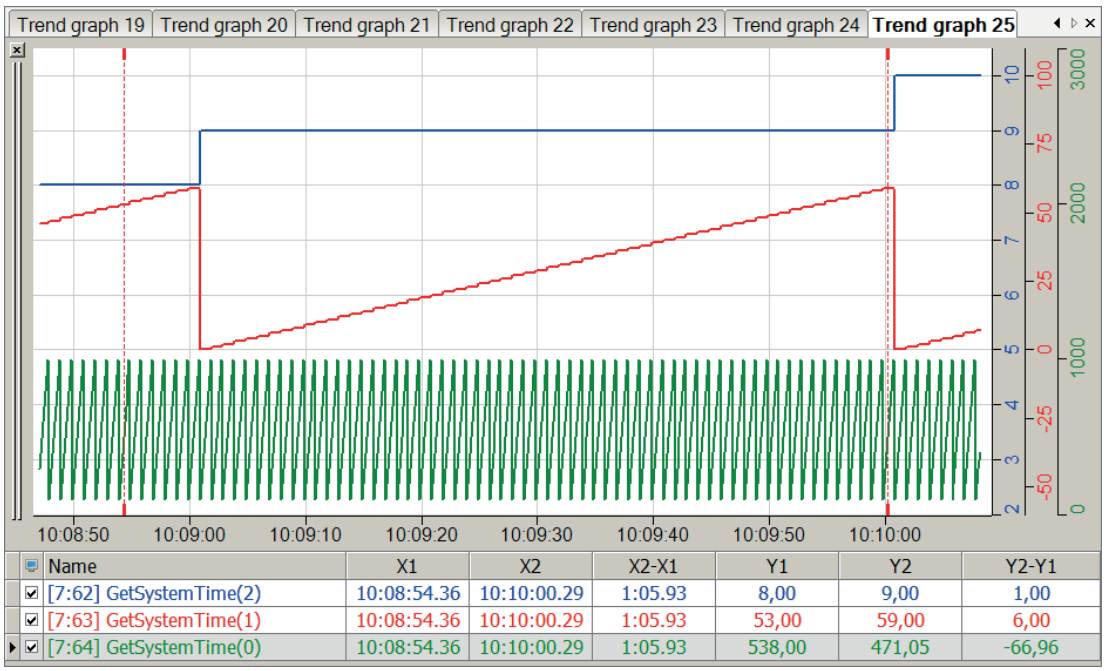
In *ibaQPanel*, the day of the week (Part 8) can easily be displayed with its name by using a multi-state display. For the other parts, the normal label works fine. Use a gauge to display the time on an analog-like clock in *ibaQPanel*.

Example

Output of the minute part, second part and millisecond part of the system time.

Solution

In the figure below the blue curve shows the minutes of the system time ('part' = 2) and the red curve shows the seconds of the system time ('part' = 1). The green curve shows the milliseconds of the system time ('part' = 0).



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[7:62] GetSystemTime(2)	10:08:54.36	10:10:00.29	1:05.93	8,00	9,00	1,00
[7:63] GetSystemTime(1)	10:08:54.36	10:10:00.29	1:05.93	53,00	59,00	6,00
[7:64] GetSystemTime(0)	10:08:54.36	10:10:00.29	1:05.93	538,00	471,05	-66,96

2.8.17 GetSystemTimeAsText

```
GetSystemTimeAsText('DateTimeFormat*="yyyy-MM-dd HH:mm:ss"', 'UTC*=0', 'Enable=1')
```

Arguments

'Argument'	Description	
'DateTimeFormat'	yy, yyyy	Year 2 or 4 digits
	M, MM	Month M: 1, 2, 3, 12; MM: 01, 02, 03, ...12
	MMM, MMMM	Name of month
	d, dd	Day d: 1, 2, 3, ...31; dd: 01, 02, 03, ... 31
	ddd, dddd	Day of week
	h, hh,	Hour 12-hour notation
	H, HH	Hour 24-hour notation
	m, mm	Minutes
	s,ss	Seconds
	f, ff, fff	Fractions of a second up to 0.1, 0.01, 0.001 (millisecond), ... precision
	tt	AM/PM indicator; only available if ibaPDA service runs in a culture that supports AM/PM (e.g. US-EN)
'UTC'	0 (default)	Date/time local
	1	UTC date and time
'Enable'	1	Optional argument; if 'Enable' is provided, then date/time will only be emitted when 'Enable' = 1 or true. If the argument is omitted, date/time will always be emitted.

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns the current date and time. Depending on the parameter 'UTC', date and time will be returned as local or UTC date/time.

The calculation can be controlled using the optional parameter, 'Enable.' If you want to take advantage of higher precision by using fractions of a second, you should provide the 'Enable' parameter and set it on 1 or true.

Examples

Examples for Tuesday, August 07, 2023, 17:05 (CEST)

Formula	Result
`GetSystemTimeAsText("yyyy-MM-dd HH:mm:ss",0)`	2023-08-07 17:05:42
`GetSystemTimeAsText("yyyy-MM-dd HH:mm:ss",1)`	2023-08-07 15:05:42
`GetSystemTimeAsText("yy-MMM-dd hh:mm:ss",1)`	23-Aug-07 03:05:42
`GetSystemTimeAsText("yyyy-MM-dd HH:mm:ss fff",0)`	2023-08-07 17:05:42 478

2.8.18 GetWeekOfYear

```
GetWeekOfYear('Rule*=0', 'FirstDayOfWeek*=0')
```

Arguments

'Rule*'	Definition rule for first week of the year	
	'Rule' = 0	The first week of the year is the one that includes at least 4 days of that year. (ISO 8601)
	'Rule' = 1	The first week of the year starts on January 1st.
	'Rule' = 2	The first week of the year starts on the first day of the week on or after January 1st.
'FirstDayOf-Week*'	Definition of first day of the week	
	'FirstDayOf-Week'= 0	Monday (ISO 8601)
	'FirstDayOf-Week'= 1	Tuesday
	'FirstDayOf-Week'=2	Wednesday
	'FirstDayOf-Week'=3	Thursday
	'FirstDayOf-Week'= 4	Friday
	'FirstDayOf-Week'= 5	Saturday
	'FirstDayOf-Week'= 6	Sunday

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns the number of the current week in the current year. The week numbering is done according to 'Rule'.

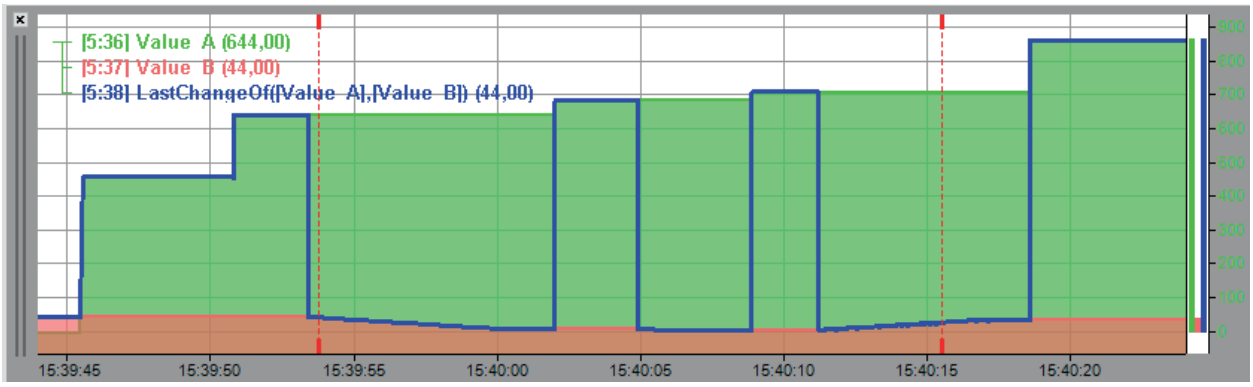
2.8.19 LastChangeOf

```
LastChangeOf('Expression1','Expresion2',...)
```

Description

This function returns the most recently changed value of a set of expressions (‘Expression1’,’Expression2’,...). If multiple source expressions change their value simultaneously, the result of this function will be the value of the first expression in the sequence of arguments whose value has changed.

Example



The expressions *Value_A* (green) and *Value_B* (red) are changing independently from each other. The result of *LastChangeOf* with these values follows always the value which changed at last.

2.8.20 LimitAlarm

```
LimitAlarm('Expression','Limit','DeadBand','Time',' Reset=0’)
```

Arguments

'Expression'	Measured value	
'Limit'	Limit value, from which the function returns TRUE	
'DeadBand'	Specification of a dead zone below the limit value, within which the function does not reset to FALSE	
'Time'	Specification of the time, for which the measured value must be above the limit value until the function returns TRUE	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation	
	'Reset' = 0	Perform calculation
	'Reset' = 1	Stop calculation, reset and set result to 0
	'Reset' = 2	Stop calculation and keep result

Description

This function monitors the measured value ('Expression') and sets the result to TRUE if the measured value is longer than the specified time ('Time') above the ('Limit') limit value. The result of the function becomes FALSE again if the measured value falls below the limit value by the value specified under the ('DeadBand') deadzone.

Tip

The LimitAlarm function can also be used for a lower limit For this purpose, only the measured value and the limit value must be flipped, i.e. multiplied by (-1).

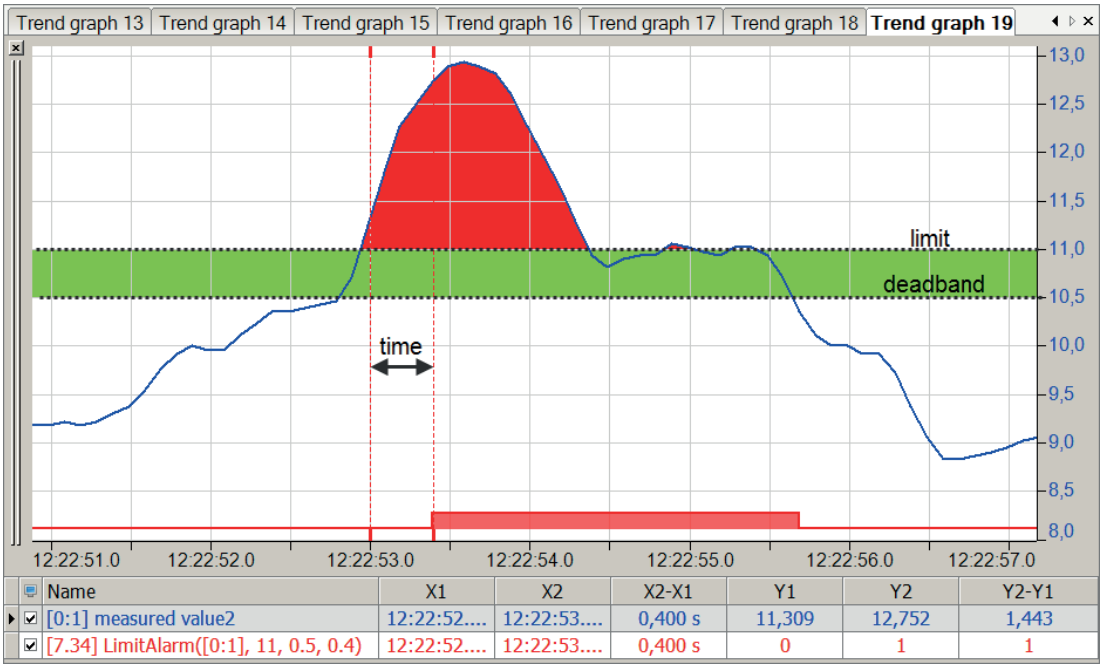
e.g: LimitAlarm([0:1] ***(-1)**, 9 ***(-1)**, 0.5, 0.4)

Example

The function should return TRUE if the measured value is longer than 0.4 seconds above the value 11. The function should then return FALSE again if the measured value has fallen below 10.5.

Solution

In the figure below the blue curve shows the measured value and the red bar shows when the limit is exceeded. The green band shows the dead zone that prevents a direct FALSE setting of the function.



2.8.21 ModuleSignalCount

```
ModuleSignalCount('ModuleNo*', 'SignalType*=0', 'Direction*=0')
```

Arguments

'SignalType*'	Signal type	
	'SignalType' = 0	Analog and digital signals
	'SignalType' = 1	Analog signals only
	'SignalType' = 2	Digital signals only
'Direction*'	Input or output direction or both	
	'Direction'= 0	Inputs and Outputs
	'Direction'= 1	Inputs only
	'Direction'=2	Outputs only

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns the number of active signals in a module with module number 'ModuleNo'. With the parameter 'SignalType' you can determine, which signal types are to be considered. Signals, which are configured in the module but not checked as *Active*, are not taken into account.

2.8.22 PulseFreq

```
PulseFreq('Expression','Omega=0*','EdgeType=2*' )
```

Arguments

'Expression'	Pulse counter signal	
'Omega*'	Filter frequency	
'EdgeType*'	Edge type to be counted	
	'EdgeType' = -1	Falling edges only
	'EdgeType' = 0	Rising and falling edges
	'EdgeType' = 1	Rising edges only
	'EdgeType' = 2	'Expression' is a pulse counter

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function computes the frequency of 'Expression' pulse counters or pulses. The unit of the result is pulses/sec or Hz.

A low-pass filter with an 'Omega' limit angular speed is applied to the result. If 'Omega' is 0 then the low-pass filter is deactivated. 'EdgeType' determines which edges of pulses should be counted.

Zero is returned as the calculated frequency if no pulse occurs in 1000 samples.

This function was especially created for using the WAGO incremental encoder 750-631. You may use the function to calculate the speed based on the pulse counter signal from the encoder. The pulse counter value is differentiated taking into consideration possible counter overflows. As the result of the differentiation may include interfering frequencies or noise, a low-pass filter can then be used. The filter frequency to be set should be slightly above the maximum pulse frequency.

2.8.23 RestartAcquisition

```
RestartAcquisition('Trigger')
```

Description

The acquisition is restarted when a rising edge on 'Trigger' occurs. The function returns the value 1 when the acquisition is restarted.

2.8.24 SampleAndHold

```
SampleAndHold('Expression','Sample','Initial=0)
```

Arguments

'Expression'	Measured value
'Sample'	Parameter that determines whether the function follows the measured value (1) or holds the last measured value (0). 'Sample' can be a condition itself or be determined by a different function.
'Initial'	Optional parameter (default = 0), which determines the initial value of the function when 'Sample' is inactive at the start of the measurement.

Description

This function is a sample-hold function. The output follows 'Expression' when 'Sample' = TRUE. It remains unchanged when 'Sample' = FALSE. With the optional 'Initial' parameter, the initial value of the output can be specified if the function is on "Hold" when called.

Example

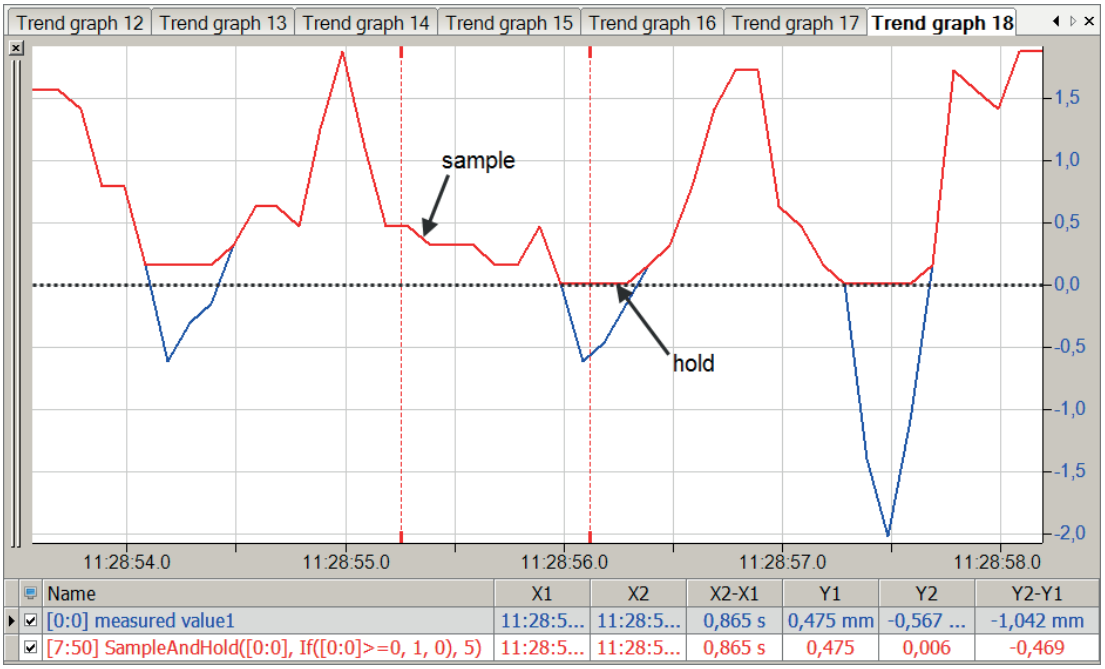
The function should follow a measured value if it is positive and be held as soon as the measured value becomes negative.

Task description

An If query as the 'Sample' parameter returns the value 1 for positive values and the value 0 for negative values.

Solution

In the figure below the blue curve shows the measured value and the red curve shows the output value with held measured values.



2.8.25 SampleOnce

```
SampleOnce('Expression','Sample')
```

Arguments

'Expression'	Measured value
'Sample'	Parameter which determines when the function should follow the measured value 'Expression' for exactly one sample. 'Sample' can be a condition itself or be determined by a different function.

Description

This function is a sample-once-function. The result of the function follows 'Expression' for the duration of one sample when a rising edge occurs on 'Sample'. For all other timestamps the result is NaN if 'Expression' is a numeric signal and an empty string if 'Expression' is a text signal.

2.8.26 Sign

```
Sign('Expression')
```

Description

This function returns the sign of 'Expression' as its result.

'Expression' > 0 --> +1

'Expression' = 0 --> 0

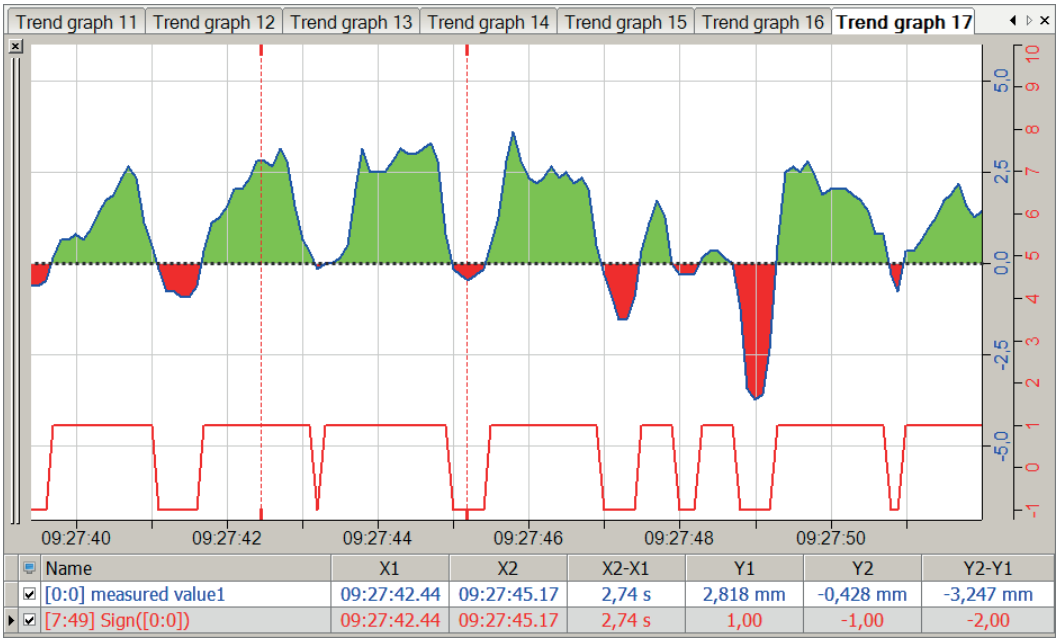
'Expression' < 0 --> -1

Example

Only the sign from a measured value is relevant.

Solution

In the figure below the blue curve shows the measured value and the red curve shows the sign of the measured value.



2.8.27 T

```
T()
```

Description

This function returns the time elapsed since the start of measuring (in seconds).

Using the time function, time-dependent virtual variables, e.g. sine waves, can be calculated.

Example of a sine wave of 0.5 Hz: sin(2*PI()*0.5*T())

Tip

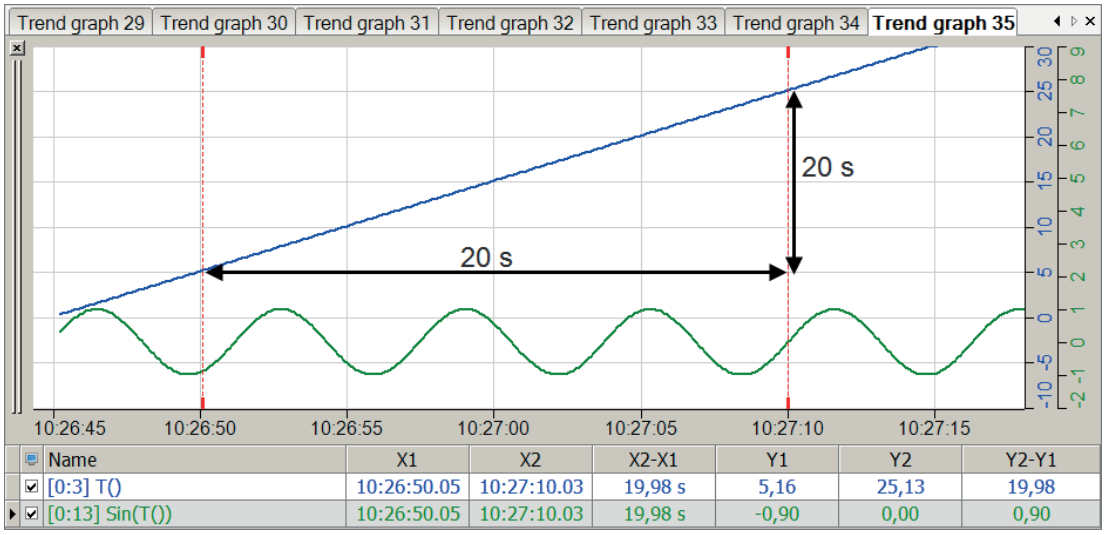
The time function can be entered In the expression editor simply by double clicking in other expressions, as is the case with a signal. The two brackets behind T are required for syntax reasons, as in the case of the number Pi.

Example

Presenting the time function and a time-dependent, virtual variable

Solution

Calculation of the time function and a time-depending sine curve



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:3] T()	10:26:50.05	10:27:10.03	19,98 s	5,16	25,13	19,98
[0:13] Sin(T())	10:26:50.05	10:27:10.03	19,98 s	-0,90	0,00	0,90

2.8.28 VarDelay

```
VarDelay('Expression','Delay', 'MaxDelay=30*')
```

Arguments

'Expression'	input signal
'Delay'	Time delay in seconds
'MaxDelay'*	Optional parameter (default = 30) to determine the maximum delay permitted in seconds

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function delays the 'Expression' signal by the 'Delay' time. In contrast to the Delay function, the delay time may change over time. *'MaxDelay'* specifies the maximum delay permitted and is preset to 30 s by default.

Example

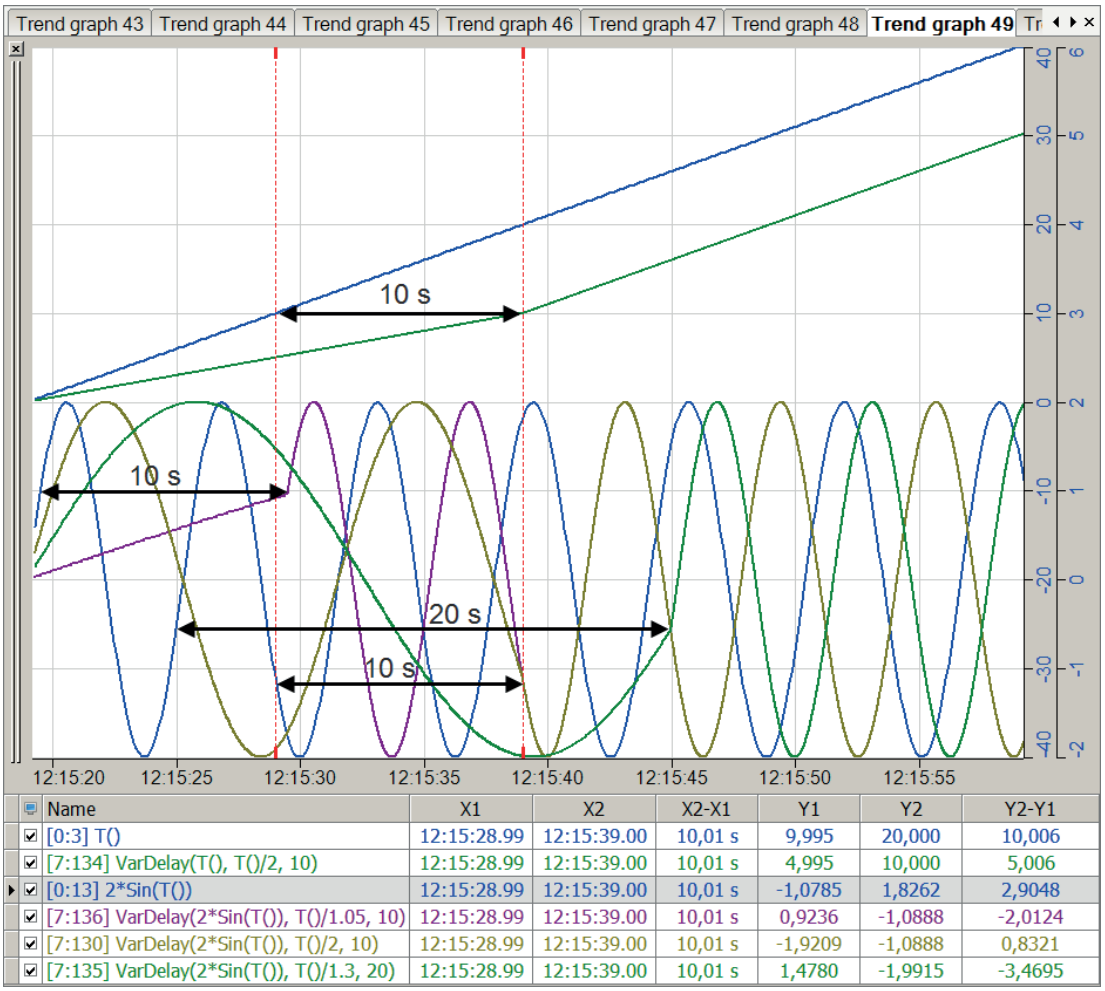
Temporally variable delay of a signal reaching the maximum permitted delay

Solution

The time function T() is used as the variable delay. This and a sine signal are set to this variable delay and the resulting curves are recorded.

As soon as the maximum delay is reached, the delay remains constant whereas the resulting curves only represent the shifted output signals.

In the figure below the top part shows the delay of the time signal by a variable shift with a maximum value of 10 seconds. The bottom part shows the delay of a sinusoidal function with three different delay and maximum delay values.



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:3] T()	12:15:28.99	12:15:39.00	10,01 s	9,995	20,000	10,006
[7:134] VarDelay(T(), T()/2, 10)	12:15:28.99	12:15:39.00	10,01 s	4,995	10,000	5,006
[0:13] 2*Sin(T())	12:15:28.99	12:15:39.00	10,01 s	-1,0785	1,8262	2,9048
[7:136] VarDelay(2*Sin(T()), T()/1.05, 10)	12:15:28.99	12:15:39.00	10,01 s	0,9236	-1,0888	-2,0124
[7:130] VarDelay(2*Sin(T()), T()/2, 10)	12:15:28.99	12:15:39.00	10,01 s	-1,9209	-1,0888	0,8321
[7:135] VarDelay(2*Sin(T()), T()/1.3, 20)	12:15:28.99	12:15:39.00	10,01 s	1,4780	-1,9915	-3,4695

Tip

Negative values for 'Delay' do not produce an error message but are treated as 0, i.e. there is no delay.

2.8.29 WindowAlarm

```
WindowAlarm('Expression','Limit1','DeadBand1','Limit2','DeadBand2','Time',' Re-
set=0')
```

Arguments

'Expression'	Measured value	
'Limit1'	Upper limit value, from which the function returns TRUE	
'DeadBand1'	Specification of the dead zone below the upper limit value ('Limit1'), within which the function does not reset to FALSE	
'Limit2'	Lower limit value, from which the function returns TRUE	
'DeadBand2'	Specification of the dead zone above the lower limit value ('Limit2'), within which the function does not reset to FALSE	
'Time'	Specification of the time, for which the measured value must be greater than the upper limit or smaller than the lower limit until the function returns TRUE	
'Reset'	Optional parameter (default = 0) to stop and restart the calculation	
	'Reset'=0	Perform calculation
	'Reset'=1	Stop calculation, reset and set result to 0
	'Reset'=2	Stop calculation and keep result

Description

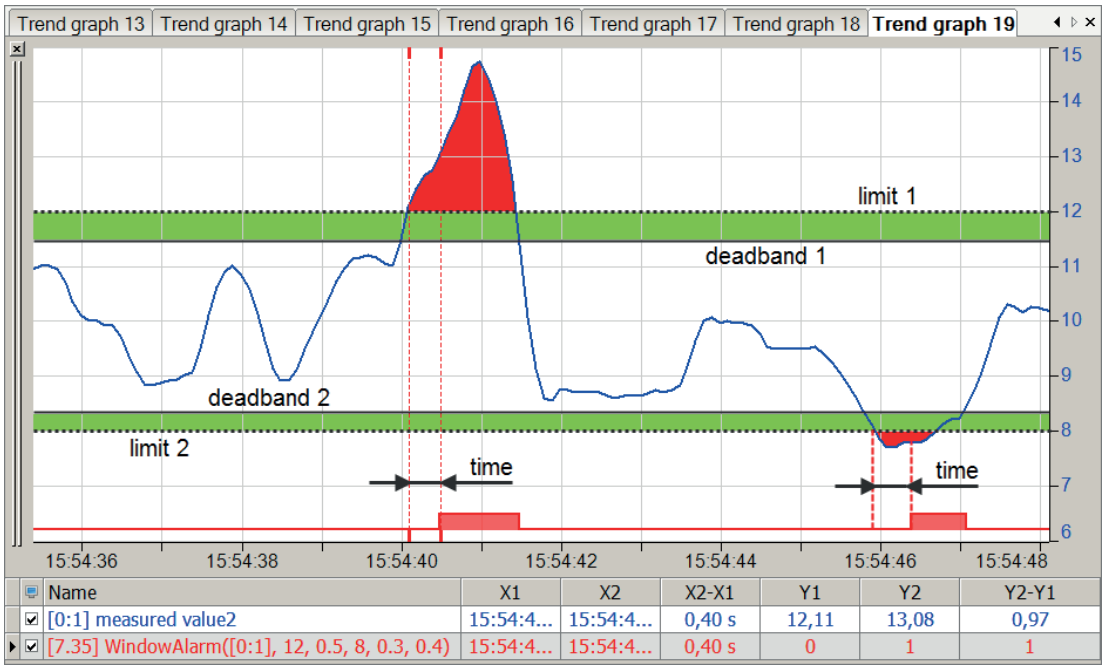
This function monitors the measured value ('Expression') and sets the result to TRUE if the measured value is longer than the specified time ('Time') outside the range between the upper limit value ('Limit1') and the lower limit value ('Limit2'). The result of the function becomes FALSE again if the measured value falls below the upper limit by the value specified under 'DeadBand1', or exceeds the lower limit by the value specified under 'DeadBand2'.

Example

The function is to be triggered if the measured value is outside the value range between 8 and 12 for longer than 0.4 s. The upper dead zone should be 0.5, the lower dead zone 0.3.

Solution

In the figure below the blue curve shows the measured value and the red bar shows the triggers the WindowAlarm function. The green areas show the dead zones of the respective limits.


Trend graph 13
Trend graph 14
Trend graph 15
Trend graph 16
Trend graph 17
Trend graph 18
Trend graph 19
limit 1
deadband 1
deadband 2
limit 2
time
time
15
14
13
12
11
10
9
8
7
6
15:54:36
15:54:38
15:54:40
15:54:42
15:54:44
15:54:46
15:54:48
Name
X1
X2
X2-X1
Y1
Y2
Y2-Y1
[0:1] measured value2
15:54:4...
15:54:4...
0,40 s
12,11
13,08
0,97
[7.35] WindowAlarm([0:1], 12, 0.5, 8, 0.3, 0.4)
15:54:4...
15:54:4...
0,40 s
0
1
1

2.9 Diagnosis functions

2.9.1 CameraStatus

```
CameraStatus('ModuleNo*','SignalNo*','Timeout=2*')
```

Arguments

'ModuleNo*'	Module number of the ibaCapture server in the signal tree	
'SignalNo*'	Signal number (camera)	
'Timeout*'	Time in seconds in which the sync signal must change; default = 2	

Parameters ending with * are only evaluated once at the start of the acquisition..

Description

This function returns the status of an ibaCapture camera.

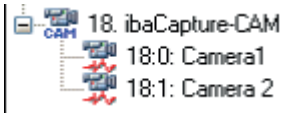
Results

0	Camera not ok
1	Camera OK

A camera is not ok when the value of the sync signal of the camera does not change within 'Timeout' seconds. Both the recording of this camera on the *ibaCapture* server as well as the synchronization between *ibaCapture* server and *ibaPDA* server must be running in order to facilitate the sync signal to change.

‘ModuleNumber’ is the module number of the ibaCapture server in the signal tree.

Example:



Status of "Camera1" with Timeout = 2 s: CameraStatus(18, 0, 2)

2.9.2 DataStoreInfo

```
DataStoreInfo('DatastoreIndex*','InfoType*')
```

Parameters ending with * are only evaluated once at the start of the acquisition..

Description

This function provides information about the selected data storage. This information may be used for controlling other functions or for display and diagnostic purposes.

For normal (PDA) data storage, use 'DatastoreIndex' >= 0.

For *ibaQDR* data storage, use 'DatastoreIndex' < 0.

The index can easily be obtained by looking at the tree structure in the configuration dialog of the data record. Index increases top-down.

Specify the information type ('InfoType') you want to receive.

The following information types are supported:

Information types	Possible results
0: recording status	0 = stopped 1 = waiting for trigger 2 = recording 3 = posttrigger recording
1: Saving in the backup directory:	0 = base directory is used 1 = backup directory is used
2: recorded time in the current file expressed in seconds	Value is updated every second.
3: the free space on the current hard disk expressed in MB	Value is updated every minute.
4: is ibaQDR synchronized?	0 = ibaQDR is NOT synchronized 1 = ibaQDR is synchronized
5: Image triggers storing to backup directory	-1=No active or configured image triggers 0=All image triggers are using the base directory 1=All image triggers are using the backup directory 2=Some image triggers are using the base directory and some are using the backup directory
6: Number of overlapping data files	Actual value

Table 5: Information types and possible results of the DataStoreInfo function

2.9.3 DataStoreInfoDB, ...Influx, ...Kafka, ...MindSphere, ...MQTT

```
DataStoreInfoDB('DatastoreIndex*','InfoType*')
DataStoreInfoInflux('DatastoreIndex*','InfoType*')
DataStoreInfoKafka('DatastoreIndex*','InfoType*')
DataStoreInfoMindSphere('DatastoreIndex*','InfoType*')
DataStoreInfoMQTT('DatastoreIndex*','InfoType*')
```

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function provides information about the selected database or cloud data stores. This information may be used for controlling other functions or for display and diagnostic purposes.

Identify the desired DB/Cloud data storage with the 'DatastoreIndex*' > = 0.

The index can easily be obtained by looking at the tree structure in the configuration dialog of the Data Storage. Index increases top-down.

Specify the information type ('InfoType') you want to receive.

The following information types are supported:

Information types	Possible results
0: Recording status	0 = stopped 1 = waiting for trigger 2 = recording 3 = posttrigger recording
1: Data throughput	Value in kB/s
2: Is server connected?	0/False = no 1/True = yes
3: Recorded time since last start trigger	Value in seconds; this will be constant 0 for continuous recording.
5: Current buffer usage	Value in %
6: Current file buffer usage	Value in %
7: Unprocessed bytes in file buffer	Value in %

Table 6: Information types and possible results of the DataStoreInfoDB function

2.9.4 DataStoreInfoHD

```
DataStoreInfoHD('DatastoreIndex*','InfoType*')
```

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

Note

With ibaPDA v8.4 his function has been subject to a breaking change. In order to support all recording states of the triggered HD recording, the DataStoreInfoHD diagnostic function now uses the same values for the status (info type 0) as the other DataStoreInfo functions. If you have already used this function, please check your evaluation of the status values.

This function provides information about the selected HD records. This information may be used for controlling other functions or for display and diagnostic purposes.

Identify the desired HD data records with the 'DatastoreIndex*' > = 0.

The index can easily be obtained by looking at the tree structure in the configuration dialog of the data record. Index increases top-down.

Specify the information type ('InfoType') you want to receive.

The following information types are supported:

Information types	Possible results
0: Recording status	0 = Stopped 1 = Waiting for trigger 2 = Recording 3 = Post trigger recording
1: Data throughput	Value in KB/s
2: Is server connected?	0/False = no 1/True = yes
5: Current buffer usage	Value in %
6: Current file buffer usage	Value in %
7: Unprocessed bytes in file buffer	Value in %

Table 7: Information types and possible results of the DataStoreInfoHD function

2.9.5 DongleInfo

```
DongleInfo('InfoType*')
```

Parameters ending with * are only evaluated once at the start of the acquisition.

Note

This function is still supported for backward compatibility purposes. Please use instead the *LicenseInfo* function, which serves the same purposes while supporting soft licenses as well.

Description

This function provides information about various properties related to the dongle. This information can be used for display and diagnostic purposes.

Specify the information type ('InfoType') you want to receive. This type of information is determined and output at the start of the measurement. If you want to receive more information from the dongle then you need to configure the function several times, with a different type of information each time.

The following information types are supported:

Information types	Possible results
0: Dongle available	TRUE = dongle available FALSE = no dongle or dongle defective
1: Dongle time limit in days	Value of the remaining lifetime of the dongle
2: Demo time limit in days	Value of the remaining lifetime of the dongle for trial periods / demo versions
3: ibaQDR acquisition time limit in seconds	Value of the remaining time which continues to run the ibaQDR system after the dongle has been removed.
10: Dongle inserted - counter	Number of times the dongle was inserted
11: Dongle removed - counter	Number of times the dongle was removed
12: Dongle changed - counter	Number of times the dongles were changed

Table 8: Information types and possible results of the DongleInfo function

Note

By evaluating information type 0, you can monitor whether a license container is present or available. In practice, you can use this function in *ibaQDR* to generate an email when it is detected that the license is missing (e.g., dongle removed). With *ibaQDR*, data collection continues for up to 4 days without a license to give you the opportunity to procure a new license container without jeopardizing production. This grace period does not exist with *ibaPDA*.

2.9.6 FobDLinkStatus

```
FobDLinkStatus('BoardNo*','LinkNo*')
```

Arguments

'BoardNo*'	Board number (0 to 7)	
'LinkNo*'	Link number (refer to the signal tree in the I/O Manager)	

Parameters ending with * are only evaluated once at the start of the acquisition..

Description

This function returns the status of an ibaFOB-D board link.

The card number (from 0 to 7) can be obtained from the 7-digit display on the card or the graphical presentation in the I/O Manager. The link number can be obtained from the signal tree in the I/O Manager.

Results

0	Link not active
1	Link OK
2	Link interrupted
3	Link RX ok, but flex ring interrupted

Prerequisites

This function only returns a value if active input modules are installed and configured in the I/O Manager.

When using a bus monitor (such as ibaBM-DPM-S), at least one active slave must be configured in the I/O Manager.

2.9.7 FobFastLinkStatus

```
FobFastLinkStatus('BoardNo*','LinkNo*','Filtered*')
```

Arguments

'BoardNo*'	Board number (0 to 3)	
'LinkNo*'	Link number (refer to the signal tree in the I/O Manager)	
'Filtered*'	Filter settings (TRUE or FALSE)	

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns the status of a link of an ibaFOB-X card in 32-bit mode (32 Mbit/s).

The card number (from 0 to 3) can be obtained from the 7-digit display on the card or the graphical presentation in the I/O Manager. The link number can be obtained from the signal tree in the I/O Manager. ‘Filtered’ can be TRUE or FALSE. The filtered link status ignores any change in link status that is faster than 40 ms.

Results

0	Link not active
1	Link OK
2	Link interrupted

Prerequisites

This function only returns a value if active input modules are installed and configured in the I/O Manager.

When using a bus monitor (such as ibaBM-DPM-S), at least one active slave must be configured in the I/O Manager.

2.9.8 FobFlexDeviceStatus

```
FobFlexDeviceStatus('BoardNo*','LinkNo*','Address*')
```

Arguments

'BoardNo*'	Board number (0 to 7)	
'LinkNo*'	Link number (refer to the signal tree in the I/O Manager)	
'Address*'	Address of the device in the Flex ring (1 to 15)	

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns the status of the Flex device with the 'Address' address at the 'LinkNo' link of a 'BoardNo' ibaFOB-D card.

The card number (from 0 to 7) can be obtained from the 7-digit display on the card or the graphical presentation in the I/O Manager. The link number can be obtained from the signal tree in the I/O Manager.

The device address can be set on the rotary switch of the device. Address values 1 to 15 are possible.

Results

0	Device not configured
1	Device OK
2	Device not connected

2.9.9 FobFLinkStatus

```
FobFLinkStatus('BoardNo*','LinkNo*')
```

Arguments

'BoardNo*'	Board number (0 to 3)	
'LinkNo*'	Link number (refer to the signal tree in the I/O Manager)	

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns the status of a link of an FOB-S or an FOB-X card in 3-Mbit mode (3.3 Mbit/s).

The card number (from 0 to 3) can be obtained from the 7-digit display on the card or the graphical presentation in the I/O Manager. The link number can be obtained from the signal tree in the I/O Manager.

Results

0	Link not active
1	Link OK
2	Link interrupted

Prerequisites

This function only returns a value if active input modules are installed and configured in the I/O Manager.

When using a bus monitor (such as ibaBM-DPM-S), at least one active slave must be configured in the I/O Manager.

2.9.10 FobMLinkStatus

```
FobMLinkStatus('BoardNo*','LinkNo*')
```

Arguments

'BoardNo*'	Board number (0 to 3)	
'LinkNo*'	Link number (refer to the signal tree in the I/O Manager)	

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns the status of a link of an FOB-S card running in FOB-M mode (5 Mbit/s).

The card number (from 0 to 3) can be obtained from the 7-digit display on the card or the graphical presentation in the I/O Manager. The link number can be obtained from the signal tree in the I/O Manager.

Results

0	Link not active
1	Link OK
2	Link interrupted

Prerequisites

This function only returns a value if active input modules are installed and configured in the I/O Manager.

2.9.11 FobPlusControlLinkStatus

```
FobPlusControlLinkStatus('BoardNo*')
```

Arguments

'BoardNo*'	Board number (0 to 3)	

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function provides the link status of an ibaFOB-PlusControl board.

The card number (from 0 to 3) can be obtained from the 7-digit display on the card or the graphical presentation in the I/O Manager. The board only has one link.

Results

0	Link not active
1	Link OK
2	Link interrupted

Prerequisites

This function only returns a value if active input modules are installed and configured in the I/O Manager.

2.9.12 FobSDLinkStatus, FobSDexpLinkStatus

```
FobSDLinkStatus('BoardNo*')
FobSDexpLinkStatus('BoardNo*')
```

Arguments

'BoardNo*'	Board number (0 to 3)	

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns the link status of an ibaFOB-SD board (PCI) or ibaFOB-SDexp board (PCIe).

The card number (from 0 to 3) can be obtained from the 7-digit display on the card or the graphical presentation in the I/O Manager. The board only has one link.

Results

0	Link not active
1	Link OK
2	Link interrupted

Prerequisites

This function only returns a value if active input modules are installed and configured in the I/O Manager.

2.9.13 FobTDCLinkStatus, FobTDCexpLinkStatus

```
FobTDCLinkStatus('BoardNo*')
FobTDCexpLinkStatus('BoardNo*')
```

Arguments

'BoardNo*'	Board number (0 to 3)	

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns the link status of an ibaFOB-TDC board (PCI) or ibaFOB-TDCexp board (PCIe).

The board number (from 0 to 3) can be obtained from the 7-digit display on the card or the graphical presentation in the I/O Manager. The board only has one link.

Results

0	Link not active
1	Link OK
2	Link interrupted

Prerequisites

This function only returns a value if active input modules are installed and configured in the I/O Manager.

2.9.14 ICPSensorStatus

```
ICPSensorStatus('ModuleNo*','SensorNo*')
```

Arguments

'Module-No*'	Module number of a Padu-8-ICP module
'SensorNo*'	Sensor number (between 0 and 7)

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function monitors the status of an ICP sensor. The first 'ModuleNo' parameter indicates the module number of an ibaPADU-8-ICP or ibaMS8xICP-/ IEPE module in the signal tree. The second parameter, ‘SensorNo’, specifies which sensor should be monitored. The sensor number goes from 0 to 7.

Results

0	Sensor OK
1	open loop detected

2.9.15 InterruptCycleTime

```
InterruptCycleTime('Type=0*')
```

Arguments

'Type*'	Optional parameter (default = 0), with which the type of the result value can be determined.	
	'Type' = 0	The actual value is returned.
	'Type' = 1	The minimum (smallest measured cycle period) is returned.
	'Type' = 2	The maximum (largest measured cycle period) is returned.

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns the Interrupt cycle time in microseconds. The value is not updated with each Interrupt, but with a lower cycle.

2.9.16 InterruptTime

```
InterruptTime('Type=0*')
```

Arguments

'Type*'	Optional parameter (default = 0), with which the type of the result value can be determined.	
	'Type' = 0	The actual value is returned.
	'Type' = 1	The minimum (smallest duration measured) is returned.
	'Type' = 2	The maximum (largest duration measured) is returned.

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns the Interrupt time (Interrupt duration) in microseconds. The value is not updated with each Interrupt, but with a lower cycle.

2.9.17 IsProjectModified

```
IsProjectModified()
```

Description

This function returns the result true, if there are unsaved modifications in the current project.

2.9.18 LicenseInfo

```
LicenseInfo('InfoType*')
```

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function provides information about various properties related to the license container (dongle or soft license). This information can be used for display and diagnostic purposes.

Specify the information type ('InfoType') you want to receive. This type of information is determined and output at the start of the acquisition. If you want to receive more information from the license container then you need to configure the function several times, with a different type of information each time.

The following information types are supported:

Information types	Possible results
0: License container available	TRUE = license container available FALSE = license container missing or defective
1: License time limit in days	Value of the remaining lifetime of the license
2: Demo time limit in days	Value of the remaining lifetime of the license for trial periods / demo versions
3: Time limit in seconds before the acquisition will be stopped because of a license removal	Value of the remaining time which continues to run the system after the license container has been removed.
10: License container inserted counter	Number of times the license container was inserted
11: License container removed counter	Number of times the license container was removed
12: License container changed counter	Number of times the license container were changed

Table 9: Information types and possible results of the LicenseInfo function

Note

By evaluating information type 0, you can monitor whether a license container is present or available. In practice, you can use this function in *ibaQDR* to generate an email when it is detected that the license is missing (e.g., dongle removed). With *ibaQDR*, data collection continues for up to 4 days without a license to give you the opportunity to procure a new license container without jeopardizing production. This grace period does not exist with *ibaPDA*.

2.9.19 MultiStationStatus

```
MultiStationStatus()
```

Description

Returns the current multistation mode.

Results

0	Standalone
1	Slave
2	Master
3	Unsynchronized station

2.9.20 PerformanceCounter

```
PerformanceCounter("'Category*'","'CounterName*'","'InstanceName*'")
```

Arguments

'Category*'	Text entry from the "Object" column of the Windows Performance Monitor
'Counter-Name*'	Text entry from the "Performance Indicator" column of the Windows Performance Monitor
'Instance-Name*'	Text entry from the "Instance" column of the Windows Performance Monitor

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

The PerformanceCounter function can read, display and record the progress of certain performance characteristics of the computer. It returns the value of the performance counter or 0 if the performance counter does not exist. With the PerformanceCounter function, one performance indicator can be read in each case. If you want to display multiple performance indicators, then you have to configure the function repeatedly.

Example

Monitoring system performance

Solution

In order to select the performance characteristics that should be monitored under Windows 7, click on the Windows icon and search for Computer Management. Then on the left side under *System – Performance – Monitoring tools* you need to select the item *Performance monitoring*. You can add performance characteristics to be monitored with the green +- symbol in the top bar.

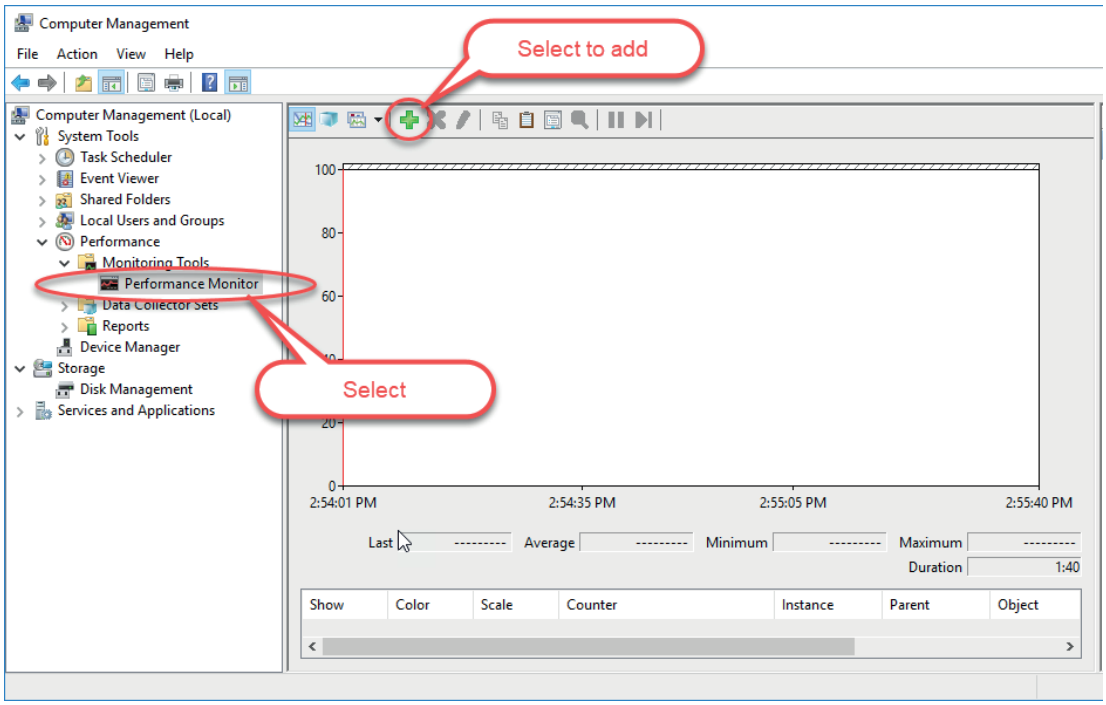


Fig. 2: Show new performance characteristics to be monitored

Select the performance indicators and then the instances. In this case, process, ibaPDA, and all instances are selected, then added and confirmed with <OK>. In the following illustration, the steps are marked and numbered.

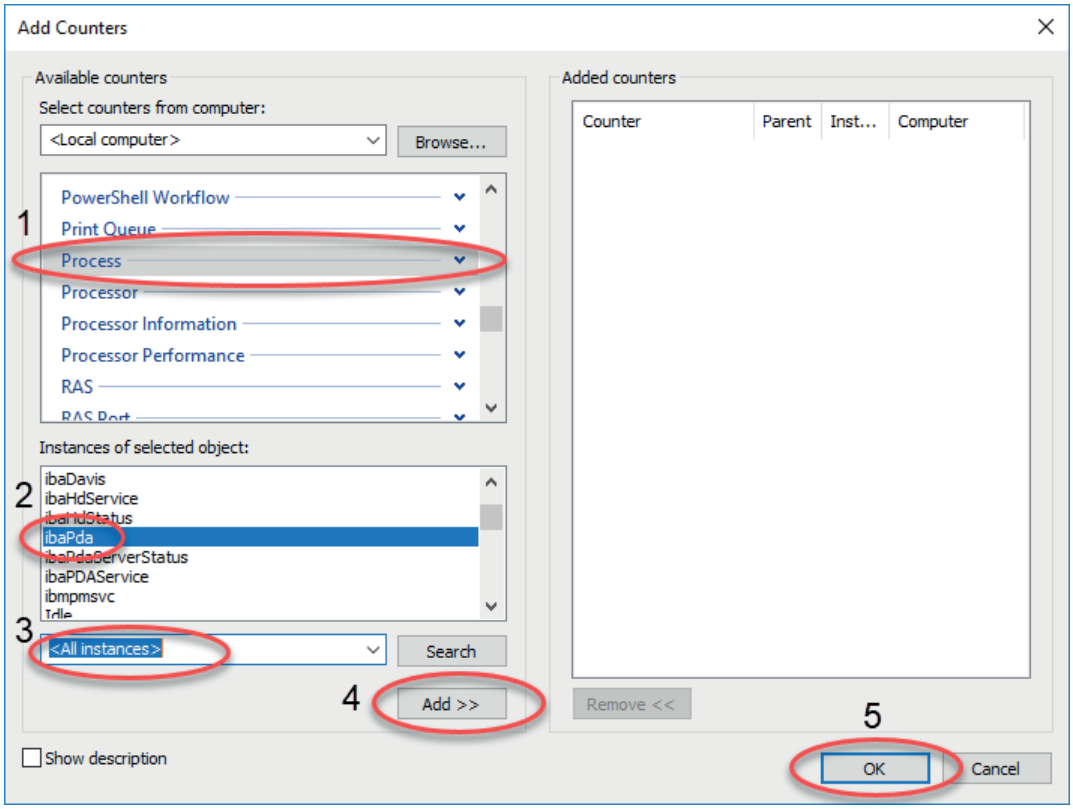


Fig. 3: Selecting the function arguments

The allocation to individual function arguments by PerformanceCounter is illustrated in the next illustration. You must accurately enter the names of the individual performance indicators, objects and instances for the entry in ibaPDA.

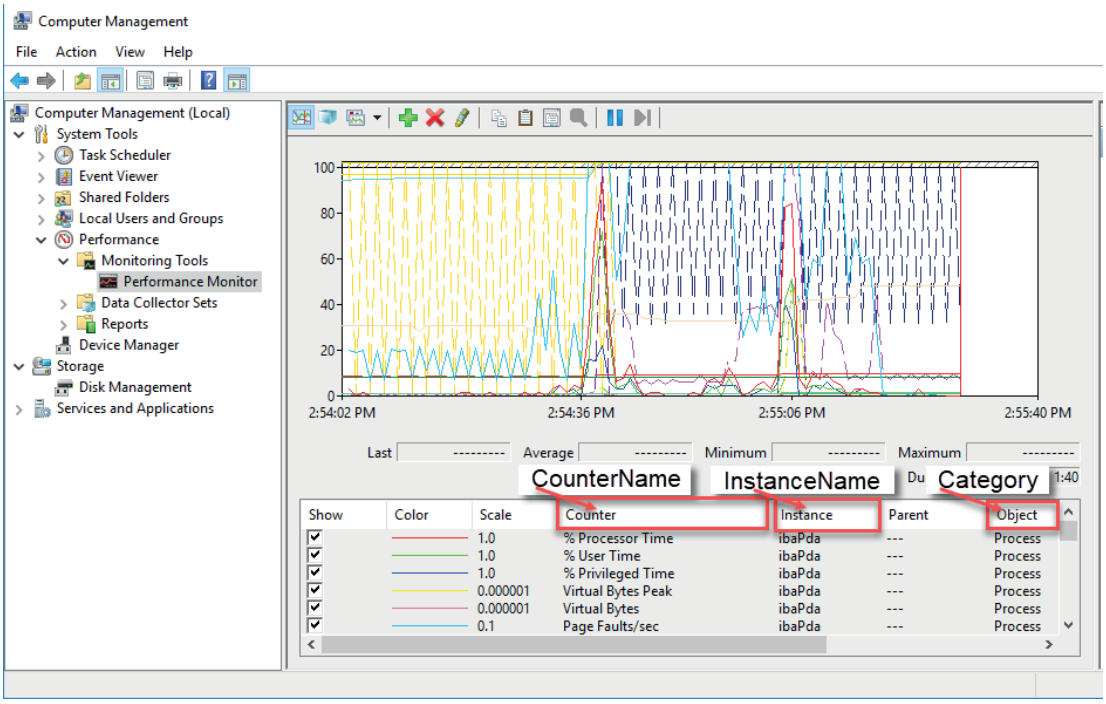
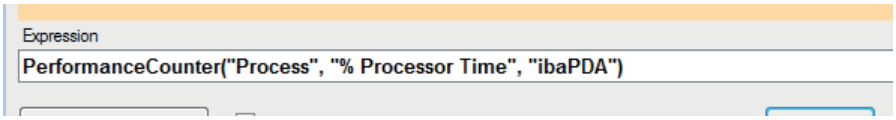


Fig. 4: Name of the function arguments

Example for the entry in ibaPDA for determining the processor time:



2.9.21 Ping

```
Ping("'Address'", 'Trigger', ' Timeout=5', ' Size=32')
```

Arguments

'Address'	IP address of the target computer
'Trigger'	Trigger signal (rising edge), with which ping is to be sent
'Timeout'	Optional parameter (default = 5) for setting the period to wait for a response before "no response" is shown as a result.
'Size'	Optional parameter: Size, in bytes, of the ping request (default = 32)

Description

This function sends a ping request to 'Address' upon a rising edge of 'Trigger'. The function returns the time, in milliseconds, needed before the ping response was received. If there is no response within 'Timeout' seconds then -1 is issued. The result is 0 as long as no ping request was sent. 'Size' determines the size of the ping request in bytes.

The function returns an analog value as the result.

Results

-1	no response to ping request within 'Timeout'
0	no ping request sent
n	Response time in ms

2.9.22 TimeSinceLastSync

```
TimeSinceLastSync()
```

Description

This function returns the time elapsed in seconds since the last time synchronization. If there was no time synchronization, the function returns -1.

Tip

Time synchronization is set in the configuration menu of ibaPDA then in the I/O Manager under the General node. Possible sources for the time synchronization are DCF77, IEC1131, PTP and HPCi via DGM200P or there is no time synchronization set.

2.9.23 TimeSyncStatus

```
TimeSyncStatus('Source*')
```

Arguments

'Source*'	Source for time synchronization	
	'Source' = -1	last source used for time synchronization
	'Source' = 0	DCF77 source 1
	'Source' = 1	DCF77 source 2
	'Source' = 2	IEC1131
	'Source' = 3	DGM200P
	'Source' = 4	PTP Slave
	'Source' = 5	ibaClock

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

This function returns the status of the selected time synchronization source.

The result can be as follows:

Results

0	Source is inactive
1	Source is active and valid
2	Source is active but not valid

Tip

Time synchronization is set in the configuration menu of *ibaPDA*, then in the I/O Manager under the General node.

2.10 Filter functions

2.10.1 BP

```
BP('Expression','Frenquency1*','Frequency2*')
```

Arguments

'Expression'	Measured value
'Frequency1*'	Specification of the lower limit of the frequency band for the filter
'Frequency2*'	Specification of the upper limit of the frequency band for the filter

Parameters ending with * are only evaluated once at the start of the acquisition..

Description

The function is a bandpass filter with a frequency band between the frequencies, 'Frequency1' and 'Frequency2'. The filter is a second-order Butterworth filter.

2.10.2 HP

```
HP('Expression','Frenquency*')
```

Arguments

'Expression'	Measured value
'Frequency*'	Specification of the limit frequency of the filter

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

The function is a high-pass filter with the 'Frequency' limit frequency. The filter is a second-order Butterworth filter.

2.10.3 LP

```
LP('Expression','Omega*','Reset=0')
```

Arguments

'Expression'	Measured value	
'Omega*'	Specification of the angular frequency of the filter, which results in the frequency of the LP function to frequency = 'Omega'/(2*PI)	
'Reset'	Optional digital parameter that can be used to deactivate the function. ‘Reset’ can be an expression as well.	
	'Reset' > 0	Filter is deactivated, input signal is displayed unfiltered
	'Reset' = 0	Filter is applied

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

The function is a low-pass filter with frequency = 'Omega'/(2*PI) applied to 'Expression'. The filter is a single-pole filter with a roll-off rate of 20 dB/decade. If the optional 'Reset' parameter is TRUE then the filter is deactivated and the result is the unfiltered 'Expression' input signal. 'Reset' can also be formulated as an expression.

Examples:

LP([0:0],10)	Filter is active ('Reset' omitted).
LP([0:0],10,If([0:0]<0,1,0))	The filter function is only used for positive values.
LP([0:0],10,[3.1])	e.g. with [3.1] = If([0:0]>10, 1, 0) The filter is deactivated as soon as the expression [3.1] returns TRUE, i.e. if the expression [0:0] exceeds the limit value 10.

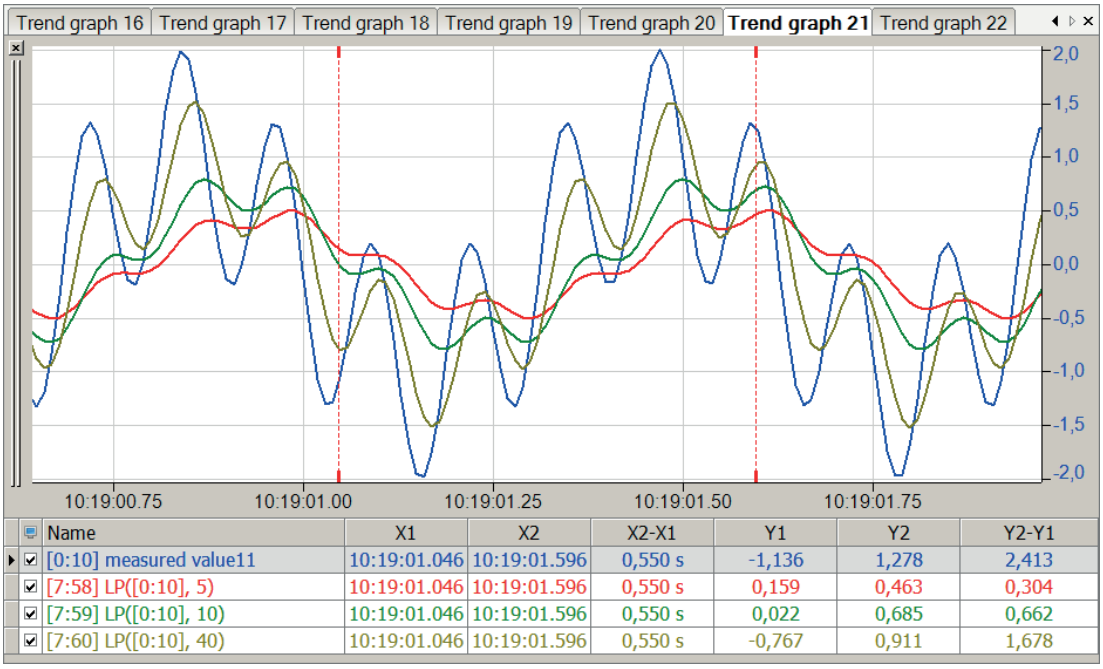
Example

Applying the filter function to a sine wave with 10 Hz, which is overlaid by a further wave with 50 Hz.

Task description

The filter function should be applied with the values 5, 10 and 40 for 'Omega'.

Solution



Name	X1	X2	X2-X1	Y1	Y2	Y2-Y1
[0:10] measured value11	10:19:01.046	10:19:01.596	0,550 s	-1,136	1,278	2,413
[7:58] LP([0:10], 5)	10:19:01.046	10:19:01.596	0,550 s	0,159	0,463	0,304
[7:59] LP([0:10], 10)	10:19:01.046	10:19:01.596	0,550 s	0,022	0,685	0,662
[7:60] LP([0:10], 40)	10:19:01.046	10:19:01.596	0,550 s	-0,767	0,911	1,678

Blue	Original signal of a superimposed sine wave	Red	Filtered signal with 'Omega' = 5
Green	Filtered signal with 'Omega' = 10	Yellow	Filtered signal with 'Omega' = 40

The lower the value selected for 'Omega', the more the signal is attenuated.

2.10.4 EnvelopeSpectral

```
EnvelopeSpectral('Expression','Frenquency1*','Frequency2*',' Cutoff frequency')
```

Arguments

'Expression'	Measured value
'Frequency1*'	Specification of the lower limit of the frequency band for the envelope calculation Value may not be less than 10% of the Nyquist frequency.
'Frequency2*'	Specification of the upper limit of the frequency band for the envelope calculation Value may not be greater than 90% of the Nyquist frequency.
'Cutoff frequency'	Optional specification of a limit frequency for the configuration of a low-pass filter which will be applied to the envelope.

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

The function calculates the spectral envelope of 'Expression' with a constant band pass between the 'Frequency1' and 'Frequency2' frequencies. The smallest bandwidth is 10% of the Nyquist frequency, i.e., 10% of half the sampling frequency.

The 'Expression' signal is rescanned at the time base of the virtual signal. No anti-aliasing filter is applied.

2.10.5 Preprocess

```
Preprocess('Expression', "'Preprocess profile name*'")
```

Arguments

'Expression'	Expression that should be preprocessed
'Preprocess profile name**'	Name of the preprocess profile or pre-processor

Parameters ending with * are only evaluated once at the start of the acquisition.

Description

The preprocess profile with the name, 'preprocess profile name,' is applied to 'Expression'.

Preprocess profiles can be configured in the pre-processor manager. You can access this manager via the expression editor of an Inspectra expert module.

2.11 Retentive functions

The functions in this group can be used in a *Virtual retentive* module. Functions with a retentive character always keep their last result value until they are deliberately reset.

When using these functions in a virtual retentive module, the most recently calculated result value can be saved by stopping the acquisition. When restarting the acquisition, this value can be reloaded as the initial value.

Thus, long-term data, such as operating hours counter and media consumption, are not lost by stopping and starting the acquisition.

Functions with this character are:

↗ *Add, page 29*

↗ *Avg, page 43*

↗ *Count, page 99*

↗ *CountText, page 90*

↗ *CountUpDown, page 102*

↗ *Int, page 34*

↗ *LastChangeOf, page 123*

↗ *Max, page 51*

↗ *Min, page 57*

↗ *SampleAndHold, page 127*

↗ *SetReset, page 20*

↗ *StdDev, page 68*

↗ *Toggle, page 24*

Note

Please note, that only functions, which support the retentive behavior, are allowed to be used in a virtual rententive module.

All other functions, including the basic arithmetic operations must not be used in a virtual rentive module and will cause errors.

2.12 Plugins

The *ibaPDA* plugin system was designed to enable *ibaPDA* users to create their own features to perform custom calculations of measurement data with *ibaPDA* in real time. The functions can be used in expressions for virtual signals just like built-in functions. They will also appear in the expression builder (plugins).

To customize functions, you have to create a NET.dll file. This dll-file can be written in any .NET language (C#, C++, VB.NET, …).

Other documentation

For a detailed description of the plugin programming and configuration, please refer to the "ibaPDA Plugin" manual.

3 Support and contact

Support

Phone: +49 911 97282-14

Email: support@iba-ag.com

Note

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

Product Security Incident Response Team (PSIRT) at iba

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

Email: psirt@iba-ag.com

Contact

Headquarters

iba AG
Gebhardtstrasse 10–20
90762 Fuerth
Germany

Phone: +49 911 97282-0

Email: iba@iba-ag.com

Mailing address

iba AG
Postbox 1828
D-90708 Fuerth, Germany

Delivery address

iba AG
Gebhardtstrasse 10
90762 Fuerth, Germany

Regional and worldwide

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

www.iba-ag.com

ibaPDA

Data Storage

Manual Part 5

Issue 8.14

Measurement Systems for Industry and Energy

www.iba-ag.com

Manufacturer

iba AG

Gebhardtstrasse 10-20

90762 Fuerth

Germany

Contacts

Main office	+49 911 97282-0
Support	+49 911 97282-14
Engineering	+49 911 97282-13
E-mail	iba@iba-ag.com
Web	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 2026, 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 and can be found in the iba help center docs.iba-ag.com.

Version	Date	Revision	Author	Version SW
8.14	06-2026	New: ibaQDR inspection line data store	rm	8.14.0

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

1 About this documentation 5

1.1 Target group and previous knowledge 5

1.2 Notations 5

1.3 Used symbols 6

1.4 Documentation structure 7

2 Introduction 8

3 Types of data storage 9

4 Datastore configuration toolbar 12

5 Data recording profiles 13

5.1 General 13

5.2 Profile types 14

5.3 Add and manage data recording profiles 15

5.4 "Time" profile 16

5.5 "Time, not equidistant" profile 17

5.6 "Time, DB/cloud" profile 20

5.7 "Time/length, ibaQDR inspection line" profile 22

5.8 "Length" profile 23

5.9 “Time, ibaHD” profile 24

6 Configuring data storage 27

6.1 Add a data storage 28

6.2 Basic settings for data storage 28

6.3 Trigger mode 30

6.3.1 Maximum file time 31

6.3.2 Start trigger 32

6.3.3 Stop trigger 37

6.3.4 Trigger pool 38

6.3.5 Advanced 40

6.4 Signal selection 41

6.4.1 Assigning a signal to a storage profile 43

6.4.2 Checking the profile assignment 43

6.4.3 Disabling the profile assignment 43
6.5 Video triggers 44
6.6 Image trigger 45
6.7 Data file information 46
6.7.1 Data file name 47
6.7.2 Data file location 49
6.7.3 Subdirectory organization 51
6.7.4 Files - Info fields 52
6.7.5 Data files - Cleanup strategy 53
6.7.6 Post-processing 55
6.8 Apply new data storage configuration 55
6.9 Configuration file of data storage 56
6.10 Loading a backup configuration file 58
7 Diagnostics for data stores 59
8 Stop prevention 62
9 Certificates 64
10 Support and contact 65

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 they are capable of assessing safety and recognizing possible consequences and risks on the basis of their specialist training, knowledge and experience and knowledge of the standard regulations.

1.2 Notations

In this manual, the following notations are used:

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

1.3 Used symbols

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

Danger!

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

Observe the specified measures.

Warning!

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

Observe the specified measures.

Caution!

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

Observe the specified measures.

Note

A note specifies special requirements or actions to be observed.

Tip

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

Other documentation

Reference to additional documentation or further reading.

1.4 Documentation structure

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.

Part 1	Introduction and installation	General notes, license policy and add-ons Installation and program start User interface, system architecture, client server User management, printing
Part 2	I/O Manager	Basic information about the I/O Manager, general settings Groups and vector signals, text signals, outputs, configuration files
Part 3	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.
Part 4	Expression builder	All functions for calculating virtual signals
Part 5	Data storage	Types of data recording, recording profiles, signal selection
Part 6	Data visualization	All display modes for live data, their operation and settings
Part 7	Appendix	Various additions, error listings, etc.

2 Introduction

Besides the display of data, the storage of measured data is one of the main features of *ibaPDA*. If the measured data are stored and available as data files (*.dat), you can view, analyze or post-process the data conveniently with *ibaAnalyzer*.

The data files generated by *ibaPDA* have a special signature that prevents the measured data being read or post-processed using third-party software. To maintain calculation accuracy, data types such as 32-bit integers or 64-bit floating points (double) are stored in the data file unchanged so that calculations can be carried out in *ibaAnalyzer* with full resolution. The datatype 64-bit floating point is also supported for virtual signals as results of calculations in *ibaPDA*.

The data storage management of *ibaPDA* offers a variety of options which enable you to customize the data storage according to your needs in order to get the maximum benefit.

The dialog for data storage configuration can be opened in the *Configuration – Data Store Manager...* main menu or by clicking on the button in the main tool bar.

3 Types of data storage

Depending on the licenses registered in the dongle or soft license, different types of data storage are available for configuration in the dialog.

Time-based data store
The data storage occurs with this type in the form of data files (*.dat), which can be evaluated with the program *ibaAnalyzer*. The data files generated in this way can be protected from unauthorized opening via password.

Usually, a (time-based) data storage and a blue link "Add data store" are shown in the data storage tree. With the basic version of *ibaPDA*, you can add one additional data store. You can purchase additional licenses if you should need more data storage:

- ibaPDA-Data-Store
  This license allows you to add 2 additional data stores per license.
- ibaPDA-Ultra-Data-Store
  This license allows you to configure up to 255 data stores with an average number of up to 20 signals or less per data store. If required you can configure data stores with more than 20 signals. In this case the number of remaining data stores will be reduced accordingly. For example, if you configure a data store with 110 signals the number of remaining data stores will be reduced by 6.

ibaHD data store
The ibaHD data store is a special type of data storage for recording and visualizing historical data. For this purpose, you need the license and software for *ibaHD-Server*.

For the HD data storage, you can choose between

- ibaHD time-based data store,
- ibaHD event-based data store
- ibaHD length-based data store

In case of a time-based HD data store you can additionally define HD time periods if the connected *ibaHD-Server* supports this feature (*ibaHD-Server* v3.1 or higher). Furthermore, HD time-based data stores can be triggered, e.g. in order to record data only when it makes sense (ibaHD-Server v3.2 or higher). You can use signals, time periods or system time for trigger.

In case of an event-based HD data store you'll have to define the events here.

In case of a lenght-based HD data store you'll have to define the measuring locations here.

In a basic version of *ibaHD-Server*, you will be able to set up 2 ibaHD stores. Licenses for additional ibaHD stores are available.

Other documentation

For more information about ibaHD data storage, please refer to the *ibaHD-Server* manual.

ibaQDR data store

If you use an *ibaQDR* system or have a corresponding license registered, the types *ibaQDR data store* and *ibaQDR inspection line data store* will also be displayed in the "Add data store" dialog. The dedicated data store type for inspection lines contributes to the special requirements of this kind of plants.

It is possible to configure more than one *ibaQDR* data store per system.

A special property of an *ibaQDR* data store is the time and length-based storage of measured values.

Other documentation

For more information about *ibaQDR* data store, refer to the manual for *ibaQDR*.

DB/cloud time-based data store

If you have a license for a database or cloud memory, such as *ibaPDA-DataStore-SQL-Server*, then you can use this form of data storage. iba offers different licenses graded by the number of signals.

The measured values are written (streamed) directly into a suitable database or cloud. The data is always written in a time-based manner.

The following database/cloud systems are supported: Oracle, SQL-Server, MySql, PostgreSQL, SAP-HANA, Azure SQL and MariaDB.

Other documentation

You will find additional information about DB/cloud data store in the manual on the corresponding product, such as *ibaPDA-Data-Store-SQL-Server-xxx*.

Kafka cluster time-based data store

If you have a license *ibaPDA-DataStore-Kafka*, then you can use this form of data storage. iba offers different licenses graded by the number of signals.

The measured values are written (streamed) directly into an Apache Kafka cluster. The data is always written time-based.

Other documentation

You will find additional information about Kafka cluster data store in the manual on the product *ibaPDA-Data-Store-Kafka-xxxx*.

MQTT time-based data store and Sparkplug B data store

If you have a license *ibaPDA-DataStore-MQTT* , then you can use this form of data storage. iba offers different licenses graded by the number of signals.

The measured values are written (streamed) directly to an MQTT broker. The data is always written in a time-based manner.

Other documentation

You will find additional information about MQTT data store in the manual on the product *ibaPDA-Data-Store-MQTT-xxxx*.

InfluxDB time-based data store

If you have a license *ibaPDA-DataStore-InfluxDB* , then you can use this form of data storage. iba offers different licenses graded by the number of signals.

The measured values are written (streamed) directly to an InfluxDB server. The data is always written in a time-based manner.

Other documentation

You will find additional information about InfluxDB data store in the manual on the product *ibaPDA-Data-Store-InfluxDB-xxxx*.

Note

Whenever you have changed the configuration of one or more data stores you should close the configuration dialog with the <OK> button. At this time, the recording of data stores which were subject to change will be stopped and closed. Other, unchanged data stores will continue to record. The recording of the changed data stores will restart automatically with the new configuration.

Important: Changes in the data storage configuration will not affect the data acquisition of the server. The server continues the data acquisition even if the recordings are stopped and restarted.

4 Datastore configuration toolbar

The *Data Store Manager* provides the following controls for data store configurations.

Toolbar

Symbol	Description
	New configuration; then choose whether an empty configuration (without data-store, profile *As is* only) or a new default configuration should be created. The default configuration consists of a pre-configured timebased datastore (data file). All signals in the I/O configuration will be selected and assigned to the profile *As is*.
	Opens an existing configuration file (XML-file .ds or .zip).
	Opens an existing configuration file (.ds or .zip) in simulation mode. The loaded configuration is not aligned with the current licenses an cannot be applied. The simulation mode only provides for display of any datastore configuration.
	Opens access to the automatically saved backups of the datastore configuration in the path `%ProgramData%iba\ibaPDA\Backup` on the *ibaPDA*-Server computer. In a dialog window you can select and load the desired backup. Furthermore, you can use the dialog to delete a backup.
	Saves the current, full datastore configuration in an XML-file. You can select or enter the path and filename in the following dialog. The file extension .ds is added automatically.
	These buttons are disabled, unless you can move selected datastores. Move the selected datastore by one position up or down in the tree.
	Go forward/backward buttons: Navigate to the previous or following step in the configuration.

5 Data recording profiles

The data recording profiles are used to determine how the data should be recorded. There are different options depending on the type of data recording (time-based, length-based, DB/Cloud, etc.). Basically, you should first configure the profiles for your planned data stores. Then link the signals to the desired profile.

5.1 General

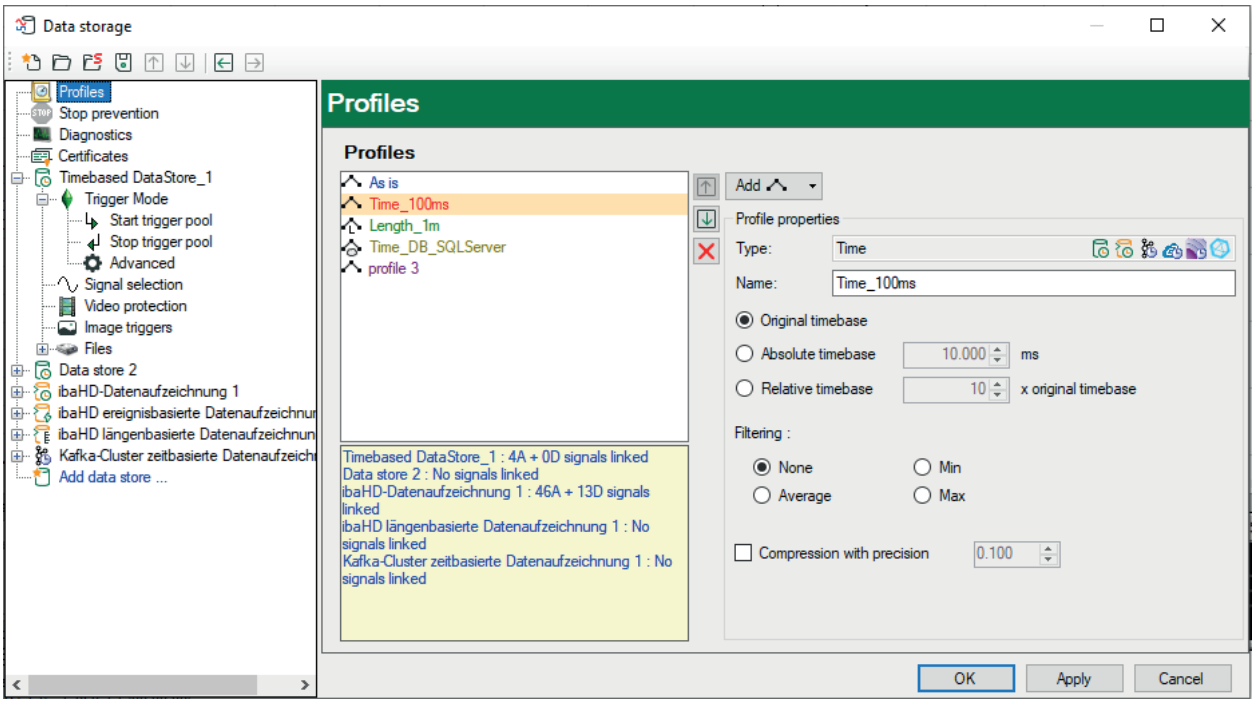
The recording profile function primarily serves to reduce data volumes for time- and length-based data stores. Even when the process phases relevant for the recording have been narrowed down with the trigger settings, still a large number of values is generated that can quickly fill up the storage media. Particularly with general long-term data storage, it is not often necessary when using measured values that they will be available at a later point in time at a high-resolution sampling speed, e.g., 1 millisecond. Additionally, the signal type also determines whether it must be recorded at high or low cycle rates. Rapidly changing signals like currents or forces are usually recorded with a smaller time base than slow-changing signals like temperatures.

Since each signal can be individually linked to a recording profile, it is also possible to differently record signals coming from the same module.

All profile types except the non-equidistant recording have the option of a statistical filter. If the signals are acquired much faster than recorded according to the profile setting, you can use the filter to determine which value (actual value, maximum, minimum or average value) should be saved in relation to the values acquired in the recording interval.

For DB/Cloud data stores, the profile is primarily used to match the target database for ID and value columns.

The following figure shows a typical time-based profile.



Click on the *Profiles* branch in the tree structure at the left of the configuration dialog. In the right-hand part, you can now see a list of available profiles (at least "As is"). Next to the list, you will find more information on the profile you have selected in the list. Below the list, you see for which data storage the selected profile is used and how many signals are assigned to it.

Any number of storage profiles can be defined. Each profile is stored with a unique name assigned by the user, so that the allocation of profiles to the signals is very clear and comprehensible.

5.2 Profile types

ibaPDA offers different recording profiles for the different types of data store. The icon behind the profile name indicates the data store type, which can use this profile.

Profile type	Symbol Profile type	Type of data store	Symbol data store
Time		time-based data store (*ibaPDA* data file)	
		ibaHD time-based data store	
		Kafka cluster time-based data store	
		MindSphere time-based data store	
		InfluxDB time-based data store	
		MQTT time-based data store	
Time, not equidistant		time-based data store (*ibaPDA* data file)	

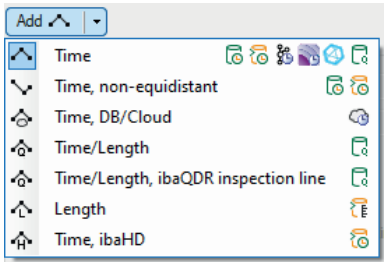
Profile type	Symbol Profile type	Type of data store	Symbol data store
Time, DB/cloud		DB/cloud time-based data store (e.g. SQL-Server)	
Time/length, ibaQDR inspection line		time and length-based data store (*ibaQDR* data file)	
Length		ibaHD length-based data store	
Time, ibaHD		ibaHD time-based data store with 3 different modes	

Table 1: Profile types and appropriate types of data store

5.3 Add and manage data recording profiles

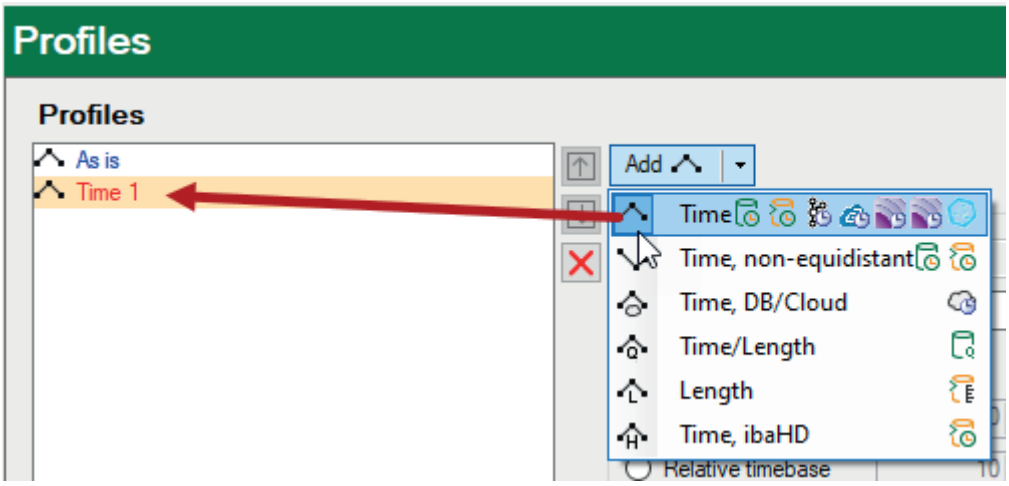
In order to add a new profile, select the *Profiles* branch in the tree structure of the data storage configuration dialog.

In the right window, you will find an <Add> button. An icon on the button indicates the profile type that would be added upon click. If the icon matches the profile type you wish to add, click the <Add> button. If you are not sure about the profile type, just click on the dropdown-list icon on the button in order to show all profile types available.



Select the desired profile type from the dropdown list.

A new profile is added to the profiles list.



The new profile is highlighted with a yellow background.

Then you should define the profile properties (see further below).

Next to the profile list, there are buttons with the following functions:

	Move the selected profile up in the sequence. The icon is not available (gray) if the operation is not possible.
	Move the selected profile down in the sequence. The icon is not available (gray) if the operation is not possible. The "As is" profile is always at the top of the list.
	Delete the selected profile.

5.4 "Time" profile

As the name says, this profile type is used for general time-based recording. The profile stores measured data with equidistant samples. The type is available by default and can be used for the following types of data store:

- Time-based data store
- *ibaHD* time-based data store
- Kafka cluster time-based data store
- MindSphere time-based data store
- MQTT time-based data store
- InfluxDB time-based data store

Profile properties

Type
Display of the profile type with the appropriate types of data store (symbols)

Name
Enter an unambiguous and understandable name here.

Time base
Choose whether you like to record the acquired data with the original time base (as configured under *General settings* in the I/O Manager) or with another time base (absolute time base). If you select *Absolute time base*, please define the desired time base in the field, given in milliseconds.

Upper limit value is 214748 (approx. 3.58 min.)

If you decide to use a *Relative timebase*, enter or set an integer factor to get a multiple of the original timebase. Upper limit is 100,000.

Filter
Select the signal value to be stored with reference to the time base set above.

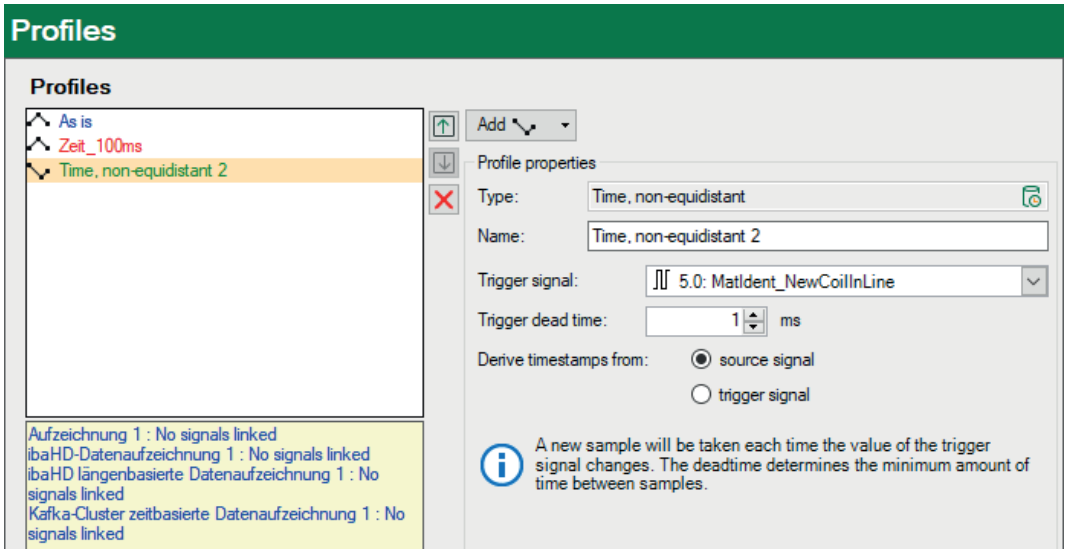
- None: The actual value will be stored
- Average: The arithmetic average of all measured values, which were acquired in the interval (according to the time base, such as 10 ms or 100 ms) is formed and saved.

- Min: Only the smallest value of all samples measured within the defined interval will be stored.
- Max: Only the largest measured value of all measured values acquired within the defined interval will be stored.

Compression with precision
Use this option to achieve a further reduction in data. Method:
every value is subtracted from the previous one. The (absolute) difference is compared to the accuracy value defined in the field nearby. The new sample will only be stored if the difference exceeds the accuracy value (based on the physical unit of the signal). If the difference is equal or less than the accuracy value, an internal counter in the data file will be incremented and the sample will not be stored. Storage space can be saved thereby.
Usually, this method is suitable for signals with a general noise or for sensors with less accuracy. The accuracy value indicates the absolute range given in physical units in which a signal sample can vary without being stored as a new value.

5.5 "Time, not equidistant" profile

This profile type has been developed for specific requirements. It is suitable for situations with an controlled irregular occurrence of measurement data, depending on the process event. It is also suitable, for instance, if you just want “snapshots” of the process data to be triggered by a control signal.



Profile properties

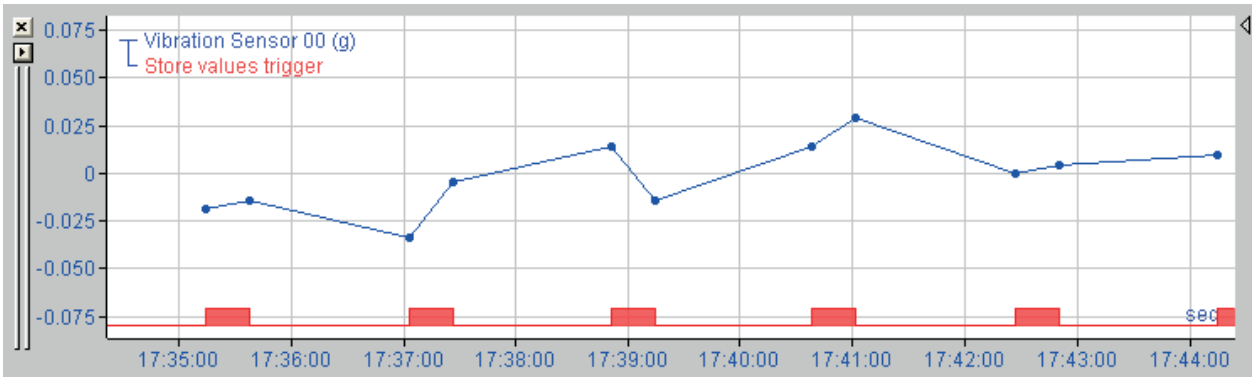
Type
Display of the profile type with the appropriate types of data store (symbols)

Name
Enter an unambiguous and understandable name here.

Trigger signal
Select a signal from the dropdown list to be used as trigger for writing samples in the data file.

Every time the value of the trigger signal changes, a new sample will be stored.

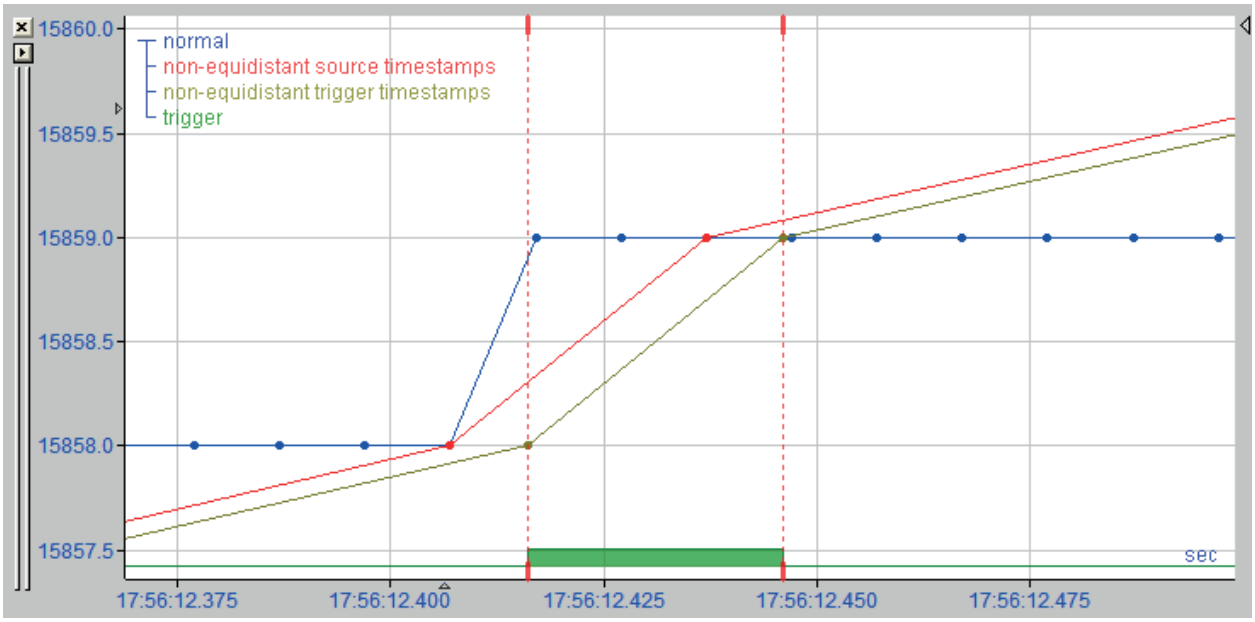
If you use an analog value for triggering, samples are with high probability recorded continuously unless the analog value remains 100 % constant over periods where no sample should be stored. Using a digital signal as trigger will store samples with every rising/falling edge, as shown in the figure below.



Trigger dead time
The trigger dead time determines the minimum period between the samples. Trigger events that occur during the trigger dead time are ignored.

Derive timestamps from source signal/trigger signal
If the trigger signal and source signal have different time bases, you can choose which timestamps to use for the non-equidistant signal. “Source signal” is every signal assigned to this profile.

Explanation
The screenshot below shows another example of the difference between the two options.



The blue signal is the source signal. The green digital signal is the trigger signal. The source signal is sampled at 10 ms. The trigger signal is sampled at 1 ms. You can see that the trigger value changes at a time that does not correspond with the sample of the source signal.

The red signal is the non-equidistant signal that uses the timestamps from the source signal. You can see that the sample points of the red signal correspond with the sample points of the original signal. If you choose source signal timestamps, the timestamp will be smaller or equal to the trigger timestamp.

The brown signal is the non-equidistant signal that uses the timestamps of the trigger signal. You can see that the sample points of the brown signal correspond with the sample points of the original signal moved to the trigger timestamps.

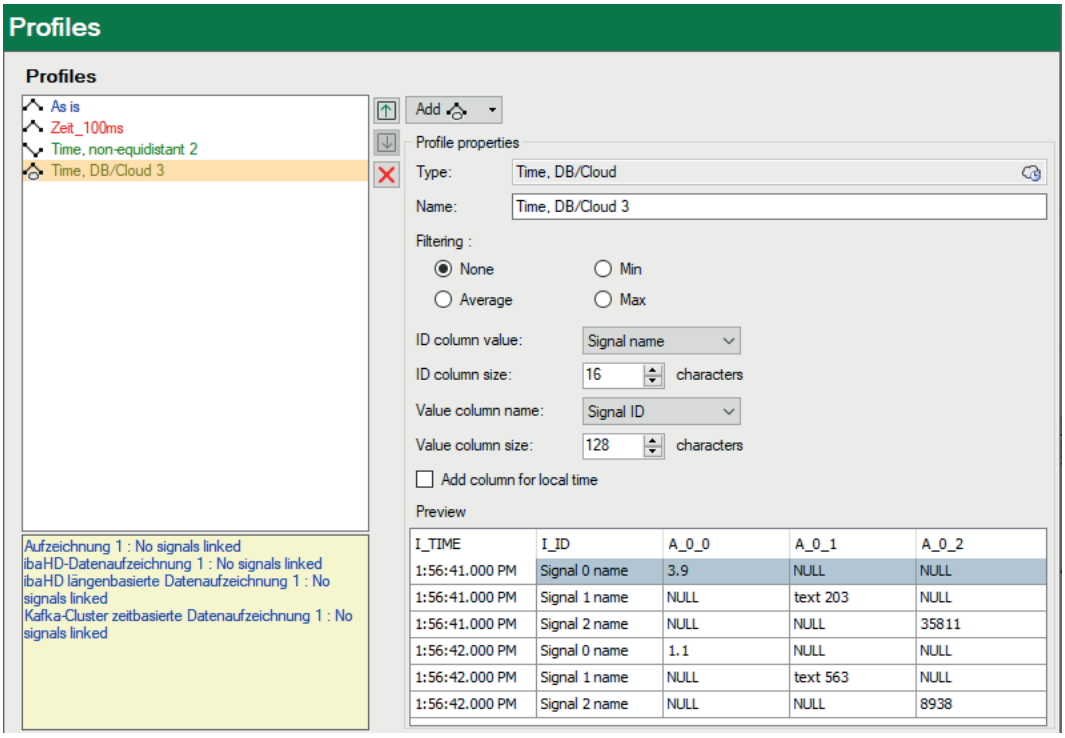
Both options have advantages as well as disadvantages.

Source timestamps
The advantage of source timestamps is that the original samples are used. The disadvantage is that if you have multiple source signals with different time bases, the sample points of the generated non-equidistant signals will have different timestamps. If you subsequently perform calculations based on these non-equidistant signals, they will be converted to equidistant signals and the calculations will be performed on equidistant signals.

Trigger timestamps
The disadvantage of trigger timestamps is that the original sample points are moved. The advantage is that all non-equidistant signals generated by the same profile have the same timestamps irrespective of their original time bases. Calculations are performed on the non-equidistant signals, and as result, a non-equidistant signal is given.

5.6 "Time, DB/cloud" profile

Use this profile if you want to write the measured data in supported databases by means of a DB/cloud time-based data store.



Profile properties

Type

Display of the type with the appropriate types of data storage (symbols)

Name

Enter an unambiguous and understandable name here.

Filtering

Select the signal value to be stored with reference to the time base set above.

- None: The actual value will be stored
- Average: The arithmetic average of all measured values, which were acquired in the interval (according to the time base, such as 10 ms or 100 ms) is formed and saved.
- Min: Only the smallest value of all samples measured within the defined interval will be stored.
- Max: Only the largest measured value of all measured values acquired within the defined interval will be stored.

Value of ID column, size of ID column

A table can also have an ID column, which can be used as the key. This column is denominated ID in *ibaPDA*. You can set whether you want to use such a column or not. If you want to use such a column, you can configure which value should be added. You can choose from the following:

- None: No ID column is used
- Fixed: A fixed text is written in the ID column. This can, for example, be used as identification of the *ibaPDA* system if several *ibaPDA* systems write in one table.
- Text signal: The value of a text signal
- Signal ID: The ID of the signal to which this profile is assigned, e.g. [10:2]
- Signal name: The name of the signal to which this profile is assigned
- Signal comment 1: The first comment of the signal to which this profile is assigned
- Signal comment 2: The second comment of the signal to which this profile is assigned

The ID column is created with the data type NVARCHAR(x), whereby x is the size that was set in the field *Size of the ID column*.

Name of the value column, size of the value column
Use this option to determine the values column in which the value of each signal assigned to this profile is written. Available for selection are:

- Fixed: A fixed column name. This means that the data of each signal is written in a separate line. This also means that you should use an ID column to be able to determine the signal to which the data belongs.
- Signal ID: The column name is derived from the signal ID.
- Signal name: The column name is derived from the signal name.
- Signal comment 1: The column name is derived from the first comment of the signal.
- Signal comment 2: The column name is derived from the second comment of the signal.

If you do not choose “fixed,” then a line will receive the data from several signals. The notation is more efficient, but not mandatory.

If you want to write text signals in the table, you must specify the size in number of characters.

Add column for local time
The default time column I_TIME shows the UTC time. If you enable this option a column showing the local time will be added.

Preview
The preview shows which columns will be created based on the ID and value columns settings.

Other documentation

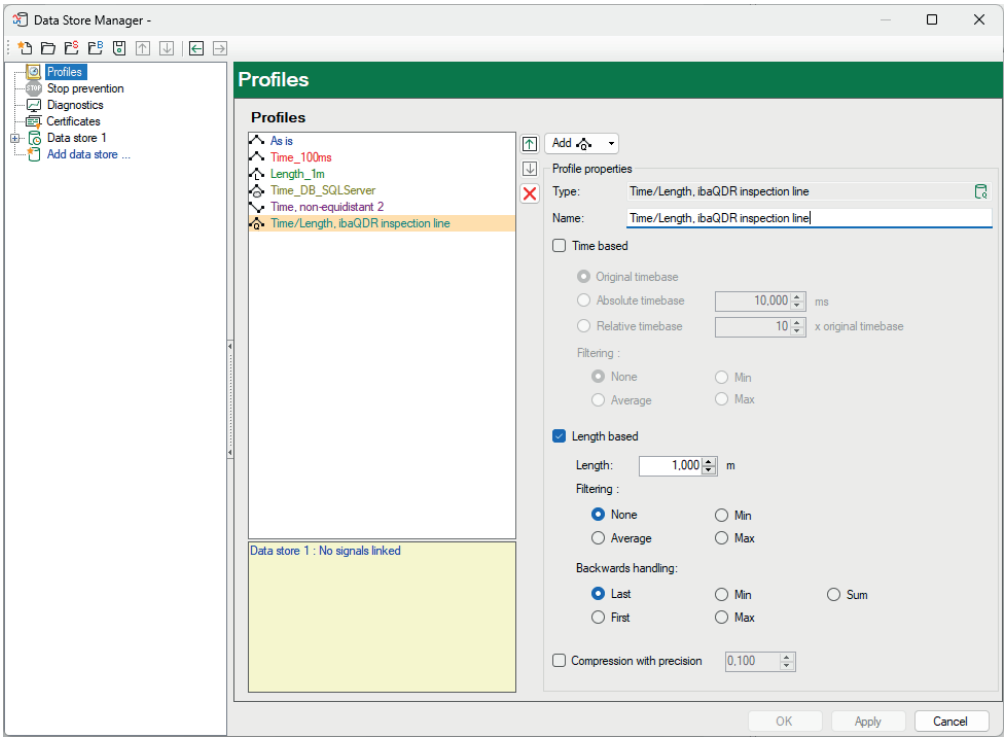
You will find more information about DB/cloud data storage in the respective manual for the product, such as *ibaPDA-Data-Store-SQL-Server-xxxx*.

5.7 "Time/length, ibaQDR inspection line" profile

With this profile, both time and length-based data is recorded in data files. Since this profile type can only be used with *ibaQDR* and when using an *ibaQDR data store* or *ibaQDR inspection line data store*, it is also called a QDR profile.

When adding an *ibaQDR data store*, two base profiles are automatically created. When using *ibaQDR*, you may add further QDR profiles if required.

The QDR profiles are not available for signal selection for other types of data storage.



Profile properties

Type

Display of the type with the appropriate types of data storage (symbols)

Name

Enter an unambiguous and understandable name here.

Time-based, length-based

By enabling one of these options or both of them, you determine how the data is written in the data file.

With the time-based recording, the settings are the same as for the “Time” profile.

With the length-based recording, the settings correspond. A length interval can be set here instead of an absolute time interval. The standard deviation is 1 m. You can enter both larger and smaller values.

Other documentation

The profile *Time/length* is only suitable for *ibaQDR data stores* or *ibaQDR inspection line stores*. You will find more information about this in the manual for *ibaQDR*.

Note

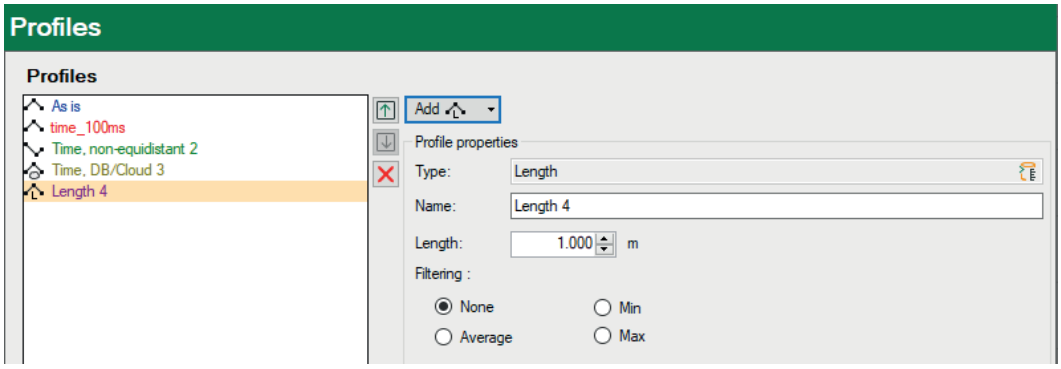
Values are always stored in REAL format as soon as a storage profile is assigned to the signals (except "As is"), even if the original samples are of INT16 format (e.g. from an ibaPADU)!

Exception:

32 bit integer values are not converted to floating point values when written to a data file.

5.8 "Length" profile

This profile type is used to record measured values in a length-based ibaHD data storage. It determines at what intervals (meters) measured values are written in an ibaHD store. The conversion of measured values to a length base is configured in the definition of the ibaHD data storage.



Profile properties

Type
Display of the profile type with the appropriate types of data store (symbols)

Name
Enter an unambiguous and understandable name here.

Length
Adjust the length of the section interval here, at which a measured value should be recorded. The default setting is 1 m.

Filter
Select which value of a signal, related to the set distance interval, is to be stored in the record.

- None: The actual value will be stored
- Average: The arithmetic average of all measured values, which were acquired in the interval (according to the time base, such as 1 m or 10 m) is formed and saved.
- Min: Only the smallest value of all samples measured within the defined interval will be stored.
- Max: Only the largest measured value of all measured values acquired within the defined interval will be stored.

Other documentation

For more information about ibaHD data storage, please refer to the *ibaHD-Server* manual.

5.9 “Time, ibaHD” profile

This profile is dedicated for timebased recording in ibaHD-Server. The profile offers 3 different modes, which support the following types of recording:

- As is
- Equidistant
- Non equidistant

Furthermore, the profile allows to adapt the aggregation factor.

Profile properties

Type
Shows the type with the appropriate data storages (symbols)

Name
Enter an unambiguous and understandable name here.

Mode
Select the the desired recording mode.

- As is
  The signals are stored as they are. This means, equidistant signals will be stored equidistantly, non equidistant signals will be stored non equidistantly. There are no further settings except the aggregation mode.

- Equidistant
  The signals are resampled. The new timebase can be configured as an absolute timebase or an integer multiple of the original timebase. Filtering can be used to determine which value will be stored. This profile has the same options as the standard “Time” profile.

  For more information about this recording mode see ↗ *"Time" profile, page 16*

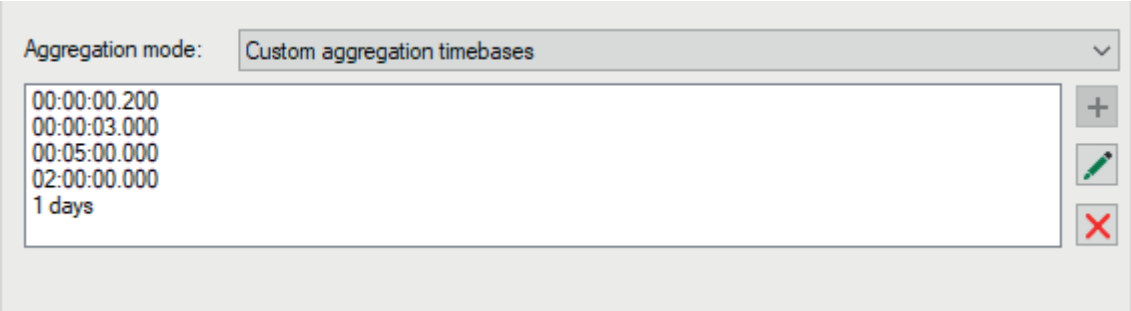
- Non equidistant
  A sample is taken and stored each time the value of the trigger signal changes. In the case of a digital trigger signal a sample will be taken on both rising and falling edges. The timestamp of the sample can be taken from the source signal or from the trigger signal. This is only relevant if the time bases of source and trigger signal are different. This profile has the same options as the standard “Time, non-equidistant” profile.

For more information about this recording mode see ↗ *"Time, not equidistant" profile, page 17*

Aggregation mode
For all 3 modes the profile offers the option to set the timebase for the different aggregation levels automatically or manually.

If you want to set the timebase manually, select *Custom aggregation timebases* in the drop down list of this field.

The current timebases will then be displayed.

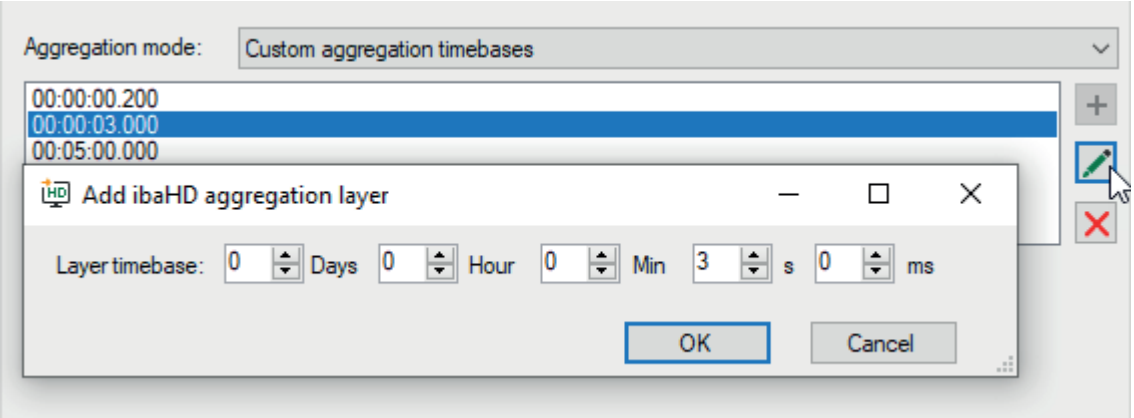


On the right side you will find a number of buttons with the following functions:

+	Add new timebase; only available if there are less than 5 timebases.
	Edit selected timebase
	Remove selected timebase

If you want to change a timebase, select it in this list and click on the <Edit> button.

You then can change the value in the following setup dialog.



Other documentation

For more information about the aggregation levels, please refer to the *ibaHD-Server* product manual.

6 Configuring data storage

The term *data storage* describes a method for storing measured data in data files or other media depending on particular trigger events or other controls. You may define and configure any number of data storages, which are capable of recording data independently from each other.

Note

The following chapters explain the key configuration steps using the example of the standard data storage (time-based in data files). Many of these descriptions also apply accordingly for other types of data storage.

Configuration steps deviating from this and settings for other types of data storage, such as *ibaQDR*, *ibaHD*, DB/Cloud, etc. can be found in the respective manuals for the software products.

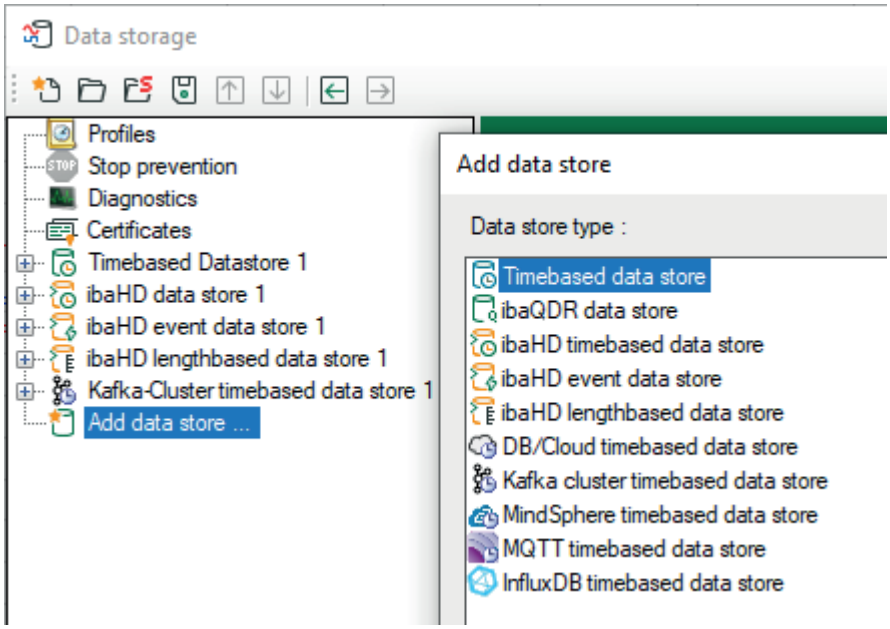
The number of independent data storages is limited by the license key (dongle). A *ibaPDA* standard package contains two data storages in data files, which can be used at the same time. With additional licenses, upgrades in steps of two additional data storages are available. Each data storage creates its own data files using its own recording parameters such as storage profiles, trigger settings and hard disk cleanup strategy. All data storages can work simultaneously and create data files. Moreover, each data storage can create an overlapping recording, consisting of two or more files. The available data storages are listed in the tree of the data storage dialog, each with its sub-branches *Trigger mode, Signal selection* and *Files*.

Note

The following restrictions apply to data storages in data files:

- Module numbers: 0 to 1048575
- Highest signal number: 524287
- Highest sub-signal number: 16383 (e.g. for bit decoder)

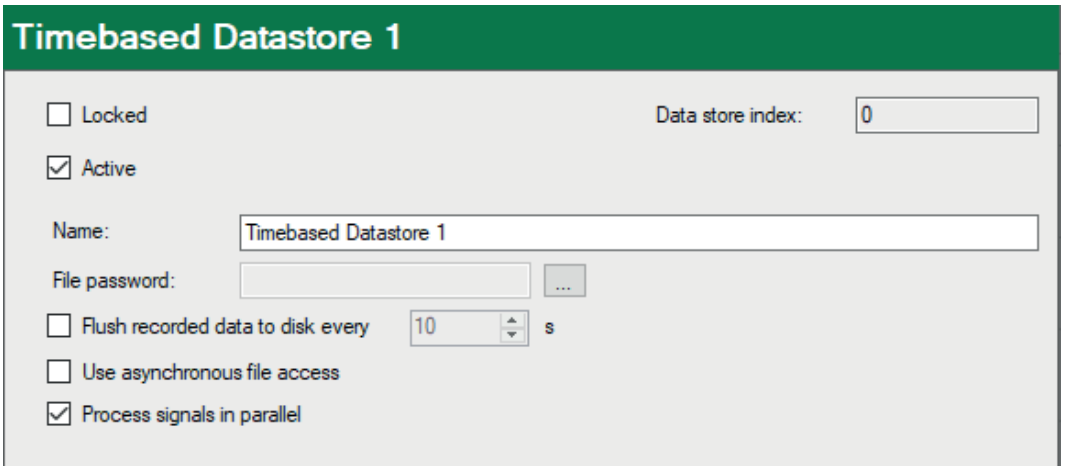
6.1 Add a data storage



There is automatically a standard data storage (time-based) available, but no signals are assigned to it. You may rename and configure this default data storage. In order to add a new data storage, click on the blue link *Add data storage* in the tree structure. You can also right-click on the data storage name in the tree and choose *Add data store* from the context menu.

If you do not assign a storage profile to the signals to be recorded, they will be assigned to the profile "As is". In this case, the signals are recorded with the original time base according to the General settings or the Module settings.

6.2 Basic settings for data storage



Locked
If you enable this option, then a data storage configuration can only be changed or disabled if the current user has the corresponding rights with the user management enabled.

Active
A data store must be enabled in order to work. The advantage of configuration and activation being separated is that you can define and configure several data storages without being obliged to use them all at the same time when you disable the storages you do not require. Just disable the data storages you do not want to use.

To enable a data storage, click on the major branch of the relevant data storage and check the *Active* option.

Data store index
This index is automatically assigned by the system and is only shown here. The index is increased by 1 every time a data store of the same type is added. The index is important if the diagnostic functions DataStoreInfo…() are to be used. For each data storage type there is a corresponding function (DataStoreInfoDB, DataStoreInfoHD etc.) In this function the index must be entered as an operand in each case in order to receive the diagnostic data from the correct data store.

Data store name
Enter here a meaningful name for the data storage here.

File password
You can protect the data file from being opened without authorization by using a password. The password must be at least 6 characters long.

To open a protected data file with *ibaAnalyzer* later, the user must know and enter the assigned password here. If no password has been set yet or if you want to change the password, click the <…> button right next to the field.

The file password is also used by the playback interface and the offline trend graph (*ibaQPanel*) as well as by *ibaDatCoordinator* (post-processing) and *ibaHD server* (import), provided it has been configured accordingly there.

Flush recorded data to disk every … s
ibaPDA writes measured data automatically into the data file while making optimal use of the integrated data compression.

Enabling this option prompts the system to write the data to the data file regularly after an adjustable time period has elapsed. This is necessary, for example, if data files will be analyzed with *ibaAnalyzer* while they are still being written. If this option is enabled, however, the data files will become larger, since data compression can no longer be used optimally.

Use asynchronous file access
By default, this option is disabled. This means that all write operations for data storage and also data acquisition are processed sequentially in one thread.

Especially when using external drives for data recording (NAS, USB disks), where access is significantly slower than with an internal, local hard disk, data acquisition may be slower and the interrupt buffer will fill up. In the worst case, the buffer will become full and data acquisition will stop. This can also occur when writing to different partitions of the same physical drive, which is still being used by other processes.

Another disadvantage is the failure of all data stores if one drive is not available, despite the fact that different drives are being used.

Enabling this option decouples data acquisition and data storage, i.e., data acquisition and data storage are then processed in different threads. This prevents data acquisition from being negatively affected by slow file access times during data storage.

When using multiple physical drives (NAS, USB) for different data stores, if one drive is ever unavailable, only the corresponding store is affected. The other stores can continue writing.

The disadvantage of this operating mode is a higher memory requirement. The buffer memory for write commands to be executed is 500 MB per data store. If this limit is reached, then the recording stops and restarts after 5 s.

Process signals in parallel
This option is especially advantageous if you want to process and store a large number of signals. Parallel processing significantly increases performance.

6.3 Trigger mode

In order to open the trigger mode dialog, click on the "Trigger mode" subbranch of the relevant data storage in the tree.

This dialog contains a number of settings relevant for data storage control. Different trigger modes for starting and stopping a data storage are available. You can specify parameters such as pre- and post-trigger time and the behavior of the data storage in specific situations.

All settings remain until the next change, even if the acquisition is stopped or the program is closed. Any change in the settings is saved to the current data storage configuration file at the time you press the <OK> button and exit the dialog.

Note

The trigger itself does not affect the measurement. It only determines when and how the data is saved.

Triggers only work when the measurement is running.

6.3.1 Maximum file time

This setting specifies the time limit of a data file. No data file of the corresponding data storage will exceed this time limit.

The limitation also applies to triggered recordings. This is particularly useful for avoiding "over-long" data files in case a defined stop trigger does not fire, e. g. due to incorrectly set levels or edges. The time limit is dominant over the stop trigger. If you want to use a stop trigger for stopping the recording, make sure that the time limit is high enough for the expected recording time.

If neither a start nor a stop trigger is set, the recording is only time-controlled. This is a continuous recording with a number of consecutive data files of the same length, i.e. recording time.

You may directly enter the values for Hour, Minute and Second or use the arrow buttons.

The maximum recording time is one week (168 h : 59 min : 59.9 s).

Please note, that the data file size can grow significantly depending on number of signals, profile and file time. The data file size is limited to 64 GB.

6.3.2 Start trigger

Trigger modes

You can choose between several trigger modes when using a start trigger for data storage:

No start trigger

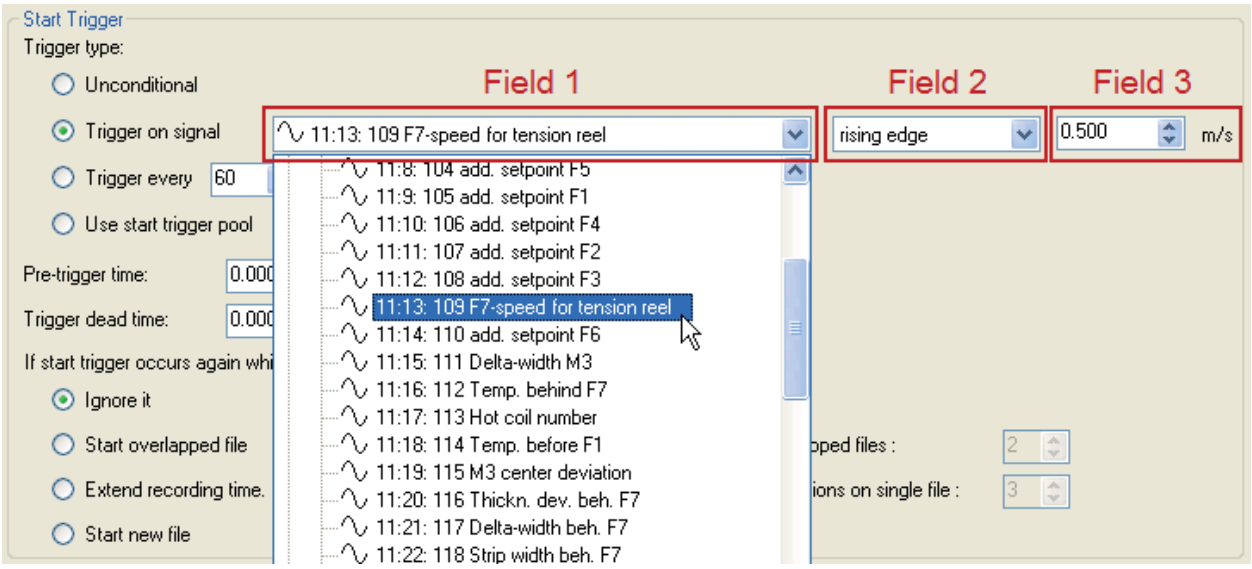
If this item is checked, no trigger is enabled. In this case, the recording will start immediately at the start of the measurement or when pressing the "Go" button.

Trigger on signal

If you want the trigger to fire on a measured signal or a virtual signal, you need to check "Trigger on signal" in the trigger mode dialog.

- Field 1 Dropdown list for signal selection
  (available analog and digital signals)
- Field 2 Dropdown list for selecting edges or levels
- Field 3 Dropdown list for selecting the trigger level value
  given in the specific physical unit
  (field 3 is only enabled in case of analog trigger signals)

Both analog and digital signals can serve as triggers. The signal to trigger on is to be selected from the drop-down lists in the fields nearby (see picture below, field 1). In the drop-down list, you will find the well-known signal tree containing available signals. Select the signal you want to use as trigger signal.

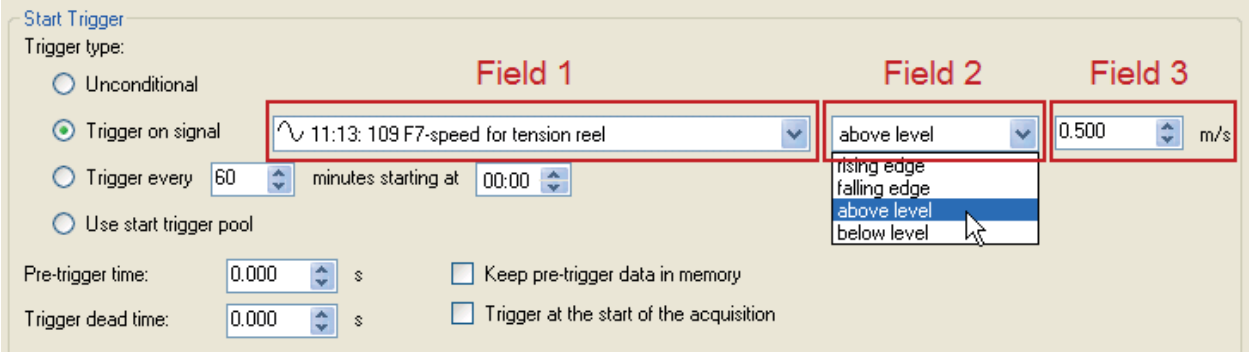


Depending on whether a digital or an analog signal was selected, the fields 2 or 3, respectively, are offered allowing the trigger event to be defined more specifically.

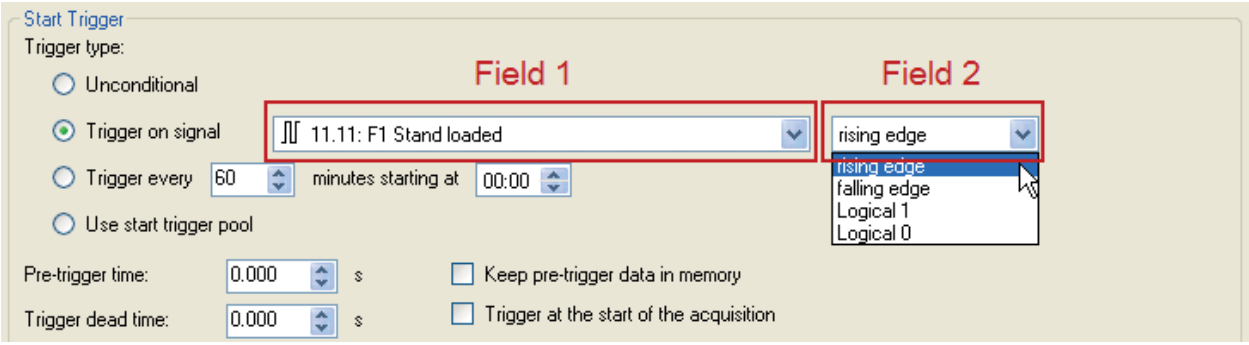
As for analog signals, you can choose between level or edge triggers including a predefined level (field 3). An edge in the context of analog values is a transit of a measured value through a predefined level.

- Rising edge: The value exceeds the level.
- Falling edge: The value undercuts the level.

The setting *above/below level* is available for start trigger only. It is used to make sure that the recording will start immediately with start of acquisition if the start trigger condition (value is above or below level) is fulfilled at this time. See also explanation in the tip below.



As for digital signals, you can choose between level or edge triggers including the 2 levels logical 0 (FALSE) and logical 1 (TRUE).



Information on the use of trigger edges

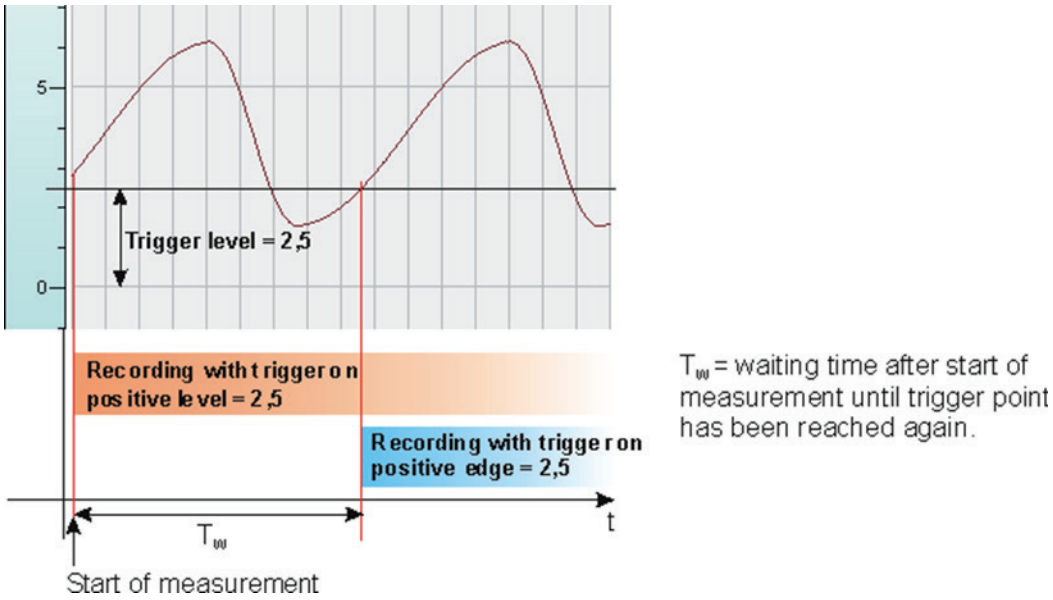
Tip

Care must be taken when using trigger edges!

In order for a trigger to fire at a positive or negative edge, the value specified in field 3 must be at least slightly exceeded or fallen below. Otherwise, no trigger will occur.

Therefore, it is often more useful to define a level trigger, which fires as soon as the signal reaches or exceeds/falls below the level defined in field 3. If this is the case already at the time the measuring starts, the recording will start immediately.

The diagram shows the differences between the recording starts, which might occur depending on the trigger conditions and the time the measurement starts.



Periodic trigger

If you want to use a start trigger always at a certain time on a periodical basis, you can check the "Trigger every … minutes starting at …" option.

This is useful if you want records to be started every 8 hours, e. g. based on a work shift.

Enter the period given in minutes, or select it from the input field. Value range is from 0 to 1440, which equals one day. Then enter or select the start time for the first trigger. Value range is from 00:00 to 23:59, which equals one day.

Example

"Trigger every 480 minutes starting at 06:00" will generate the first start trigger at 06:00, followed by further start triggers every 8 hours (= 480 min), that is at 14:00 and 22:00. The last trigger on that day will fire at 22:00. The next trigger will then fire at 06:00 on the following day.

In order to ensure the recording of data files of exactly 8 hours without a sample lost, set the "Maximum file time" to 9 hours and enable the “Start new file” option (in case another start trigger fires while the recording is already running).

If you want the recording to start immediately at acquisition start or as soon as you apply a new data storage configuration, you also need to check the "Trigger on start of the acquisition" option. The first data file created will then contain the range from the restart of the acquisition or data storage to the next trigger.

Tip

The periodic trigger at a specified start time is useful for creating data files beginning at second 0 of one minute. If you, for instance, want to create data files of 1 minute length from second 0 to 59, use the periodic trigger with the settings "Trigger every 1 minute(s) starting at 12:34" (any time).

Set the "Maximum file time” to 2 minutes and enable the "Start new file” option (in case another start trigger fires while the recording is already running).

Use start trigger pool
Enable this option if you want to use the trigger signals from the start trigger pool. See ↗ *Trigger pool, page 38*.

Pre-trigger time
Setting a possible pre-trigger time.

If this function is selected by entering a time value, recording begins immediately when the measurement starts. In this case, the measured data is stored periodically in a file, with its length of time corresponding to the pre-trigger time, or in the computer memory. If the trigger condition is met, the incoming data is added to the data buffered during the pre-trigger time. Therefore, the file start does not match the trigger time at the end of recording.

In the offline analysis with ibaAnalyzer, the trigger times of the start and stop trigger are marked by two arrows in the graph.

If you want to use the start trigger pool, you can determine that the trigger name is added to the filename of the data file. Go to "Files" in the configuration dialog of the data storage and enable the "Add start trigger name" option.

Keep pre-trigger data in memory
Enable this option if you prefer to store the pre-trigger data in the computer memory instead of a file on the hard disk. Be sure to have sufficient RAM installed when you choose this option.

Trigger dead time
This property is available for every start trigger type (signal trigger, periodic trigger and start trigger pool). The trigger dead time determines the time of suppressing subsequent triggers after a trigger occurred.

If the dead time, for instance, is set to 5 seconds, all other triggers are ignored for the duration of 5 seconds after the first trigger occurrence.

Trigger at the start of the acquisition
If you have defined a start trigger (any type) and restart the acquisition or apply a new data storage configuration to the server, the recording will not start unless the trigger fires at the same moment. Without a trigger occurrence at this time, you will have to wait until the trigger fires and the recording starts.

Check this option if you want to make sure that the recording starts immediately after restart of acquisition and/or data storage regardless of any start trigger condition.

If start trigger occurs again while file is already recording, then:

Ignore it
Selecting this option will cause the system to ignore any new start trigger during a running recording for as long as the stop trigger occurs.

Start overlapped file
This option should be used to save measured data in separate files, in case they depend on more than one independent but overlapping trigger events.

If a start trigger occurs, a file is created, in which the measured data will be saved. If another start trigger fires during a running recording, another file is created for storing the measured data. When the first stop trigger fires, the first file will be closed; the second file will be closed when the second stop trigger occurs.

So, this mode allows the start trigger of the subsequent recording to occur before the stop trigger of the recording currently running, i.e. temporarily parallel recordings in two successive files of the same file series. This mode is needed for overlapping, linked processes.

The overlapped mode is not limited to two files only. Three or more files are also possible, depending on the number of start triggers occurring before the first stop trigger fires and the defined maximum number of files.

Maximum number of overlapped files:
If you have selected the "Start overlapped files" option, you should set the number of permissible overlapped files. Usually, two are sufficient. You may, however, adjust the number according to your needs, i. e. how many start triggers you expect to occur before the stop trigger fires.

Rule of thumb: number of possible start triggers during a recording + 1 = max. number of overlapped files.

Upper limit is 10.

Extend recording time
As an alternative to the overlapping mode, choose this option if you prefer to extend the running recording in case another start trigger occurs during the running recording. Each additional start trigger extends the recording time by the time which is set as "Maximum file time" above. This occurs as often as set in the "Maximum number of extensions on single file" field. If the max. number of extensions is reached, all subsequent start triggers will be ignored. The recording will then be stopped after the last maximum recording time has elapsed. Of course, the recording is stopped immediately by any stop trigger.

Maximum number of extensions on single file
If you have selected the extended recording mode, you may set the maximum number of extensions in this field. If this limit of extensions is reached, the recording is stopped after the maximum file time at the latest.

Start new file
The current data file is closed and a new file created. So, the start trigger can serve as a stop trigger at the same time.

6.3.3 Stop trigger

The settings for the stop trigger are made in the same way as those for the start trigger. Here, both analog and digital signals can also be used as triggers.

No stop trigger
A running recording is only stopped by stopping the measurement or after the maximum file time has elapsed.

Trigger on signal
See explanation for start trigger above.

Trigger on reception of text signal
Instead of using a measured signal as stop trigger, you can also stop the recording by the reception of a text signal. In this case, the stop trigger is fired by the reception of the text signal selected in the field nearby. The contents of the text signal are not relevant for this function.

Please note the relation between this function and the naming of the data file in correspondence with the text signal in the "Files" branch of the data storage configuration ("Files - File name - Add text signal").

Use stop trigger pool
Enable this option if you want to use the trigger signals in the stop trigger pool. See ↗ *Trigger pool, page 38*.

Post-trigger time
Setting an optional post-trigger time.

If this function is selected by setting a time, recording does not stop until this post-trigger time has expired. Exceptions: if the stop trigger does not occur, the "Maximum recording time" setting will automatically stop the recording. If the stop trigger fires only shortly before the maximum recording time ends so that there is not sufficient (memory) space left for the full post-trigger time, recording is nevertheless stopped after this time, i.e. the post-trigger time is partially cut off.

In the offline analysis with *ibaAnalyzer*, the trigger times of the start and stop trigger are marked by two arrows in the graph.

6.3.4 Trigger pool

With the trigger pool, *ibaPDA* provides an easy way to configure multiple trigger signals as start or stop triggers for data storage.

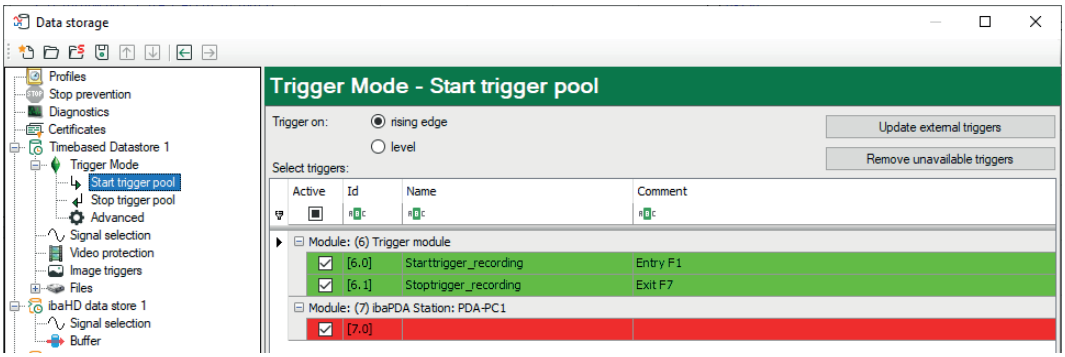
The signals of the trigger modules can be used as triggers for the recording.

For more information on trigger modules, please refer to part 3 *Trigger module*.

If several signals should be used as start or stop triggers for recordings, the setup of start trigger pools and stop trigger pools is recommended.

Each data storage has a start trigger pool and a stop trigger pool. A trigger pool is basically a list of signals taken from all trigger modules. If one of the triggers fires then the entire trigger pool is triggered as well. The trigger can fire on a rising edge of a trigger signal or at each sample with the signal = 1 (TRUE).

You will find the configuration dialog for start and stop trigger pools under the “Trigger mode” branch in the tree structure of each data storage.



Use the drop-down menu in the first column to select the trigger signals belonging to the trigger pool.

The selected trigger signals are highlighted in green color.

A red-highlighted row indicates that the trigger signal is no longer available. This is the case, e. g., if you have disabled a trigger module. Missing signals are ignored. You can remove them from the list by clicking the <Remove unavailable triggers> button.

The first row of the trigger grid can be used to filter and sort the trigger signals, just as in a spreadsheet. This can be very helpful if you want to find one particular signal in a long list of numerous trigger signals.

Clicking on the "ID", "Name" or "Comment" column headers will sort all trigger signals by the criterion selected in ascending or descending order.

If you click on the small filter icon in the upper right corner of the "ID", "Name" or "Comment" column headers (only visible by hovering the mouse), you can select one of the entries for filtering. You can also enter an ID, a name or a comment, or parts of it, in the corresponding cells.

Example:

If you know that the name of the trigger signal you are looking for starts with “Trigger…”, just start typing the name in the respective cell and all the corresponding triggers will be listed below.

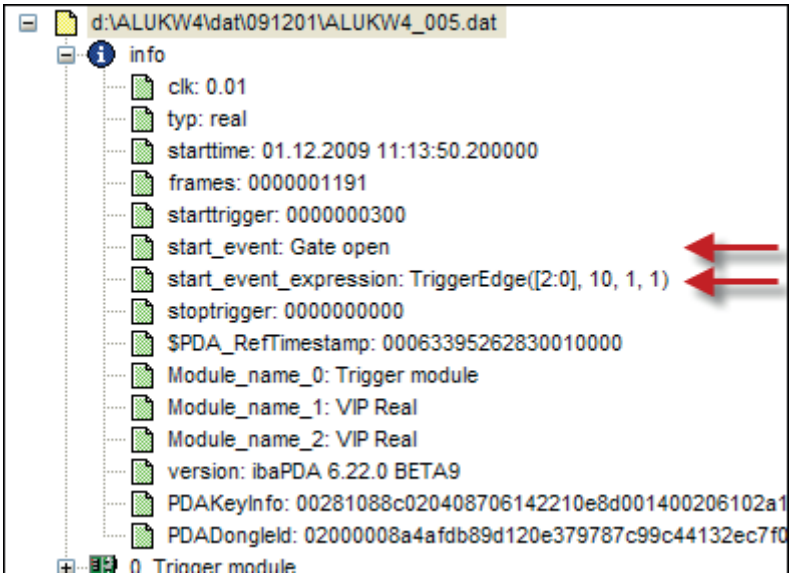
Name	Comment
Trigg	
ger module	
Trigger final speed	Triggers if final speed has been reached
Trigger rollforce	Triggers if rollforce exceeds min value
Trigger Speed acceleration	Triggers if acceleration
Trigger speed min	Triggers if speed is above min speed

The keys <Ctrl> or <Shift> allow you to enable/disable multiple signals at the same time.

If you click the “Active” column header then the value (active/inactive) of the selected row is copied to all the rows underneath it. Double-clicking on the "Trigger module" header enables/disables all the trigger signals of this module.

If you want to use a start trigger pool or a stop trigger pool as trigger, you should enable the option “Use start trigger pool” or ”Use stop trigger pool”, respectively, in the trigger mode dialog.

When using a trigger pool, recording can be started or stopped by any desired trigger within the trigger pool. The name of the triggering trigger is stored in the data file in the same way as the info fields “start_event” for the start trigger and “stop_event” for the stop trigger. The expression following the trigger is also stored in the info field “start_event_expression” or “stop_event_expression”, respectively. The name of the start trigger can also be added to the filename.



The <Update external triggers> button is only relevant in conjunction with the multistation mode where multiple *ibaPDA* systems exchange signals with one another, synchronized or un-

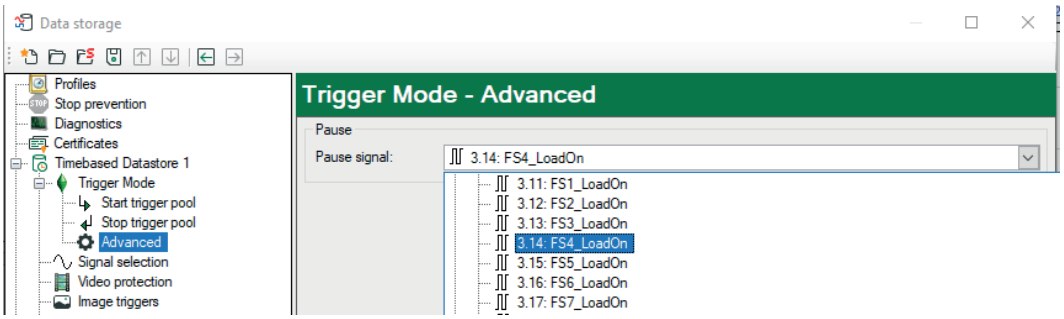
synchronized. Because for non-synchronized operation the trigger signals from other stations (external triggers) are not automatically updated, you have to click on this button in order to get the current trigger status from the other stations.

6.3.5 Advanced

Pause

Data storage can be paused by a digital signal. During long-term recordings, it might be useful to pause the data storage yet without closing the data file. For example, this could be the case in shut-down times of machines during a work shift when no measured data arise, but the data file is intended to cover the entire shift.

While a data storage is paused, all measured values in the data file are replaced with invalid values, which will be displayed as a gap in ibaAnalyzer. Signals with INT, WORD and BYTE data types are saved to the data file as 16 bit integer values. Since there is no invalid bit for these data types in the data file, these values will be replaced with 0.

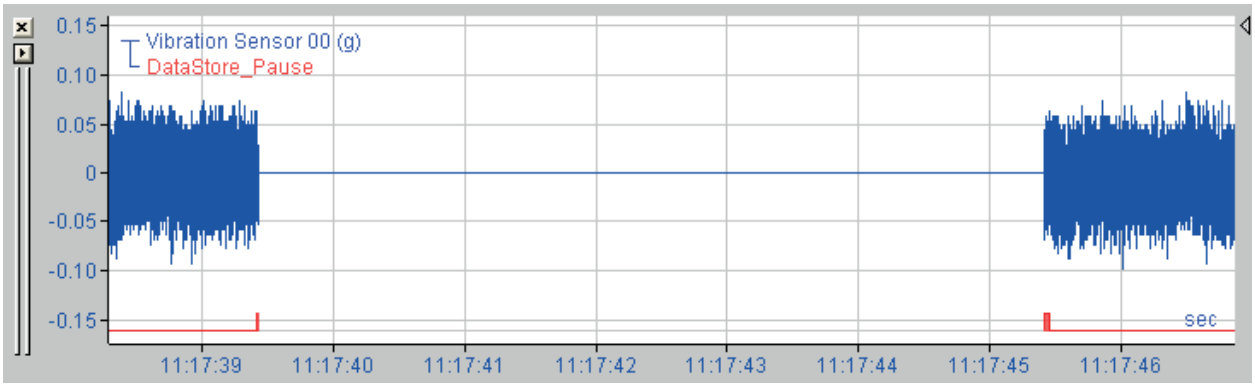


The period of the ibaPDA server acquisition loop is equal to the smallest common multiple of all module time bases, but no less than 50 ms.

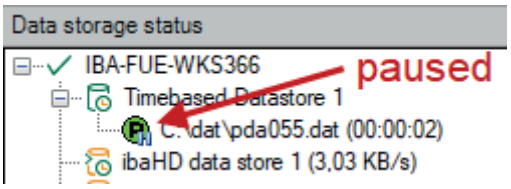
The pause signal is evaluated only once per acquisition loop. The signal is 1 (TRUE) if it remains 1 (TRUE) for the whole duration of the acquisition loop period. It is 0 (FALSE) as soon as 1 sample is 0 (FALSE) during the period of the acquisition loop.

This means that the data storage will be paused a certain time x after the rising edge of the pause signal. The time x is between 0 and the acquisition loop period.

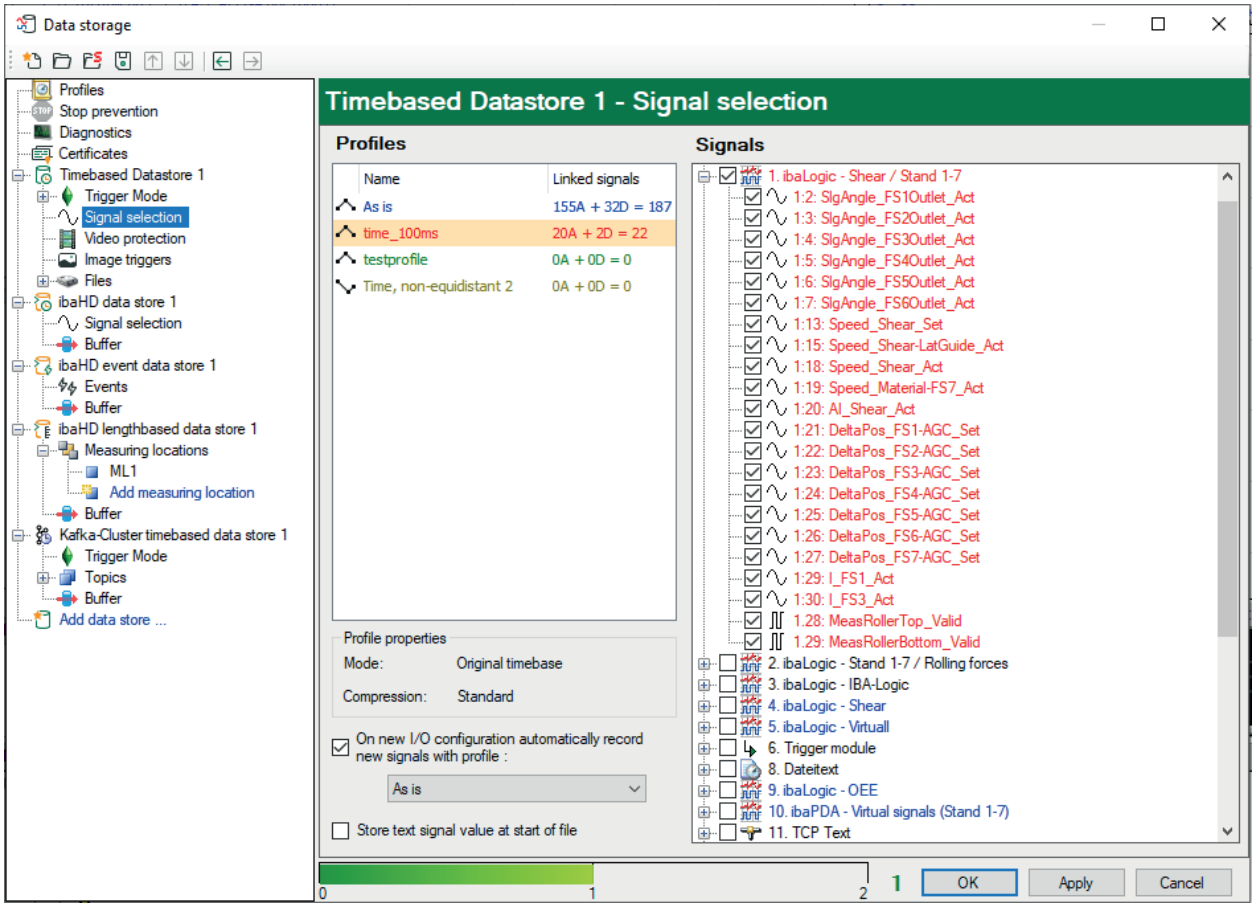
This also means that the data storage will be resumed a certain time x before the falling edge of the pause signal.



The data storage status tree shows a pause icon when the recording is paused.

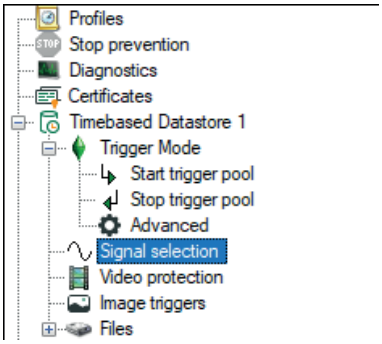


6.4 Signal selection



Signals must be assigned to a data storage profile in order to be recorded in a recording profile. A simple way is to assign all signals to be recorded to the "As is" standard profile. However, this could result in very large data files, depending on the number, sample rate and maximum recording time limit.

In order to select signals for recording, mark the *Signal selection* sub-branch in the tree of the data storage dialog.



In the right part of the dialog window, the available recording profiles are listed. Next to it on the right is the signal tree with all available, i.e. enabled, signals. Only the signals shown here can be assigned to a recording profile.

Note

A signal can only be assigned to one profile per data store.

If a signal will be recorded with different profiles, different data recordings must be defined for this purpose or copies of the signal must be generated via virtual modules.

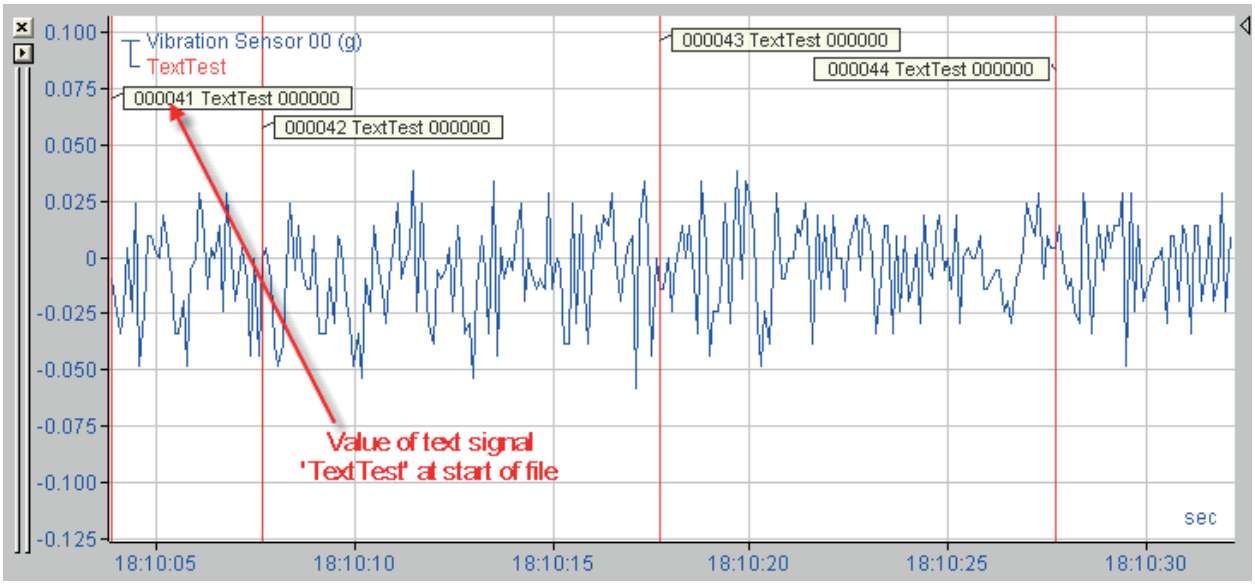
You may, however, use as many profiles as you like in a data store.

On new I/O configuration, automatically record new signals with profile:
In the lower part of the dialog, you can choose a profile from a pick-list which will be used as default profile for new signals. If you add new signals to the I/O configuration, these signals will be assigned automatically to this default profile and recorded.

Store text signal value at start of file
Enable this option if you want to ensure that the value of an assigned text signal is always saved already at the start of each measurement file. The value of the text signal will then be available right from the beginning of the measurement file for subsequent evaluation.

If you do not enable this option, the values of the text signals are only saved in the measurement file then, i.e. when the value changes. Therefore, if the value of a text signal does not change during the recording, no text appears in the file. If the value changes during recording, the value of the text signal would only be recorded from that point on.

The following figure shows the result with this option enabled in *ibaAnalyzer*.

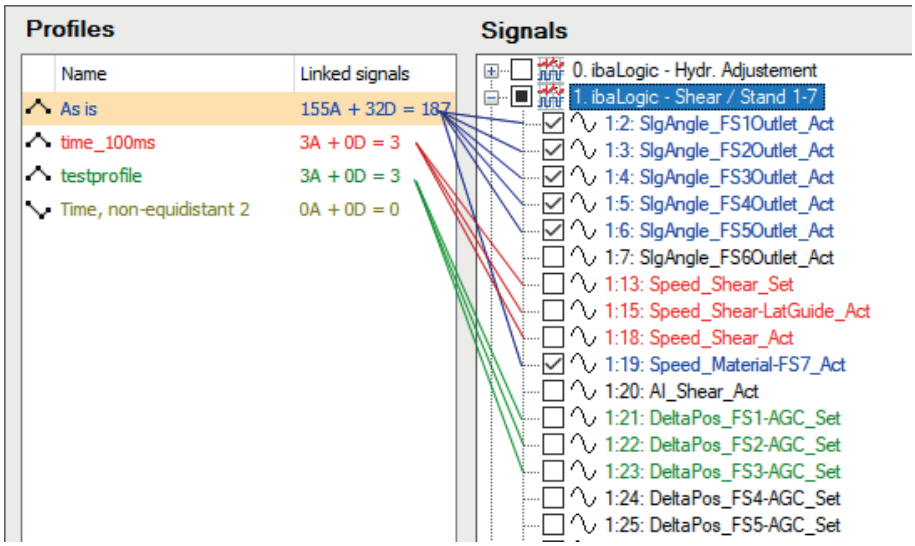


6.4.1 Assigning a signal to a storage profile

A tick in the selection box beside a signal shows that this signal is assigned to a profile marked in the left section of the dialog window. In order to assign a signal, the profile must be selected first and then the signal. If you want to assign all signals of a module to a profile at once, the relevant module needs to be checked in the selection box of the module. All signals of this module will be checked, too.

Signals not selected are either assigned to a different profile or not assigned at all. Whether a signal is already assigned to a profile can be checked by clicking on each profile and checking the selection box of the relevant signal. If the signal is not checked, it is not assigned.

More convenient is the color check: the colors of the signal names correspond to the colors of the profiles they are assigned to. If you extend a module in the tree, you see at once which signal is assigned to which profile. A black signal name indicates that the signal has not been assigned to a profile yet.



6.4.2 Checking the profile assignment

If you like to check quickly whether a storage profile is assigned to a signal, select the profile in the profile list.

Below this list, you will find information about the number of analog and digital signals assigned to this profile.

At the same time, you will see checkmarks at the modules, whose signals are assigned to this profile, in the signal tree right to the profile list.

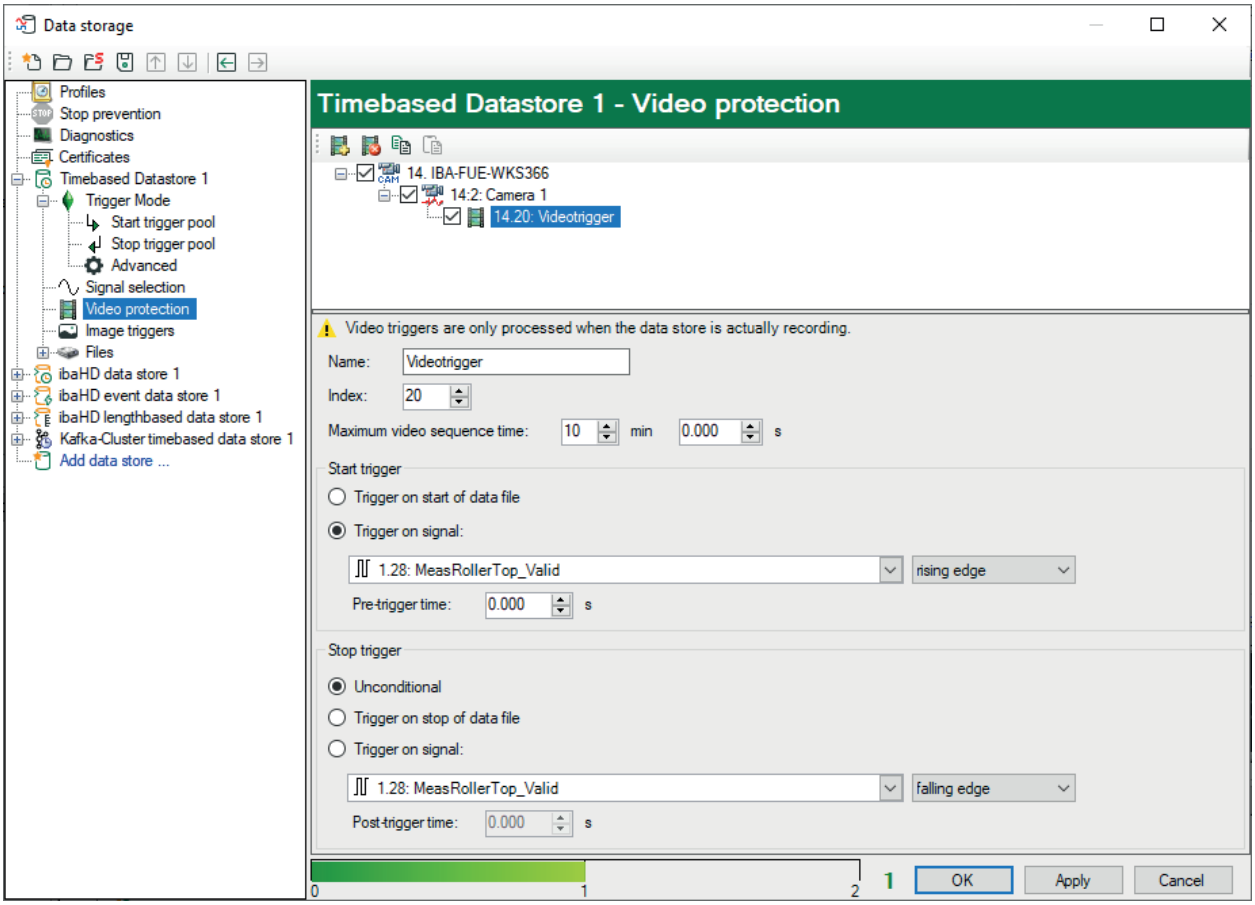
A gray square within the selection box means that only a part of the signals of the corresponding module is assigned to this profile.

A checkmark on a white selection box indicates that all signals of the corresponding module are assigned to this profile.

6.4.3 Disabling the profile assignment

In order to remove a profile assignment, just uncheck the corresponding signal or module.

6.5 Video triggers



All *ibaCapture* modules configured in the I/O manager, i. e. all *ibaCapture* servers and their associated cameras, are listed in this tree.

For each camera you can add, delete and configure video triggers for a reliable video recording.

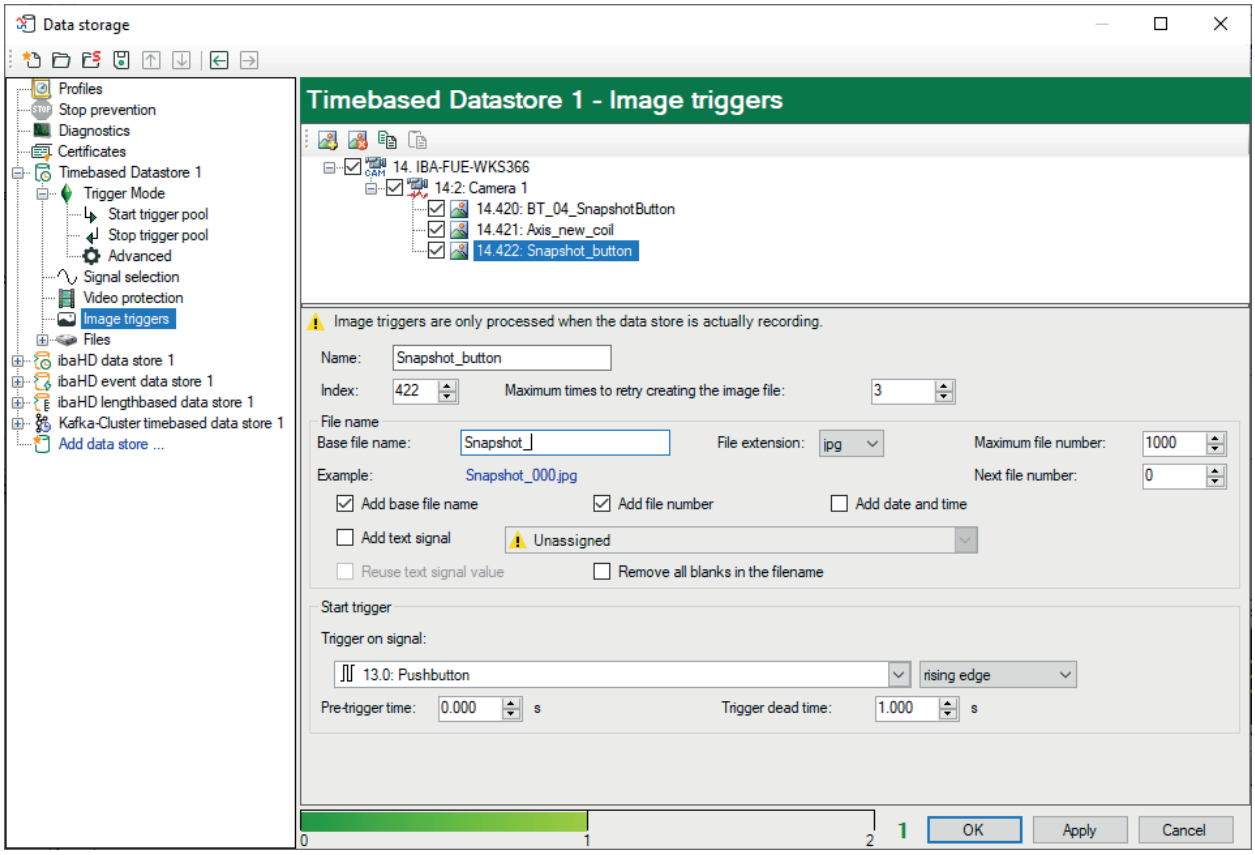
Every video trigger consists of a start trigger and an optional stop trigger. The start and stop trigger can be any measured signal, virtual signal or trigger module signal.

Other documentation

For a detailed description of video triggers and their configuration, please refer to the corresponding *ibaCapture* manual.

6.6 Image trigger

With image triggers you can store event-controlled snapshots or stills from video recordings as an image file.



All *ibaCapture* modules configured in the I/O manager, i. e. all *ibaCapture* servers and their associated cameras, are listed in this tree.

For each camera, you can add, delete and configure image triggers.

Every image trigger is given a name and index. The file name and type of the image file to be generated can be configured individually for each image trigger. You define the trigger for the snapshot with the start trigger. You can use any measured signal, virtual signal or trigger module signal as the start trigger.

There is a default path configured as base directory for the image files:

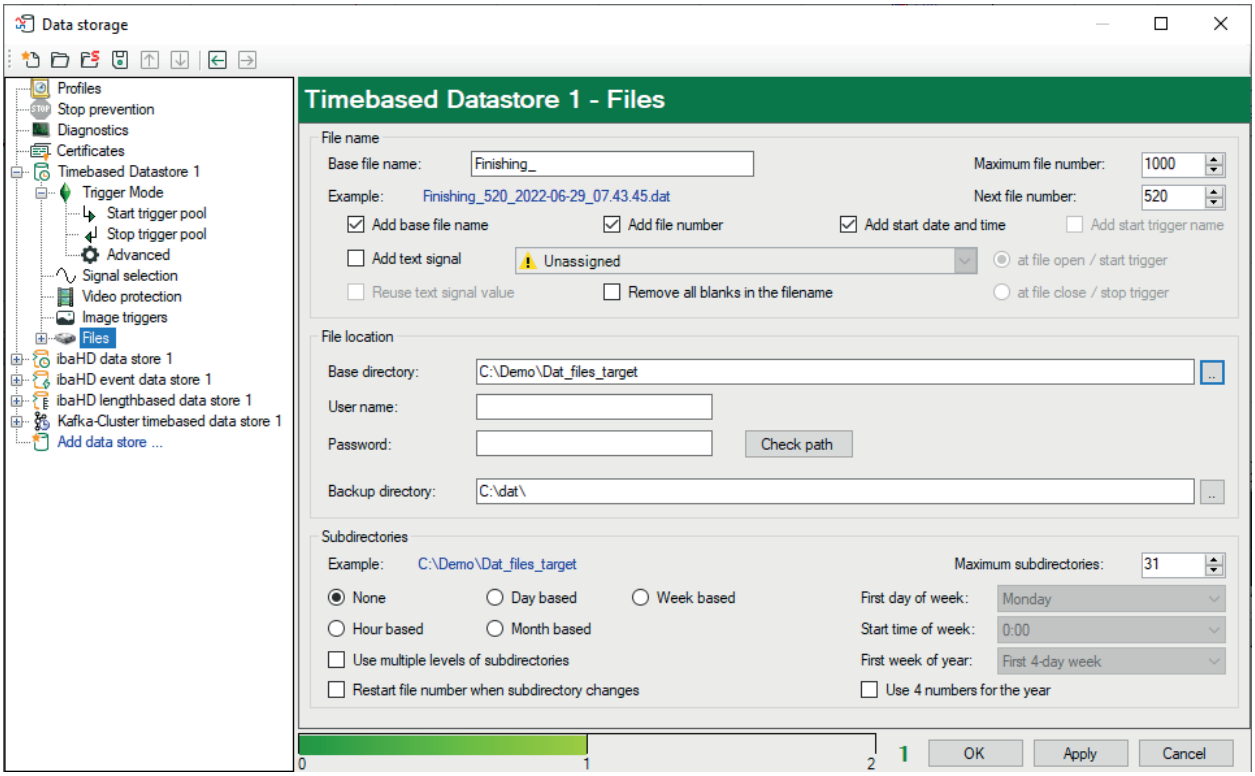
`c:\iba\ImageData`

Sub-directories are automatically created for *ibaCapture*-Server, cameras and image triggers.

Other documentation

For a detailed description of image triggers and their configuration, please refer to the corresponding *ibaCapture* manual.

6.7 Data file information



Naming and administration of data files must be configured prior to the first recording. In order to configure these settings, click on the "Files" sub-branch of the corresponding data storage in the data storage dialog. On this level, you can configure file name and storage location.

6.7.1 Data file name

Base file name
Enter a file name prefix which is always the same, e.g. a term with relation to the data storage creating the files. The base file name will be supplemented automatically by an incrementing number if more than one data file with the same name exists.

Check the “Add base file name” selection box if you want the base file name to be also (part of) the file name. Uncheck it if you want other information to be shown in the file name.

However, it is always advisable to enter a base file name even if you do not intend to use it. In case, for instance, you use the text signal only for naming the files and the text signal is not available due to an error, *ibaPDA* will use the base file name instead (plus numbering, if necessary).

Optionally, you can also select other information to be part of the file name:

- File number
- Start date and time
- Start trigger name
- Text signal

Simply enable the option you wish to be added to the filename.

Below the base file name, an example of a file name is shown in blue letters. Whenever you enable or disable the naming options, you can immediately see the effect it would have on the next file name.

You cannot change the order of the optional information in the file name. If you want to customize the data file names differently, you should use the text signal.

Add file number
If this option is enabled, the file number will be added based on a sequential number between 0 and the maximum number of files.

- Maximum number of files
  This setting specifies the number of the files provided for the recording. If the maximum number of files is reached, the next recording will overwrite the first file with the same name (then, e.g., again with index 0) unless other restrictions have been specified in the settings under "Cleanup strategy".
- Next file number
  The index of the next file created. If necessary, this index can also be set manually on any value between 0 and the maximum number of files.

Add start date and time
If enabled, the current date and time will be added to the file name.

The format is _yyyy-mm-dd_hh.mm.ss

Please note that this is the date and time at the start of the data file. The values may differ from the Windows file date, which is the date at file closure.

Add start trigger name
This option is only available if the start trigger pool is used as trigger mode.

If enabled, the name of the trigger signal from the start trigger pool which fires first will be added to the file name. So, you can easily see which trigger event started the recording.

Add text signal
If you choose this option, a text signal section selected from the dropdown list nearby will be added to the file name.

In case you disable all other naming options, the data file will be stored with a name derived from a text signal and the file will be saved under this name.

If you check this option, additional options and an entry field are activated:

- In the field right next to the selection box, you can select the appropriate text signal from a dropdown list. Of course, the text signal must have been defined in the I/O Manager beforehand.
- Select whether the contents of the text signal should be read at file opening (= start trigger) or at file closing (= stop trigger). Which selection is best depends on the process you are measuring, i.e. the time the correct value of the text signal is available.
- Reuse text signal value
  If this option is disabled, the text signal value will only be used once right after reception. If no new text signal value is received for the next data file, the base file name will be used instead.
  By enabling this option, you determine that the last received text will be used again for the next data file. In this case, *ibaPDA* automatically supplements the file name by an index to prevent overwriting of the last file.
- Remove all blanks in the file name
  When a text signal is configured with the “Fixed width” separating mode, it may happen that some characters are filled in with spaces. These blanks might have a negative effect on the a file name. Therefore, you can enable this option in order to remove blank characters in the file name.

6.7.2 Data file location

Base directory
The base directory is the main directory where the data files are stored. There is a default path for timebased data stores:

`c:\iba\Datastores`

The system will create automatically a sub-folder with index for each configured data store (Store1, Store2 etc.)

You may change the base directory. Then, enter the main path for storing the data files of this data storage. A valid drive and path must be entered. You may use the browser button <...> to find available drives and paths.

You can either select a local or network drive.

Note

For example, if you enter drive C:\, the local hard disk of the server computer is addressed. The browser shows the perspective of the server in terms of drives and paths.

Note

Basically, you can write the data files directly on a network share or NAS. However, the performance will be significantly lower, compared to the local harddisk.

Thus, iba recommends always configuring the local harddisk as base directory. By means of the software *ibaDatCoordinator* the terminated data files can be copied or moved automatically and reliably to another storage medium.

If, even so, you want to store the data files on a remote drive of another computer, you need to configure the user account which is used for running the *ibaPDA* server on the server computer also on the remote computer. This user account should have administrative rights. If it is not possible to set up the same account on both computers, you may enter *user name* and *password* of another user with the appropriate rights on the remote computer. *ibaPDA* then uses this login information to log on the remote computer in the attempt to write data files.

User name and password
You only have to enter this login information if you want to store the data files on a remote computer, on which the user account used by *ibaPDA* on the server computer is not configured (see above).

<Check path> Button
This button may be used for checking the base directory as entered above in terms of availability under consideration of the user account data.

Backup directory
In this field, you may enter another drive or path for file storage, just in case the base directory is not available, e. g. due to network problems. If *ibaPDA* cannot write on the base directory, it

automatically tries to write on the backup directory. So, it is advisable to enter a path to a local drive on the server computer as backup directory. The system checks every 15 minutes if the base directory is available again. If so, *ibaPDA* will store the next file in the base directory.

6.7.3 Subdirectory organization

Subdirectories can be used for a more structured storage of the data files. For every option and combination you choose, an example of the path is given immediately in blue letters.

None
The files are stored in the path specified in *Base directory*.

Hour, day, week or month based
Further subdirectories are added to the path specified as base directory. These contain parts of a complete file series. The names of the subdirectories are automatically derived from the date and/or time, depending on your selection. Each subdirectory then contains those files that were recorded in the relevant period of time.

The format of the subdirectories follows the pattern

- `yymmddhh` or `yyyymmddhh` for an hourly, daily or month-based setup
- `yyww` for a week-based setup

A "daily" subfolder, for example, would be named as follows: 050221 for February 21, 2005. The next "Daily" folder would then be 050222 and so on.

In case of a "Weekly" storage type, the folders would be named according to the calendar week number. For determining the week number, the Windows calendar is used. Since the week number may vary depending on region and country, several parameters need to be set:

- First day of week (e. g. Germany = Monday, USA = Sunday)
- Start time of week (the time at which the first day of the new week begins)
- First week of year (the week around January 1, the first complete week of the year or the week with four days in the new year).

You should adjust these settings to your regional conditions in order to get a proper week counting.

Maximum subdirectories
This setting is only relevant if a subdirectory organization was selected. It determines the maximum number of subdirectories according to the number of hours, days or months in the base directory.

Example:

"Hourly" organization, number of subdirectories = 2; up to two folders are created (these contain the file series); when the second hour has passed, a third folder is initially created; however, the first folder with all its contents will be deleted automatically, so that only two folders exist.

Use multiple levels of subdirectories
Check this option if you prefer separate subdirectories for year, month, week, day and hour.

The format of the subdirectories then follows the pattern

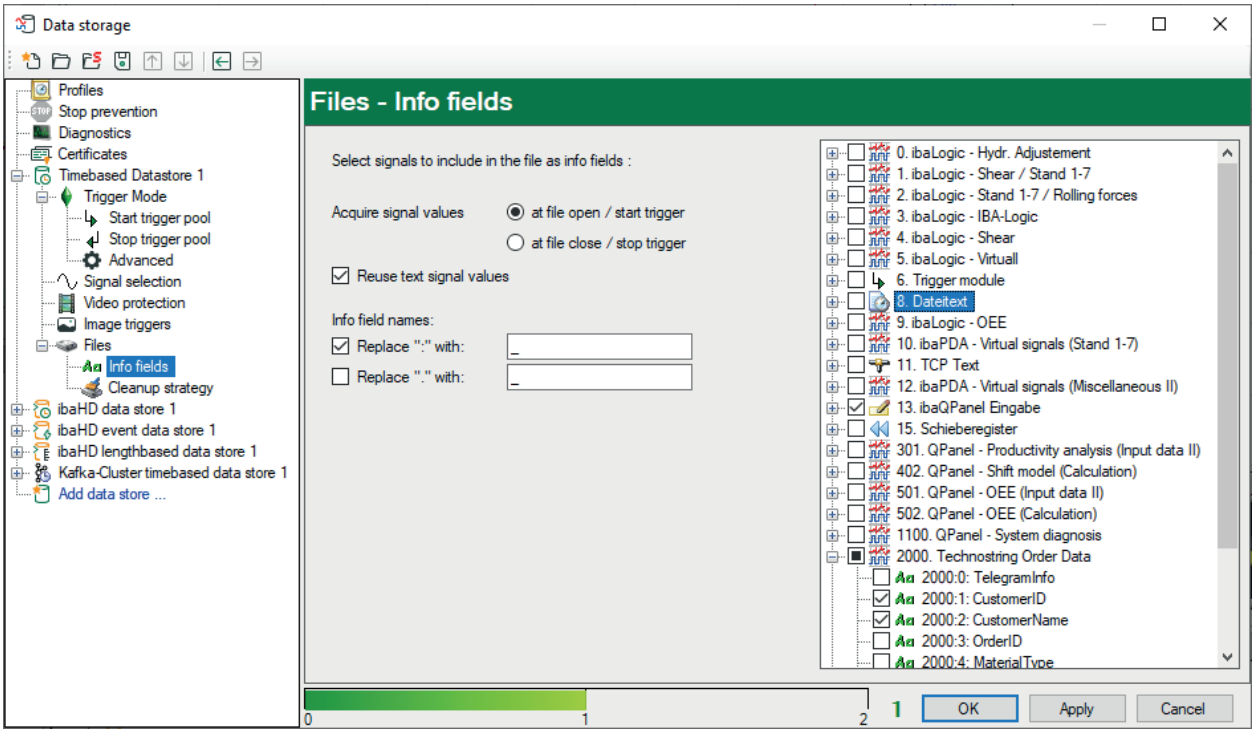
- `(yy)yy\mm\dd\hh` for an hour-, day- or month-based organization
- `(yy)yy\ww` for a week-based organization

Restart file number when subdirectory changes
If this option is enabled, the file number will start again at 0 in every new subdirectory.

Example:

If the length of a data file is set to 1 hour with the storage type set to "Daily", you can use the file numbering for naming the file. Enable the option for resetting the file number in order to browse every "daily" subdirectory for file names ranging from xyz_0 to xyz_23.

6.7.4 Files - Info fields



Click on the *Info fields* node of the *Files* branch of the corresponding data storage to open the dialog. Here, you can set the signals (analog, digital or text signals) to be stored as info fields in the data files and the time they should be acquired.

Select signals to include in the file as info fields:
In the right part of the signal tree, you will find all modules with analog, digital and text signals as you defined them in the I/O Manager.

Enable the selection boxes in order to select individual signals or complete modules to be stored in the data file.

When analyzing with *ibaAnalyzer*, you will find the information of the info fields in the info header of the data file.

In the info fields, only one current value per data file is entered and saved, either at the start or at the end of the data file recording (see below).

Acquire signal values
By choosing this option, you determine when the values are used by an analog, digital or text signal for recording.

- At file open / start trigger: The current signal values are transferred into the data file at the start of a recording. Signal value changes during a recording process are taken into consideration for the next file at the earliest.
- At file close / stop trigger: The signal values are first transferred into the data file at the end of the recording. Signal values can be received, changed and sent again during a running recording. The last signal value received before closing the file will be taken. This option should be selected if the desired information is not available at the start of the recording.

Reuse text signal values
This option prompts *ibaPDA* to keep the received text information after reading instead of deleting it as usual. So, you make sure that new data files are supplied with the same text signal values in case no new text has been received.

When using the text signal for naming the file, *ibaPDA* automatically adds an incrementing index in order to provide unambiguous data file names.

Info field names: Replace ":" with ... and replace "." with ...
The use of ":" and "." in info fields can cause problems in the course of further processing and analysis of the data by other systems. In *ibaPDA*, these characters are used, e.g., in the signal IDs (analog [Module:Channel], digital [Module.Channel]).

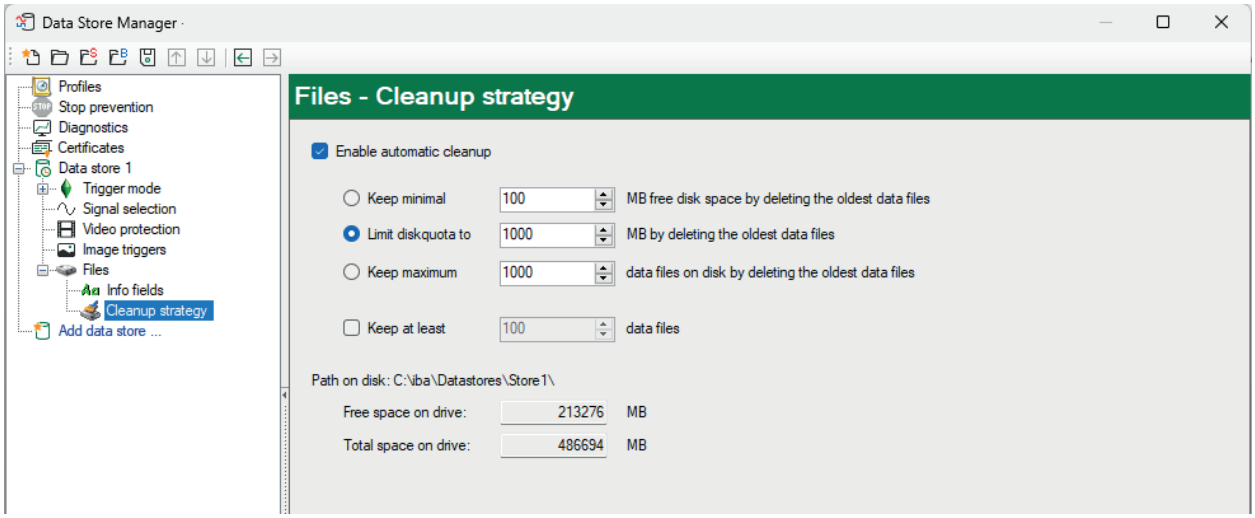
To avoid such problems, you can have the characters ":" and/or "." replaced by other, permissible characters here.

Enable the desired option and enter the new character in the field next to it.

6.7.5 Data files - Cleanup strategy

ibaPDA offers the possibility to check the available space on the storage medium and to initiate cleanup measures, if required.

Click on the "Cleanup strategy" node of the "Files" branch of the corresponding data storage to open the dialog.



We strongly recommend enabling this option in order to prevent the hard disk (HD) from "spilling over" together with the risk of system interruptions unless you use other methods to ensure always enough free disk space for new data files.

The cleanup strategy can be used to limit the memory for measured data.

The actual values of total and free memory space on the base directory drive is displayed in the center of the dialog window.

Enable automatic cleanup
Check this option in order to enable the automatic cleanup.

Keep minimal ... MB free disk space by deleting the oldest data files
If this option is enabled, the oldest files will be deleted in order to restore sufficient memory space as soon as the defined limit of free memory space is breached. Enter an appropriate value to ensure the system integrity.

The indication of the remaining free memory space refers to the total memory capacity of the drive, just as displayed under "Free space...".

Note

When using several *ibaPDA* systems that write on a common drive (file server), the permissible memory space should be the same for all *ibaPDA* systems. In this case, the "Limit disk quota..." option would be more preferable.

The same applies when running multiple data storages at the same time. Of course, you must enable this option for every data storage.

Limit disk quota to... MB by deleting the oldest data files.
This option limits the total disk space used by all dat files of the corresponding data storage in the base directory, including subdirectories. Select this option and enter a sufficient value for the disk space the expected data files presumably require.

Basically, we recommend using this option.

Keep maximum ... data files on hard disk by deleting the oldest data files.
If this option is enabled, the system will save as many data files as determined in this setting. In case this limit is exceeded and there is no other rule causing a deletion, the oldest data files will be overwritten. The number of files may fall below the specified number, but it can never exceed it.

Keep at least ... data files
If this option is enabled, as many old files as set here are saved from being deleted. The number of files refers only to the files created by the respective *ibaPDA* system.

This is particularly helpful, for example, if two *ibaPDA* systems or several recordings write on the same hard disk at the same time. In the long term, the number of files prevents that one of the two *ibaPDA* systems or data storages "goes short" in case only the MB limit is enabled.

This option can be activated in addition to the first two options, however, it has a higher priority.

6.7.6 Post-processing

As of *ibaPDA* v7.3.0, the *Files – Post-processing* branch is no longer available for configuring new data stores.

Instead, we recommend using the program *ibaDatCoordinator* for the automated further processing of data files. This software offers a diverse suite of features and also ensures greater reliability.

In cases where the post-processing feature was configured using a software version <v7.3.0, which was then upgraded to version ≥v7.3.0, the feature will still be available.

However, you must now also enter the user name and password in the configuration. The post-processing command is no longer executed automatically on the basis of the user profile associated with the *ibaPDA* service.

Instead, a dedicated user must be created who is authorized to perform post-processing.

6.8 Apply new data storage configuration

You can set up multiple data storages and profiles without leaving the dialog in between.

After you have finished the setup, click on the <Apply> or <OK> button.

Note

Whenever you have changed the configuration of one or more data stores you should close the configuration dialog with the <OK> button. At this time, the recording of data stores which were subject to change will be stopped and closed. Other, unchanged data stores will continue to record. The recording of the changed data stores will restart automatically with the new configuration. In any case, the samples will be retained. In accordance with the storage profile, the new data stores will directly start with the next sample.

Important: Changes in the data storage configuration will not affect the data acquisition of the server. The server continues the data acquisition even if the recordings are stopped and restarted.

6.9 Configuration file of data storage

All configuration data is stored in an XML file. The *ibaPDA* system uses three different configuration files for I/Os, data storage and view (layout). The *CurrentDataStoreConfig.ds* file is used for data storage configuration. The file is stored in a sub-folder of the `ProgramData` folder (path: `c:\ProgramData\iba\ibaPDA...`).

This file is loaded with each start of the system. It is updated whenever you change the configuration or leave the configuration dialog of the data storage by click on the <OK> button. The current configuration is therefore always stored in this file without explicitly saving it. Each time a new configuration is applied, the previous one is automatically saved in the `...\Backup` folder. The last ten configurations are always available in the backup folder.

Tip

You can access the saved configuration files in the backup folder directly by a special button in the toolbar of the data storage manager:

If the most recently used configuration file is not found, the server automatically loads the most recent backup configuration.

If you have applied a valid and functional configuration, it is recommended to also store the configuration file under another name in a protected folder or disk of your choice in order to avoid unintended modifications or loss of configuration data. You can also use the *Save project...* function in the *File – Save project...* menu.

Saving a data storage configuration

1. Open the data storage dialog after making sure that the current configuration is correct.
2. Click the button (save configuration) in the toolbar of the data storage dialog.
3. In the dialog box, select the path where the file should be saved. We recommend choosing the path `c:\ProgramData\iba\ibaPDA`, as this is the default path of the data storage configuration files. But of course you may choose any other folder.
4. Enter a file name (prefix).
5. Enter a file extension (.ds or .zip).
   If you select .ds then a single `file name.ds` file is created. If a file already exists with the same name then it is overwritten.
   If you choose .zip then a `filename.zip` ZIP file is created containing a `CurrentDataStoreConfig.ds` file.
   If a file already exists with the same name, the file within the ZIP file will be replaced. This also applies if the ZIP file contains an *ibaPDA* project file in which more files are stored.
6. Click on <Save>.

Loading an existing data storage configuration

1. Open the data storage dialog.
2. Click the button (open) in the toolbar of the data storage dialog.
3. Select the desired configuration file by using the browser. Usually, you will find the configuration files in the folder `c:\ProgramData\iba\ibaPDA\Server`. The browser only shows files with the `.ds` extension.
   The `CurrentDataStoreConfig.ds` file contains the latest configuration.
4. Press <Open>. The configuration will be loaded and compared with the signals, drives and the license information available in the dongle. If no conflicts occur, the configuration is considered to be valid and the recording starts.

Loading an existing configuration file in simulation mode

Similar to the I/O Manager, it is also possible to load a data storage configuration in simulation mode during data recording. A data store can therefore be configured without the I/O configuration being accepted by the server. The iba support department can better diagnose user problems by using the simulation mode because the configuration is displayed exactly as on the user's original system.

To load a data storage configuration into simulation mode, use the appropriate button in the toolbar of the data storage configuration dialog:

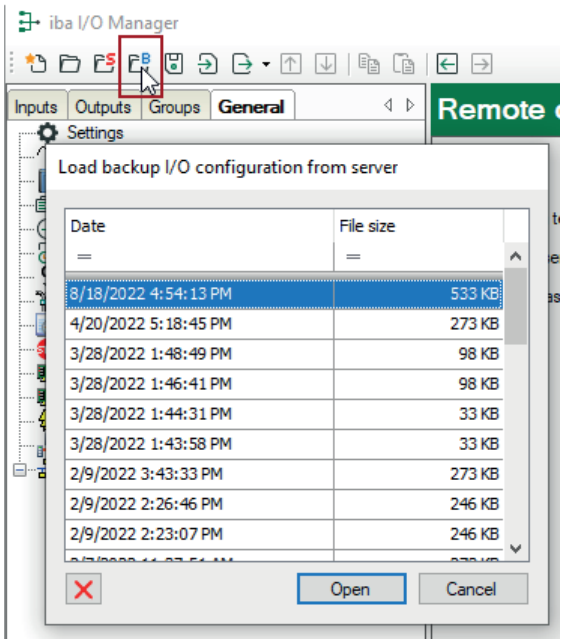
In the following dialog, you can choose the desired configuration file (.ds) or load an *ibaPDA* project file (.zip).

If the file is loaded, "SIMULATION" and the full path of the file are displayed in the header of the dialog. The <Apply> and <OK> buttons are not available.

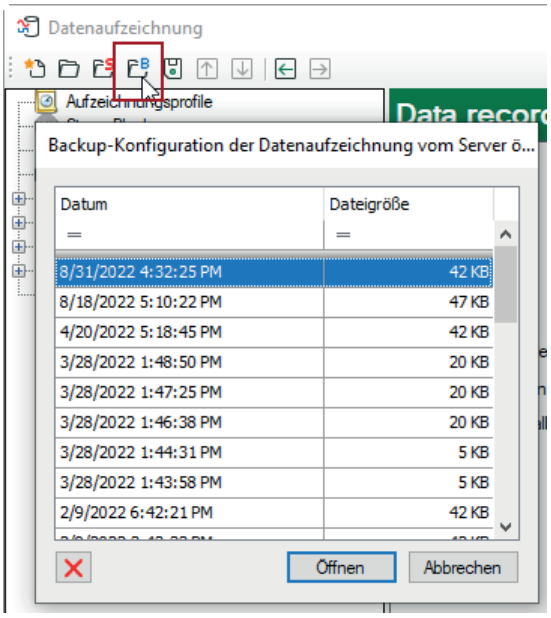
Alternatively, you can drag the file from Windows Explorer into the tree of data storage dialog in order to activate simulation mode.

6.10 Loading a backup configuration file

Each time a new I/O configuration or datastore configuration is applied, *ibaPDA* automatically creates a backup of the previous configuration. These backups are stored in the path `%ProgramData%\iba\ibaPDA\Backup` on the *ibaPDA* server computer. In the toolbar of the I/O Manager or data storage manager there is a dedicated button which provides direct access to the backups.



Open backup in I/O Manager



Open backup in datastore manager

If you click on this button, a dialog window will open, which shows a list of the available backups with their date and time of creation and file size.

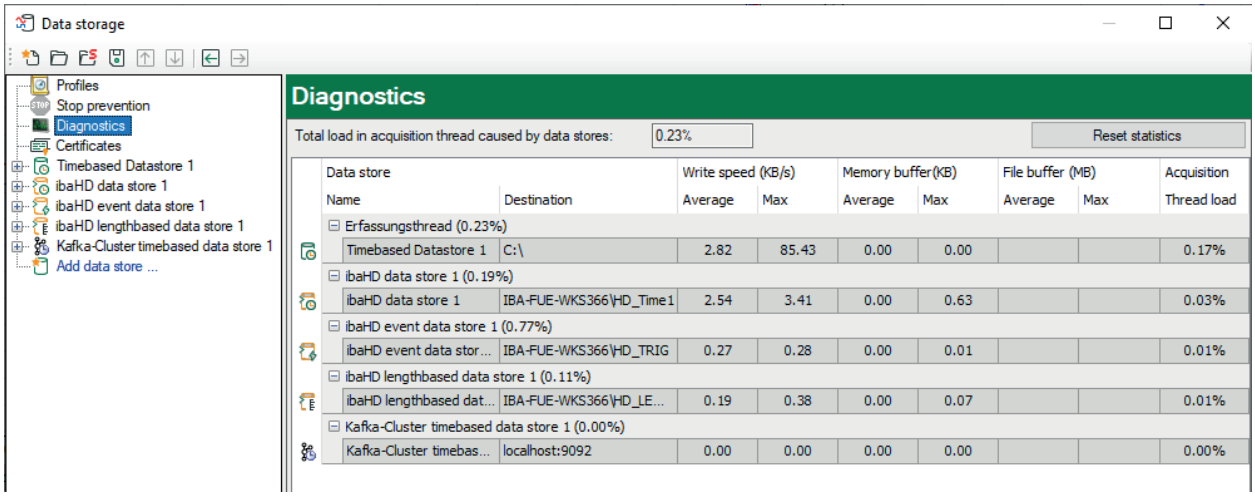
By selecting a row and clicking on <Open> or by doubleclicking on a row you can open the corresponding backup in the I/O Manager or data storage manager respectively.

If you want to apply the configuration from the backup, then click on <Apply> or <OK>.

In the dialog window with the backup list there is a button to remove a selected backup, if needed.

7 Diagnostics for data stores

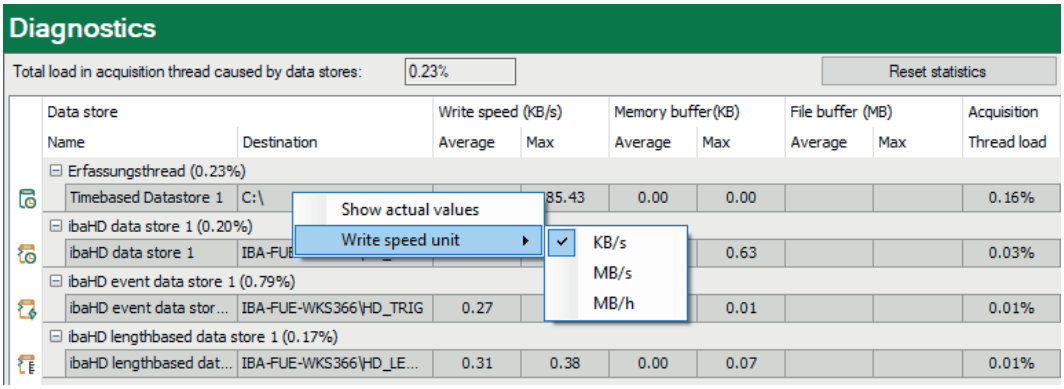
In the tree of data stores, there is one *Diagnostics* node, which provides some information about the system load due to the data stores. The measurement must be running meanwhile.



The performance values of all data stores are shown in the table. There is one row per data store. The rows are grouped according to the threads that write the data.

In each group row is the name of the thread and (in brackets) its share of the load. The load average is displayed by default. But, you can switch between the average and actual value using the context menu.

In the same menu, you can also adjust the display unit of the write speed.



In addition to the write speed, you will also see information on the occupancy of the main memory ("memory buffer") and the file memory ("file buffer"). Values in the "File buffer" columns only exist for HD data stores and for databases, cloud and MQTT data stores if the file buffer has been enabled for these data store types. For further information, please refer to the manuals for ibaHD-Server and the data stores (Data-Stores).

Depending on the data store type, the data displayed can be interpreted differently:

Normal time-based data stores with synchronous file access
The rows of this store type show the write speed of the data on the hard disk. The columns for the buffered data show zeros because the data are written directly on the disk. The column of the acquisition thread load shows the proportion of time needed for the run-length encoding and writing on the disk.

Time-based data stores with asynchronous file access
The rows of this store type show the write speed of the data on the hard disk. The columns for the buffered data show the amount of data that still has to be written on the disk. The column of the acquisition thread load shows the proportion of time needed for the run-length encoding. The actual writing process is carried out in the write thread, the load of which you can see in the group row.

ibaQDR **data storage**
The rows of this type of store show the write speed of all the input coil files from all measuring locations. The write speed of product coils and input product coils is not displayed. The group rows of the QDR product thread show the load due to the generation of the product coil files. The columns for the buffered data show the amount of length-based data that has not yet been used for a product coil file. The column of the acquisition thread load shows the share of the load due to saving time-based data in input coil files and generating length-based data at all measuring locations.

ibaHD **data stores (time, length and event-based)**
The rows of this store type show the write speed of the data on the ibaHD server. The columns for the buffered data show the amount of data that still has to be written onto the ibaHD server. These values should usually be close to zero. If the connection to the ibaHD server is interrupted, these values increase rapidly. If the file buffer has been activated for HD recording and the normal buffer is full, the data is stored in files on the hard disk ("file buffer").

The column of the acquisition thread load shows the proportion of time needed to provide the data to be written on the ibaHD server. This period includes:

- Run-length encoding for time-based stores
- Event trigger calculation for event-based stores
- Generating length-based data for length-based stores

DB/Cloud, MQTT and MindSphere data stores
For this storage type, the rows show the write speed of the data to a DB or cloud server or an MQTT broker. The columns for the buffered data show the amount of data that still needs to be written to the disk. These values should usually be close to zero. If the connection to the target server or broker is interrupted, these values increase rapidly. If the file buffer has been activated for HD recording and the normal buffer is full, the data is stored in files on the hard disk ("file buffer").

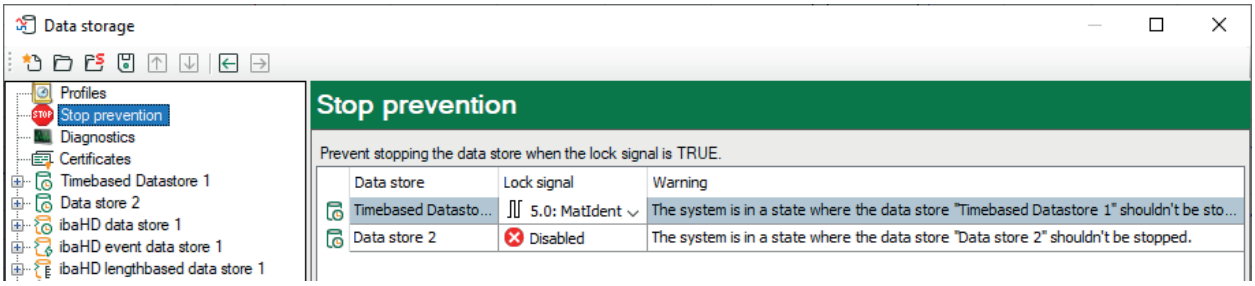
The column for the acquisition thread load shows the proportion of time needed for the evaluation of the triggers and generation of the row data.

At the top, above the table, the total load of the acquisition thread due to the data store is displayed. This is the sum of all the values in the *Acquisition thread load* column. It is a measure of how much time is needed to process the data which should be stored. If this value increases to 100%, alarm generation and client updates are suspended. If the value remains at 100% for a long time, the *ibaPDA* system becomes overloaded and acquisition is stopped when the driver buffer overflows.

Tip

With the *DataStoreInfo…* functions in the expression builder, you can make the relevant diagnostic data available as virtual signals for each data storage type. You can then visualize and record these signals.

8 Stop prevention



By enabling the “Stop prevention” function you have the possibility to prevent the data storage from being stopped by accident.

Stop prevention only affects only standard and QDR data stores (data files).

You can enable the stop prevention by selecting any digital signal. If attempts to stop the storage occur on an *ibaPDA* client while the signal is TRUE, a message box pops up asking for confirmation. The data storage will only be stopped if you answer this question with “Yes”. Using this feature you can protect a running data storage with signals from the plant, the process or by an operation.

If stop prevention is enabled, i.e., a lock signal has been assigned, a red STOP icon appears in the stop prevention node. If no lock signal is assigned, the icon is gray.

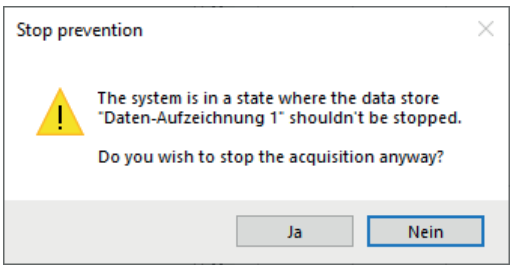
Lock signal
Click on the drop-down icon in this field and select the desired signal from the signal tree.

All digital signals can be used as lock signal, e. g.

- Digital input signals from sensors or PLC
- *ibaQPanel* inputs
- Virtual signals

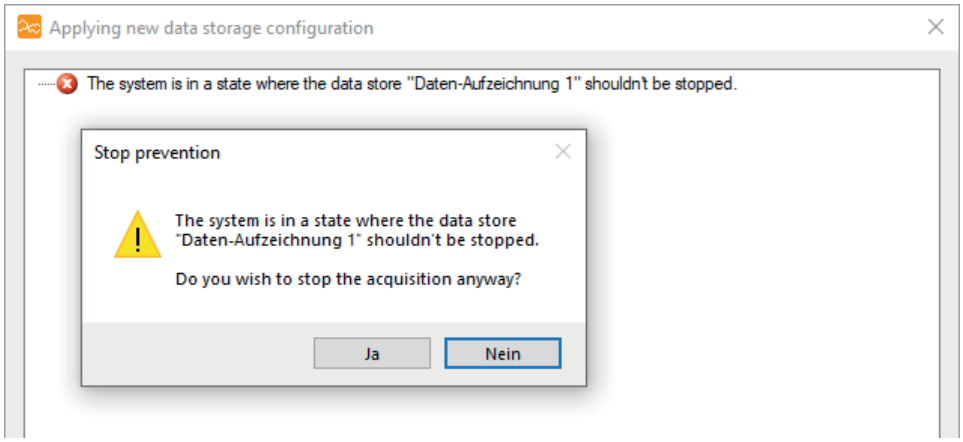
Warning text
You can overwrite or modify the standard text in this field. This helps you to include practical advice, directions or contact data in the warning message.

If the stop prevention is triggered by clicking on the stop trigger button, the warning message pops up immediately.



You can only acknowledge this message. If you still want to stop the data storage, you must deactivate the stop prevention by deleting the lock signal. Alternatively, you can stop the acquisition, in which case the same message will also appear. Nevertheless, you then have the option to end the acquisition and thus also the storage.

If the stop prevention is triggered by a change of the data store configuration, the warning message pops up, supplemented by an entry in the validation dialog.



If you click <Yes> at this point, then the storage in question is stopped, whereas the new configuration is applied and the storage is restarted. Meanwhile, the acquisition continues.

9 Certificates

The *Certificates* node gives you access to the central certificate store. This node is available in both the I/O manager and the data storage configuration dialog. From both locations, you can access the same pool of certificates and can manage the relevant certificates.

A detailed description of the certificate store and the handling of certificates can be found in the *ibaPDA* manual part 2, chapter *Certificates*.

10 Support and contact

Support

Phone: +49 911 97282-14

Email: support@iba-ag.com

Note

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

Product Security Incident Response Team (PSIRT) at iba

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

Email: psirt@iba-ag.com

Contact

Headquarters

iba AG
Gebhardtstrasse 10–20
90762 Fuerth
Germany

Phone: +49 911 97282-0

Email: iba@iba-ag.com

Mailing address

iba AG
Postbox 1828
D-90708 Fuerth, Germany

Delivery address

iba AG
Gebhardtstrasse 10
90762 Fuerth, Germany

Regional and worldwide

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

www.iba-ag.com

ibaPDA

Data visualization

Manual Part 6

Issue 8.13

Measurement Systems for Industry and Energy

www.iba-ag.com

Manufacturer

iba AG

Gebhardtstrasse 10-20

90762 Fuerth

Germany

Contacts

Main office	+49 911 97282-0
Support	+49 911 97282-14
Engineering	+49 911 97282-13
E-mail	iba@iba-ag.com
Web	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 2026, 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 and can be found in the iba help center docs.iba-ag.com.

Version	Date	Revision	Author	Version SW
8.13	02-2026	Layouts can be locked	rm	8.13.0

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

1 About this documentation 8

1.1 Target group and previous knowledge 8

1.2 Notations 8

1.3 Used symbols 9

1.4 Documentation structure 10

2 Introduction 11

3 Client preferences General 12

4 General handling of views 14

4.1 Add view 14

4.2 How to remove a view 14

4.3 How to rename a view 14

4.4 How to move a view 14

4.5 Arranging views by context menu 15

5 Selecting and displaying signals 17

5.1 Selecting signals 17

5.2 How to search for signals 18

6 Trend graphs 19

6.1 Controls of a trend graph 19

6.1.1 Toolbar 19

6.1.2 Context menu 21

6.1.3 X-axis buttons 23

6.1.4 Markers and signal table ("marker grid") 25

6.1.4.1 Markers 25

6.1.4.2 Signal table (marker grid) 26

6.2 Move signals 26

6.3 How to remove a signal from display 28

6.4 Show and hide signals 28

6.5 Using common or separate Y-axes 30

6.6 Bar graph on Y-axis 31

6.7 How to shift a graph 31

6.8 Remove Graphs 31
6.9 How to scale the axes 32
6.10 How to shift scales 33
6.11 Title for signal chart (curve) 33
6.12 Properties and preferences trend graph 34
6.12.1 Legend 38
6.12.2 Marker grid 39
6.12.3 Signals 40
6.12.4 X-axis 43
6.12.5 Y-Axis 45
6.12.6 Advanced 48
7 FFT view 49
7.1 Opening an FFT view in ibaPDA 49
7.2 FFT view overview 51
7.3 Main window 54
7.3.1 Waterfall 55
7.3.2 Contour view 57
7.3.3 Zoom 58
7.3.4 Legend 58
7.4 Spectrum graph and spectrum table 60
7.4.1 Spectrum graph 64
7.4.2 Spectrum table 68
7.5 Time graph and time table 70
7.5.1 Time graph 71
7.5.2 Time table 72
7.6 Spectrum parameter table 73
7.7 Slice view 74
7.8 Marker spectrum display 76
7.9 Markers 78
7.9.1 Interactive marker 79
7.9.2 Configured marker 83
7.10 Properties and preferences FFT view 86

7.10.1 Visuals ... 89
7.10.2 Bands ... 90
7.10.2.1 Band spectrum ... 94
7.10.3 Markers ... 94
7.10.4 Base axes ... 95
7.10.5 Value axes ... 97
7.10.6 Z-axis ... 103
7.10.7 InSpectra ... 104
7.10.8 Spectrum graph ... 105
7.10.9 Time graph ... 106
7.10.10 Spectrum parameter table ... 106
8 Scope view ... 107
8.1 General ... 107
8.2 Trigger ... 108
8.3 State indicator ... 109
8.4 Markers ... 109
8.5 Signal grid ... 109
8.6 Memory length ... 110
8.7 Overview ... 110
8.8 Chart ... 110
8.9 Axes ... 111
8.10 Background image ... 112
8.11 Properties and preferences scope view ... 113
8.11.1 Signals ... 114
8.11.2 Base axis ... 114
8.11.3 Value axis ... 115
8.11.4 Trigger ... 116
9 ibaQPanel ... 117
9.1 General ... 117
9.2 The ibaQPanel view ... 118
9.3 The display objects ... 119
9.4 Properties and settings Qpanel view ... 123

10 ibaCapture camera view 125

10.1 General 125

10.2 The camera view 126

10.3 Properties and preferences camera view 127

11 HD trend graph 130

11.1 Connecting to an HD server 130

11.2 Adding an HD trend graph 131

11.3 Operation of the HD trend graph 131

11.4 Properties and preferences HD trend graph 132

12 HD event table 133

12.1 Connecting to HD server 133

12.2 Configuration of events 133

12.3 Adding an event table 133

12.4 Operation of the event table 134

12.5 Properties of the HD event table 136

12.5.1 Basic settings 137

12.5.2 Alarms 140

12.5.3 Columns 141

12.5.4 Commands 144

12.5.5 Visuals 149

13 HD time periods table 153

13.1 Connecting to an HD server 153

13.2 Adding an HD time periods table 153

13.3 Operation of the time periods table 154

13.4 Properties of the HD time periods table 156

13.4.1 General Settings 156

13.4.2 Columns 158

13.4.3 Commands 159

13.4.4 Visuals 161

13.5 Configuration of time period queries 163

14 Numeric digital display ... 168

14.1 Properties Numeric digital display ... 171

15 Text digital display ... 173

16 Watch view ... 174

17 PQU spectrum ... 177

18 Phasor view ... 178

19 Orbit view ... 179

20 Circle view ... 180

21 Cycle view ... 181

22 Zoom function ... 182

22.1 Zooming in a trend graph view ... 182

22.2 Zooming in the FFT view ... 183

23 Managing layouts ... 184

23.1 Definition ... 184

23.2 Internal structure and file storage ... 184

23.3 Layout Manager ... 185

23.4 Toolbar for basic functions ... 187

23.5 General settings ... 189

23.6 Display style ... 191

23.7 Toolbars and menus ... 191

23.8 Layouts ... 193

23.8.1 Layout pool ... 194

23.8.2 Layout structure ... 198

23.8.3 Startup layout ... 199

23.8.4 User assignment ... 201

23.9 Save layout in project ... 204

23.10 Automatic scaling of the QPanel window ... 204

24 Support and contact ... 206

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 they are capable of assessing safety and recognizing possible consequences and risks on the basis of their specialist training, knowledge and experience and knowledge of the standard regulations.

1.2 Notations

In this manual, the following notations are used:

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

1.3 Used symbols

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

Danger!

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

Observe the specified measures.

Warning!

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

Observe the specified measures.

Caution!

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

Observe the specified measures.

Note

A note specifies special requirements or actions to be observed.

Tip

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

Other documentation

Reference to additional documentation or further reading.

1.4 Documentation structure

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.

Part 1	Introduction and installation	General notes, license policy and add-ons Installation and program start User interface, system architecture, client server User management, printing
Part 2	I/O Manager	Basic information about the I/O Manager, general settings Groups and vector signals, text signals, outputs, configuration files
Part 3	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.
Part 4	Expression builder	All functions for calculating virtual signals
Part 5	Data storage	Types of data recording, recording profiles, signal selection
Part 6	Data visualization	All display modes for live data, their operation and settings
Part 7	Appendix	Various additions, error listings, etc.

2 Introduction

ibaPDA offers different types of views for different ways of representing data. The term "view" basically refers to the frame containing certain display types and relevant data. Views can be arranged in different ways on the screen, as further described below.

Available view types:

- Trend graph
- FFT view
- Watch view
- Scope view
- QPanel
- Camera view
- HD trend graph
- HD event table
- HD time periods table
- Numeric digital display
- Text digital display
- PQU spectrum
- PQU phasor view

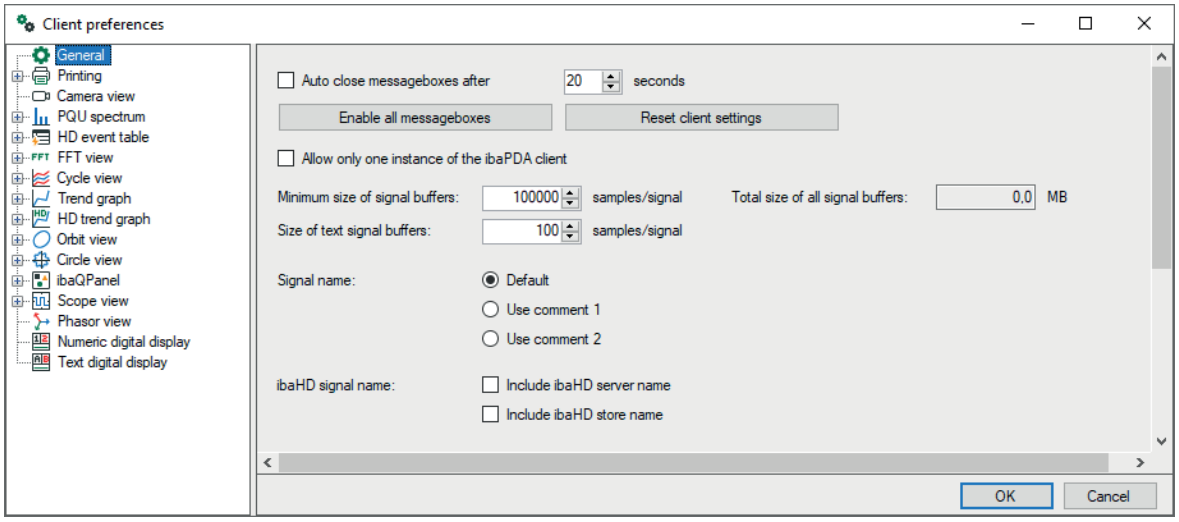
+ license-depending views:

- Orbit view
- Cycle view
- Circle view

Things such as screen layout, docking windows arrangement, number and arrangement of views, number and presentation of individual signals, view settings and different layouts are saved in the layout configuration file.

This section describes the general handling of the views and their settings and properties.

3 Client preferences General



Auto close messageboxes after ... seconds
Here you can choose if and when messages posted by the *ibaPDA* program (e.g. status or error messages) should be closed. If this option is disabled then the messages must be confirmed manually.

<Enable all messageboxes> button
Press this button if you want to be sure that all message boxes are enabled even those which had been disabled before.

<Reset client settings> button
Press this button if you want to reset all preference settings made on this client back to factory settings. The client will restart after.

Allow only one instance of the ibaPDA client
If you select this option, *ibaPDA* client is prevented from launching multiple times on the same computer. Technically this corresponds to the use of the "/reuse" switch when starting via the command line.

Minimum size of signal buffers / Size of text signal buffers
These settings determine the size of each signal buffer, or text signal buffer, on an *ibaPDA* client. The signal buffer contains a number of samples corresponding to a certain time with regard to the sample time. A setting like the 100000 default value, for example, will enable the *ibaPDA* client to show a trend graph for 100 seconds for a signal sampled with 1 ms. If you would like to see a longer period over the x-axis, e. g., 10 minutes, you should increase the buffer size accordingly.

Keep in mind that this setting applies to all signals. If you use large buffers for displaying lots of signals in multiple views, you can run into "Out of Memory" errors. If this happens, reduce the buffer sizes.

Signal name: default / use comment 1 / use comment 2
By selecting one of these options, you can use other sources for alternative signal names in the signal tree and views. Default value is the signal name as configured in the I/O configuration.

Please note that the signal name is replaced by comment 1 or comment 2 if you check one of the options *Use comment 1* or *Use comment 2*. Comment 1 or comment 2 in return will then contain the original signal name. This will not affect the I/O Manager but the search functions and signal legend contents.

Tip

This feature can be used to give signals different names in different languages. An integrator can use English signal names while his Chinese customer can use Chinese signal names, for example.

HD signal name: including HD server name / including HD store name
By selecting one or both of these options, you can determine how the signal names of historical data should be displayed in the views.

Note

The preferences do not allow the selection of dedicated signals. Only in the properties of the views/displays you can assign signals to the views/displays.

4 General handling of views

4.1 Add view

A new view can be opened by clicking on one of the buttons in the main toolbar or by selecting the desired view type in the menu *View – Add view*. The new view opens in its own window.

4.2 How to remove a view

If there is only one view in the window (one tab), click on the little cross in the upper right corner.

If there is more than one view in the window (several tabs), select the view to be removed first (indicated by bright black letters in the tab and in the foreground). Then click on the little cross in the upper right corner.

Or right-click on the tab of the relevant view and select *Remove* from the context menu in order to remove a view.

If there is no tab and liitle cross, you can display them by disabling *Hide tabs* in the *View* menu.

If you just want to hide a view to save space, the best way is to shift it to another view so it can be in the background as a tab.

4.3 How to rename a view

You can name and rename a view. The name is displayed on the tab. Right-click on the tab of the relevant view and choose "Rename" from the context menu. Enter the name into the "New name" dialog field and click <OK>.

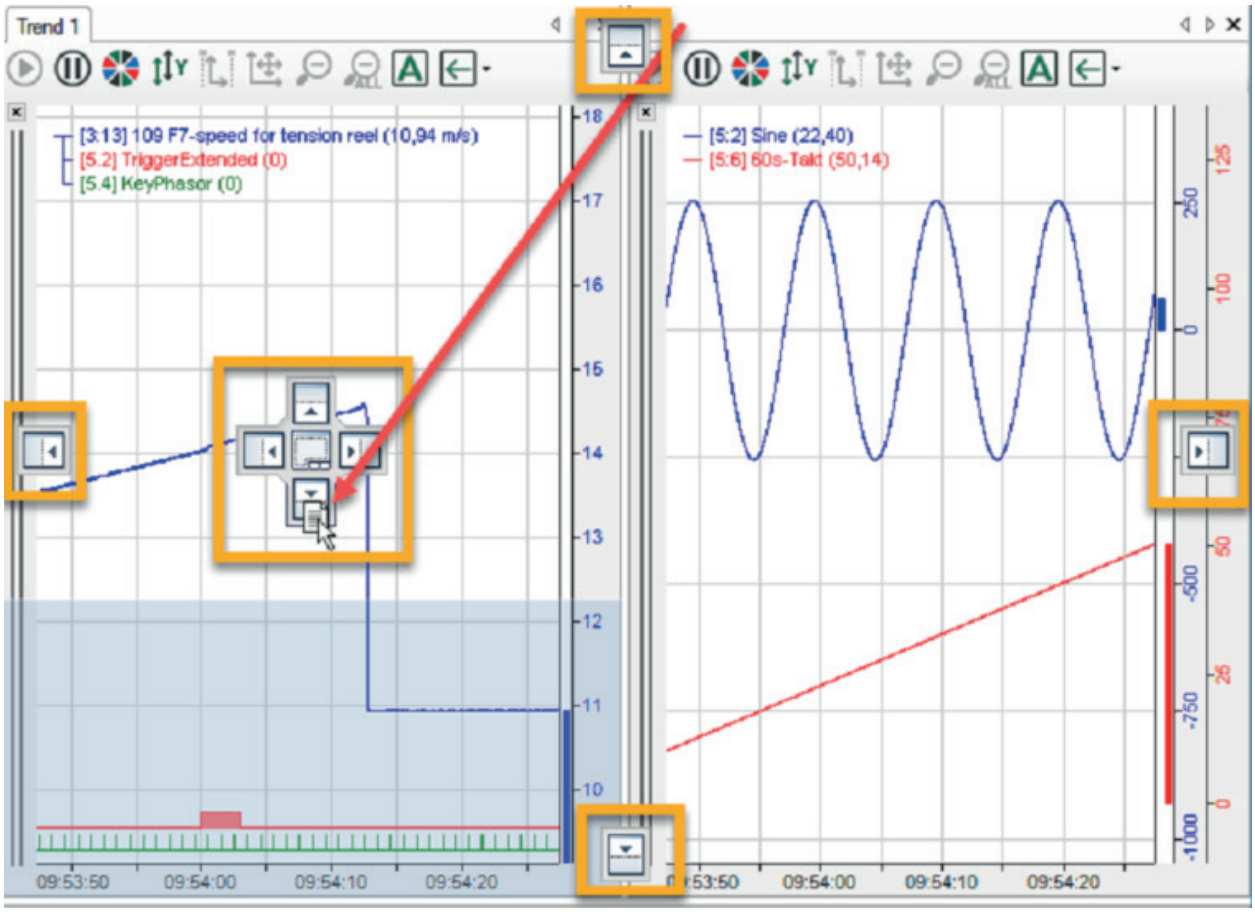
4.4 How to move a view

All visible windows in the *ibaPDA* user interface are dockable windows. So are the signal monitors (views). You can move, drag or squeeze the views as you like. Or they can be arranged on top of one another as tabs.

For doing so, use the drag and drop option or the context menu.

Combining multiple views in one window (creating a monitor group)

Click on the tab of the view you would like to move, hold the mouse key and drag the view in the middle of the requested window until a cross-shaped group of docking pads appears. Place the mouse over the central pad. A blue area indicates where the view will be placed. Release the mouse button and the view becomes part of the monitor group. Use the same method for moving the views to the side, above or beneath other views.

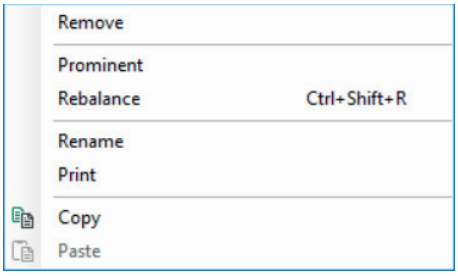


Separating views

In order to separate views from a joint window, just click on the corresponding view's tab and drag it to a position out of the center of the current window. As soon as a blue frame appears in the upper, lower, left or right part of the window, let it drop. The view is opened in a new window. All windows are automatically rescaled to provide optimal fit on the screen.

4.5 Arranging views by context menu

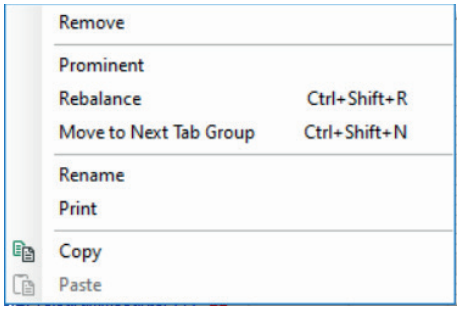
Right-click on the tab label of the view to open the context menu.

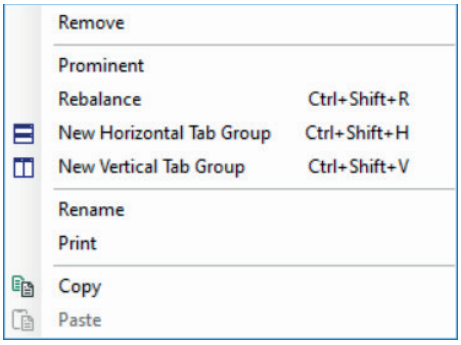


Beside the *Close*, *Rename*, *Print*, *Copy* and *Paste* commands, there are also *Prominent* and *Rebalance*.

Select *Prominent* if the respective view is to be shown in full size in the window. This hides all other views, such as trend graphs, FFT, oscilloscope or QPanel views. To switch back to the previous view, click *Rebalance*.

Depending on the number of views and their arrangements, the options in the context menu differ.





As soon as you have multiple tabs arranged, the *Move to Next Tab Group* command or *Move to Previous Tab Group* command appears. "Next" and "Previous" refer to a clockwise manner. If you click on one of these commands, the tab in question is added to the next or previous tab group.

5 Selecting and displaying signals

The methods of selecting and displaying measured signals is optimized for mouse keys. Many actions can be performed by drag & drop or via the context menu (right mouse button).

5.1 Selecting signals

There are two basic methods to select signals for display:

- A double-click on the desired signal in the signal tree.
  - If no view has been opened yet, a new trend graph containing this signal is opened.
  - If one or more trend graph(s) are already opened, the signal will be added to the focused trend graph.
  - If another view is active, e. g. FFT or scope view, the signal will be added to this view.

  You can also drag and drop the signal into the display window. This method also applies to many other views, not just to the trend graph.

With regard to the trend graph, these two methods offer further helpful variants for daily use.

Often, it does not make sense to open a separate chart for each signal, as the display window is filled quiet quickly and therefore hard to read. So, you can position several signals in one trend view and decide whether the signals get their own Y-axis or a joint one.

- Select the first signal:
  Drag the desired signal into an open range of the trend graph.
- Select another signal to be displayed in a new signal chart:
  Drag the desired signal in the area of the X-axis in the display window or double-click on the signal name in the tree.
- Presenting another signal in an existing graph:
  Simply drag the desired signal into the area of the desired graph. The signal is displayed in the same graph but with its own Y-axis.

Tip

If you want multiple signals to be displayed quickly in a chart with separated Y-axes, press <Ctrl> while double-clicking on the relevant signal names.

- Select another signal to be displayed in an existing chart, but with respect to the same Y-axis as the existing signal:
  Drag the desired signal in the area of the Y-axis of the requested chart. Both signals now use the same Y-axis. A new color is automatically assigned to the new signal. In the upper left corner of the chart, you will find the signal names (legend). Signals with a common axis are linked by a line.

Tip

If you want multiple signals to be displayed quickly in a chart with a common Y-axis, press <Shift> while double-clicking on the relevant signal names. Each new signal is added to the Y axis of the lowest signal in the legend.

5.2 How to search for signals

For finding and selecting signals to be displayed, there is a search function available in the signal tree window.

Just click on the *Search* tab at the bottom.

Enter a complete signal name or parts of it in the entry line.

Then press <RETURN>.

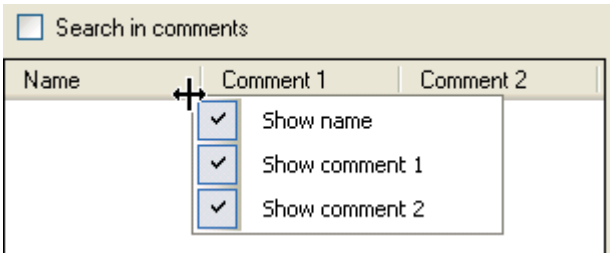
The result list shows all signals containing the search item entered.

If there are too many results, enable the "Search in previous results" option, refine your search term and restart the search.

If you want to keep the results of several search processes, check the "Add to previous results" option before any further search.

Additionally, you can search for the specified string in the comments 1 and 2.

Besides the "Name" column in the result list, you can show or hide the columns for comments using the context menu.



The signals can be dragged directly from the result list into the view.

6 Trend graphs

A trend graph in the context of *ibaPDA* is a display of a certain combination of signal charts and curves. Similar to a recorder, trend graphs are used to show live measured data.

Each trend graph view can basically consist of an arbitrary number of curves. The signal curves can be zoomed in and the scales changed.

The pause button ⏸ can be used to stop a current trend graph, for example, in order to consider the measured values. The data storage continues in the background.

The default configuration shows a trend graph view without a graph, X or Y axes. After you configured the signals in the I/O manager, they are immediately available for display. For better clarity, you may open several views, which can be arranged in different windows side by side or in tabs.

In addition to the normal trend graph, there are also trend graphs for displaying data from the *ibaHD server* ("HD-trend graph") and existing measurement files ("offline trend graph"). The basic operation of trend graphs is the same for all types.

6.1 Controls of a trend graph

6.1.1 Toolbar

Each view has a couple of control buttons in its own toolbar.

Depending on the display type, the toolbars for trend graph, oscilloscope, FFT view, camera view or QPanel differ.

The following describes the trend graph toolbar. For description of the other tool bars, please refer to the corresponding chapters.

The functions are (from left to right):

Start scrolling
Only enabled if <Pause scrolling> button was pressed before. Restarts the scrolling of the display at the current time.

Shortcut key: <F6> (toggle)

Pause scrolling
Only enabled if the display is scrolling. Stop the continuous display. After clicking this button, the display stops scrolling. A signal table ("Marker table") appears below the signal chart and two rulers are shown in the signal display. The rulers can be moved with the mouse along the X-axis. The corresponding Y-values are displayed in the table below the trend view. The X-axis can be moved with the mouse. This enables the user to see values from the past. For more pause mode control options, see ↗ *X-axis buttons, page 23*.

Shortcut key: <F6> (toggle)

Automatically assign signal colors
All curves of this display are colored per signal chart according to the standard scheme.

Also see *Settings*, chapter ↗ *Properties and preferences trend graph, page 34*

Autoscale all
All curves of this display are automatically scaled per signal chart and Y axis.

Shortcut: <F5>

Zoom and auto scale between markers
Only enabled in pause mode.

View zooms into range between the markers X1 and X2 and the Y-values are auto scaled.

Restore manual scale
Only enabled if a manual scaling has been defined in the settings of the signal chart. The manual scaling are restored (where defined) after an autoscaling or zooming action.

Zoom out one step
Only enabled if the display has been zoomed. Return to the previous zoom factor (reduce).

Shortcut: <F3>

Zoom out all
Only enabled if the display has been zoomed. Will zoom out to initial (automatic) zoom level (no zoom) according to axes settings.

Shortcut: <F4>

Change legend style
With every click on this button, the legend in the trend graph changes between transparent, opaque and invisible.

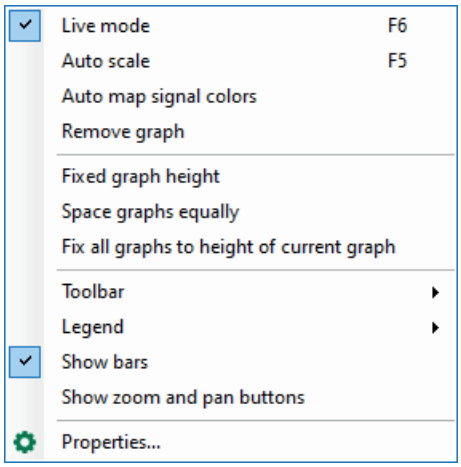
Change orientation of graphs
Drop-down menu for selection of scrolling orientation.

The entire toolbar as well as single toolbar buttons can be hidden. In order to do so, use the context menu (see next chapter).

6.1.2 Context menu

You can open a context menu by right-clicking in the area of a trend graph.

The context menu provides further control options for the view and may look as follows:



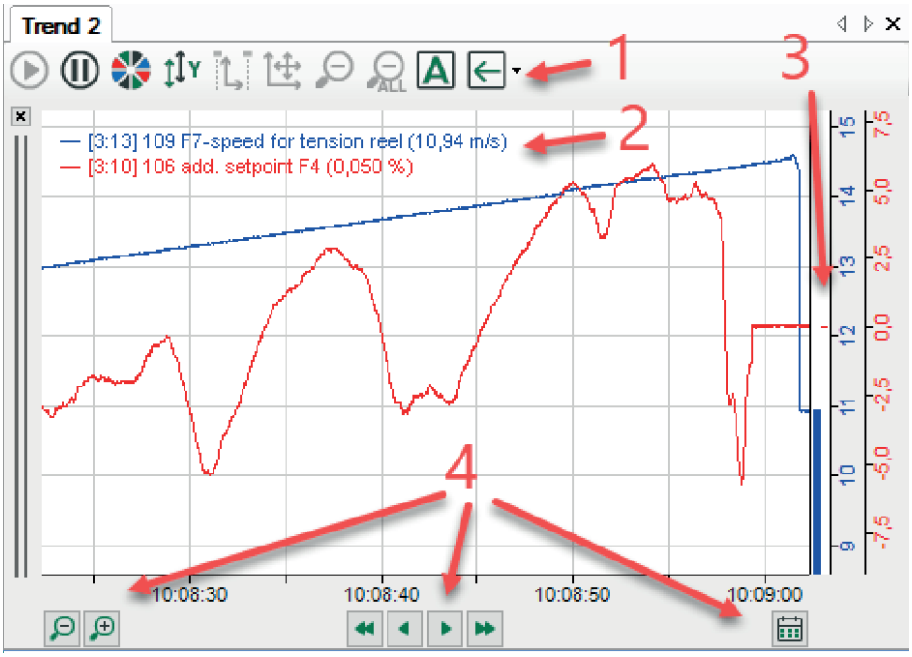
Some menu commands correspond to the functions of the toolbar buttons as described above, like *Live mode/Pause*, *Auto scale* and *Auto map signal colors*. However, the first 4 commands from top only apply to the trend graph the context menu was opened in.

Fixed graph height
In order to avoid up or downsizing of a trend graph while resizing the view window, you can fix the height of the signal chart individually. When enabling this option, the current height of the signal chart is maintained.

In case of resizing a window vertically, all charts are subject to resizing except for the signal charts, whose heights are defined by this option. If this option is enabled for all signal charts of a view, the minimum height of the display window will always comply with the overall height of all graphs.

Space graphs equally
If you click this command, all charts in the view lose their fixed heights and are automatically scaled to fit equally in the view window.

In the next block of the context menu, you can display or hide the indicating elements of a signal monitor:



Show toolbar (1)
Show or hide the toolbar of the view where the context menu was opened. In the sub-menu you can hide either the entire toolbar or single buttons.

Show signal values in legend (2)
Show or hide the signal values at the end of the signal legend. Another option is to choose a customized legend text (which is to be configured in the *properties*). This command applies to all trend graphs within the view where the context menu was opened.

Show bars (3)
Show or hide the value bars along the Y axes. This commend applies to all trend graphs within the view where the context menu was opened.

Show zoom and pan buttons (4)
Show or hide the zoom and pan buttons on the X axis of the view where the context menu was opened.

For more information on controlling zoom and pan buttons, please refer to ↗ *X-axis buttons, page 23*.

Properties...
If you click on this command, you will enter the *Properties* dialog for setting up of the trend graph view.

For more information, see ↗ *Properties and preferences trend graph, page 34*.

6.1.3 X-axis buttons

On the X axis there are various buttons for the navigation. For immediate showing or hiding the buttons, you can use the context menu of the view (see above).

	Zoom function
	Scroll
	Jump to a date

You can enable and configure these buttons in the properties of the X axis.

Zoom function

In the pause mode, the <+> button enlarges the center of the X axis and in the live mode the last part of the X axis. The <–> button reduces accordingly. The default value of the zoom factor is 2. This means if the section of the X axis is enlarged, it is half as large as the original section, if it is reduced, the section of the X axis is twice as large as the original section. The zoom factor can be modified via the X axis properties.

For more information, see ↗ *X-axis, page 43*.

Scroll

	Scrolls in large steps to the left, starting from the current position
	Scrolls in small steps to the left, starting from the current position
	Scrolls in small steps to the right, starting from the current position
	Scrolls in large steps to the right, starting from the current position

The step can be set via the X axis properties. The default setting for little steps is relative 0.75 and for large steps relative 2. Alternatively, the step can be set as absolute value in hh:mm:ss time units for time-based values and in meter for length-based values.

Example: if the X axis indicates a range of 100 s and if the time stamp goes to 10:00:00, a small step jumps to the left by 75 s on 09:58:45 and a large step to the left by 200 s on 09:56:40.

For more information, see ↗ *X-axis, page 43*.

Panning is possible in paused mode only. When pressing one of the pan buttons in live mode, the display will switch to paused mode automatically.

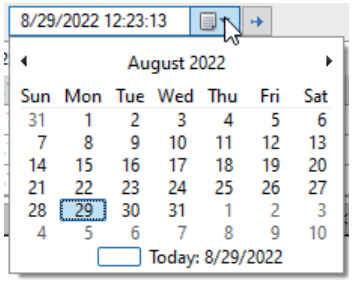
The functions zoom and scroll can be carried out with the help of the keyboard:

Button	Key	Description
	<+>	enlarging
	<–>	reducing
	Page up	small step to the right (only in pause mode)
	No key function	large step to the right (only in pause mode)
	Page down	small step to the left
	No key function	large step to the left

Jump to a date

Clicking the button opens a calendar where you can enter a date and time on second basis.

With a click on the arrow to the right, the display of the trend graph jumps to this time point.



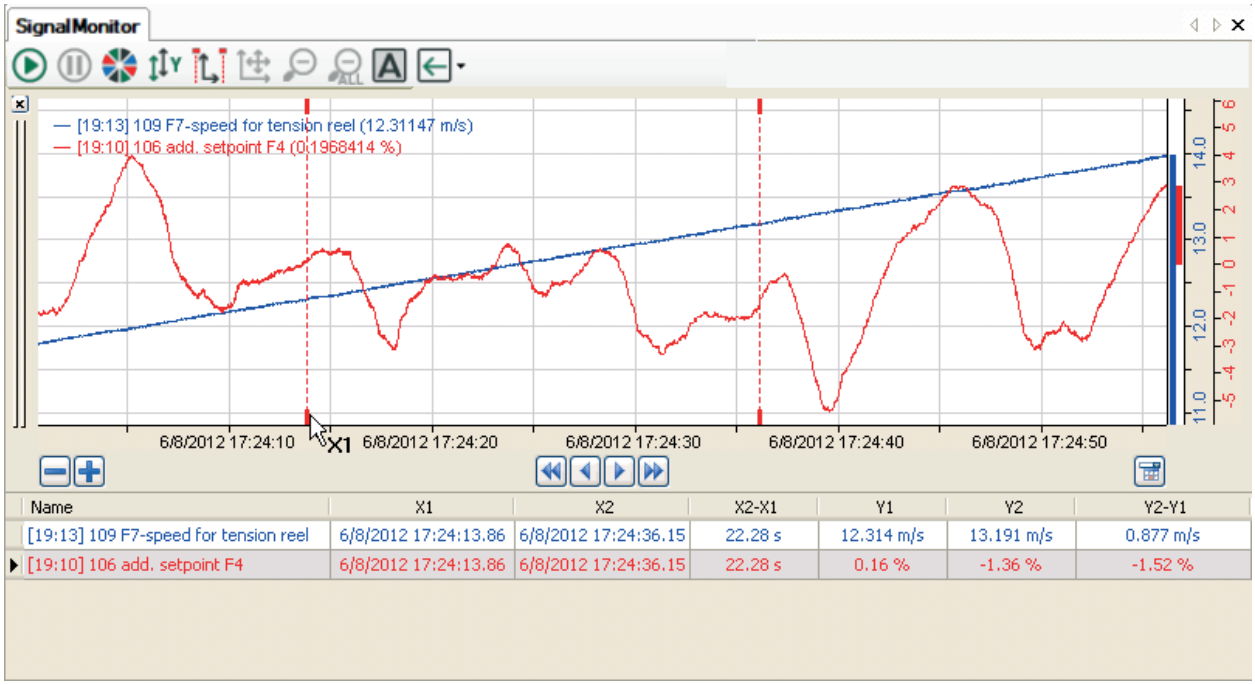
This function is also available on length-based HD trend graphs because the time stamp is always stored along with the length-based data.

A time leap is possible in paused mode only. When pressing the jump button in live mode, the display will switch to paused mode automatically.

6.1.4 Markers and signal table ("marker grid")

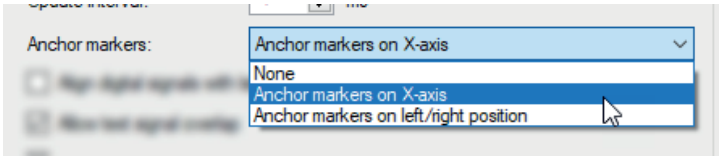
Here, you'll find information about the marker and signal table settings.

6.1.4.1 Markers



In pause mode, there are two markers, i.e. X1 and X2, available in the graph. You can move the markers individually in order to check the Y values and differences between the X and Y values of the markers.

There are two modes to control the position of the markers. The markers can be anchored to the X axis or not. These settings can be changed in the trend graph properties.



If the *Anchor markers* option is disabled (*None*), the markers will remain at the same pixel while moving the X-axis and are positioned on a new time stamp, e. g. when dragging or panning.

If the option is enabled (default: *Anchor markers on X-axis*), the markers will remain at the same time stamp. So, if the X axis is moved, the markers will also move. If the X axis is moved so far that the time stamp of the marker is no longer visible, the marker will be moved to the edge of the graph and the grab handles will become lighter. The table still shows the correct position of the marker. You can drag the marker from the edge of the graph back into the current X axis range at any time.

If the option *Anchor markers on left/right position* is enabled, then X1 will be positioned at the left end and X2 at the right end. Hence, the entire visible trend is always between the markers.

6.1.4.2 Signal table (marker grid)

The signal table is disabled in live mode by default. It pops up automatically as soon as the view switches into pause mode.

However, you can change this behavior in the preferences or properties of the trend graph.

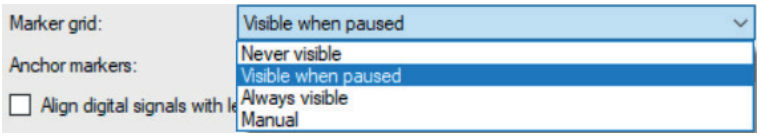
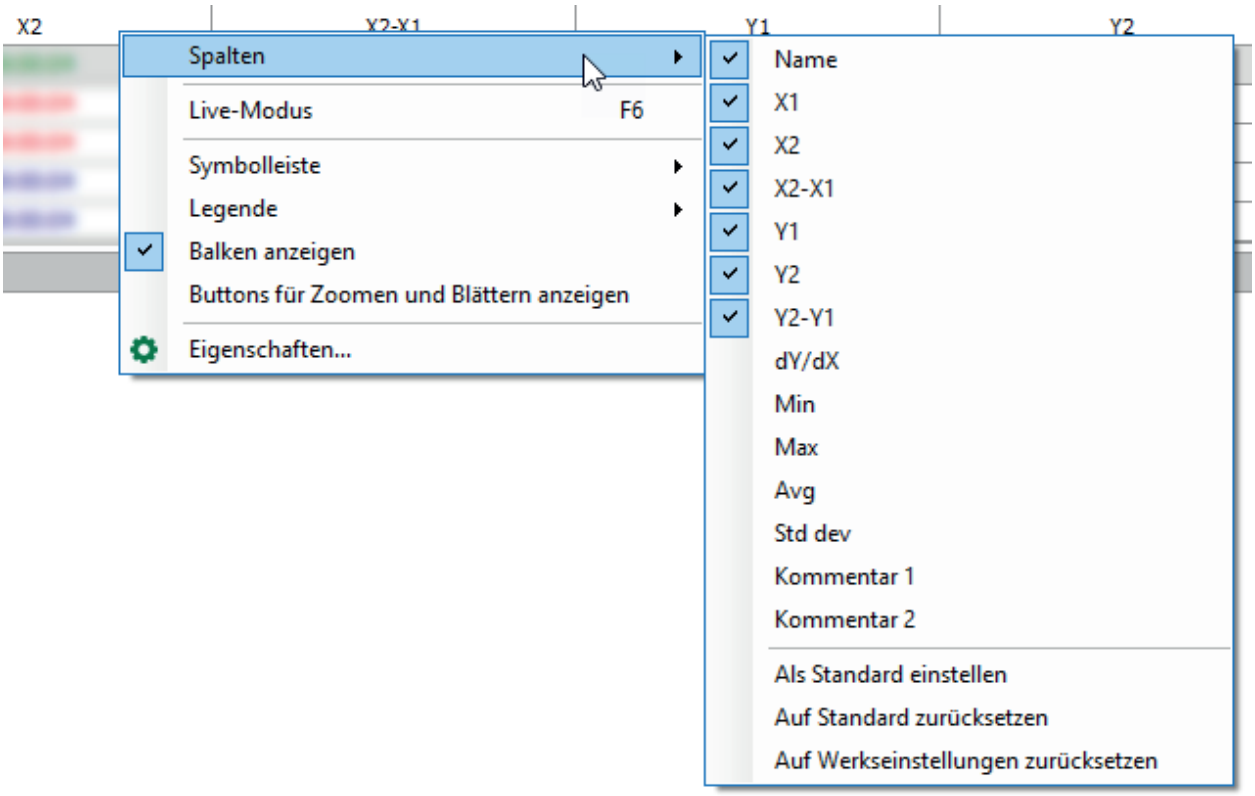


Table columns

The factory default setting shows some essential table columns. If needed, you can add or remove more columns. Just make a right click on any column header to open the context menu.



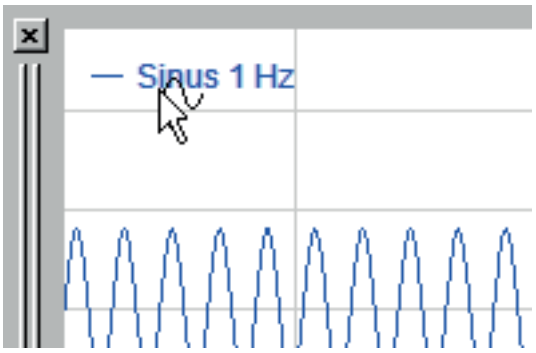
After you've found a new setting, you can save this as a new default setting by *Set as default*. If you make temporary changes to the column setting you can return to the last saved default setting at any time by *Reset to default*. Finally, you can also restore the factory setting.

6.2 Move signals

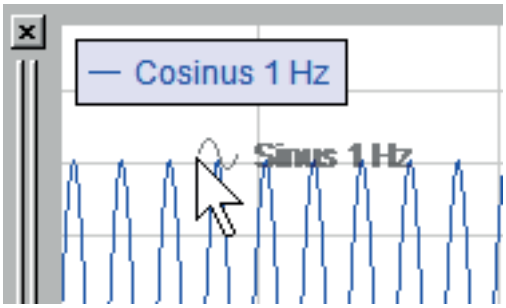
Signals can be shifted between the graphs and even beyond the limits of the window. This means that a signal can be dragged from one graph to another graph that already has a signal.

Here is how it works:

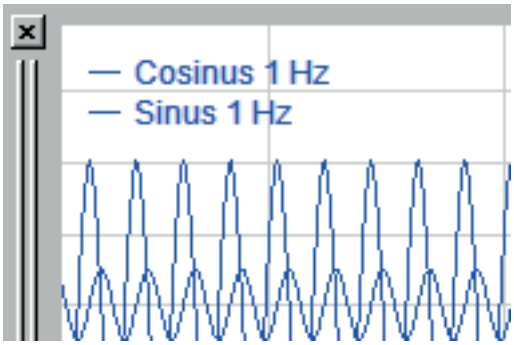
1. In the graph, put the mouse pointer on the signal name to be moved. The cursor indicates with a wavy line that it has acquired the signal.



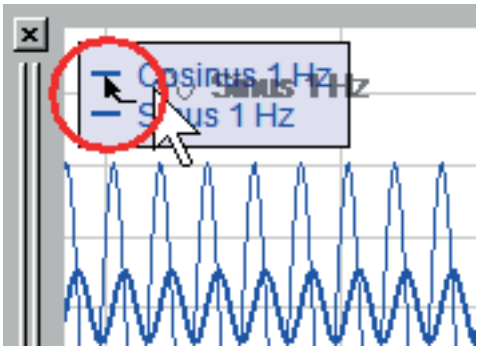
2. Now press the mouse key, hold it and drag the signal into the other graph and let it drop in a free area.



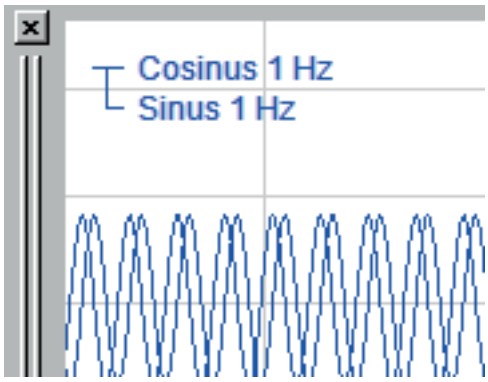
3. The result: two signals, each with its own Y-axis.



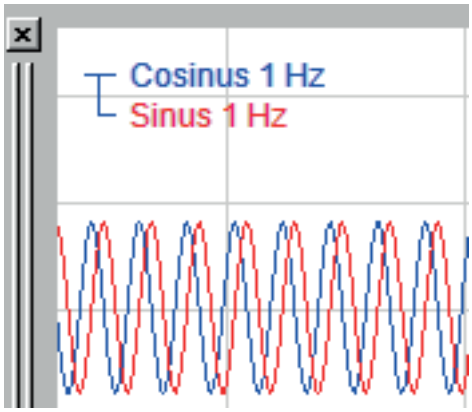
4. If, in step 2, the signal is not dropped, but dragged on the already existing signal (cursor shows a little arrow), the shifted signal will be assigned to the same Y axis.



5. The result: two signals with a common Y axis.



6. By shifting a signal from one graph to the other, the color will not change automatically. In order to get different colors, click on the “Auto color assignment” toolbar button.



7. If you want to separate signals, just click on the corresponding signal in the legend and drag it into the free area of the display window's X axis. A new graph is opened with that signal.

6.3 How to remove a signal from display

In order to remove a signal, right-click on the signal name in the legend and choose "Remove signal" from the context menu.

Alternatively, you can make a right mouse click on the Y axis of the corresponding signal and choose "Remove Y axis" from the context menu ("Remove axis" in case of FFT and oscilloscope view).

But remember: by deleting the Y-axis, all signals assigned to this axis are deleted.

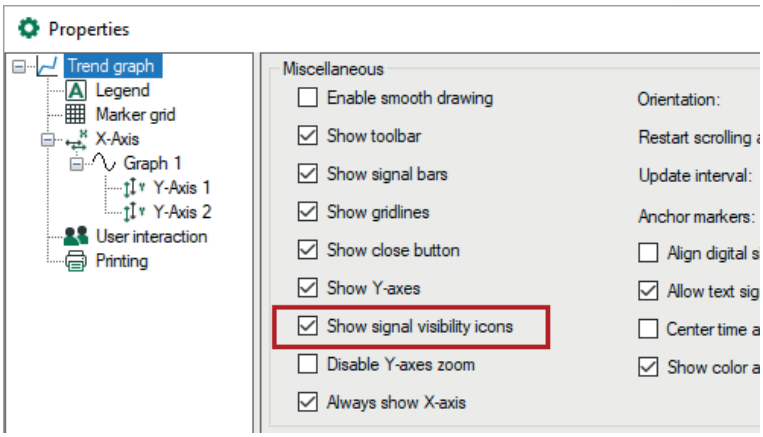
6.4 Show and hide signals

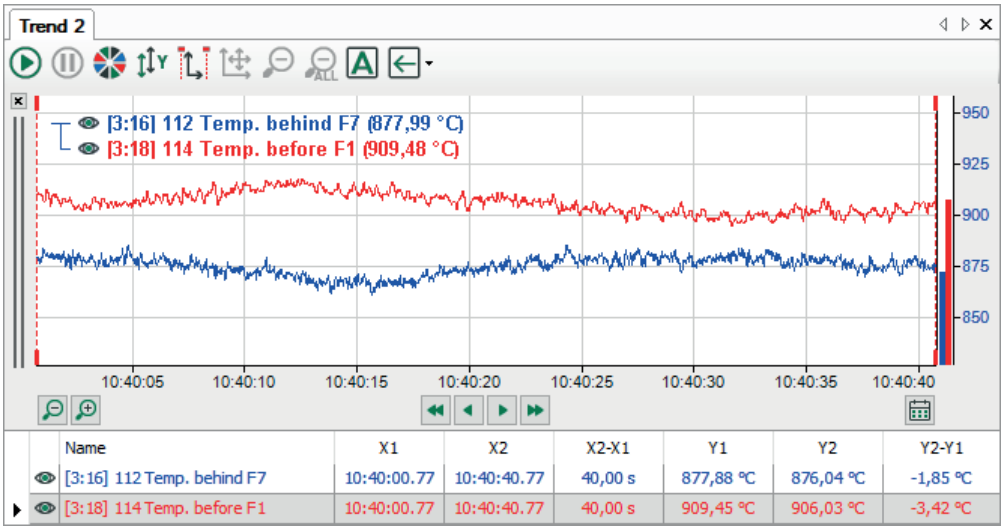
If you do not want to remove signals in a trend curve but just temporarily hide them, you can use the function for hiding and displaying a signal.

You have the following options:

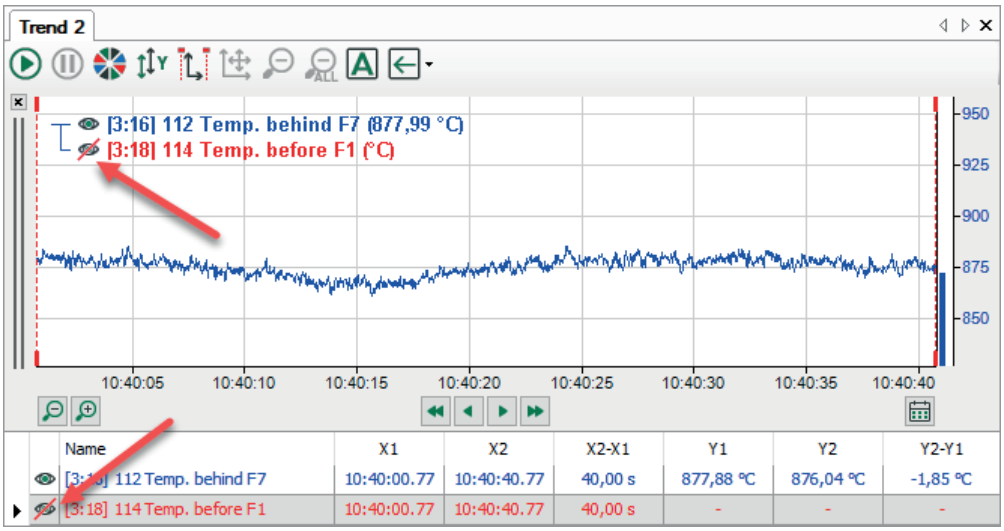
- By right-clicking on the signal name in the legend, you can display or hide a signal in the context menu.
- If the signal table below the trend curve is open, remove the check mark in the far left column (monitor icon).

- If you have enabled the "Show icons to hide signals" option in the properties of the trend display, you can show and hide the signal by clicking on the monitor icon in the signal legend.





The figure above shows both signals visable, the figure below shows the red curve hidden.



If a signal is hidden then the signal bar is also empty and no marker or actual values are displayed in the signal table.

6.5 Using common or separate Y-axes

This description only applies to trend graphs.

You can link a signal to another one, as described in the "Shifting signals" section, to get a common Y axis.

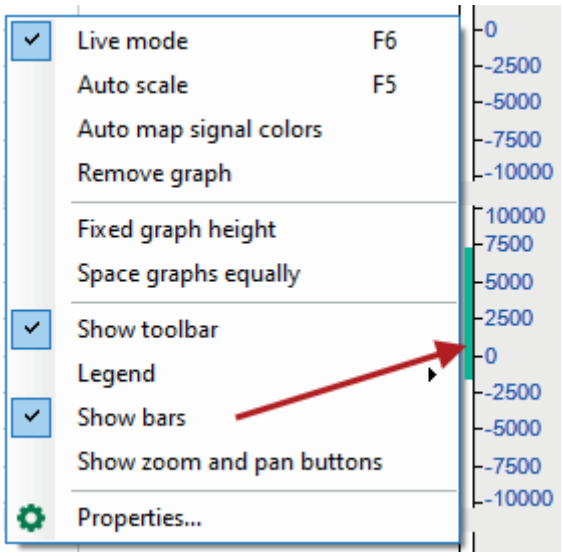
In order to separate Y axes, drag the signal into the free area of the graph, then let it drop.

6.6 Bar graph on Y-axis

This description only applies to trend graphs.

Bar charts on the Y axis are another option of signal display.

You can enable the bar graph display either by using the context menu by clicking in the trend graph and select "Show bars", or in the "Properties" dialog *Trend graph – Show bars*.



6.7 How to shift a graph

This description only applies to trend graphs.

You may change the top-down order of the charts inside a trend graph view. For a vertical scrolling orientation of the trend graph, the following applies to the right-left order of the charts.

1. Click on the caption bar (gray double line) of the chart you want to move. Hold the mouse key and slightly move the mouse until a bright black frame around the chart appears.
2. Keep mouse key depressed and move the chart up or down, for example, until the frame touches another chart. At first, only the black frame moves and shows above which chart the moved one will be inserted if you drop it.
3. Release the mouse button and the order is changed.

6.8 Remove Graphs

This description only applies to trend graphs.

You can remove a graph from a trend graph view with the following methods:

- Click on the little cross top left of the caption bar of the chart.
- Right-click in the free area of the graph to open the context menu and choose *Remove graph*.

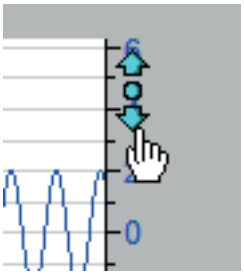
6.9 How to scale the axes

Autoscaling

In order to display a signal between its amplitudes in a graph, use the autoscale function. Make a right click in the graph you would like to scale and choose "Autoscale". All signals, i.e. all Y axes of this graph, will be scaled with respect to the highest and lowest amplitude. If you would like to autoscale all graphs in a view, press <F5> or click on the button in the view's toolbar.

Scaling with the mouse

You can change the Y scale with the mouse. Position the mouse cursor near the upper end of the Y axis until two blue arrows and a dot appear.



Mouse click (hold down) on the up-arrow: the scale is stretched (zoom in).

Mouse click (hold down) on the down-arrow: the scale is compressed (zoom out).

Mouse click on the dot: the scale is autoscaled.

If you use a mouse with scroll-wheel, you only need to place the mouse cursor on the Y scale and move the wheel up or down for changing the scale. This also works on the X axis. There is also such a function for the FFT and oscilloscope views.

Scaling in the graph's properties

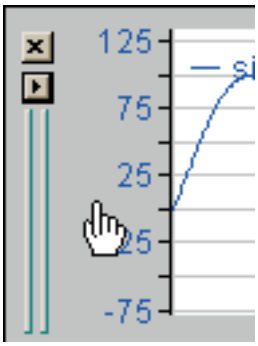
Right-click in the graph you want to rescale. Choose *Properties* from the context menu. A dialog opens which shows the settings of this graph. On the Y axis node you can set a scale by manually entering a lower and upper limit. If a graph has separate Y axes, there is one node for every Y axis in the dialog.

Note

You can open the dialog for the display properties also via the *Configuration - Preferences* main menu. The settings you make there will generally apply to new views and graphs. Views and graphs already there will not be changed.

6.10 How to shift scales

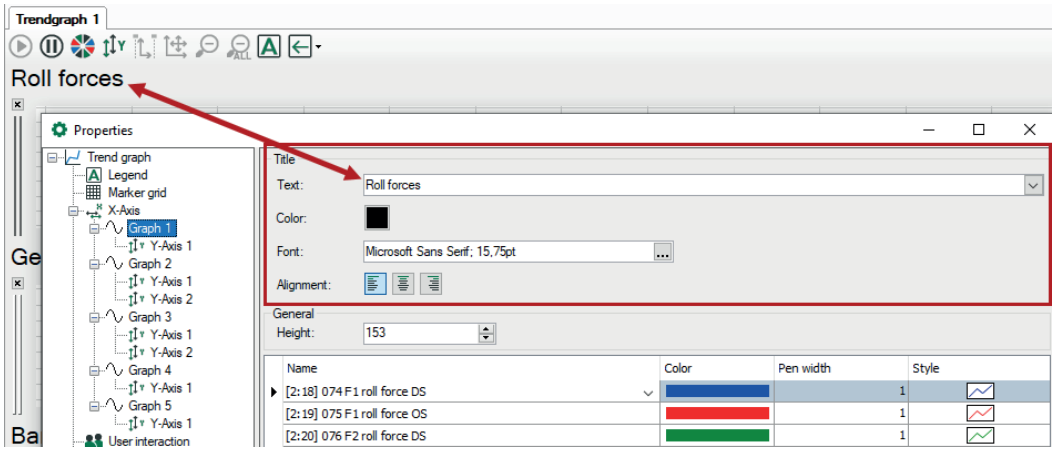
Move the mouse cursor to the Y axis until the hand symbol appears. Keep the left mouse key depressed to move the scale up or down. In the pause mode of the display, the X-axis can be moved in the same way.



6.11 Title for signal chart (curve)

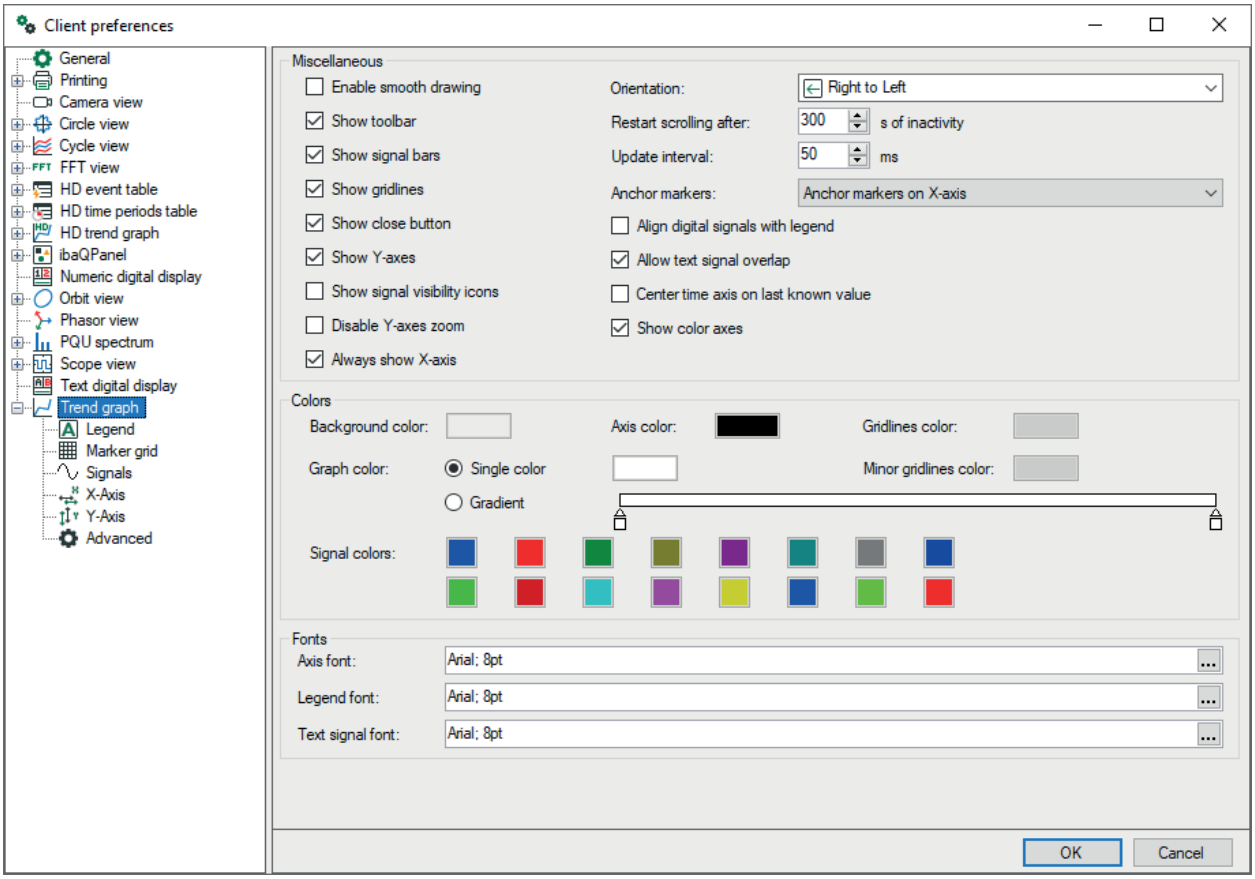
You can define a title within a trend graph view for each signal chart (graph).

To do this, right-click in the desired signal chart and select *Properties...* In the Properties dialog, select the graph for which you want to define a title. Now, you can enter a title and determine the font, color and alignment of the title bar.



More curves, which are part of the trend display, can be reached in the same dialog. Each graph or graph group in a trend display can receive its own title.

6.12 Properties and preferences trend graph



The settings in the "Preferences" dialog basically equal the ones in the "Properties" dialog, except for the "Advanced" node, which is not available in the "Properties".

The "Signals" branch in the preferences tree corresponds to the "X graph" branch in the properties tree. The settings for the Y-axis are available separately in the Properties of a trend display for each curve (Y-axis x).

Miscellaneous

Enable smooth drawing

The curves will be graphically smoothened if this option is selected.

Show toolbar

This option determines whether or not the toolbar for display control of each view is displayed below the tab.

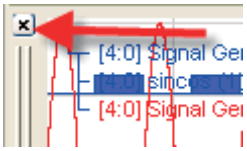
Show signal bars

By checking this option, a bar graph is displayed near the Y axis. Each signal is displayed by a bar in the corresponding color.

Show gridlines

By enabling/disabling this option, you can show or hide the grid lines in the display. Enabled by default.

Show close button
By enabling/disabling this option, you can show or hide the small buttons for closing all signal charts of a display.



Hiding the button provides some more space on the screen and prevents the unintended closing of a signal chart. Enabled by default.

Show Y-axes
By enabling/disabling this option, you can show or hide the Y axes of all signal charts of a display. Enabled by default.

Show signal visibility icons
If you enable this option then a small eye symbol is displayed in the signal legend. By clicking on this symbol you can temporarily hide a signal curve without removing it from the trend display. Clicking again makes the curve visible once more.

Disable Y-axis zoom
If you enable this option then all the actions for zooming into the display are only applied in the X direction; the Y direction remains unaffected.

Always show X-axis
If you select this option then it is ensured that the x-axis is always displayed under the lowest fully-visible signal chart. If this option is not enabled, the x-axis can scroll downwards out of sight when scrolling within the display (if you have created many signal charts).

Orientation
The orientation of the "paper feed" in the signal display can be selected from this pick-list.

Restart scrolling after … s of inactivity
Enter a value in seconds between 10 and 6000. In case the pause button was pressed in a monitor to stop scrolling of the charts scrolling will restart after the given time if no operation (incl. mouse movement) has been made since pressing the pause button.

Update interval
Enter a value given in milliseconds between 50 and 10000 for the refresh rate of the trend graph display. A fair fluent movement can be achieved with 100 ms and below.

Anchor markers on X-axis
If you enable this option then the markers stay at the x-axis position where they were placed with the mouse even if you then zoom into the curve or scroll along the x-axis. The markers can therefore disappear from sight. If you do not select this option, the markers always remain visible in the curves window, even if you zoom or scroll. However, they do leave their configured x-axis position.

The markers are only visible in pause mode.

Align digital signals with legend
Normally, the digital signals are displayed on the bottom of a signal chart. By enabling this option, you can assign the digital signals with the top of the signal chart.

Allow text signal overlap
When displaying text channels in an *ibaQPanel* trend curve, it can happen that, depending on the feed rate and change frequency of text channels, the texts are written one above the other so that they are no longer readable or even overwrite the curves. This cannot always be prevented in the online mode of the display. This option is available for the pause mode.

If you enable this option then overlapping text channel displays also appear overlapped in pause mode. If you disable this option then the texts appear cleanly separated as long as there is enough space.

Center time axis on last known value
If this option is enabled and you switch to a different layout during the measuring, the position of the time-axis is saved with the last value captured. If you then later retrieve or load the layout then the trend display in pause mode stays exactly the same as when you left the layout.

If this option is not enabled, the trend display continues to run after a layout change and does not go into pause mode.

Show color axes
By enabling/disabling this option you can show/hide the color axes in a 2D-vector representation.

Colors

This dialog can be used to adjust the colors used for the user interface of the program and for the curves. In order to change the colors, click on the colored buttons and select a color from the palette.

Background color:
Color of the area around the graphs, within the view.

Axis color:
Color of the scales, for the X axis also color of the labeling (values).

Gridlines color:
Color of the grid in the graphs.

Graph color:
Background color inside the graphs. Default is monochrome. You can even define a color gradient or shaded background. In order to configure a gradient, double-click on the little squares at the end of the thin color bar in the dialog and select the two colors. If required, double-click on the color bar to add more color tags and color them; they can also be moved. If you want to remove a color tag, simply click on it to select it (black arrow) and press the <DEL> button.

Signal colors:
These are the 16 pen colors for the curves. These colors represent the order (top-down) used by the automatic signal color map function. You will find the same colors in a graph's "Properties" dialog, "Signal" tab, for coloring the signals.

Fonts
For labeling the axes and legend, you can select different fonts here. To change a font, click on the browser button in the corresponding input field and choose a font.

6.12.1 Legend

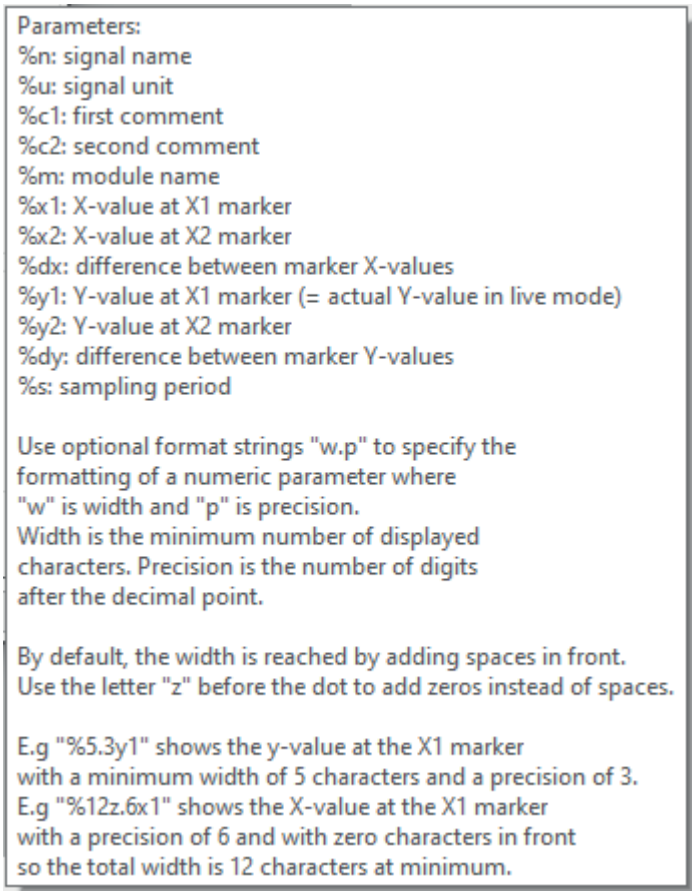
You can make various settings for presenting the signal legend in the *Legend* node.

Content
Select here what should be displayed in the signal legend.

- Signal name: Only the channel ID and signal name are displayed
- Signal name and value: As above, including the value and unit of measurement in parentheses. The refresh rate of the value display is lower than that of the graphs (approx. 1 s).
  - ... with automatic formatting: The value is automatically formatted.
  - ... with custom formatting: The value is formatted according to the settings of *Precision* (number of decimals), *Width* (minimum number of all digits incl. decimal point) and the option *Show leading zeros*. If you disable the option *Show leading zeros* then empty integer digits will be filled with blanks.
- Custom: User-defined compilation of information according to parameter string.

To create a user-defined legend, select the "Custom" option and click in the input field.

A tool tip appears that shows you all the options for designing the contents of the legend.



One by one, enter the desired parameters in the input field.

Then, click on <Apply> or <OK>.

Tip

It is best to make these settings in the Properties of an active trend graph and then to click on <Apply> to check the appearance of the legend.

- Text signal: Choose this setting, if you want to use the value of a text signal in the legend. Click on the entry field beside and select the desired text signal from the signal tree.

Style
Choose between 3 styles of legend display:

- Transparent: Only the legend text is displayed.
- Opaque: The legend is provided with a background that also covers the curves.
- Invisible: The legend is completely invisible.

Use this color for all digital signals in the legend
If you enable this option, then the legend texts of all digital signals will displayed in the color, set by the color field beside.

The digital signals in the trend graph will keep their original color.

Tool tip
In addition to the legend content, you can also configure a tool tip that appears whenever you are hovering the mouse on the signal legend.

The two signal comments are included in the tool tip by default. Alternatively, you can create a custom tool tip for which you can use the same parameters as for the legend contents (see above). This allows you to keep the signal legend short and still obtain further information about the signal if necessary.

6.12.2 Marker grid

The marker grid is a table, which is shown underneath a trend graph, displaying values of the signals in the graph referring to the positions of the two markers X1 and X2.

On the node *Marker grid* in the preferences or properties of the trend graph you determine the look of the table and when it is to be displayed.

Header
Select the desired foreground color here. This will be the color of the characters in the column headers of the table.

General

Visibility

Chose from 4 options for displaying the marker grid:

- Never visible
- Visible when paused
- Always visible
- Manual

If you choose "Manual", an additional button will appear in the toolbar, which you can use to show or hide the table at any time.

Background color, Gridlines color, Font

Here, you can set up the background color of the table cells and the color of the grid lines as well as font type and size.

Focused row

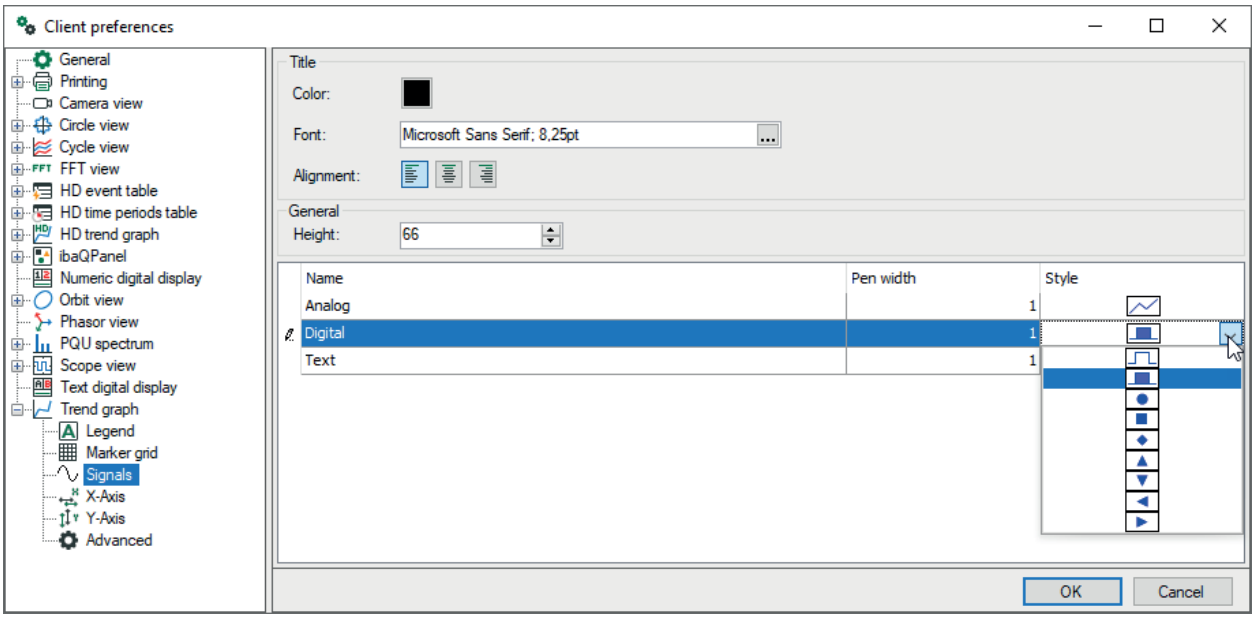
Background color

By setting the background color you determine how a table row will be highlighted when selected with the mouse.

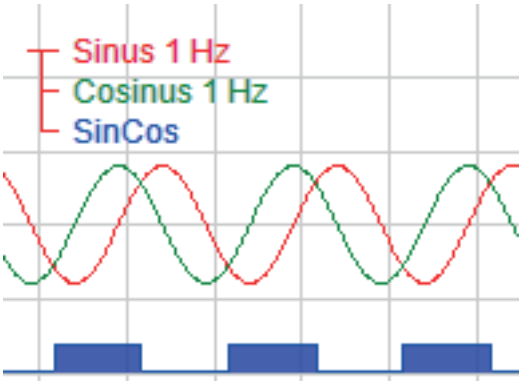
Foreground color

This setting determines the font color in case a row is selected. Typically, this is the signal color (default). However, you may choose a different color, in order to generate more contrast in conjunction with another background color, for instance.

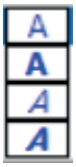
6.12.3 Signals



In this dialog you can generally define how the curves should be initially displayed. The default setting for curves of analog values is a simple line (line thickness 1), and for digital signals a filled area. Text signals will be displayed in standard font.

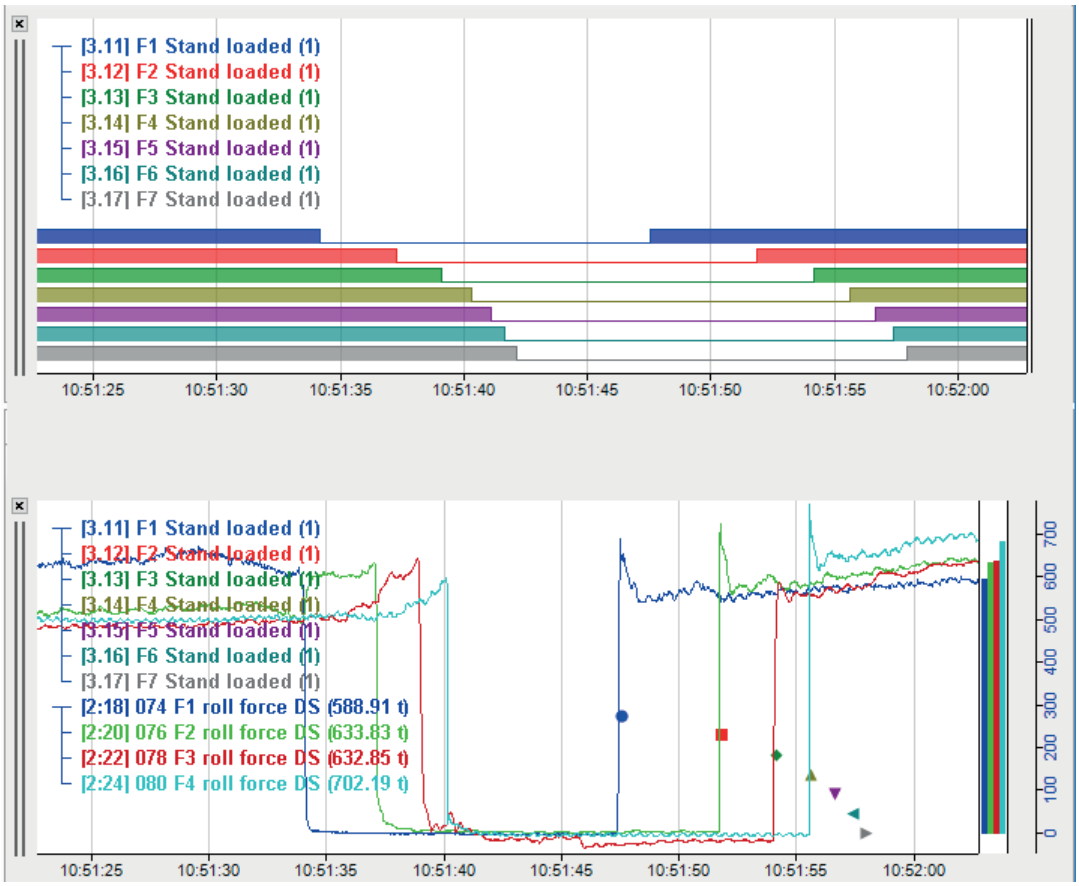


You can also select other styles for analog, digital and text signals. Click in the "Style" column. A selection list opens there of the different styles.

Symbol	Description
Analog signals	
	Line; Curve filled up to zero line; Curve filled to bottom; Curve filled to top
Digital signals	
	Line; Curve filled up to zero line Geometric symbols are displayed at the rising edge of the signal
	Normal; bold; italic; bold & italic

If you select a symbolic style for a digital signal then this symbol is only displayed on the x-axis where there is a rising edge of the signal. The remaining signal curve (TRUE or FALSE) is not displayed.
Example: strip entry in a 7-stand rolling mill

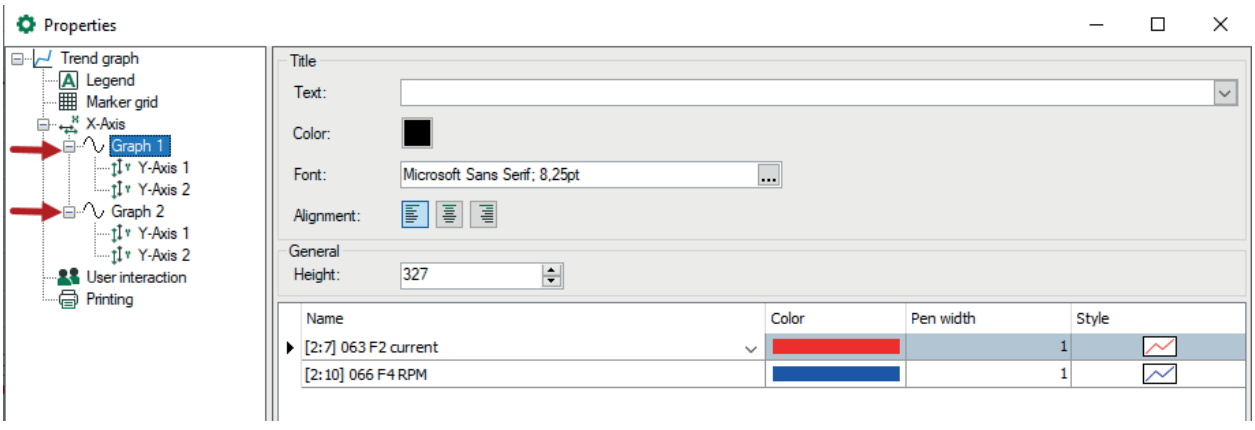
The figure below shows in the upper part a trend of digital signals in standard style, the lower part shows the digital signals in symbol style together with some analog signals.



The symbolic style can be useful if the trend display would be overloaded in the standard style, especially when digital and analog signals are presented mixed.

Transparent colors are used for filled curves to avoid one hiding behind the other. The line width can be adjusted in pixels.

In the "Properties" dialog of a trend graph view, you can set up the style for each curve individually. There you will find one or more "Graph x" branches. X is the numbering of the different graph charts within a trend graph view. Under the "X-axis" branch, you will find signals with their properties, grouped by the charts in the view. The following figure shows the properties of a trend graph (view) with two graph charts (graph 1 and graph 2) with two Y-axes each. Graph 1 contains two signals.



In the "Color" column, you may change the color of a signal by selection from a pick-list.

In the preferences you will find these settings under *Trend graph - Signals*.

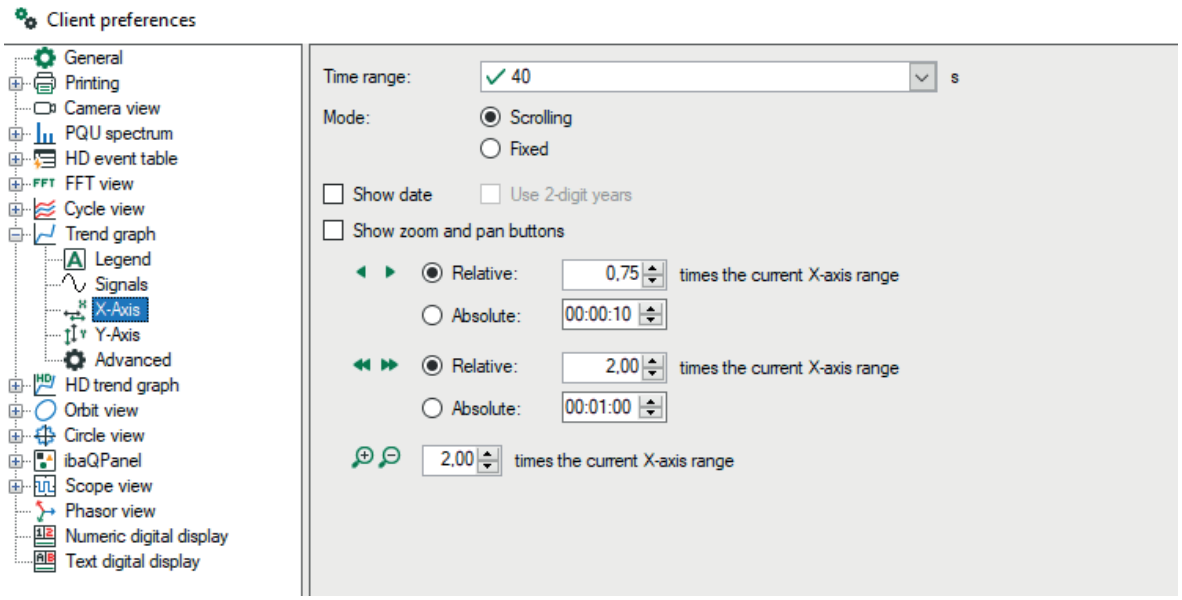
Set up the graph height

In the *General* area you can determine the minimum height of a graph for its display in a trend graph view by setting the *Height* parameter. The value is given in pixel. A default value of 66 is given in the preferences.

However, a graph will be displayed with maximum possible height when a signal is dragged into an empty trend graph view. Adding more trend graphs will divide the height in equal shares automatically as long as you did not set the height of one or more graphs manually. If there are more trend graphs open in a trend view than space available with regard to the minimum height, a scroll bar appears.

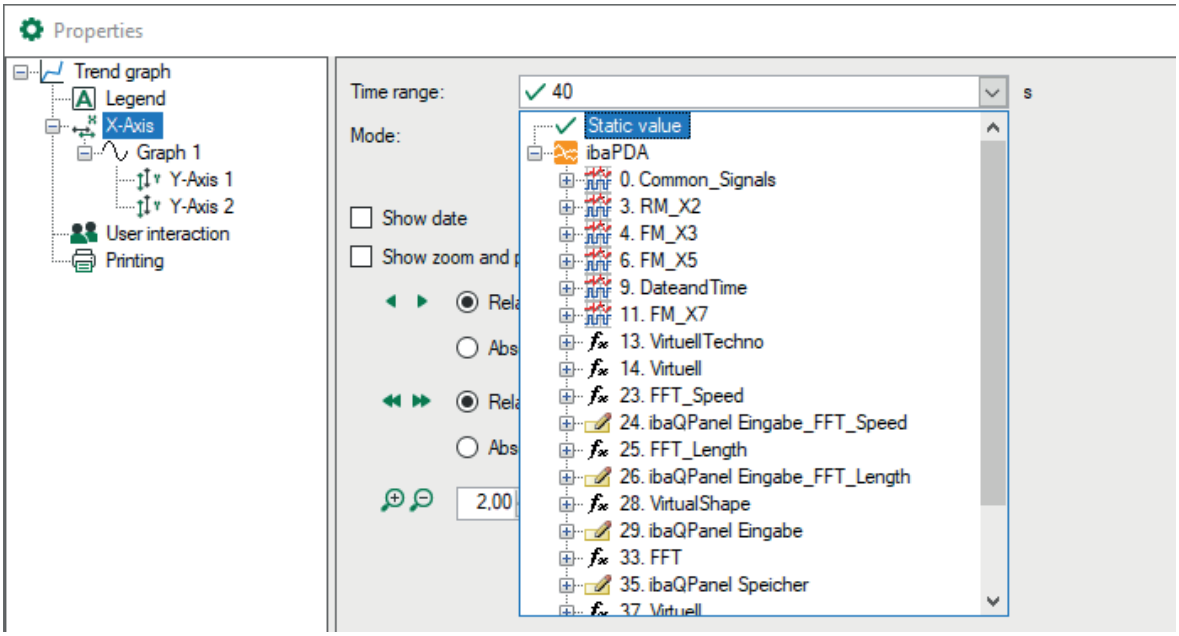
You can adapt the height for each graph in the properties later.

6.12.4 X-axis



Time range

You can configure the time range represented by the X-axis. If you reduce the value, the signal will move faster on the display and changes to an expanded form. If you increase the value, you will see more measuring points in the graph and the signal will move slower. In the preferences, you can only set a static value. In the properties of a specific trend graph view, you can select an analog signal controlling the length of the X-axis.



Mode
Scrolling: Usually, the time axis moves along with the incoming samples (default).

Locked: If you enable this option, the signal will be written into the static graph. Once the signal curve has reached the end of the visible time range, the display shifts to the next (empty) time unit and continues to write samples.

Show date
By enabling this option, the date (MM/DD/YYYY) is displayed at the ticks of the X-axis. Optionally, you can reduce the year by 2 digits (MM/DD/YY).

Show zoom and pan buttons
If you enable this option, the zoom and pan buttons will be displayed along the X-axis.

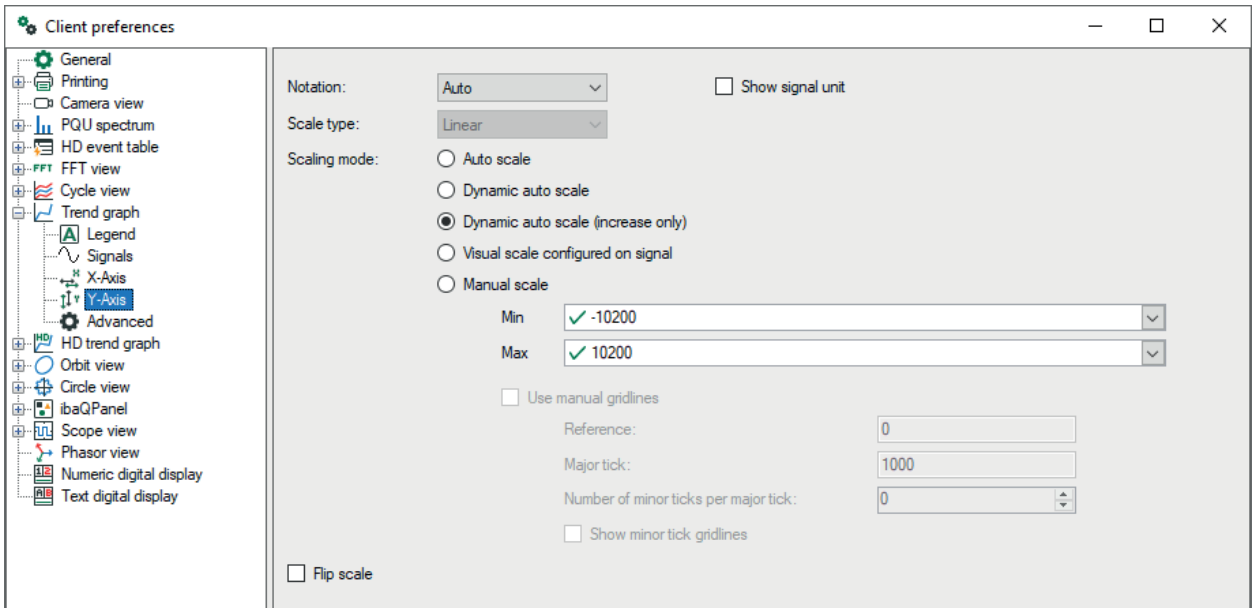
Default settings are:

- for small panning steps: 0.75, relative
- for large panning steps: 2.00, relative
- for zooming 2.00

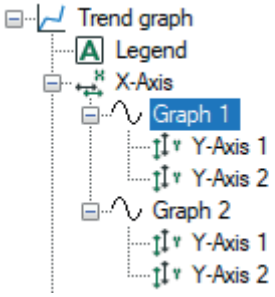
Alternatively, the panning steps can be set as absolute value in the HH:MM:SS time unit.

6.12.5 Y-Axis

In the preferences, you will find the branch for setting up the Y axes.



If you open the "Properties" dialog of a trend graph view, you will find one branch for each graph in the view as well as subordinated branches for each Y axis of the corresponding graph (see picture below).



Thus, you can set up all Y axes individually.

Scientific notation
Choose from the pick-list in this field:

- Auto: Depending on the size of the scale values (number of places before or after the decimal point), the scales are labeled in scientific notation (power of 10) or not.
- Standard: Scale values always with digits before and behind the decimal point
- Scientific: Scale values in powers of 10

Optionally you can enable the indication of the signal unit.

Scaling mode

- Autoscale: this is the default setting; if one or more signals are displayed, the Y-axis of the graph is scaled in accordance with the smallest and largest of all values occurring.
- Dynamic auto scale: if you enable this option, the scaling is always adjusted to the highest and lowest signal amplitudes in the graph (in both directions).
- Dynamic auto scale (only for enlarging): When you enable this option, the scaling is continuously adjusted with the highest signal amplitudes. If the amplitudes go beyond the trend view again, the scaling still remains unchanged.
- Manual scale: if this option is selected, the Min and Max scale values can be manually entered or selected from the field next to it. Beside a static value, you can also use any known measured or virtual analog signal configured in the I/O manager. In order to select a signal for minimum or maximum scale, open the pick-list of the corresponding field and make the selection in the signal tree.

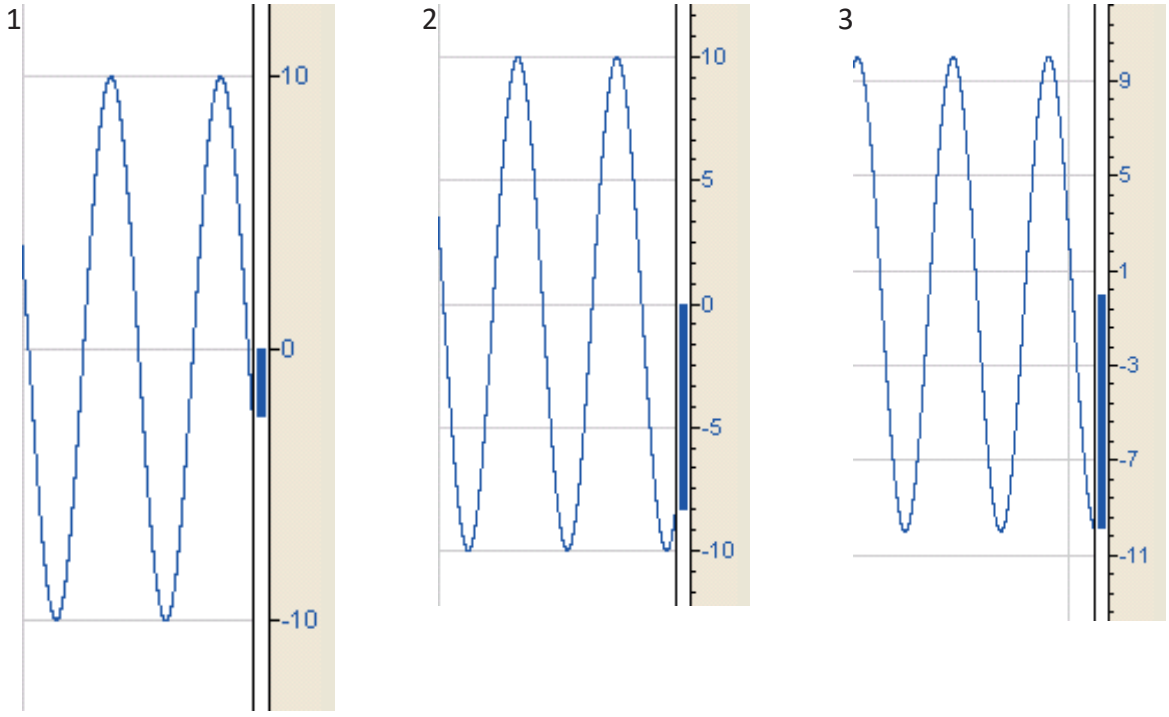
Using manual grid lines

This option is only available if the manual scaling is enabled.

With the parameters to be entered, you can alter the Y axis and grid lines.

- Reference
  Enter any value to be considered as reference for the basic grid line. Usually, the basic grid line is at value 0 (zero). Depending on your needs, the basic grid line can also have, for example, the value 10 or -25.
- Major tick
  Enter a value for the major ticks' distances on the Y-axis. Every major tick is linked to another grid line. The major ticks determine the horizontal grid lines in the graph. Depending on the total scaling, a major tick can be set every 10 s, 100 s or every 0.1 s.
- Number of minor ticks per major tick
  Enter a value complying with the number of minor ticks between two major ticks. This determines the granularity of the Y-axis. Optionally, you may enable grid lines for minor ticks.

Example



	Reference	Major ticks	Minor ticks per major tick
1	0	10	0
2	0	5	5
3	1	4	4

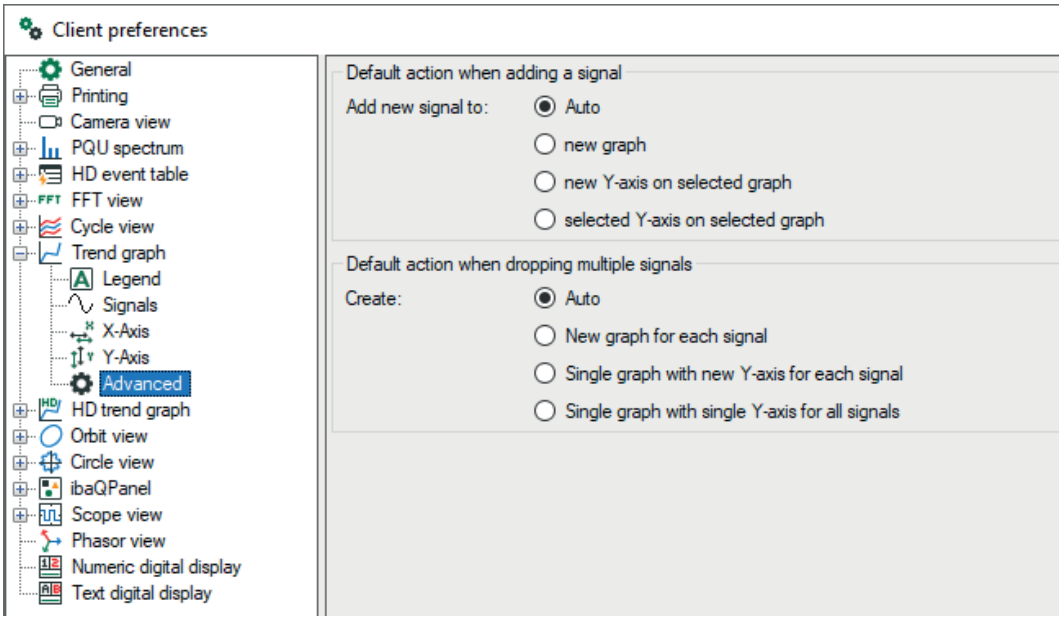
Trend graph settings for reference, major ticks and minor ticks (example)

Flip scale
By enabling the option *Flip scale* you can swap the scale top-down.

Color (only available in the properties)
You have the possibility to assign a certain color to a Y-axis. When set to automatic mode (default) the Y-axis always has the same color like the related signal. If multiple signals are attached to the same y-axis, its color is determined by the topmost signal in the legend.

6.12.6 Advanced

The advanced options allow you to change the behavior when adding signals to a trend graph by either double-clicking in the signal tree or by drag and drop. The "Advanced" branch is only available in the preferences.



The default behavior is "Auto". If you add a signal with no key combination pressed, a new graph and a new Y axis will be created for the signal. If you hold the <Ctrl> key while adding a signal, a new Y axis is created in the currently selected graph chart and the signal is added to that new Y axis. If you hold the <Shift> key while adding a signal, the signal is added to the currently selected Y axis.

If you drop multiple signals, this behavior is repeated for all signals.

By selecting one of the advanced options, you can override this default behavior and select one specific behavior that will be used always no matter which key combination is pressed.

7 FFT view

The FFT view is a spectrum analyzer included as a standard view in *ibaPDA*. It can be used to display the frequency spectrum of one or more signals on multiple value axes.

The FFT view can be used for normal analog signals as well as the modules of *ibaInSpectra* add-ons. Some display features are only available in conjunction with an InSpectra module, as indicated in the corresponding section.

If you use *ibaInSpectra*, you can drag a complete InSpectra module from the signal tree to an FFT view. The settings and mathematical parameters for the FFT view will then automatically taken from the InSpectra module settings.

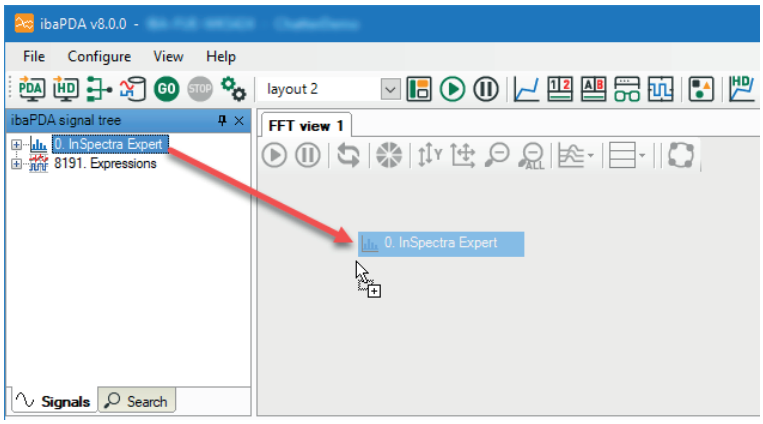
Other documentation

Detailed information on *ibaInSpectra* is contained in the *ibaInSpectra* product manual.

7.1 Opening an FFT view in ibaPDA

Use the button to add a new FFT view:

You can move individual or several highlighted signals from the signal tree to the main view of the FFT view using drag & drop. In the case of an InSpectra module, you can drag the entire module into the FFT view. In doing so, relevant parameters for the FFT view are copied from the module settings.



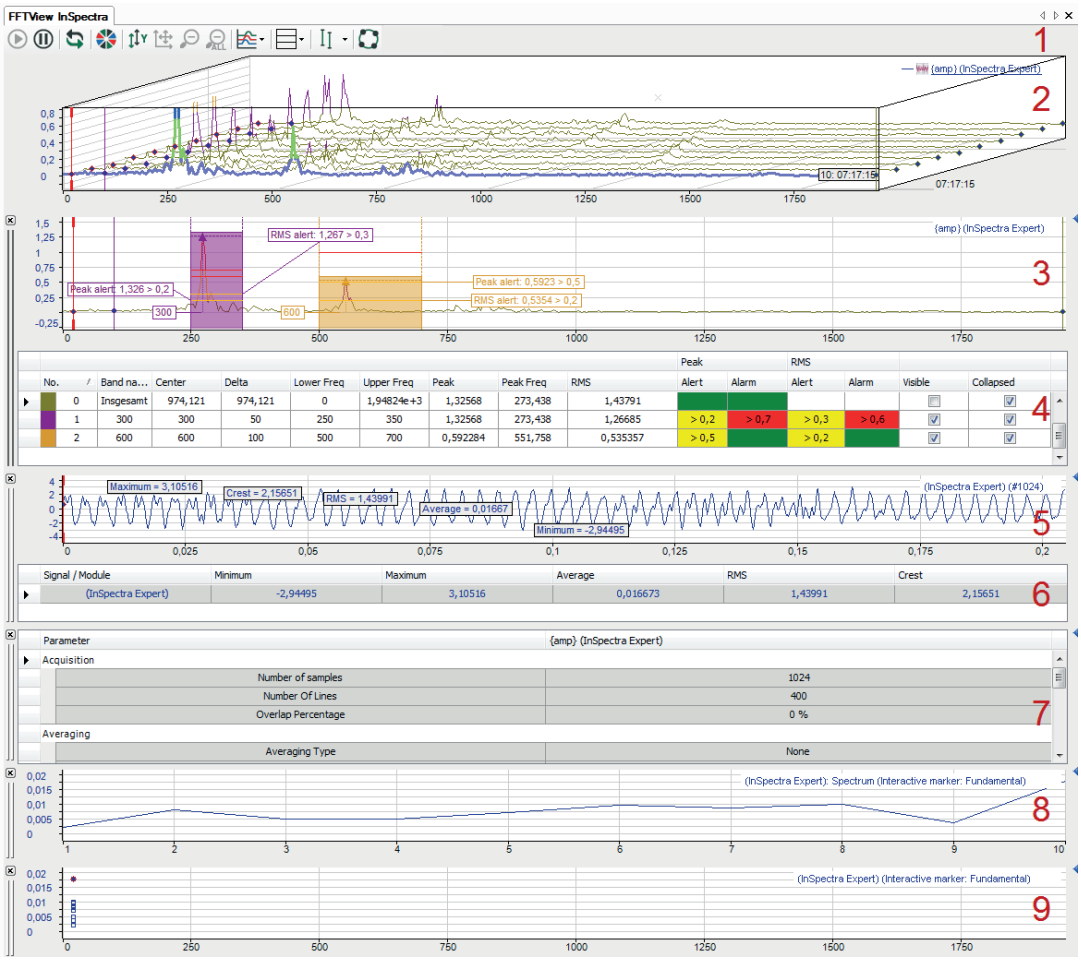
The following hotkeys are available for dragging new signals into a FFT view:

- <Shift>: When you press the <Shift> button while dragging several signals into the FFT view, all signals are placed on a joint Y-axis.
- <Ctrl>: When you press the <Ctrl> key when dragging one or more signals into the FFT view, the existing signals are replaced with new signals. If there are more signals in the view than new signals, the first signals will be replaced. If there are more new signals, the additional signals will be appended.

The description of the FFT view can be found in chapter ↗ *FFT view overview, page 51*

7.2 FFT view overview

The FFT view offers a number of special graphs and tables, which can be individually displayed or hidden as needed.



Legend

1 Toolbar

2 Main window, signal spectrum of the input signal

3 Spectrum graph (bands, frequency domain) [1]

4 Spectrum table (data, frequency domain) [2]

5 Time graph (graph, time domain)

6 Time table (table, time domain)

7 Spectrum parameter table

8 Slice graph[3]

9 Marker spectrum graph[3]

[1] With ibaInSpectra, additional static values, warning and alarm limits

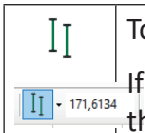
[2] Can only be used with ibaInSpectra

[3] Slice graphs and marker spectrum graphs can exist several times

The main window is always displayed at the top. The additional windows for graphs and data of spectrum and time domain are grouped in pairs. In analogy to normal trend views, their position can be changed at the header bar by using the mouse. You can display or hide the individual graphs and tables within the FFT view by means of the buttons as shown in the above figure.

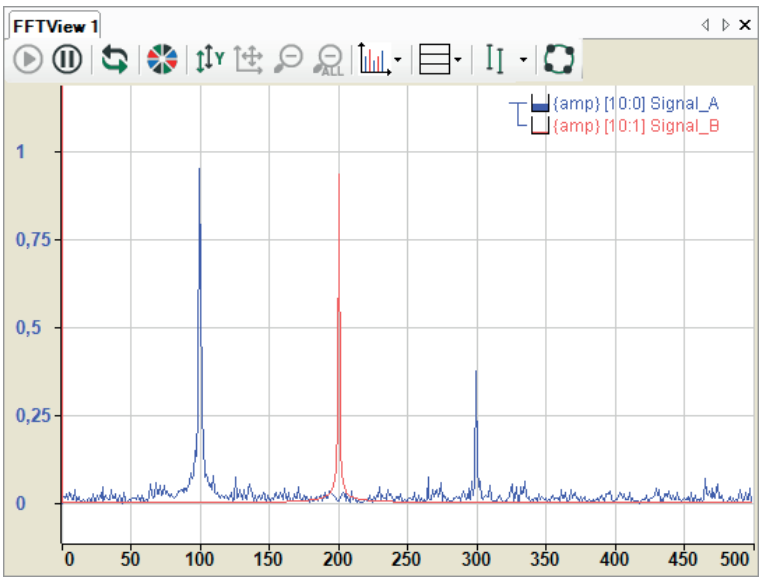
Toolbar

	Start FFT view / Pause FFT view (only *ibaPDA*) Stop or continue the FFT display update
	Reset all displayed data (only *ibaPDA*) The display is cleared only once and all values are set to zero until the next FFT calculation is completed.
	Determine planecount automatically (only *ibaAnalyzer*)
	Auto color signals
	Auto scale value axis
	Restore manual scale [1]
	Zoomout one level / Zoomout all [1]
	Toggle display type in the FFT main window (single spectrum / waterfall / contour)
	Open the sub menu for showing/hiding the windows Main window with/without waterfall (graph, frequency domain) Sectrum graph (graph, frequency domain) Spectrum table (table, frequency domain) Time graph (graph, time domain) Time table (table, time domain) Display spectrum parameter table Add slice ... Add marker spectrum ...

	Toggle interactive marker visibility, center or configure the marker If the interactive marker is visible, a value field will be displayed beside the button in the tool bar. The field shows the actual position of the marker on the X-axis (Hz). But you can also enter a value into the field to position the marker accordingly. Add InSpectra speed marker (the speed marker corresponds to the speed signal configured in the module) No function for configured markers
	Switch to the order spectrum (if speed signal and parameters of the order calculation are configured)
	[1)] Individually affects the main window, spectrum or time graph, depending on the focus

7.3 Main window

In the main window, the result of the FFT of the signal to be examined is shown in the frequency domain. The standard view for the main window is the individual spectrum.



You can enable an interactive marker that you want to use to read frequency values and the associated amplitudes along the X-axis.

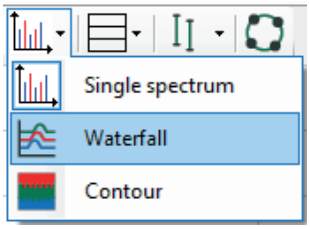
When switching to the waterfall or contour view, the individual results of the frequency analysis are displayed spatially offset. This provides an overview of the history of the frequency response curve.

Detailed information can be found in chapter ↗ *Waterfall, page 55*.

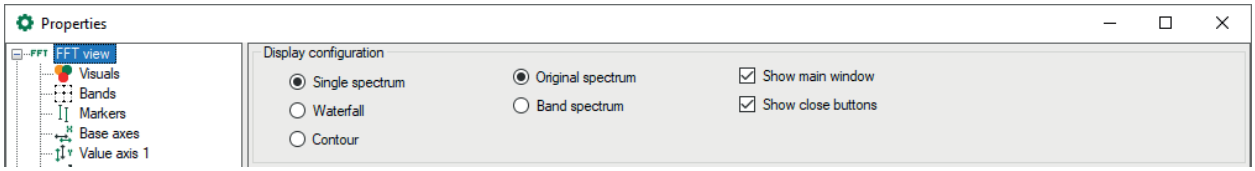
7.3.1 Waterfall

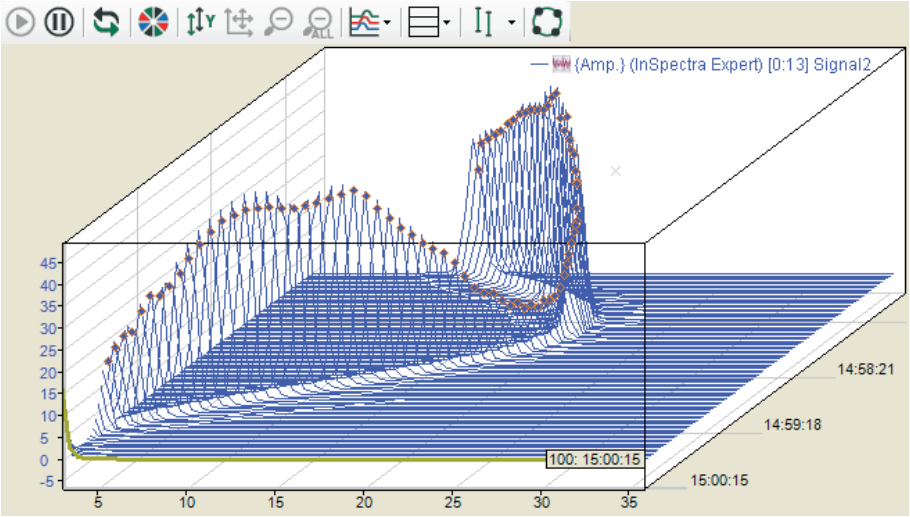
The main window of the FFT view can be converted to an isometric perspective. In this mode, the successive FFT events of a spectrum are displayed on a Z-axis, with the newest result closest to the axes origin, in order to create a waterfall effect. The display is restricted in *ibaPDA* to 262144 data points, in *ibaAnalyzer* configurable via memory usage per FFT. However, note that using a waterfall display requires more resources than a single spectrum.

You can switch to the waterfall perspective via the corresponding button in the toolbar of the FFT view.



Alternatively, you can switch perspectives in the properties dialog of the FFT view as well.





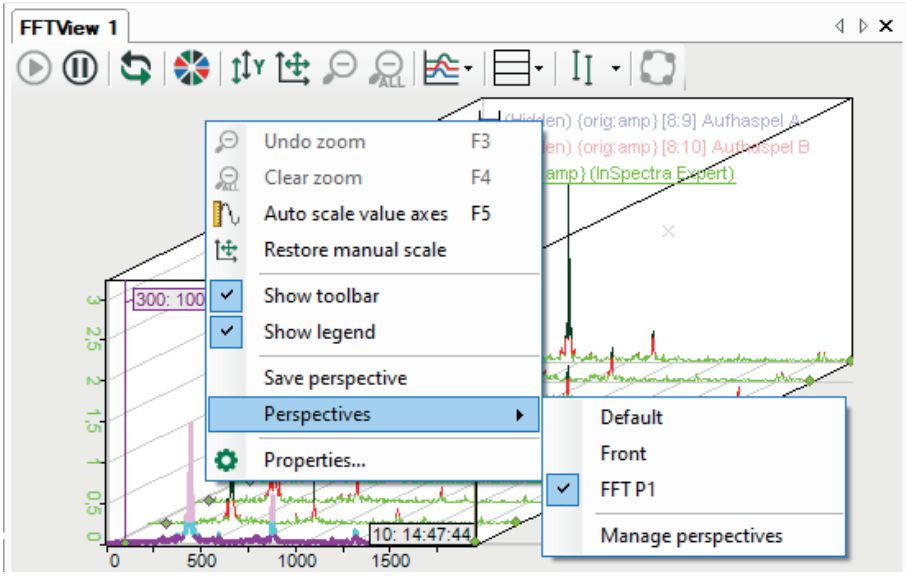
In the above figure, you can see the results of the last 100 calculations and it is clearly visible how the spectrum changed in the course of time.

By using the <Up> and <Down> cursor keys or by scrolling with the mouse wheel, you can move through the planes and have displayed the related spectra and parameters.

When moving the mouse with the <Ctrl> key pressed, you can change the angle and perspective of the view. If you press the <Shift> key at the same time, then the display pans to 0 degrees. The axis position settings are overwritten in this mode.

If you have set the desired perspective, you can save it. To do this, select *Save perspective* in the context menu and enter a name. Saved perspectives also appear in the context menu and can

be re-enabled at any time. When you select *Front* perspective, the display pans to 0 degrees. Selecting *Default* returns you to the default setting. See also chapter ↗ *Properties and preferences FFT view, page 86*



It is possible to apply the contour color scheme to the waterfall view. The amplitude range is split up in a configurable number of bands to which a color is assigned. This allows the amplitude level to be represented by colors. Up to 50 different colors bands are supported.

Scales are always displayed at the side of the chart not overlapping with the perspective flow direction. Several spectra can have different sample rates or bin values and thus the clock in which the FFT results are available may vary. That is why it is pre-set that every spectrum moves on the Z-axis at its own pace.

However, there is the option to synchronize the Z-planes across several spectra. With this option enabled, the FFT view will not allow a spectrum to advance over the Z planes until all spectra have generated a new FFT result. While the view is waiting for certain spectra to generate results, the other spectra keep showing their newest results on the front plane.

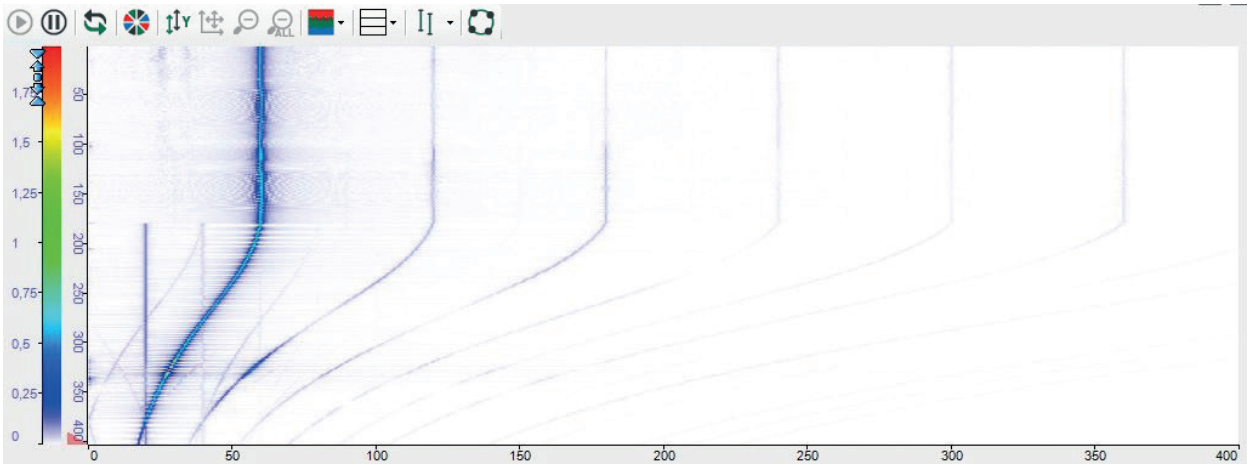
While the waterfall perspective is enabled, the label, marker and zoom rectangle functionality is limited to the foremost plane.

The sorting of the levels is configurable, i.e. you can specify which level is displayed at the front. You can choose between chronological sorting and sorting according to a signal value. The distance between the planes then depends on the time difference or the signal value difference between the planes.

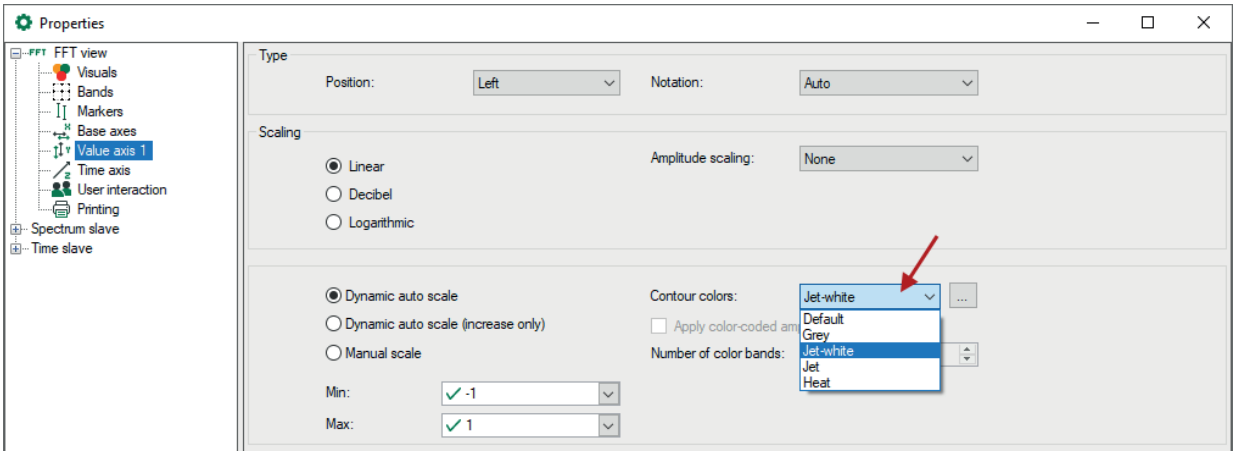
The appearance options of the waterfall display is determined in the properties window in the node *Z-axis*. See chapter ↗ *Z-axis, page 103*.

7.3.2 Contour view

The contour view corresponds to a 2D top view of the waterfall, where the amplitude height is represented by colors.



The color scheme can be configured in the properties of the value axis. Both pre-defined schemes can be selected here and customized color schemes can be created.

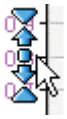


7.3.3 Zoom

The scale of an axis can be manipulated in three ways.

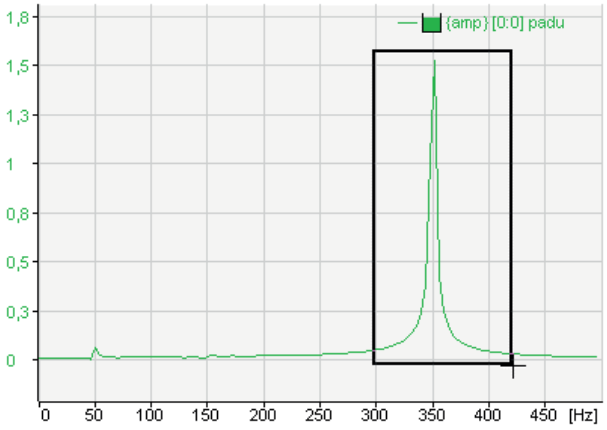
- Autoscale
  You can perform an autoscale via the context menu of the axis or by using the middle mouse button to click on the axis.
- Shift
  You can shift an axis by dragging it with the mouse.
- Zoom
  Using the mouse wheel, you can zoom in and out in the area of the cursor.

You can change the scale via the pop-up buttons on the axis too. These buttons appear when you move the mouse over the right side of a horizontal axis or over the top of a vertical axis.



The outermost symbols halve/double the scale range based on the average. The arrows have a similar function, but with a smaller zoom factor. The button in the middle autoscales the axis.

In addition, you can zoom into a certain area of the diagram using the zoom rectangle (click with mouse and drag, see fig. below). The zoom rectangle enables the zoom buttons in the view toolbar, which allow you to return to previous zoom levels.

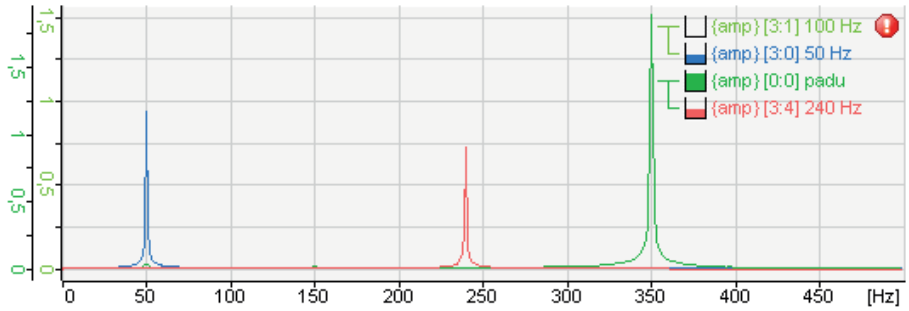


7.3.4 Legend

The legend indicates which signals are added to the view. The legend contains the following parts:

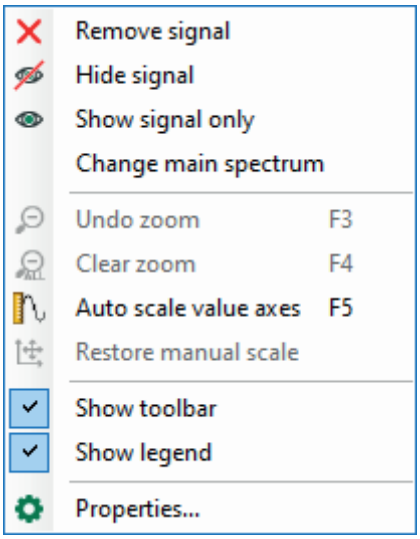
- Tree structure of the value axis. This shows which spectra are shown on which axis.
- Visual representation of the percentage buffer fill grade for each signal
- Spectrum type (band or original spectrum, amplitude spectrum, power spectrum, cepstrum or power cepstrum)
- Signal ID and signal name
- Calculation mode

If a signal is invalid, this is indicated by an exclamation mark at the end of the signal row. The example below shows a legend with 4 signals in 2 groups with the first signal (100 Hz) being invalid.



The legend has a drag & drop function. This way, a spectrum can be laid upon different value axes. While dragging the spectrum, an arrow appears in the value axis tree pointing to the tree that will contain the spectrum when it is dropped. If a spectrum is not dropped inside a legend row, the spectrum will be laid upon a new axis.

Right clicking in a legend row makes the context menu of the legend appear.



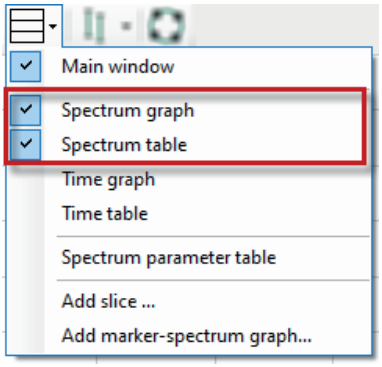
Clicking on “Remove signal” removes the corresponding signal. Clicking on “Hide signal” hides the signal and shows the signal name transparently. The signal is only temporarily hidden and can always be displayed again. By clicking on "Show signal only” in the context menu, only the selected spectrum remains in the display and all other spectra are hidden. Clicking on “Change main spectrum” makes the selected spectrum the main spectrum.

In the context menu under “Properties,” you can display the selected settings for the spectra.

In the properties of the FFT view (main window), you can also configure and enable a separate legend that contains additional information, such as name, comments and sampling time of the input signal, marker values or any literal text.

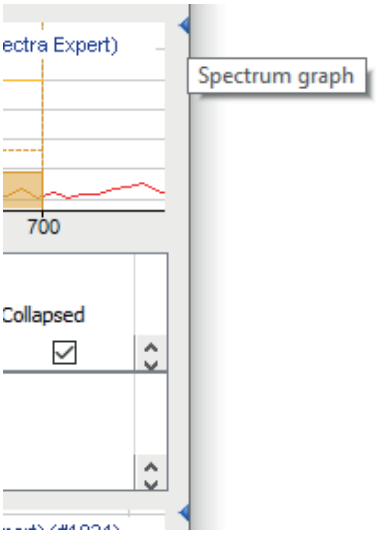
7.4 Spectrum graph and spectrum table

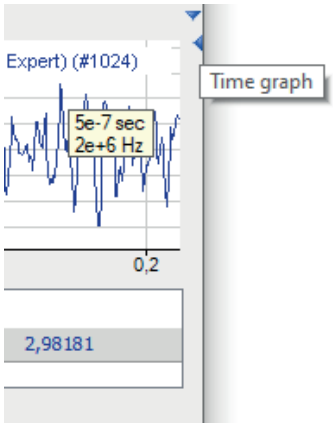
In addition to the main window, you can open a graphical and/or tabular display of the data of the frequency spectrum. Click on the button for the window menu in the toolbar of the FFT view for this purpose.



Graphical display and data table form one group, as the table always provides the data matching the spectrum in the graph. However, the graph and table can be individually displayed or hidden.

In addition, the graph and data table can be minimized (see following figure right side) or displayed together (see following figure left side). To do this, simply click on the small triangle on the right margin of the display:

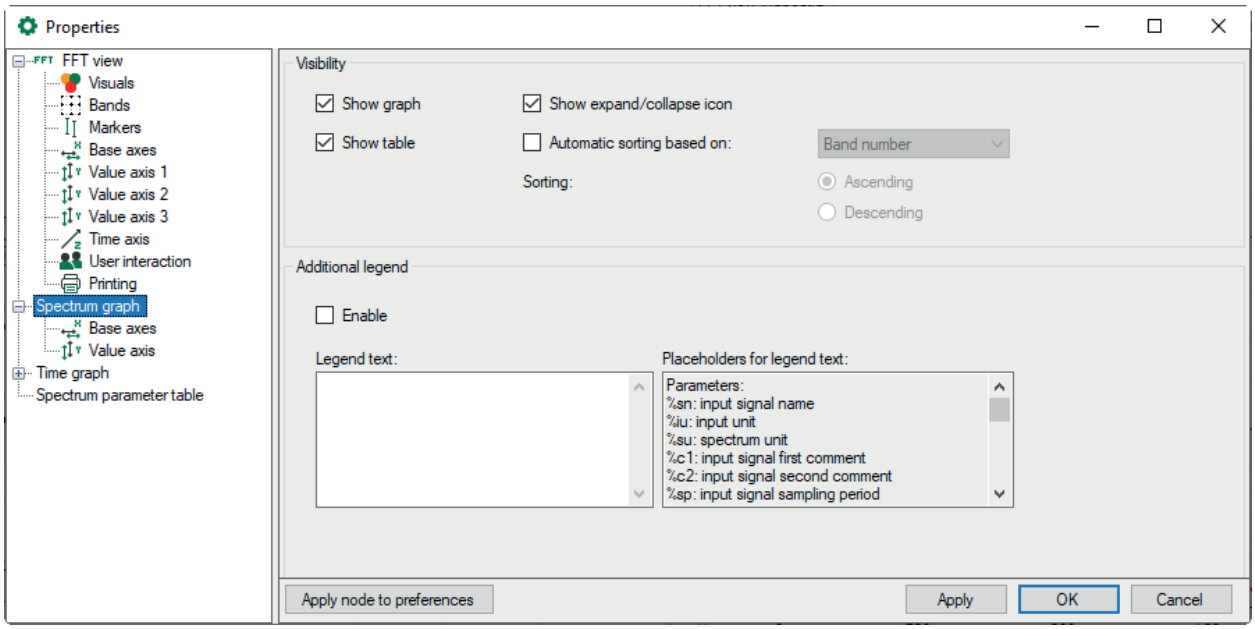




Note

Without *InSpectra* modules, the spectrum graph shows the same information as the individual spectrum in the main window and the table does not contain any data.

You define general display properties in the properties dialog of the FFT view in node *Spectrum graph*.



Visibility

You can define here whether the graph and the data table for the spectrum graph are shown by default. Even if a view is disabled here, it can be re-enabled later in the FFT view toolbar.

You can also specify whether the small triangles for expanding and collapsing the graph and data table should be displayed.

The data table can be sorted automatically. Define the parameter (column) here according to which and in which sequence the table is sorted.

Additional legend

When this option is enabled, another legend is displayed in the spectrum graph window in addition to the normal signal legend. You can define the content of this legend yourself. For example, you can enter a detailed multi-line text, in which placeholders for dynamic information can also be used. The following placeholders are available:

- %sn: Input signal name
- %iu: Input unit
- %su: Spectrum unit
- %c1: Input signal first comment
- %c2: Input signal second comment
- %sp: Input signal sampling time
- %x: X-value at interactive marker
- %y: Y-value at interactive marker
- %xmouse: X-value at mouse cursor
- %ymouse: Y-value at mouse cursor
- %tmouse: Z-value at mouse cursor
- %xmv: X-value of the nearby marker position

- %ymv: Y-value of the nearby marker position
- %tmv: Time value of the nearby marker position
- %nmv: Name of the nearby marker position
- %imn: InSpectra Expert module name
- %n: Band name
- %nb: Band number
- %r: RMS value
- %p: Peak value
- %pf: Peak frequency
- %c: Center frequency
- %d: Delta frequency
- %l: Lower frequency
- %ats: Alert status text
- %ams: Alarm status text
- %events: Alert & alarm status text
- %atrmsl: Limit value for RMS alert
- %atpeakl: Limit value for peak alert
- %amrmsl: Limit value for RMS alarm
- %ampeakl: Limit value for peak alarm
- %atrmspc: Quota of limit value for RMS alert
- %atpeakpc: Quota of limit value for peak alert
- %amrmspc: Quota of limit value for RMS alarm
- %ampeakpc: Quota of limit value for peak alarm
- %v: the monitored value
- %mtype: monitored type: either "rms" or "peak"
- %etype: Event type: either “alert” or “alarm”
- %s: Event status text
- %lim: Limit value of the event
- %pc: Quota of event

By default, all signal-related placeholders are determined based on the first spectrum. To identify another spectrum, use a colon followed by the word “spectrum” and the index of the spectrum, e.g. "%sn:spectrum1", in order to refer to the first spectrum.

Band-related placeholders can be referred to by the band name (e.g. "%p:bandname") or by the band number (e.g. "%p:band0").

Event-related placeholders can be referred to as follows:

- %lim:band0:atpeak (Peak alert)
- %lim:band0:ampeak (Peak alarm)
- %lim:band0:atrms (RMS alert)
- %lim:band0:amrms (RMS alarm)

You may use as well the event name instead of “atpeak”, “ampeak” etc.

Use the optional formatting string “w.p” to specify the format of the numeric parameters, where “w” is the width and “p” is the precision. The width is the minimum number of the characters shown. Precision is the number of decimal places.

Example: "%5.3y1" indicates the Y-value for marker X1 with a width of 5 characters and a precision of 3.

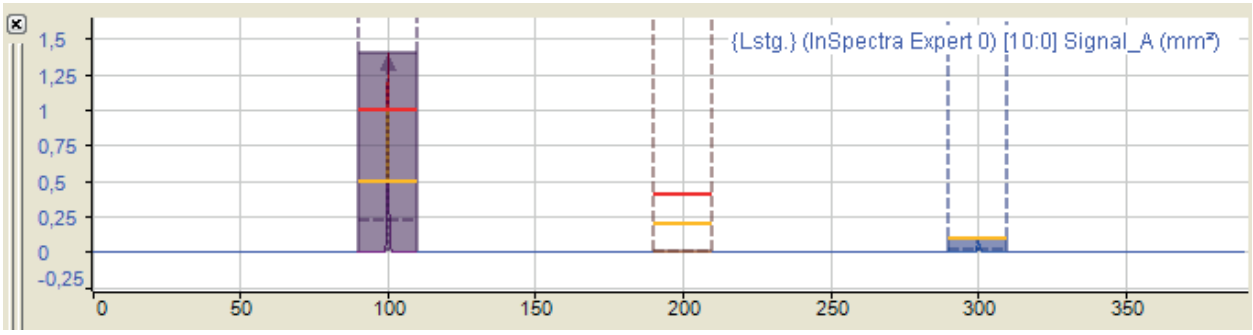
You can combine most of these techniques:

E.g. "%10.5p:spectrum1:someband" E.g. "%.6pc:spectrum2:otherband:eventname"

7.4.1 Spectrum graph

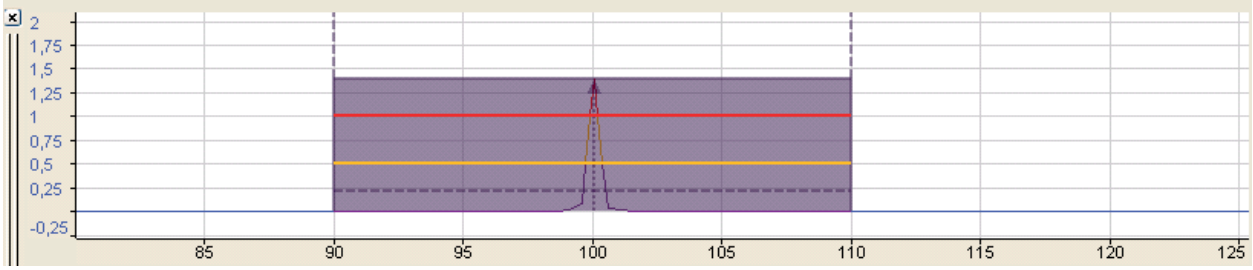
The graphical display of the frequency spectrum always shows the last result of the FFT in two-dimensional appearance or the spectrum selected in the waterfall or contour plot. (The selected spectrum in the waterfall view is shown with a different color, marked in the contour plot with a triangle):

- Spectrum
- Frequency bands
- Value bands
- InSpectra bands
- Characteristic values of the InSpectra bands
- Limits of the InSpectra bands



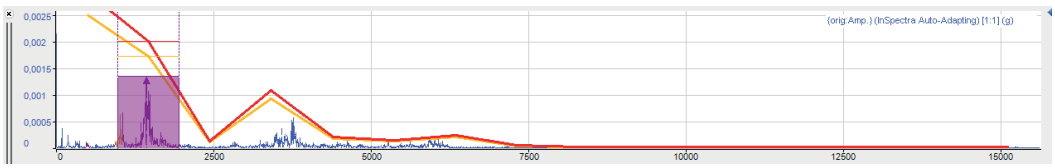
In the figure above the display shows three different frequency bands and at least a part of the spectral graphs from the main window. You can add additional graphs by dragging and dropping them from the main window or from the signal tree via drag & drop. The displays are linked so that all graphs in the small spectral display can also be seen in the main window.

When zooming in, more details can be seen.

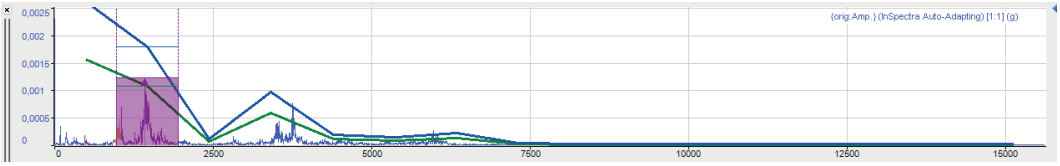


The most important parameters of the frequency bands and of the spectrum are shown with dotted and colored lines. You are shown the respective values when you position the cursor on the lines (hovering).

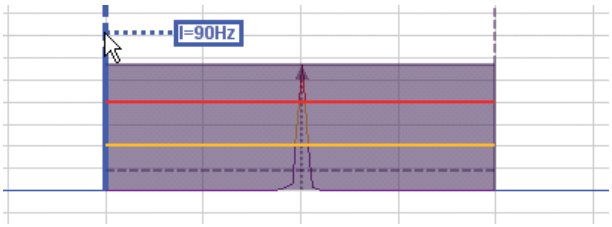
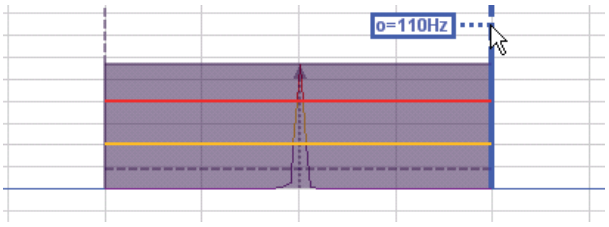
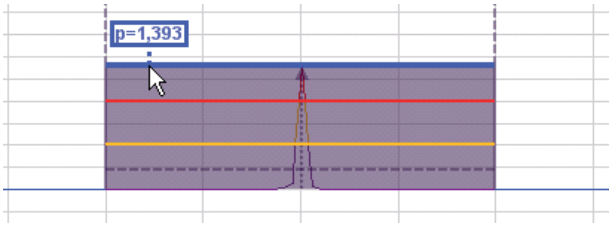
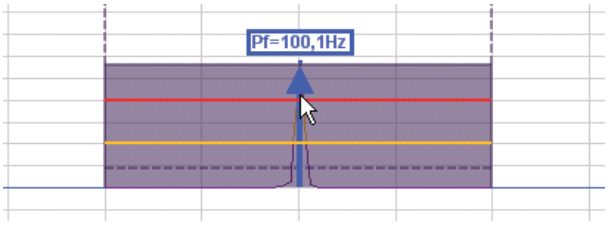
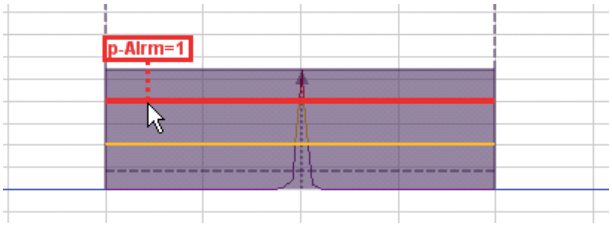
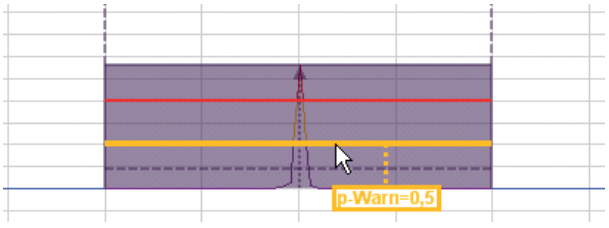
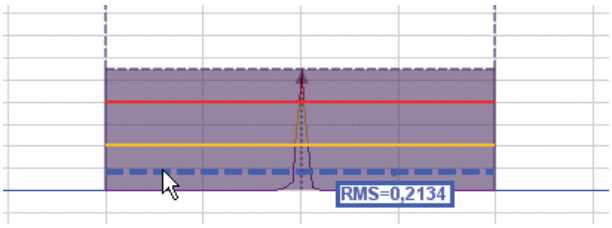
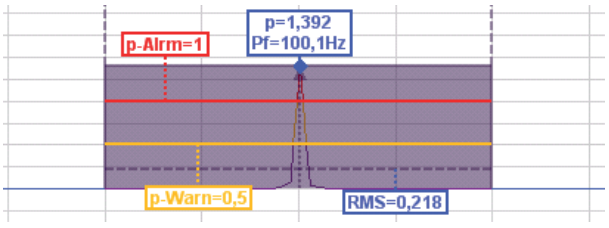
Limit curves can also be displayed for Auto Adapting modules. If “Warning and alarm” is used as the event mode in the profile, the warning limit is displayed in yellow and the alarm limit in red.



If “Lower and upper” is used as the event mode, the lower limit is displayed in blue and the upper limit in green.



Information in the spectrum graph:

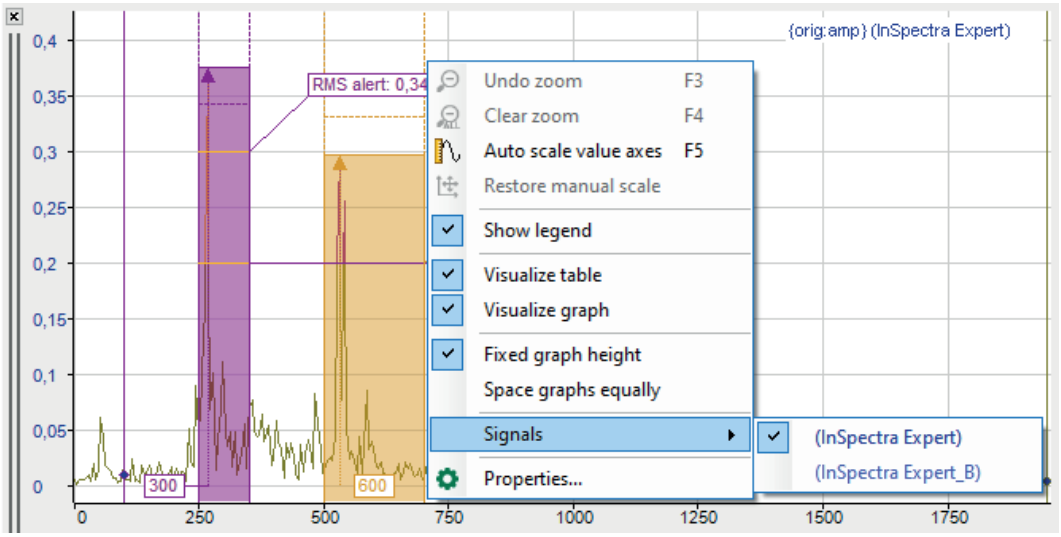
 Lower cutoff frequency	 Upper cutoff frequency
 Peak (peak amplitude)	 Peak frequency
 Alarm limit	 Warning limit
 RMS	 Display of all descriptors by selection in the spectrum properties

Note

If limit values and reference values are not displayed, you can activate the *Show events/references on dynamic auto scale* option in the *Value axis* node. The scaling is then automatically adjusted so that events and reference values are displayed.

The configuration of the bands is described in chapter ↗ *Bands, page 90*.

You can decide in the spectrum properties (by using the context menu of the display) which markings and parameters are to be displayed (permanently) and whether the graph should change the color when exceeding the alarm limits.



If there are several spectra in the display, individual display properties can be assigned to every spectrum.

If the spectrum graph has the focus (after a mouse click on the header bar), the tool buttons for zooming out and restoring the manual scale relate to this graph and not to the main window. The same applies to the assigned function keys <F3>, <F4> and <F5>.

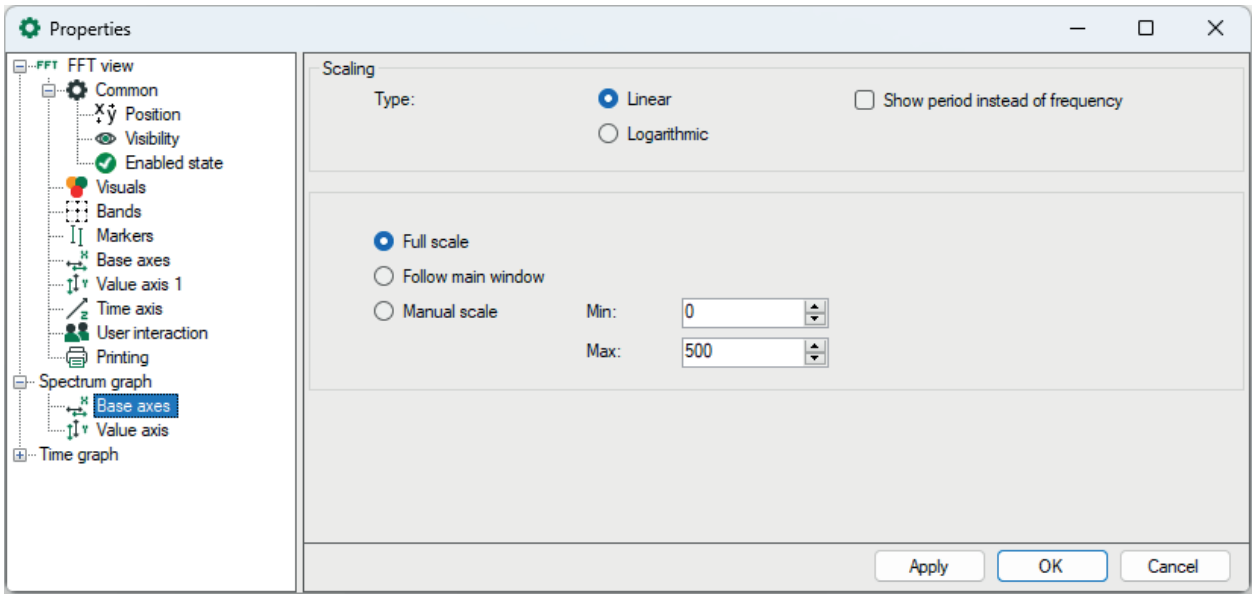
Base axis

The display has a base axis corresponding to that of the main window. You can still modify the settings of the base axis in the display properties, for example, to select a logarithmic division instead of a linear division or to provide manual scaling. In addition, you can display the period instead of the frequency.

If *Cepstrum* is selected as the spectrum method in the InSpectra profile, the result of a *Millstrum* analysis can be displayed.

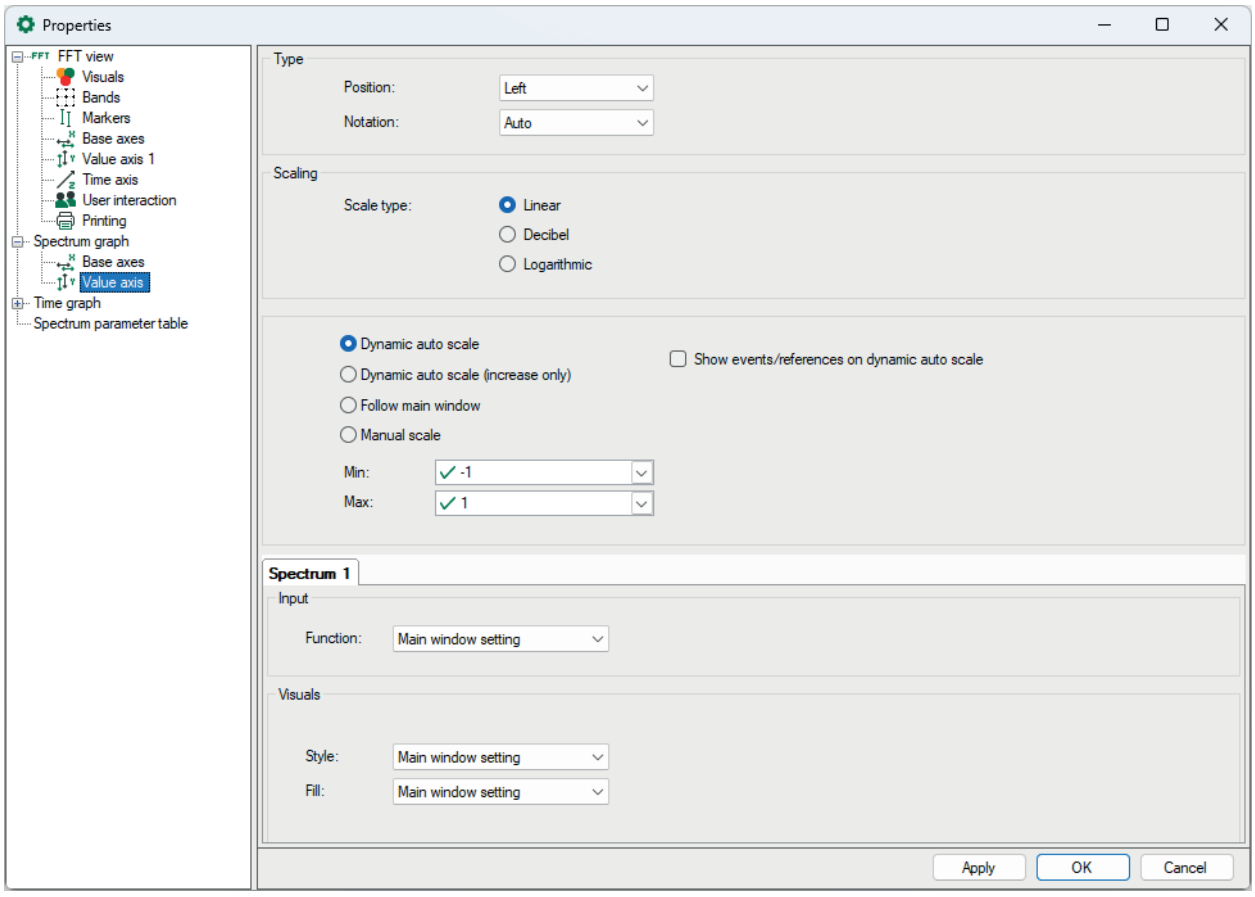
You can choose to scale manually or display the full range.

If you zoom in the spectrum graph or in the main window, this is usually independent from each other. By using the "Follow main window" option, you can determine that a zoom action in the main window also affects the spectrum graph, but only in horizontal direction.



Value axis

The spectrum graph has only one value axis. All charts in the display are displayed on the same scale of values. You can change the settings of the value axis in the properties of the display.



For scaling the value axis, you can choose between linear, decibel and logarithmic.

Scaling mode
The options of the value axis of the main view are available for selection, see chapter ↗ *Value axes, page 97*, and in addition:

- Follow main window: With this option, a zoom action in the vertical direction in the main window also affects the spectrum graph.
- Show events/references on dynamic auto scale: the scaling is automatically adjusted so that events (such as limit values) and reference values are always displayed.

Spectrum x
In the *Spectrum x tab*, you can determine the display properties for style and filling for each spectrum separately. You can adopt the main window setting or select individual settings from the respective drop-down menu.

You can also set the display properties for style and fill for each spectrum separately. You can apply the main setting of the window or select individual settings from the respective drop-down menu.

7.4.2 Spectrum table

The data table for the frequency spectrum only contains data if it is an InSpectra module. In case of a simple analog signal, the table remains empty.

In the table, a line is automatically created for every defined band of the displayed InSpectra module.

							Peak		RMS			
No.	Band ...	Center	Delta	Peak	Peak Freq	RMS	Alert	Alarm	Alert	Alarm	Visible	Collapsed
Show bands Enable collapsed bands : (InSpectra Expert)												
0	Insge...	974,121	974,121	1,03141	283,203	1,31708					☐	☑
1	300	300	50	1,03141	283,203	1,10277	> 0,2	> 0,7	> 0,3	> 0,6	☑	☑
2	600	600	100	0,629698	561,523	0,612792	> 0,5		> 0,2		☑	☑

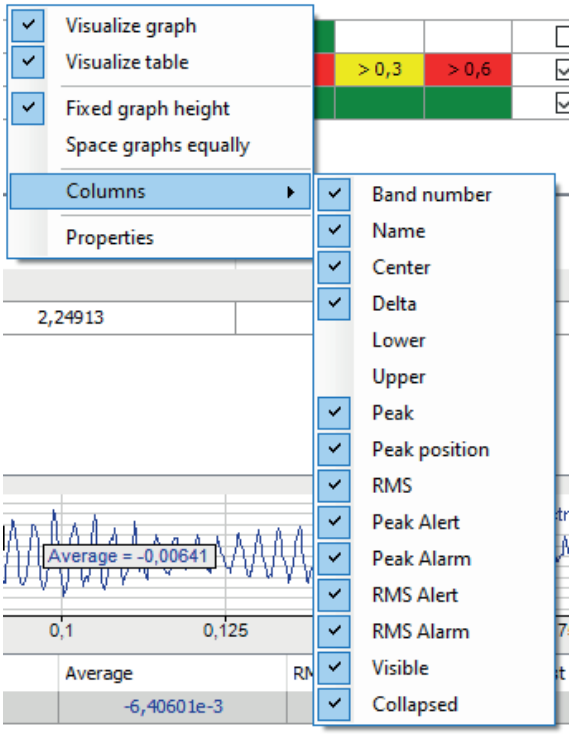
			Events	
No.	Name	Value	Alert	Alarm
InSpectra Module: (InSpectra Expert)				
0	P1+P2	1,66111		

The parameters and – if configured – the results are shown for each band. Results and alarms for characteristic values are displayed in the area below. A line is created for each parameter.

If there are several InSpectra modules in the spectrum graph, the table also shows the data for the bands of the other spectra, see in the figure below in the case of two InSpectra modules.

									Peak		RMS			
No.	Band...	Center	Delta	Lower F...	Upper F...	Peak	Peak Freq	RMS	Alert	Alarm	Alert	Alarm	Visible	Collapsed
Show bands Enable collapsed bands : (InSpectra Expert)														
0	Insge...	974,121	974,121	0	1,94824...	1,48727	458,984	1,68337					☐	☑
1	300	300	50	250	350	0,074371	297,852	0,130026					☑	☑
2	600	600	100	500	700	0,100702	517,578	0,217807			> 0,2		☑	☑
Show bands Enable collapsed bands : (InSpectra Expert_C)														
0	Insge...	974,121	974,121	0	1,94824...	0,653055	449,219	0,918155					☑	☐
1	400	400	100	300	500	0,653055	449,219	0,748773	> 0,5		> 0,2	> 0,4	☑	☐

You can display or hide the parameter columns via the context menu (right mouse click in the heading).



In every parameter column, the displayed values can be sorted by clicking on the table header. A triangle in the header indicates whether the sorting direction is ascending or descending. The order is automatically re-sorted if the order changes during acquisition.

You define the preference for sorting in the properties dialog of the FFT view in the node *Spectrum table*, see chapter ↗ *Spectrum graph and spectrum table, page 60*. The figure below shows a data table sorted by peak values.

No.	Band ...	Center	Delta	Lower Freq	Upper Freq	Peak	Peak Freq	RMS
Show bands	Enable collapsed bands : (InSpectra Expert)							
2	600	600	100	500	700	0,060893	590,82	0,105716
1	300	300	50	250	350	0,070283	297,852	0,08079
0	Insges...	974,121	974,121	0	1,94824e+3	0,389348	390,625	0,407757

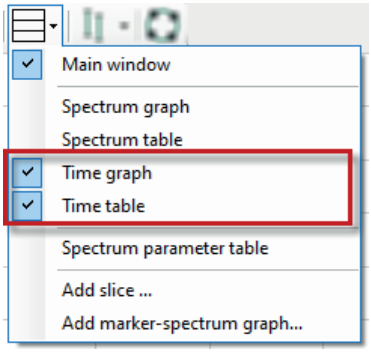
Show bands / enable collapsed bands
Use this option to globally decide for all bands whether these are displayed in the spectrum graph and whether they can be shown as collapsed bands.

If the option *Show bands* is enabled, the display of individual bands in the *Visible* column can be determined separately.

If the option *Enable collapsed bands* is marked, the display of the individual bands in the *Collapsed* column can be determined separately. Collapsed bands are indicated by a triangle at the center frequency.

7.5 Time graph and time table

In addition to the main window, you can open a graphical and/or tabular display of the data of the input signal in the time domain. Click on the button for the window menu in the tool bar of the FFT view.

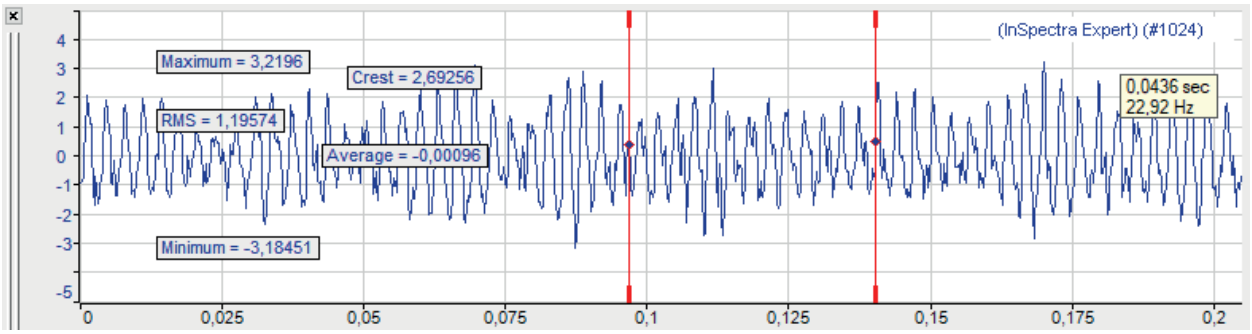


Graphical display and data table form a group, as the table always provides the data matching the graph in the display. However, the graph and table can be individually displayed or hidden.

In addition, the graph and data table can be minimized or displayed together. To do this, simply click on the small triangle on the right margin of the display.

7.5.1 Time graph

In the time graph, the time curve of the input signal is graphically displayed. The displayed section contains exactly the samples of the input signal which were included in the FFT calculation.



If the averaging function was enabled in the calculation settings of the profile, then the display shows the time signal of the last internal FFT calculation. The displays of the FFT results in the main window and frequency range, however, are also based on prior values of the input signal.

Basically the input signal of the InSpectra module is displayed. However, you can also drag further signals from the signal tree into the time graph. If there are several signals in the main window already, you can select those in the context menu of the graph.

If the time graph has the focus (after a mouse click on the header bar), the tool buttons for zooming out and restoring the manual scale relate to this graph and not to the main window. The same applies to the assigned function keys <F3>, <F4> and <F5>.

Markers

You can also enable markers via the context menu of the display (*Show markers*). The two markers can be moved independently of each other in the display. The legend on the right shows the time difference between the markers and the corresponding frequency.

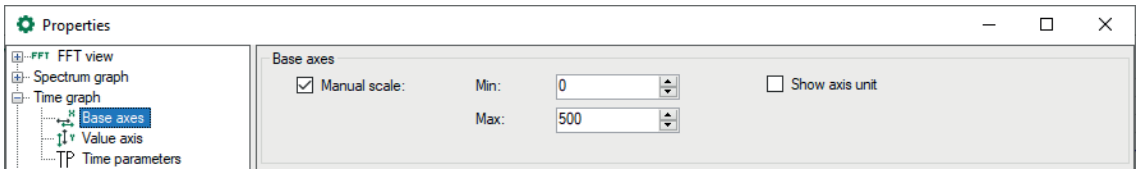
Legend

The legend of the display contains various information:

- Name of the InSpectra module (if present)
- channel number of the input signal
- Name of the input signal
- Number of samples for the FFT, unit of the input signal

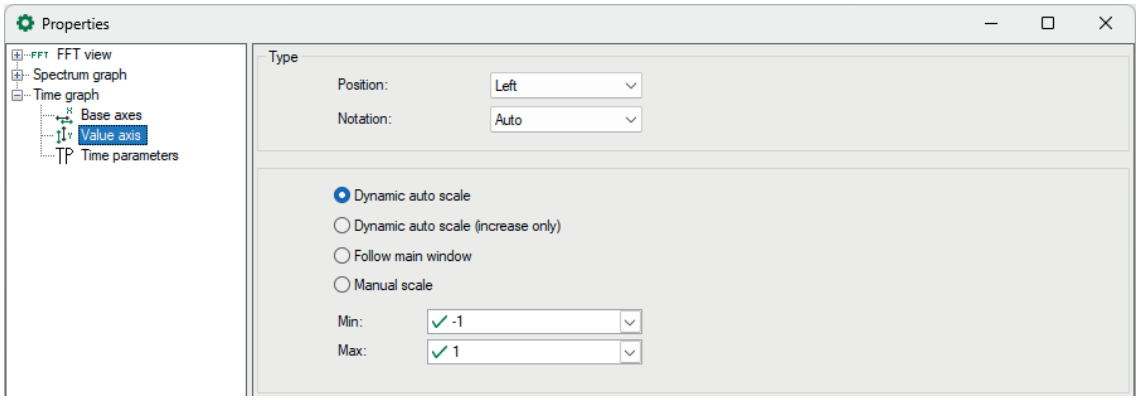
Base axis

The time graph has a base axis. When autoscaling, the length of the base axis results from the number of samples and the sampling time. You can modify the settings of the base axis in the properties of the graph.



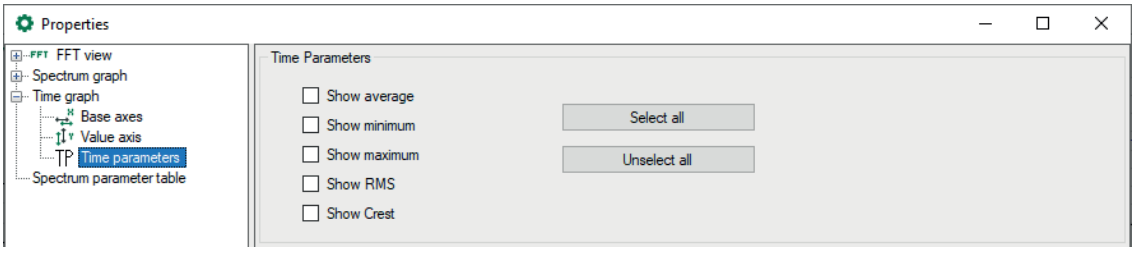
Value axis

The time graph has only one value axis. All curves in the graph are displayed on the same scale of values. You can modify the settings of the value axis in the properties of the graph.



Time parameters

The statistical values (average, minimum, maximum, RMS, crest) determined for the input signal in the shown time range can be displayed in the graph. For this purpose, select the desired parameters in the properties dialog of the graph.



Note

Sometimes, the crest factor and RMS value is not immediately visible in the graph, as it can be significantly higher or lower than the values of the signal curve. Change the scale of the value axis to see the crest factor.

7.5.2 Time table

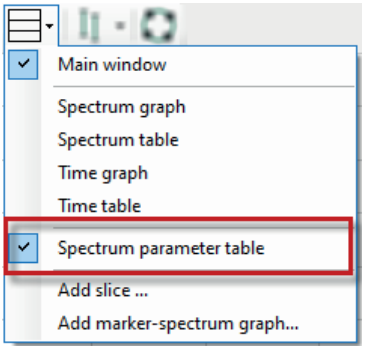
The data table of the time domain shows the same statistical values of the input signal which were described as time parameters above.

Signal / Module	Minimum	Maximum	Average	RMS	Crest
(InSpectra Expert)	-2,40479	1,85547	4,17233e-4	0,899231	2,0634

7.6 Spectrum parameter table

The spectrum parameter table is used to display the FFT calculation parameters. This allows you to display the calculation parameters you wish to observe without having to open the properties dialog of the FFT view.

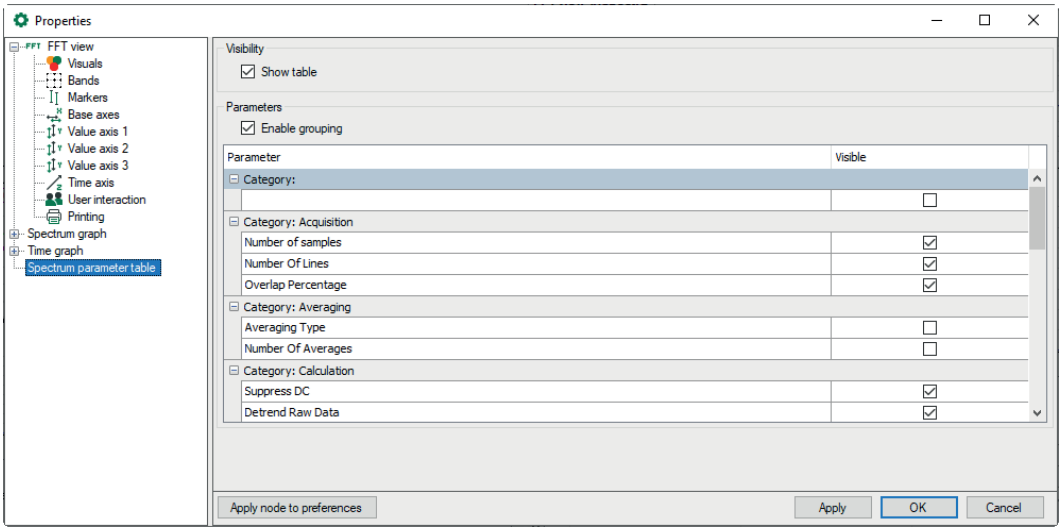
You can add the spectrum parameter table to the display using the drop-down menu.



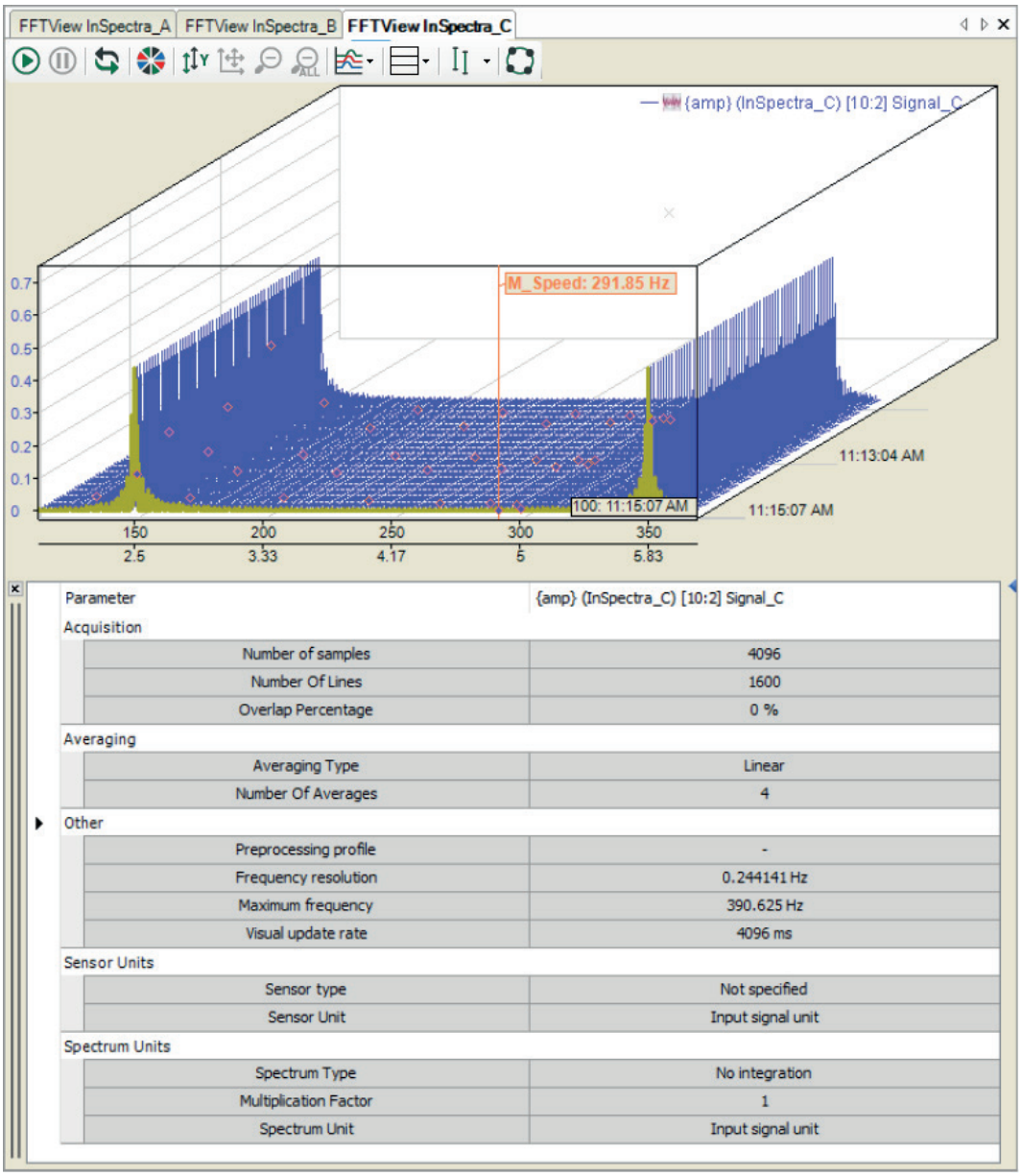
Before doing so, you should specify what information will be displayed in the table, as not all parameters are of interest and you can save some space by reducing the number of parameters.

For example, if you do not want to use an order spectrum, the order parameters can remain hidden.

You can configure the settings in the spectrum parameter table node in the properties of the FFT view. All parameters from the calculation profile are available for selection.

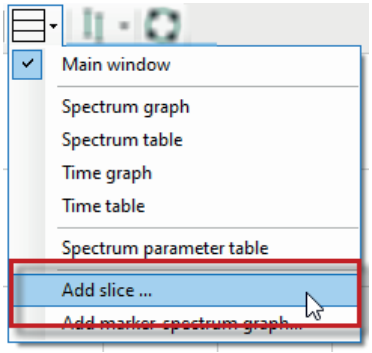


The result might look like this:



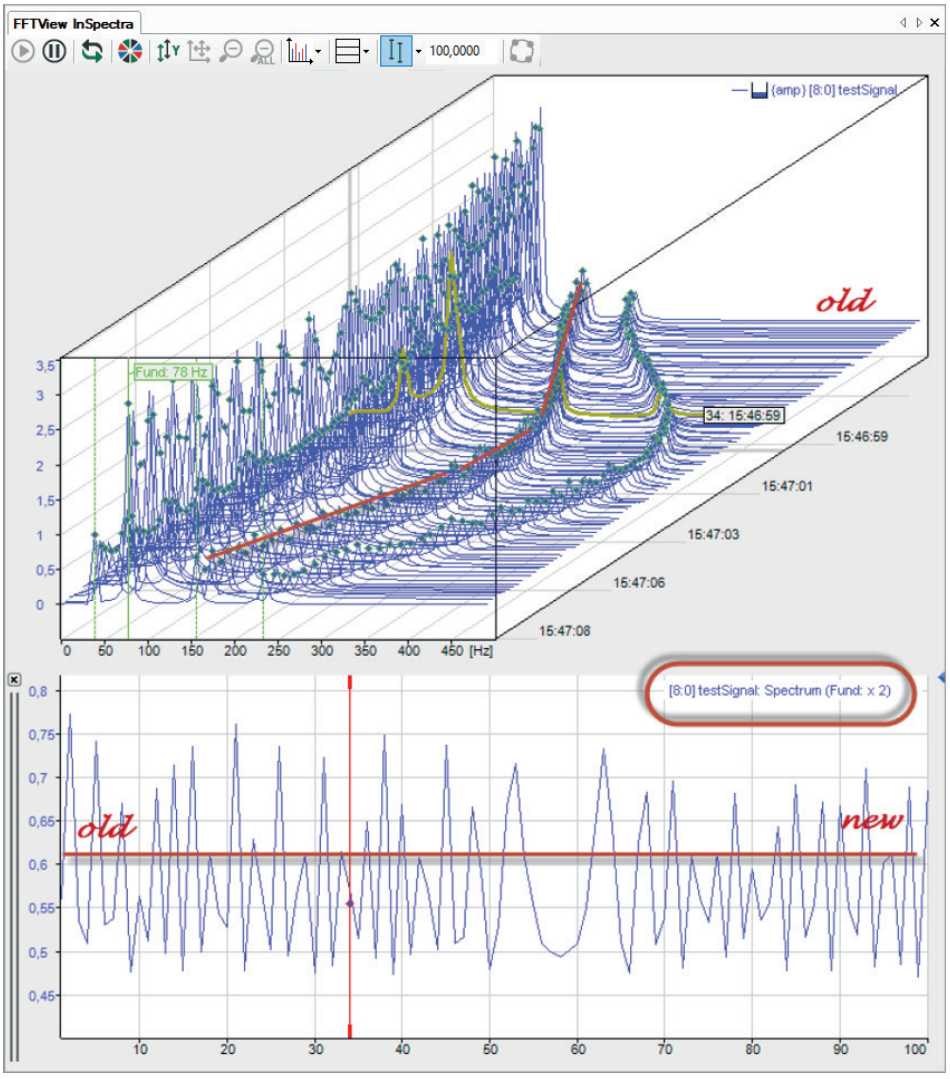
7.7 Slice view

With a slice view, you can essentially represent the chronological sequence of an FFT for a selected marker position. The amplitude profile of a frequency therefore becomes clear, especially in conjunction with the isometric waterfall view. You add a slice view using the drop-down menu for the FFT display.

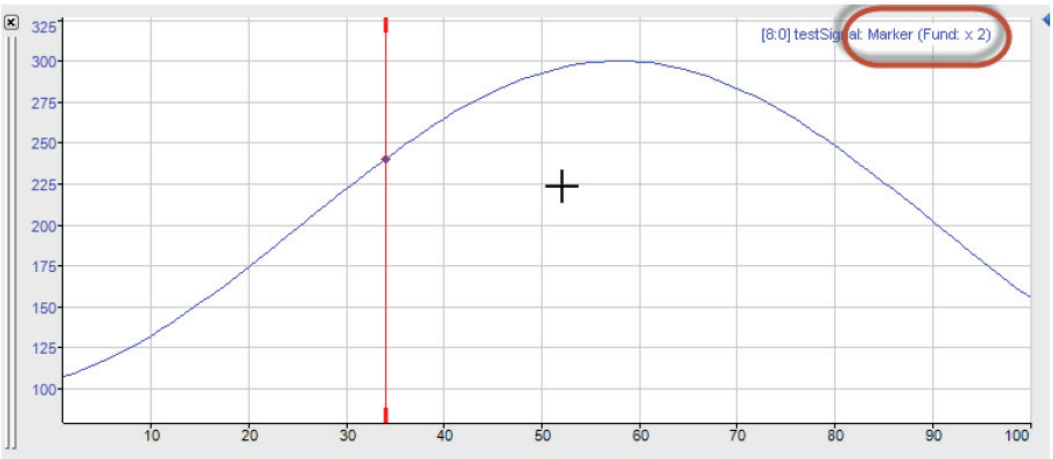


The slice view can operate in two modes:

- In **Spectrum Mode** you can monitor a spectrum value that changes over time:
  - The temporal dimension corresponds to the number of planes in the waterfall view. The highest-numbered plane contains the most recent data (front plane). The scale of the X-axis shows the plane number.
  - The frequency dimension is specified by an interactive marker or a configured marker, which is connected to a signal, e.g., a speed signal.



- In **Marker Mode** you can monitor a frequency value that changes over time:
  - Here again, the temporal dimension corresponds to the number of planes.
  - Application example: Tracking a speed marker to show the speed history.



The mode of the slice view is also displayed in the signal legend.

You can add multiple slice views for different applications.

Once defined, the slice views are listed in the drop-down menu and can also be displayed, hidden and deleted there.

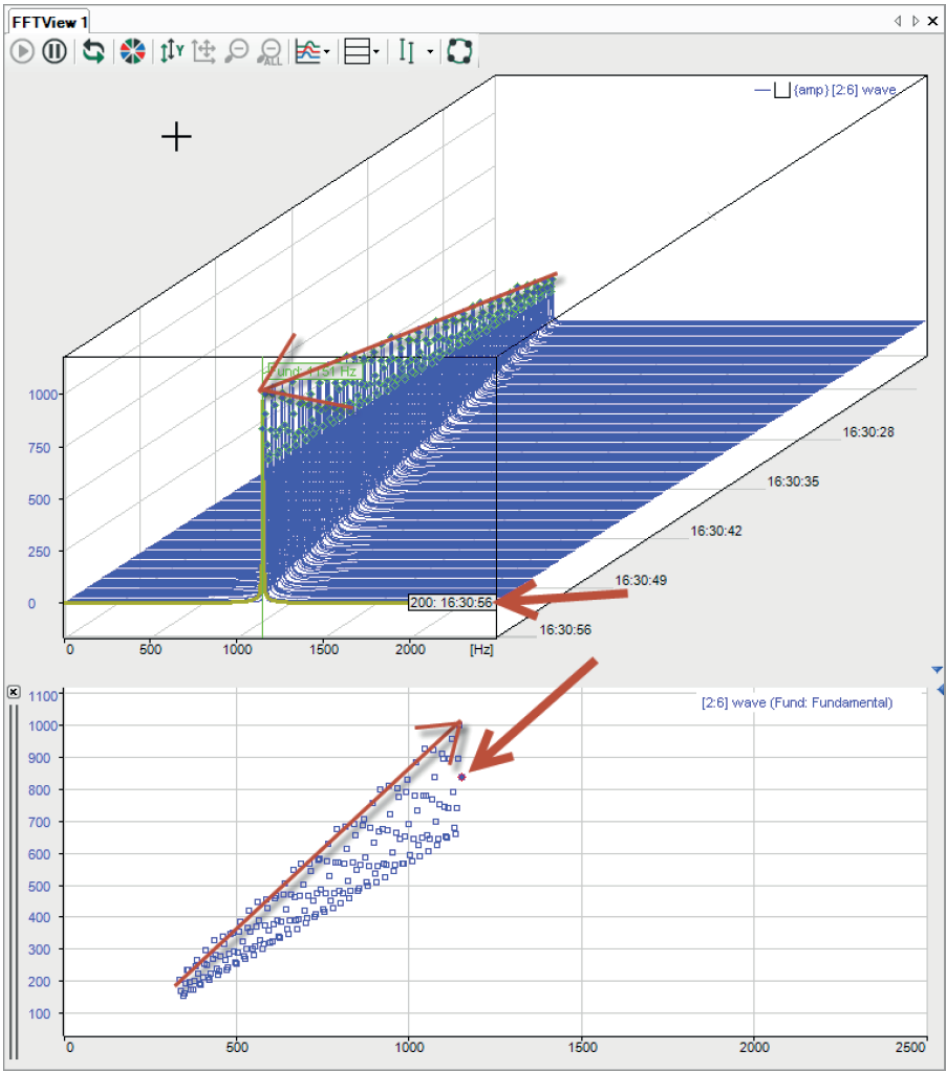
The slice view is specified by a marker. In the properties of the slice view, you can select any defined marker, including any available harmonic markers. You can also quickly switch between the different markers in the context menu on the slice view.

In addition, each slice view has its own interactive marker. You can link the interactive marker with the currently selected plane in the waterfall view with the “Link markers with waterfall” option. Note that the position of the interactive markers in the slice view always corresponds to one plane in the waterfall view.

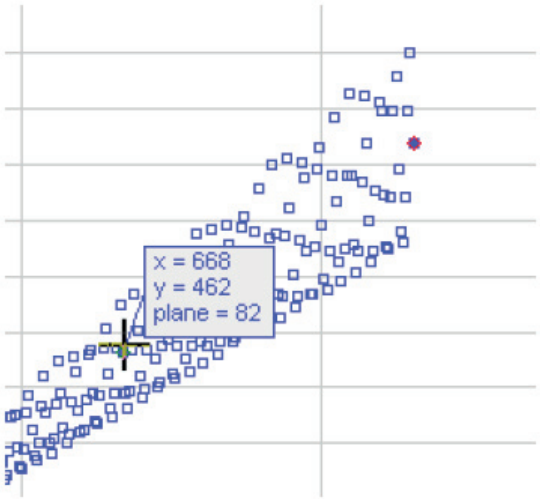
7.8 Marker spectrum display

The marker spectrum display is used to represent the relationship between a dynamic marker (horizontal axis in Hz) and the associated spectrum value (vertical axis).

For each plane of the waterfall view, a point for the value pair is entered in the graph.



This display does not have its own interactive marker. When you hold down the <M> key, the mouse moves over the points and the corresponding values (X, Y, and plane) are displayed in a pop-up.



The point within the currently selected plane is highlighted with a red circle.

You can add multiple marker spectrum displays and configure them differently.

7.9 Markers

For a better evaluation of the frequency analysis, markers can be displayed in the main window and in the spectrum graph. The markers mark frequency values along the x-coordinate. Frequencies of interest can be, for example, a constant or variable fundamental frequency, known resonance frequencies or the harmonic components.

There are several types of markers having different functions:

- Interactive marker
  There is one interactive marker. This marker can be switched on or off and manually moved. In the time graph, only this type of marker is available. See ↗ *Interactive marker, page 79*
- Configured marker
  Several markers of this type can be used in a display. This marker cannot be moved manually but its position is not necessarily fixed. The marker position can be set to a constant value or controlled by a signal. See ↗ *Configured marker, page 83*
- InSpectra marker
  This marker is configured in the InSpectra Expert module and cannot be moved manually.
- Tracking marker
  The tracking marker can be used to track the peak value of a spectrum in the contour plot.
- Tracking marker
  The tracking marker can be used to track the peak value of a spectrum in the contour plot.

For all markers, harmonic markers and sideband markers can be additionally configured.

You can enable or disable the display of the interactive marker by clicking the button in the tool bar of the FFT view. Depending on the focus, the button refers to the main window and the spectrum graph or to the time graph.

You enable or disable the display of the configured markers and the InSpectra marker solely in the properties dialog of the main window.

The markers are configured in the properties of the FFT view (main window).

For all markers, you can set that the factors are shown in the label of the harmonic markers. And it is possible to display sideband labels that show the offset.

In the waterfall display, you can connect the marker dots between the planes.

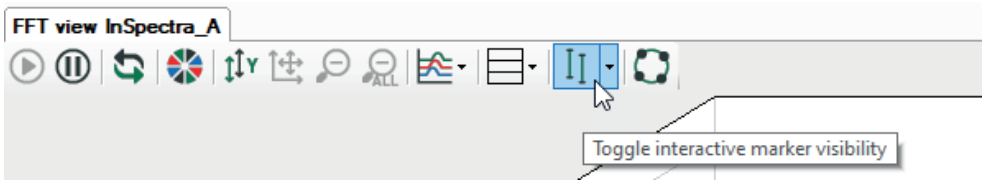
The intersections of the markers with the spectrum are displayed by small diamonds. You can hide these with the option *Hide marker dots*. If *Connect marker dots between planes* is also selected, the markers are shown as a line in the waterfall and in the contour view.

The marker lines can also be hidden in the waterfall display.

For better differentiation, you can assign your own line patterns or bold type to the different markers (first harmonic, harmonic markers and sideband markers).

7.9.1 Interactive marker

The interactive marker is used for spontaneous reading of X and Y values in a spectrum display. It can be shown or hidden at any time.

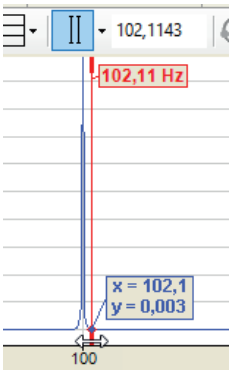


When activating for the first time, the marker is displayed at the position 1 Hz. Every time the marker is switched off and on again, it memorizes the last position.

You can change the marker position either by clicking on the thick ends at the top or at the bottom of the marker or by using the cursor keys:

Keys	Function
<Cursor to the left>/<cursor to the right>	Normal step width
<Shift>+<cursor to the left>/<cursor to the right>	Large steps
<Ctrl>+<cursor to the left>/<cursor to the right>	Small steps

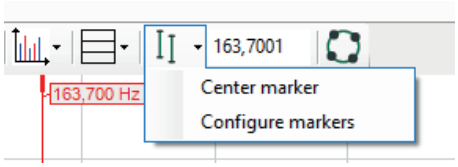
When you move the mouse over the thickened end of the marker, the cursor changes to a double-arrow symbol. You can then move the marker. In the label with the marker color (default: red), the X value is displayed on the base axis. If there are several base axes, you must specify in the settings of the base axes which axis the marker should refer to (Marker Axis). In addition, X and Y values are displayed at the intersection of the marker with the spectrum.



Center markers

Since the marker has a certain position on the frequency axis, it is possible that it is not visible in the image anymore after zooming. Switching the marker off and on to bring it back into the image is useless, as it does not change its position because of that.

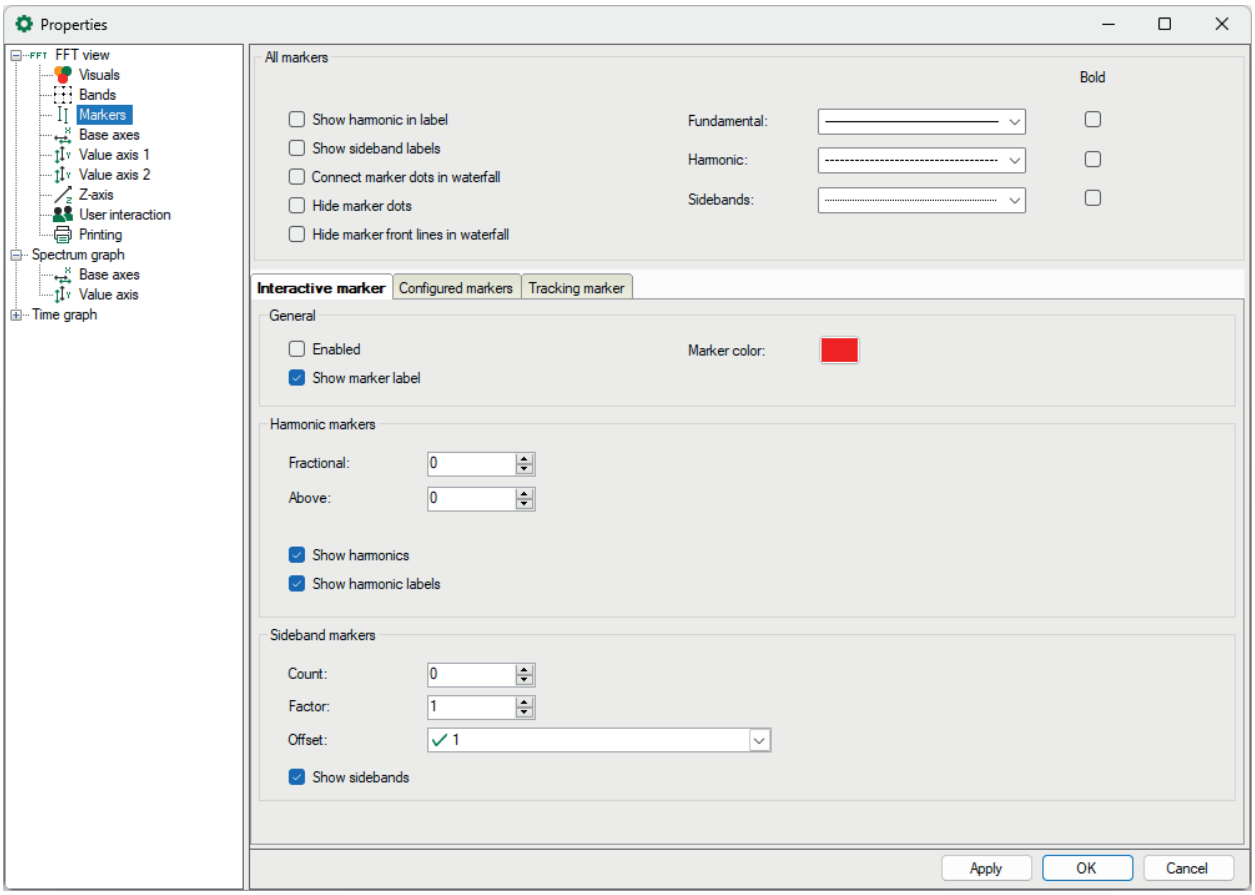
This is what the *Center Marker* function is for. With this function, you place the marker in the center of the section currently visible.



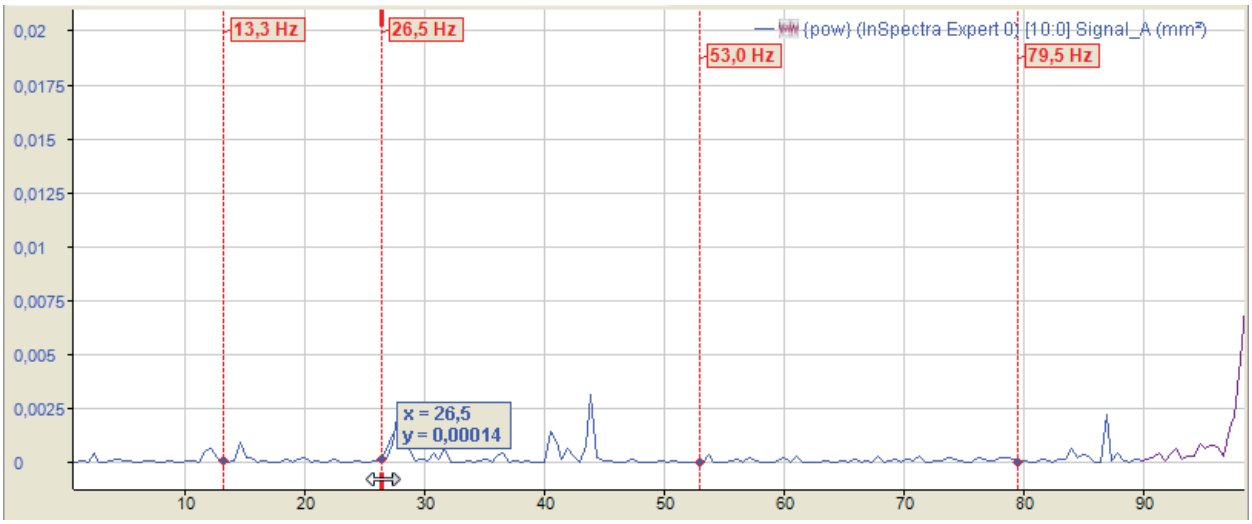
Click on the arrow symbol at the marker button in the tool bar and then on "Center marker."

Configure marker

In addition to general properties such as color and label, you can also configure harmonic markers and sideband markers in the settings.



Harmonic markers always have a frequency that corresponds to an integer multiple of the main marker. For the harmonic markers, determine the requested number of the harmonic components below and above the current marker frequency. For the harmonic frequencies, further lines are displayed. Additionally, in the *Markers* branch, enable the option *Show harmonic in label* to display the frequency values on the markers.

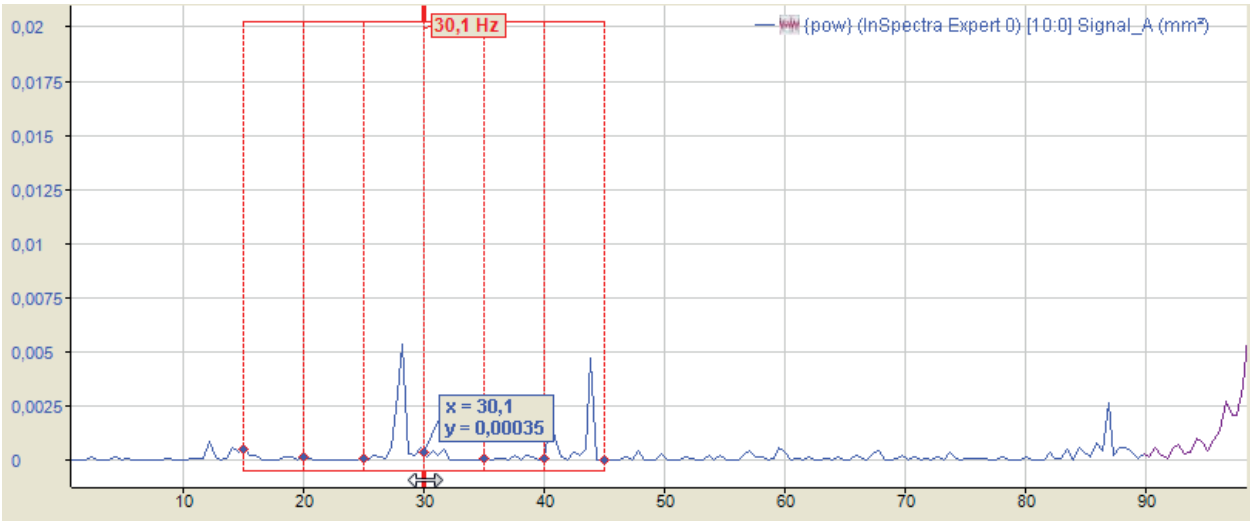


The above figure shows an interactive marker with 1 harmonic component below and 2 harmonic components above the marker frequency of 26.5 Hz.

The values of the harmonic markers are displayed at the maximum height of the value axis. The unit of these values corresponds to the unit of the base axis (see the chapter ↗ *Base axes, page 95*). The view can be configured to show only the frequency of the main marker.

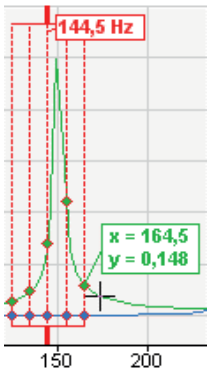
An adjustable number of sideband markers is added symmetrically right and left of the main marker. The distance to the main marker and the neighboring sidebands is the sideband offset, represented in units of the base axis. The sideband offset can be a constant value or an analog signal. The offset can also be changed with the mouse by touching one of the outer markers with the cursor and moving it to the left or right with the mouse button pressed down.

You can enter a factor if the sideband marker is to be at a multiple or fraction of the marker frequency. When the position of the marker in the display is changed by dragging, the factor is updated.

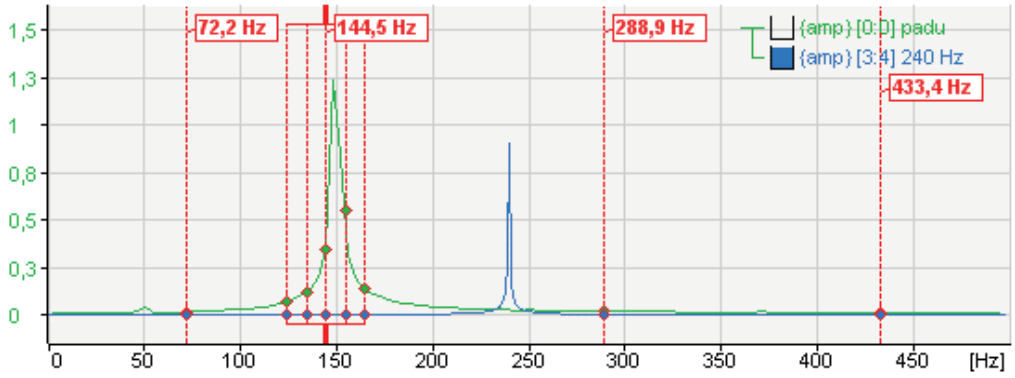


The above figure shows an interactive marker with 3 sidebands and offset of 3 Hz each.

Small diamonds indicate where markers and spectra intersect. If the mouse pointer is moved near a diamond, its coordinates (X and Y values) become visible.



Harmonic component and sideband markers can be displayed in combination, too. The figure below shows a view with a harmonic marker below and two above the main marker. The sideband offset is set to 10 Hz.



Note

If the sideband offset is specified by a signal, the value of this signal always has to be >=0. If the value is negative, the offset = 0 and no sideband markers are displayed.

Note

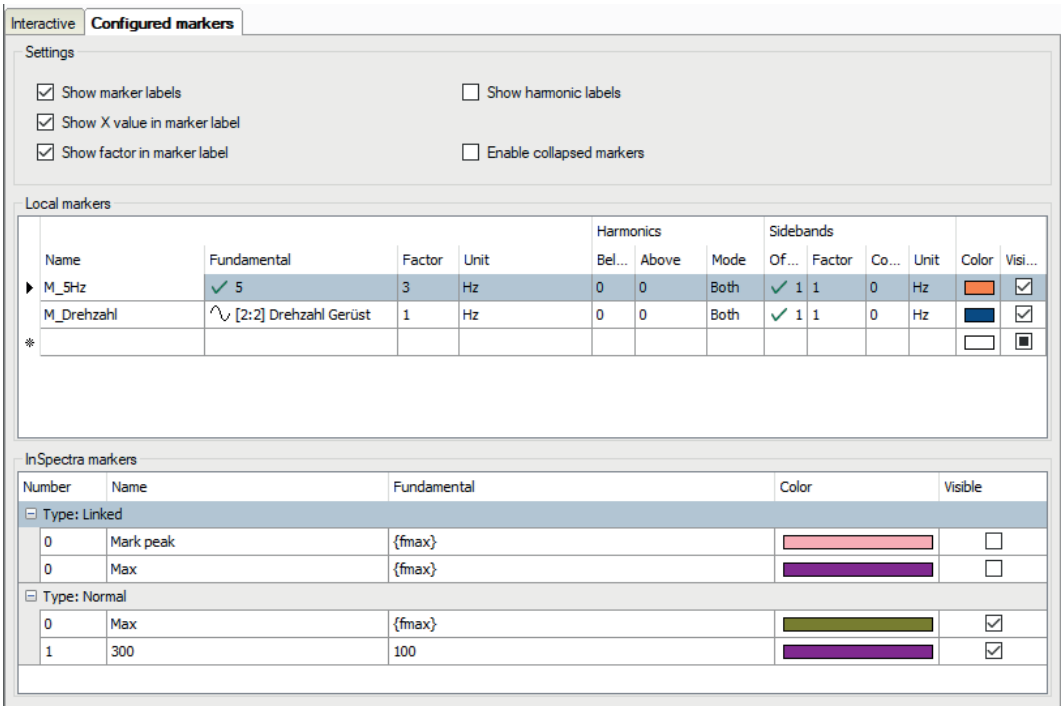
You can make the general settings of the markers in the preferences. You will find individual settings for the FFT views in the properties of a view.

For further information, see chapter ↗ *Properties and preferences FFT view, page 86*

7.9.2 Configured marker

The so-called configured markers can either be anchored at certain positions on the base axis with fixed values or moved dynamically along the base axis by means of analog signals.

The markers must first be defined and configured. You can configure the markers in the *Properties* dialog of the FFT view, in the “Marker” branch.



Settings
Here you define whether a marker label is displayed and which information is to be displayed additionally: Value on the X-axis and/or factor.

To create a marker, you simply have to enter the required information in the table line. As soon as you click in the empty space below, a new, empty line is added.

Name
Enter a clear name to be able to easily identify the marker. The name is shown in the display later on, too.

The entries for fundamental frequency, factor and unit determine the position of the marker on the base axis. The marker position is calculated by multiplying these three parameters.

Fundamental
You can enter a fixed value or select a signal for the fundamental oscillation or fundamental frequency. To select a signal, click in the table line and then on the arrow symbol. Select the signal from the signal tree.

If you want to use a signal for controlling the marker position, select a signal complying with the frequency you want to monitor.

In the example of the above image, we have selected a velocity, more precisely the speed of a drive in rpm, to control the marker. In doing so, frequencies of interest can be easily tracked,

e. g. during the acceleration and braking phase of a machine. This is especially easy to see in the main window's waterfall and contour view.

Note

If the signal for the fundamental frequency is negative, the marker is not displayed.

Factor
The default value of the factor is 1. You can enter another factor if, for example, the marker is to be positioned at a multiple or a fractional part of the fundamental frequency.

If the position of the configured marker in the display is changed by dragging, the factor is updated.

Note

The *Add speed marker* function in the FFT display automatically generates a configured marker for the speed signal. The speed signal is configured in the InSpectra Expert or InSpectra Auto-Adapting module.

Unit
As to the unit, you can choose between Hertz (Hz) and revolutions per minute (rpm). Depending on the settings, another, internal factor is taken into consideration:

- Hz: Factor = 1
- rpm: Factor = 1/60

Order can also be selected for order spectra.

Harmonics
As with the interactive marker, you can individually determine the number of harmonic markers above or below the marker frequency for every static marker. Additionally, this mode allows you to select whether only the even or odd harmonic components are taken into consideration or both types.

Sidebands
As with the interactive marker, you can individually determine the number of sideband markers and the sideband offset for every marker, static and depending an a signal value. If you drag the sideband markers, the factor is updated during dragging.

Sidebands can have a different unit than the marker itself. You can select the unit here.

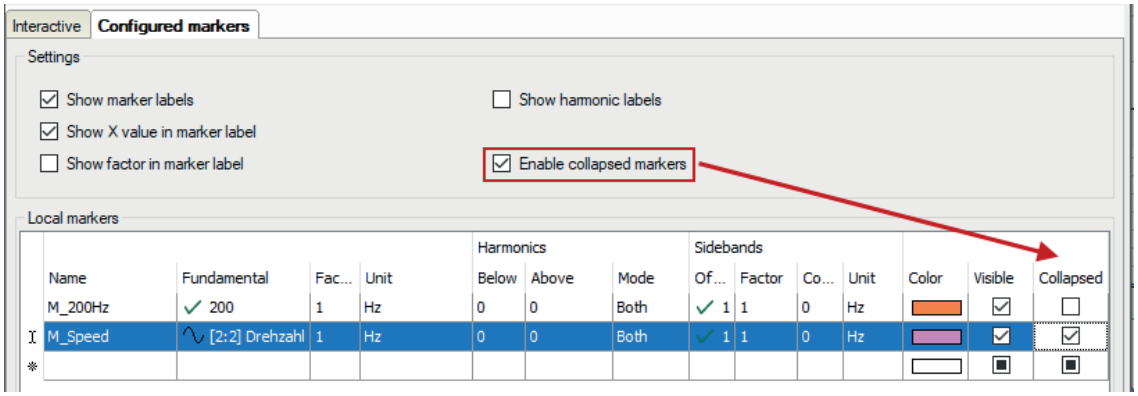
Note

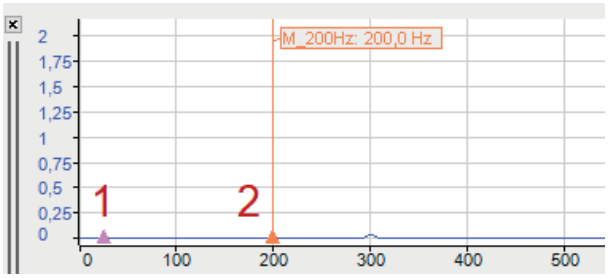
If the sideband offset is specified by a signal, the value of this signal always has to be >=0. If the value is negative, the offset = 0 and no sideband markers are displayed.

Color
Here, you can allocate an individual color to every static marker

Visible
This option decides whether a fixed marker is displayed or not. This is the only possibility of enabling or disabling static markers for the display. The marker button in the toolbar of the FFT view does not control the static markers!

“Enable collapsed markers" option
When you enable this option, an additional column appears in the marker table, in which you can decide for each marker whether it is normal, i.e. it should be displayed as a line and possibly with a label, or only as a triangle based on a spectrum.





The example above shows a collapsed marker (1) and a normal marker (2).

InSpectra marker
The InSpectra markers are shown in the table below.

InSpectra markers

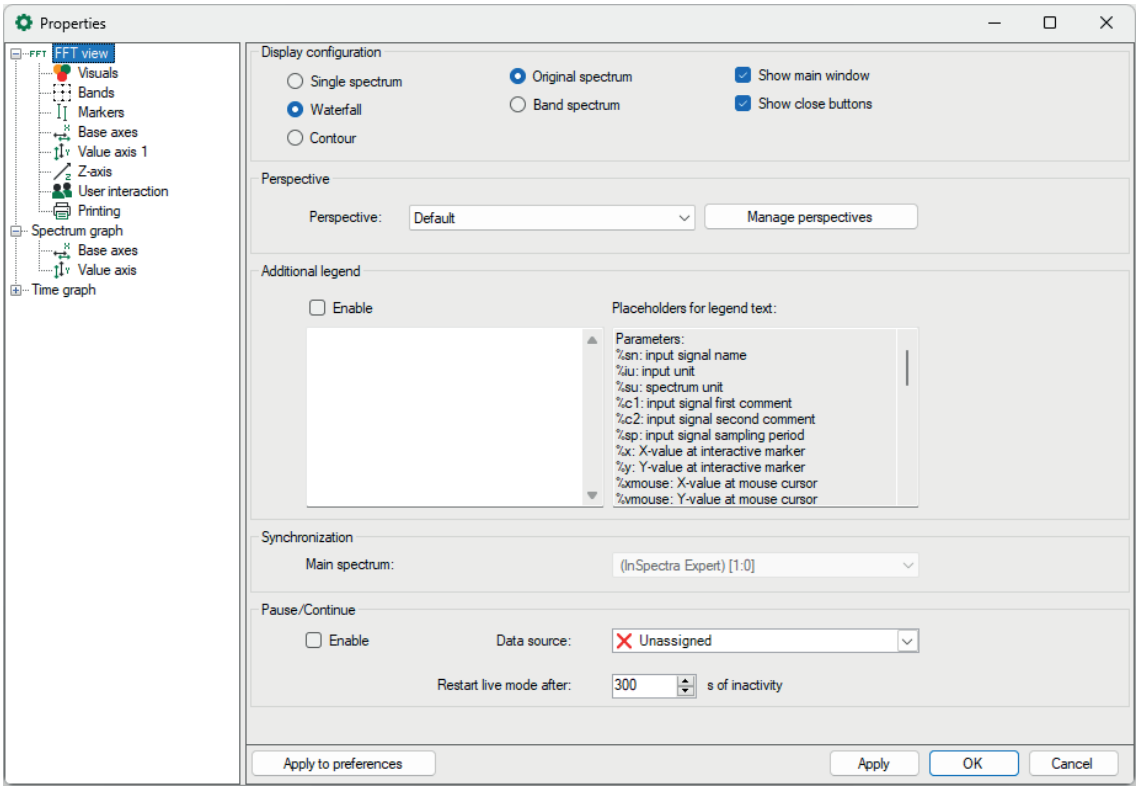
Number	Name	Fundamental	Color	Visible
Type: Linked				
0	Mark peak	{fmax}		☐
0	Max	{fmax}		☐
Type: Normal				
0	Max	{fmax}		☑
1	300	100		☑

The settings of the InSpectra markers can only be changed in the InSpectra profile. Only the visibility can be set here.

7.10 Properties and preferences FFT view

In the FFT view of *ibaInSpectra (ibaPDA)*, *ibaAnalyzer-InSpectra* and *ibaQPanel*, all settings can be adopted node by node in the preferences and are thus applied to the newly opened FFT views. Changes can be saved by pressing the button <Apply node to preferences>. The preferences cannot be viewed separately in *ibaAnalyzer*. A new FFT view must be opened in order to view preferences. In *ibaPDA*, you open the preferences via the menu *Configuration - Preferences*.

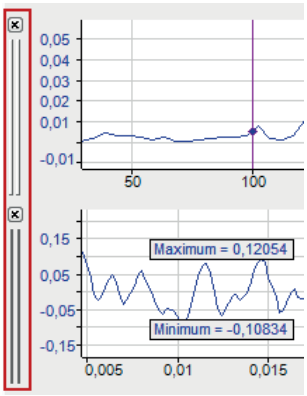
The node *FFT view* offers general settings for the display of the calculated FFTs. The following figure shows the settings in the properties of the FFT view. The preferences dialog looks similar.



Display configuration
Choose between single spectrum, the waterfall view and the contour view of the spectra. The visibility of the main window can also be set here.

You can select *Band spectrum* instead of the *Original spectrum*, then the results of the individual bands will be displayed. Settings for this are made in the *Bands* node, see chapter ↗ *Bands, page 90*.

With the *Show close buttons* option you can control the visibility of the close buttons and the lines to the left of the display.



Perspective: Drop-down list *Custom perspective*
If you have saved different perspectives for the waterfall (3D) view, then you can select one of them.

Click the button <Manage perspectives> to open the dialog for managing perspectives. This lets you delete existing perspectives, copy them to the clipboard or paste them from the clipboard. Since perspectives are always specific to one FFT view, you must copy and paste a perspective that you want to use in exactly the same way in another FFT view into the other FFT view.

The perspective is saved in the display. Once you have configured the desired perspective, select *Save perspectives* in the context menu of the main window. Give the perspective a name and close the dialog with <OK>.

Additional legend
When this option is enabled, another legend is displayed in the main window in addition to the normal signal legend. You can define the content of this legend yourself. For example, you can enter a detailed multi-line text, in which placeholders for dynamic information can also be used. The following placeholders are available:

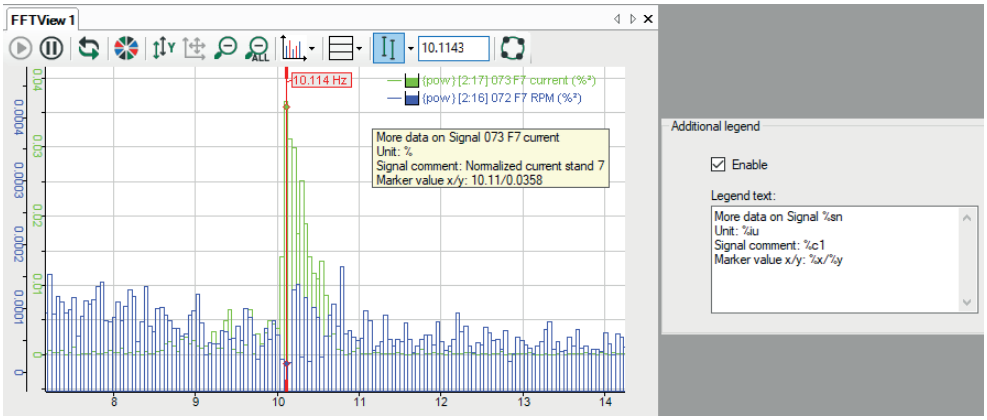
- %sn: Input signal name
- %iu: Input unit
- %su: Spectrum unit
- %c1: Input signal first comment
- %c2: Input signal second comment
- %sp: Input signal sampling time
- %x: X-value at interactive marker
- %y: Y-value at interactive marker
- %xmouse: X-value at mouse cursor
- %ymouse: Y-value at mouse cursor
- %tmouse: Z-value at mouse cursor
- %xmv: X-value of the nearby marker position
- %ymv: Y-value of the nearby marker position
- %tmv: Time value of nearby next marker position

- %nmv: Name of the nearby marker position
- %imn: InSpectra Expert module name
- %rms: RMS value of the selected plane (based on incoming values)

By default, all signal-related placeholders are determined based on the first spectrum. To identify another spectrum, use a colon followed by the word “spectrum” and the index of the spectrum, e.g. "%sn:spectrum1", in order to refer to the first spectrum.

Use the optional formatting string “w.p” to specify the format of the numeric parameters, where “w” is the width and “p” is the precision. The width is the minimum number of the characters shown. Precision is the number of decimal places. Example: "%5.3y1" indicates the Y-value for marker X1 with a width of 5 characters and a precision of 3.

The display always shows the information for the uppermost signal in the main window. The following figure shows the definition of an additional legend (right) and its display (left).



Synchronization
By default, if only one spectrum is displayed in the FFT view, identifiers, markers and zones are synchronized with this spectrum and this setting is not available. If multiple spectra are displayed in the FFT view, you can define the main spectrum here which will be used for synchronization.

Pause/Continue
This function is only available in *ibaPDA*. If this option is enabled, the visualization of the FFT is controlled by a digital signal. The FFT calculation is continued.

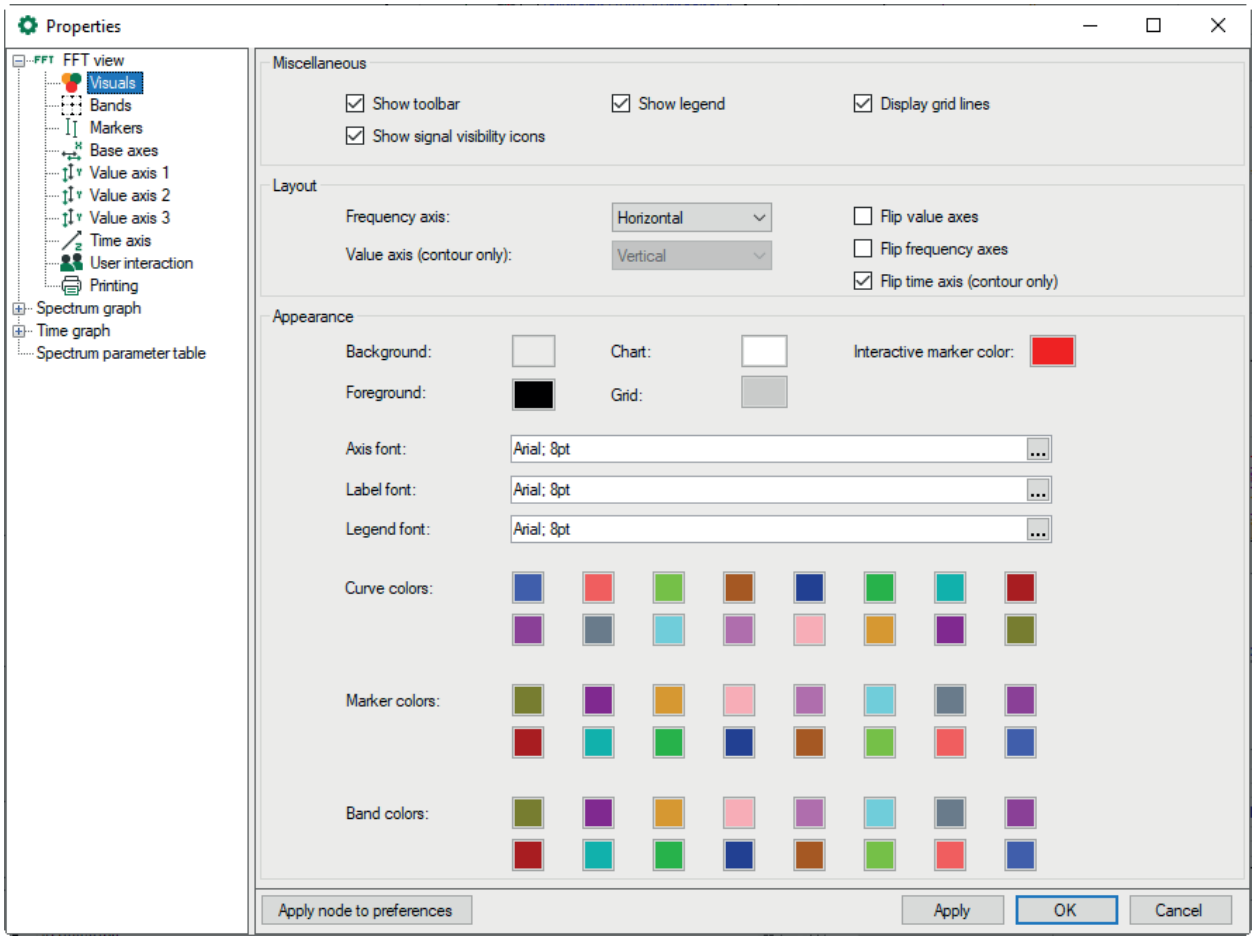
If the digital signal is TRUE (1), the visualization is paused and the display shows the frozen image of the last result.

If the digital signal is FALSE (0), the visualization continues and the display is updated regularly.

You can also enter a time in s after which live mode is automatically continued.

7.10.1 Visuals

In the dialog of the *Visuals* node, you can set the appearance and colors of the FFT view.



Miscellaneous

You can set whether the toolbar, legend and grid lines are displayed or not.

If you have checked the "Show signal visibility icons" option, you can hide and show the signal by clicking the eye icon in the signal legend.

Layout

You can change the alignment of the FFT axes from horizontal to vertical or vice versa by selecting the relevant option from the picklist *Frequency axis*. You can also flip the individual axes.

For a contour view, the color axis (value axsis) can be displayed horizontally or vertically next to it.

Appearance

Here you make the settings for colors and fonts. For the coloring of curves, markers and bands, 16 colors are available, which are automatically assigned to the corresponding items one after the other when they are added in the view.

7.10.2 Bands

The display supports frequency and value bands. These bands highlight certain parts of the spectra in a different color. Frequency bands (horizontal) have a static or dynamic average (center frequency) and a delta width. Value bands (vertical) start at a dynamic or static value and either reach upward to the next value band or positive infinity.

The frequency bands can optionally be assigned to individual spectra or to all spectra. Value bands apply to all spectra.

Bands are configured in the properties dialog in the node *Bands*. There are two types of bands:

- You can allocate *custom bands* to any spectrum or all spectra
- *InSpectra bands* are bands that were configured in the calculation profile of an InSpectra module

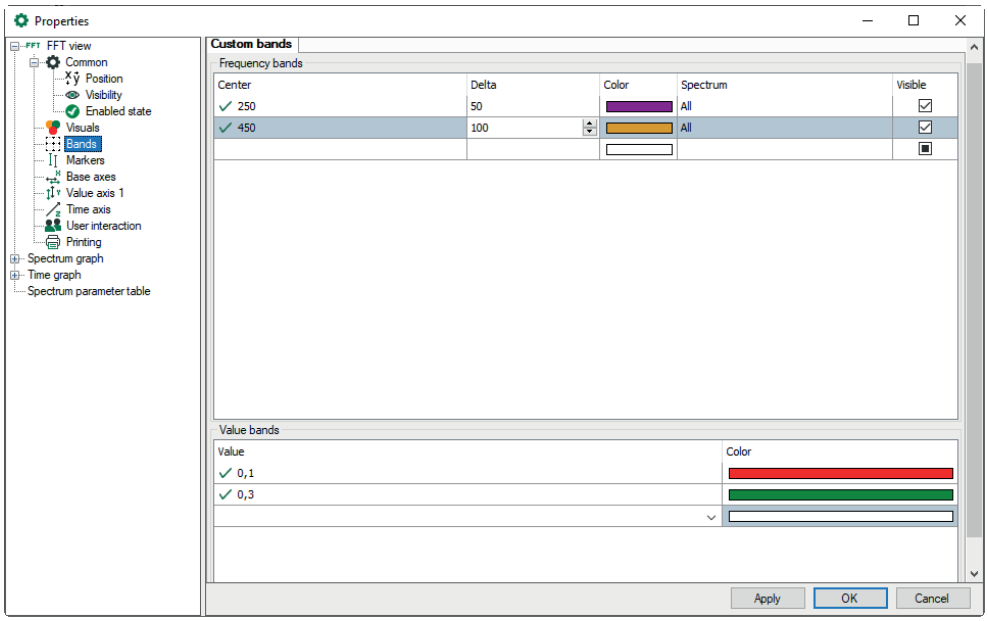
If you have selected *Band spectrum* (instead of *Original spectrum*) in the main node *FFT view*, the *Band spectrum* tab appears. For settings see chapter ↗ *Band spectrum, page 94*.

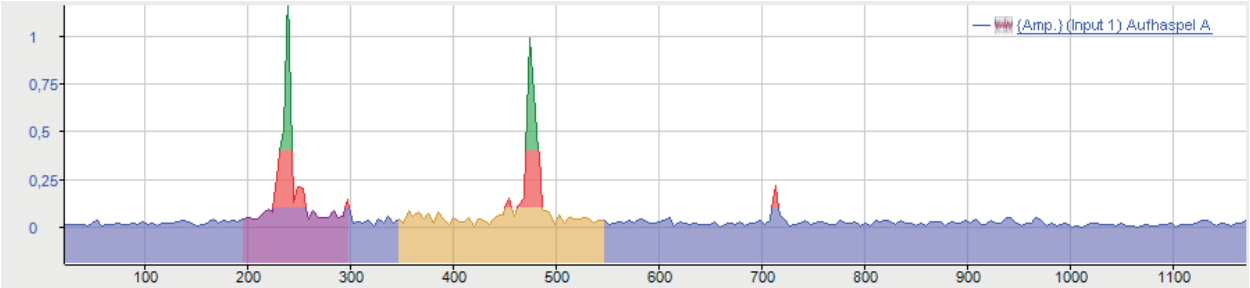
Custom bands

In the *custom bands* tab, you define the frequency bands with a static or dynamic center frequency and a delta frequency. You can allocate a color to the band and a certain spectrum or all spectra.

In the diagram below, you can see the effects of the following settings.

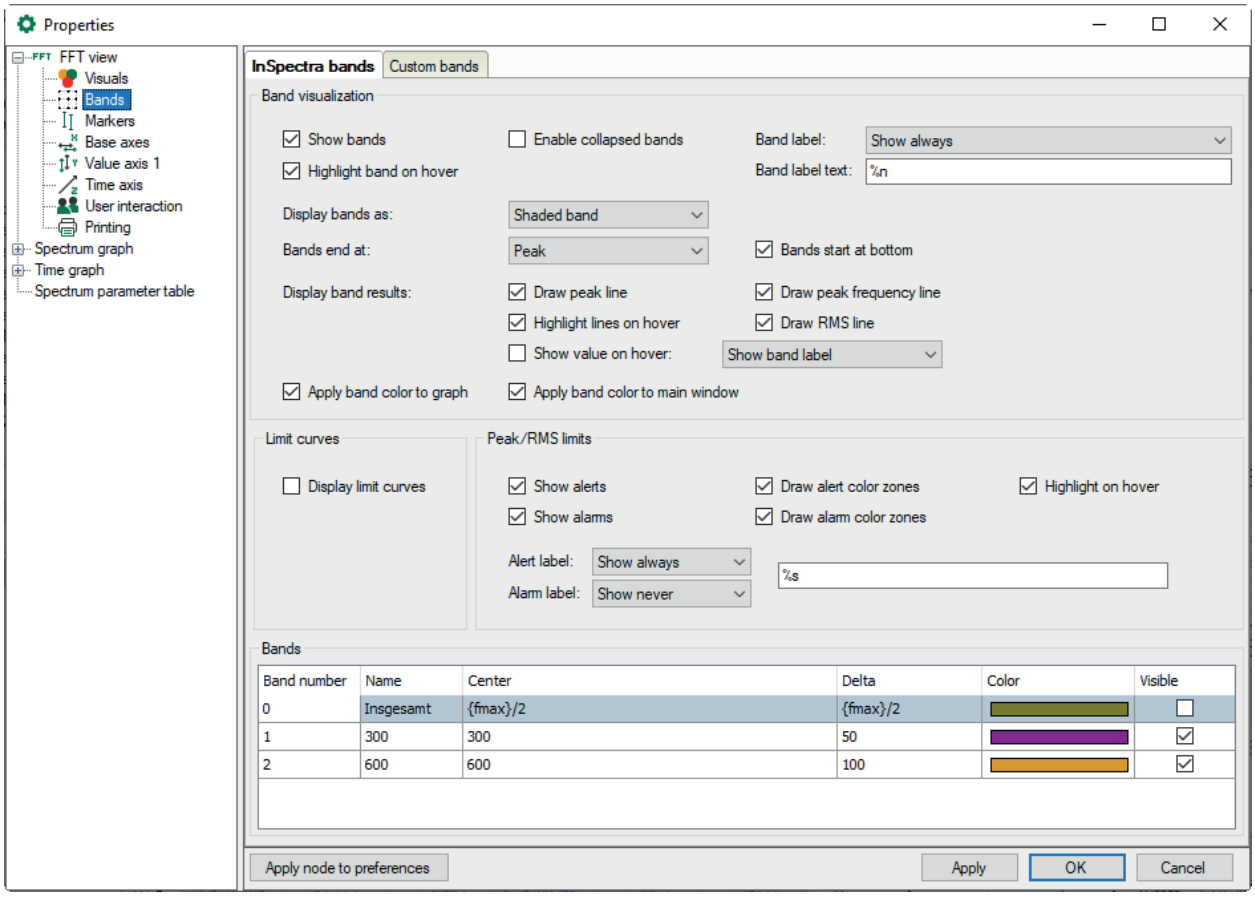
Example:





InSpectra bands

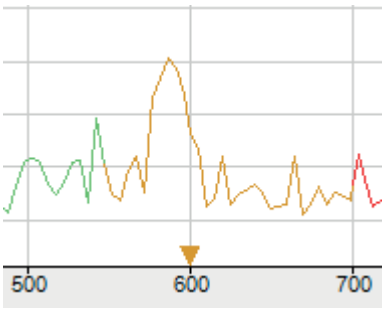
If you use the FFT view with InSpectra, an additional *InSpectra Bands* tab will appear in this dialog.



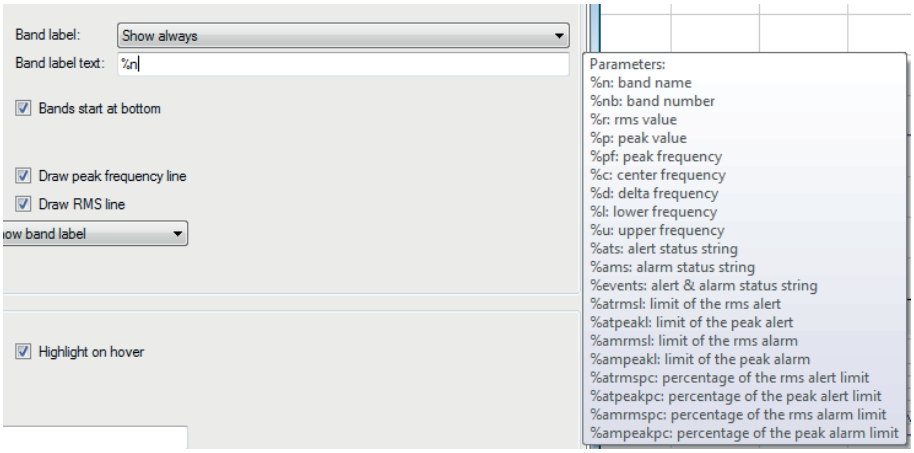
Band settings

Display properties of the InSpectra bands can be determined in the *Band settings* area.

You can enable the collapsed appearance of the bands and whether the band is highlighted on hover. If this option is enabled, the band is highlighted in the display of the frequency spectrum and in the data table.



You can determine when the band labels are displayed (never, always or on hover) and what is displayed in the label. If you click the band label text field, a list of parameters appears that you can use for dynamic information in the label text.



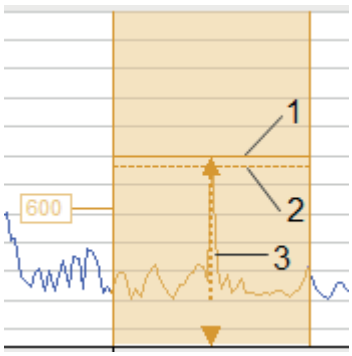
The following parameters can be used:

- %n: Band name
- %nb: Band number
- %r: RMS value
- %p: Peak value
- %pf: Peak frequency
- %c: Center frequency
- %d: Delta frequency
- %l: Lower frequency
- %u: Upper frequency
- %ats: Alert status string
- %ams: Alarm status string
- %events: Alert and alarm status string
- %atrmsl: Limit of the RMS alert
- %atpeakl: Limit of the peak alert

- %amrmsl: Limit of the RMS alarm
- %ampeakl: Limit of the peak alarm
- %atrmspc: Percentage of the RMS alert limit
- %atpeakpc: Percentage of the peak alert limit
- %amrmspc: Percentage of the RMS alarm limit
- %ampeakpc: Percentage of the peak alarm limit

You can determine whether the bands should begin at the lower margin and where they should end (at the end of graph, at the peak or at the RMS value). The frequency bands can be shown as a shaded band or non-shaded band or only as a line at the center frequency.

The characteristic values of the bands can be displayed as lines, which can be highlighted on hover. Example:



1 line for peak value

2 line for RMS value

3 line for frequency peak value, highlight on hover

In addition, the band color can be adopted as a curve color both in the spectrum as well as in the main window.

Event settings

Display properties for events (alerts, alarms) can be set in the *Event settings* area. Dynamic label texts can also be defined for events. See band settings below.

Bands

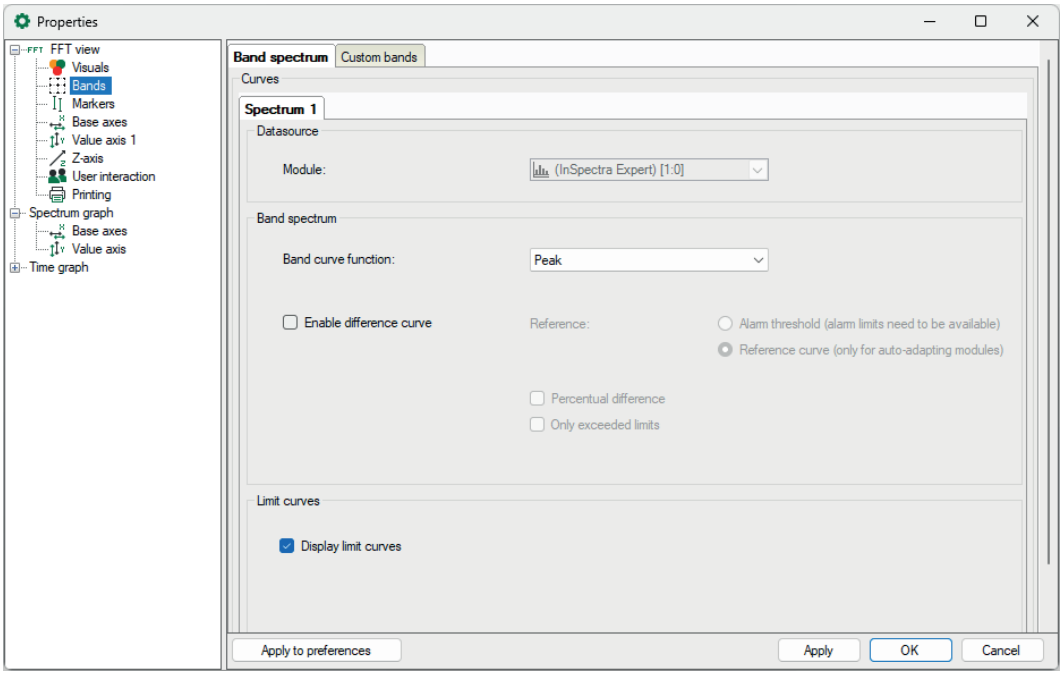
The bands configured in an InSpectra profile are shown in the table below in the dialog. The name, center frequency and delta frequency are already defined in the InSpectra profile and can no longer be changed here. The color and visibility can still be changed here.

Bands

BandNumber	Name	Center	Delta	Color	Visible
0	Insgesamt	{fmax}/2	{fmax}/2		☐
1	300	300	50		☑
2	600	600	100		☑

7.10.2.1 Band spectrum

The spectra of the individual bands can be displayed in the *Band spectrum* mode.



Data source
Display of the module

Settings
You can select the displayed curve function in the *Band curve function* drop-down menu:

- Peak, Peak fequency, RMS

Enable difference curve
If the FFT view shows an auto-adapting module, a difference spectrum can be displayed representing the difference between the reference curve or the alarm thresholds for each band.

The *Percentual difference* shows all values in percent to the reference values. If *Only exceeded limits* is enabled, the Y-value is set to 0 for all bands in which the limit is not exceeded.

Display limit curves
Enable this option to show the limit curves in the FFT view. A limit curve interconnects all band limits of the same type (warning, alarm, lower or upper).

7.10.3 Markers

You will find the description of the marker settings in chapter ↗ *Markers, page 78*.

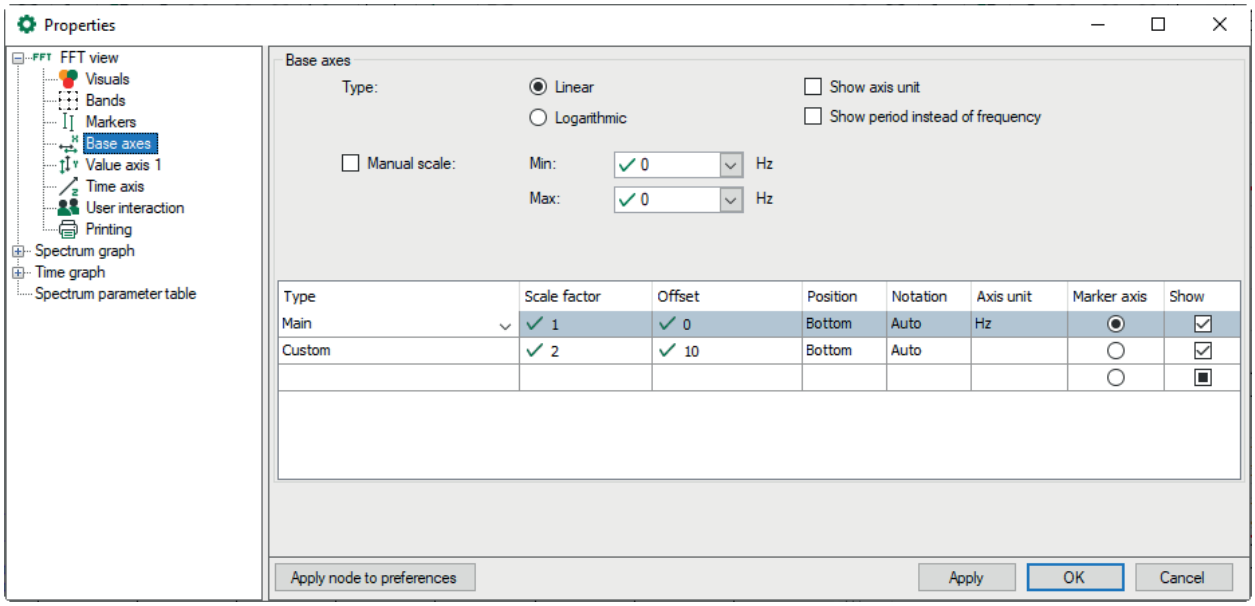
7.10.4 Base axes

You can choose between linear and logarithmic display here and whether the axis unit is displayed or not. Typically, the base axis has the unit Hz in the frequency domain and seconds in the time domain.

The base axis can be inverted so that, for example, the vibration duration (T) is shown instead of frequency (f). The following applies here: T = 1/f.

If *Cepstrum* is set as the spectrum method in the InSpectra profile, a *Millstrum* can be displayed for the base axis instead of period or frequency.

Properties



The figure above shows the properties of the base axes in the FFT view. By default, the scaling values are automatically determined. However, you can also make a manual specification.

In case of using a manual scale, beside static values you have the possibility to use a signal for determination of beginning and end values of the scale (min, max). Therefore, just click on the arrow in the *Min* or *Max* field and select the appropriate signal in the signal tree.

By default, the table of axes only shows the default Hz axis or order axis with optional setting options for the position (top/bottom), notation (auto/standard/scientific), and axis unit. To change these settings, click on the corresponding cell and select them from the drop-down list.

If you have defined several base axes, select which base axis the markers in the display should refer to in the *Marker Axis* column.

Use the *Show* option to control whether the base axis is displayed or not.

You can add and configure other base axes as needed. These additional base axes can have a different scale, reference value or unit.

An offset can be configured for a custom X-axis. It is also possible to configure a dynamic offset i.e., an offset depending on a signal value.

By default, the basic display settings for the base axes are adopted from the main window and from the spectrum graph. You can then either change some display settings for the base axes for the frequency spectrum display or synchronize them again with those from the main window.

Note

The manual “Minimum” and “Maximum” scale settings always apply only to the main axis (Hz). All other base axes enabled in the FFT view are scaled automatically.

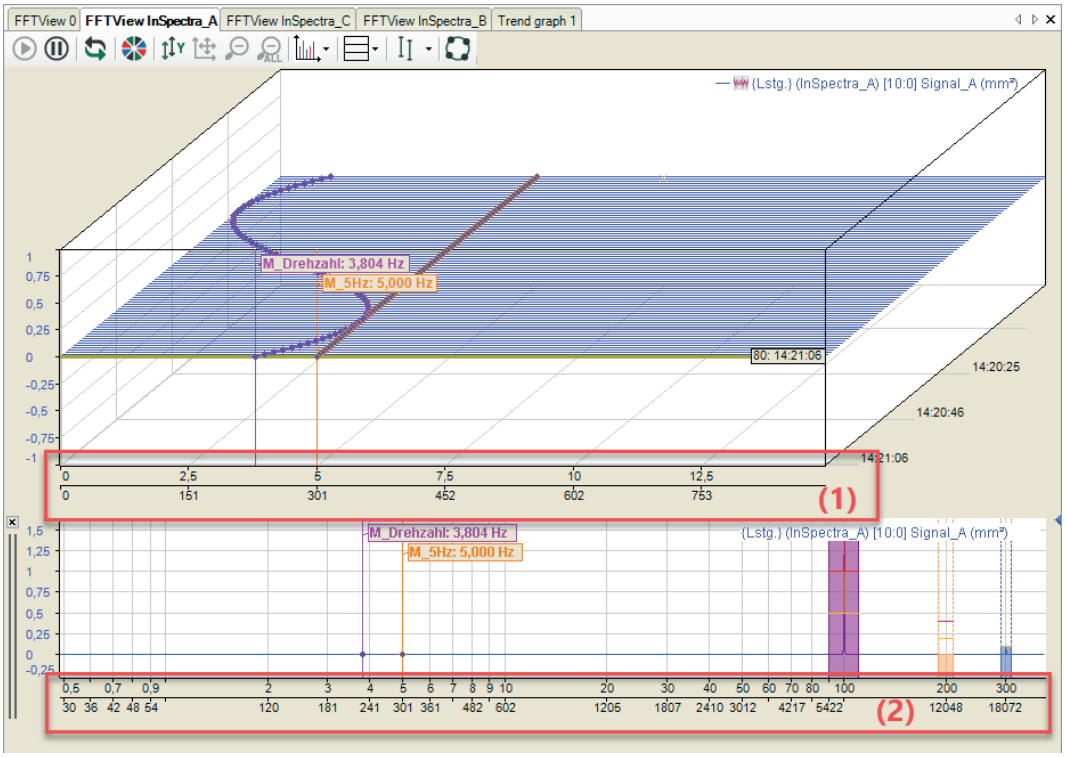
Each axis to be shown in the graph is represented by a line in the table below.

In the following example, two base axes were defined. The first shows the frequency in Hz and the second in rpm. The base axes in the main window (1) are linear. The base axes in the display frequency spectrum (2) are logarithmic.

In addition, the main window was zoomed in.

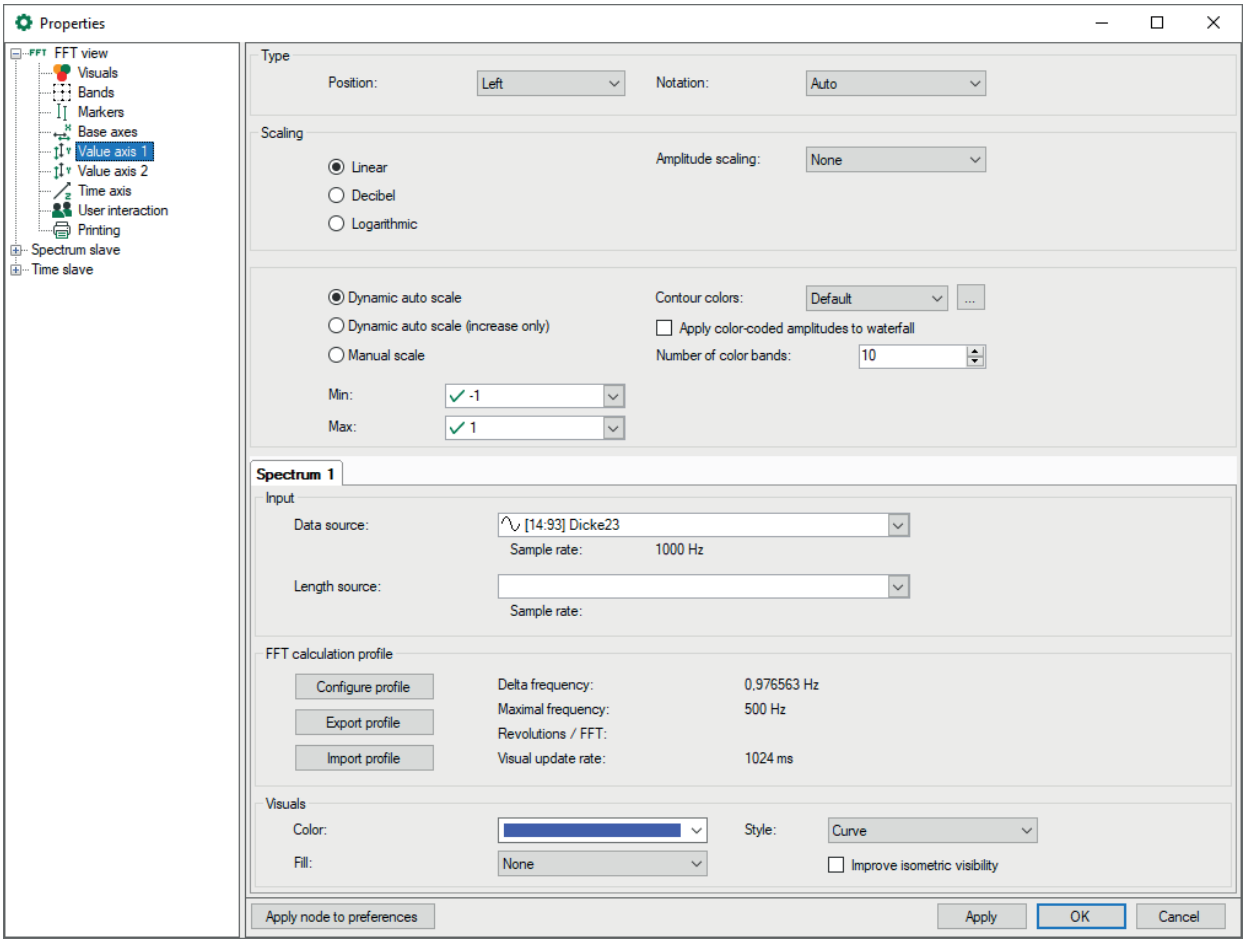
Note

Axis = 1/scaling factor



7.10.5 Value axes

A value axis can contain several spectra. Using the legend, you can change the value axis used by a spectrum by changing the sequence of the signals. A value axis can be deleted via its context menu. This also deletes all spectra on this axis. You can also display the settings for the value axis via the context menu.



Settings for type, scaling and view correspond to the usual settings in *ibaPDA* and are self-explanatory.

Scaling

Linear, decibel or *logarithmic* can be set as scale type. This scale type is applied to the appearance of single spectrum, waterfall and contour.

Amplitude scaling

Depending on the requirements for the visualization, it may be useful to either emphasize or suppress the amplitudes for the display. The following methods are available:

- Peak-to-peak
  Amplitude values are multiplied by a factor of two
- RMS
  Amplitude values are divided by the root 2, thus moving closer to the RMS value.
- Percentage of signal
  The scaling of the amplitude values is defined as a percentage of a signal. Select the signal in the *Reference signal for percentage* field.

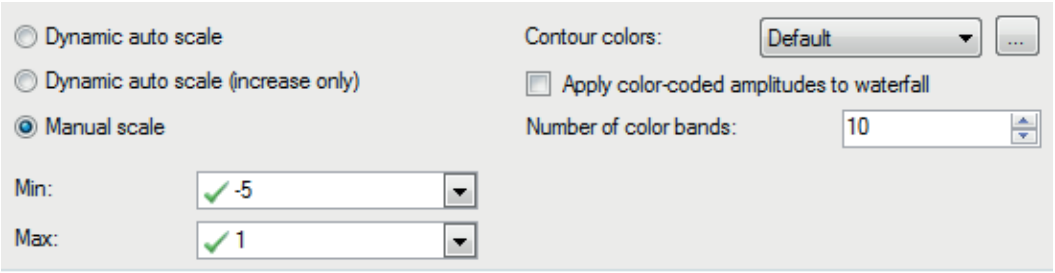
- Percentage of band peak
  The scaling of the amplitude values are based on the peak in a specific band. Enter the band number in the *Band number for percentage* field.

Scaling mode

- Dynamic auto scale: if you enable this option, the scaling is always adjusted to the highest and lowest signal amplitudes in the graph (in both directions).
- Dynamic auto scale (only for enlarging): When you enable this option, the scaling is continuously adjusted with the highest signal amplitudes. If the amplitudes go beyond the trend view again, the scaling still remains unchanged.
- Manual scale: if this option is selected, the Min and Max scale values can be manually entered or selected from the field next to it. Beside a static value, you can also use any known measured or virtual analog signal configured in the I/O manager.

Note

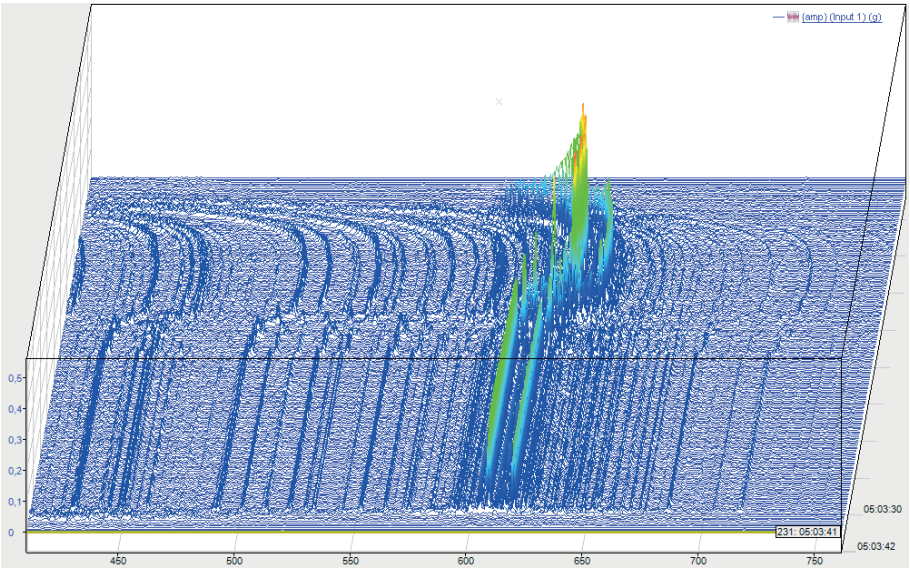
If *Decibel* is selected, the values of the manual scale nevertheless relate to the linear axis. The resulting decibel values are shown next to them.



Color settings

You can choose one of the prepared color schemes for the contour colors (default, grey, jet-white, jet or heat). If you do not like any of the offered color schemes, you can design your own color schemes. To do this, click on the <...> button and the *Manage perspectives* dialog opens. Here you can define new color schemes or change existing ones.

In addition, the colors of the contour view can also be applied to the waterfall view. To do this, enable the *Apply color-coded amplitudes to waterfall* option. The number of color bands defines the color resolution. A maximum of 50 color bands is possible.



Note

If *Apply color-coded amplitudes to waterfall* is enabled, custom value bands are displayed only in the spectrum graph.

Spectrum x

By default, a *Spectrum 1* tab is available. These settings are used to process a new signal that is dragged into the FFT view. You can drag multiple signals into an FFT view. If the signals share the same value axis, you will find a separate tab for each signal or spectrum. In the properties, the settings for each spectrum can be changed individually. If each signal or spectrum has its own value axis in the display, each spectrum in the tree structure on the left gets its own node for the value axis.

No calculation profile can be configured in the FFT view of *ibaAnalyzer-InSpectra*. A profile can only be configured for the data acquisition in *ibaPDA* without *ibaInSpectra*. This option is only used for the visualization. The results cannot be acquired. The FFT calculation profile, however, is compatible with InSpectra profiles.

Input

- Select *Data Source* to specify the signal or InSpectra module to be displayed. If you have already dragged the signal into the display via drag & drop, the field is already filled in.
- You only need to enter a *Speed Source* if you want to perform speed-dependent analyses or work with the order spectrum.

FFT calculation profile

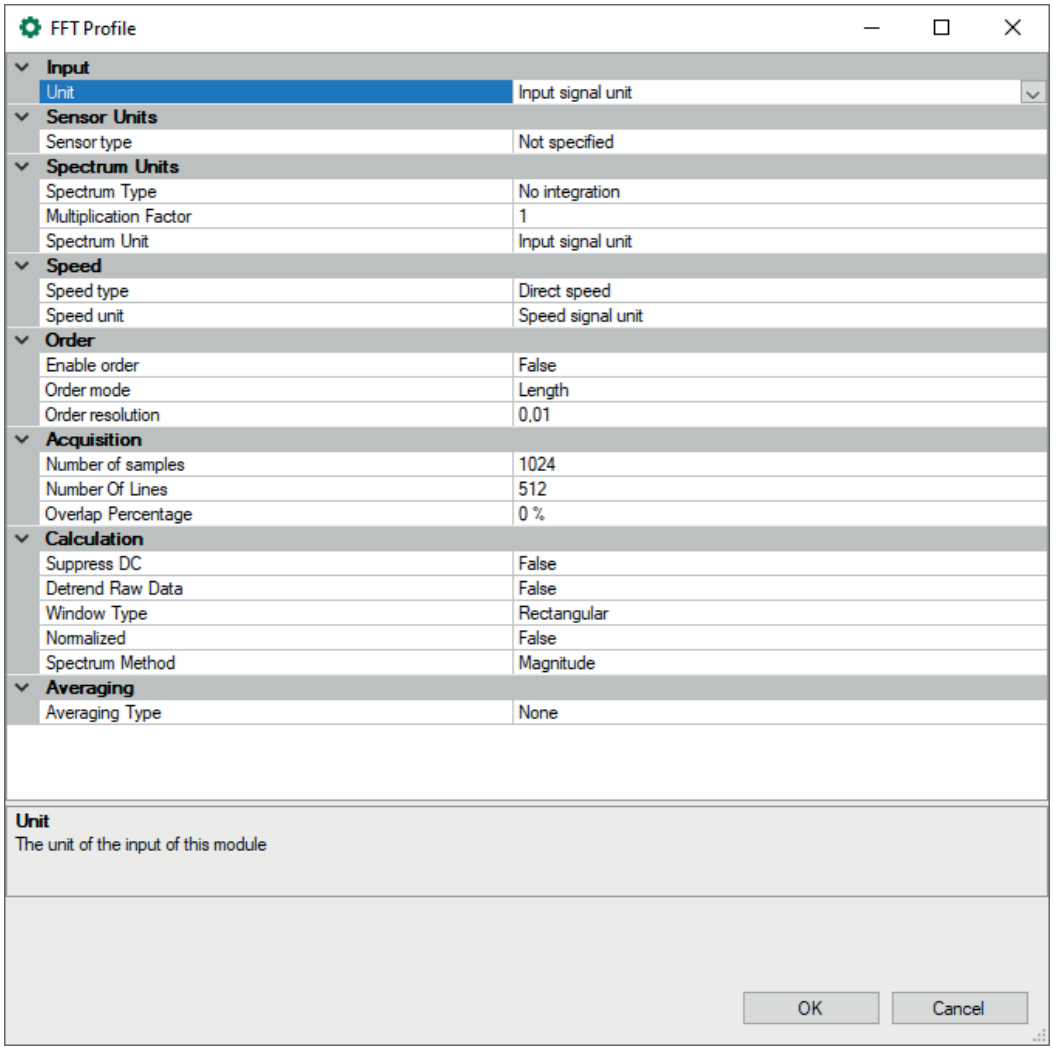
The way in which *ibaPDA* calculates an FFT is defined in so-called profiles. A profile is a collection of various parameters that are relevant to an FFT.

Each spectrum can be calculated with a different profile. You can define as many profiles as you wish and save them in the system via the export function. You can also import saved profiles into a spectrum.

In the profiles, parameters are defined, including

- Sensor data (important for vibration measurements)
- Spectrum type (e.g. integrate, differentiate)
- Speed data (important for order analysis)
- Number of samples and lines, overlap
- Basic calculation rules for the FFT (e.g. calculation mode, averaging, window type)

The button <Configure profile> opens the configuration dialog for profiles. The buttons <Export profile> and <Import profile> below can be used to export and import profiles.



The calculation parameters and their meaning are explained in the *ibaInSpectra* manual in chapter *Setting calculation parameter*.

The information next to the “Profile...” buttons describes the influence of the acquisition parameters:

- Delta frequency:
  Shows the frequency steps between the results of the division of maximum frequency by bin count.
- Max. update rate:
  Time required for update of the FFT view depending on bin count and overlap factor.

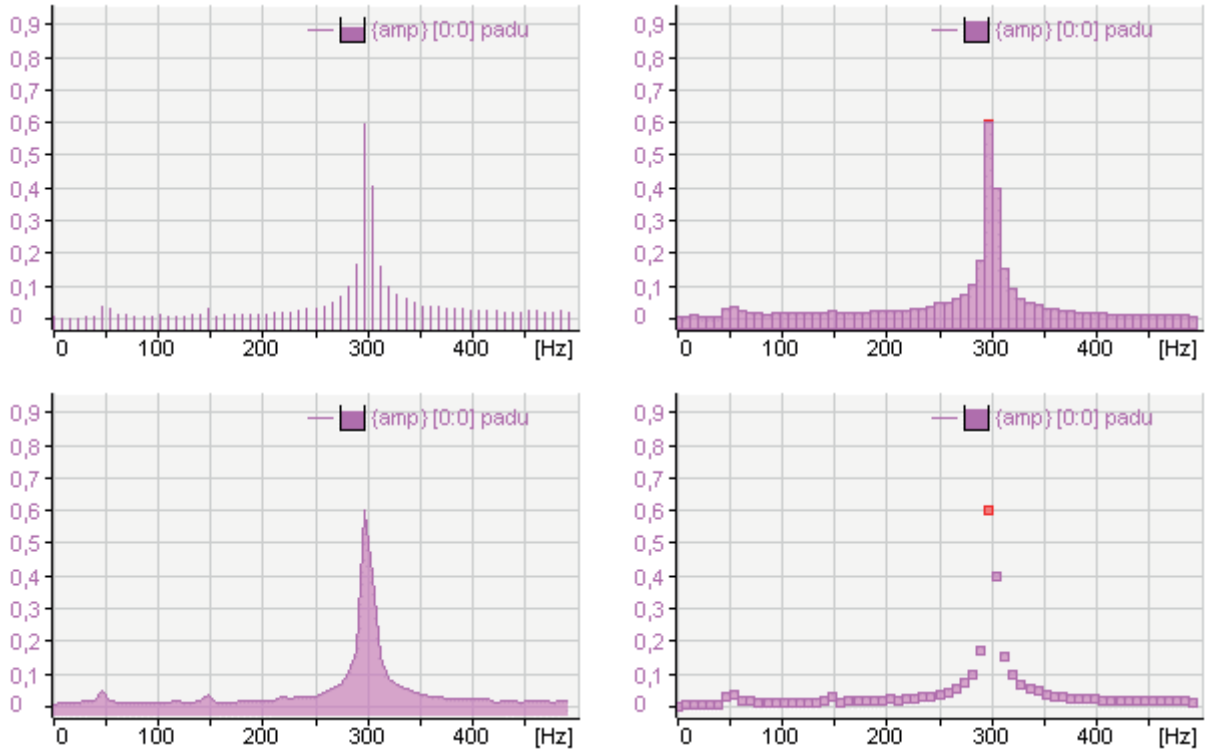
To avoid having to look at the properties to see the profile parameters, the display shows the *Spectrum Parameter Table*. This table is part of the FFT view and can be activated via the drop-down menu in the FFT view. The parameters from the calculation profile shown in the table can be defined in the *Spectrum Parameter Table* node in the FFT view properties. See chapter
↗ *Spectrum parameter table, page 73*

View

A spectrum can be visualized in four different ways:

- Lines
- Bars
- Curve or
- Dots

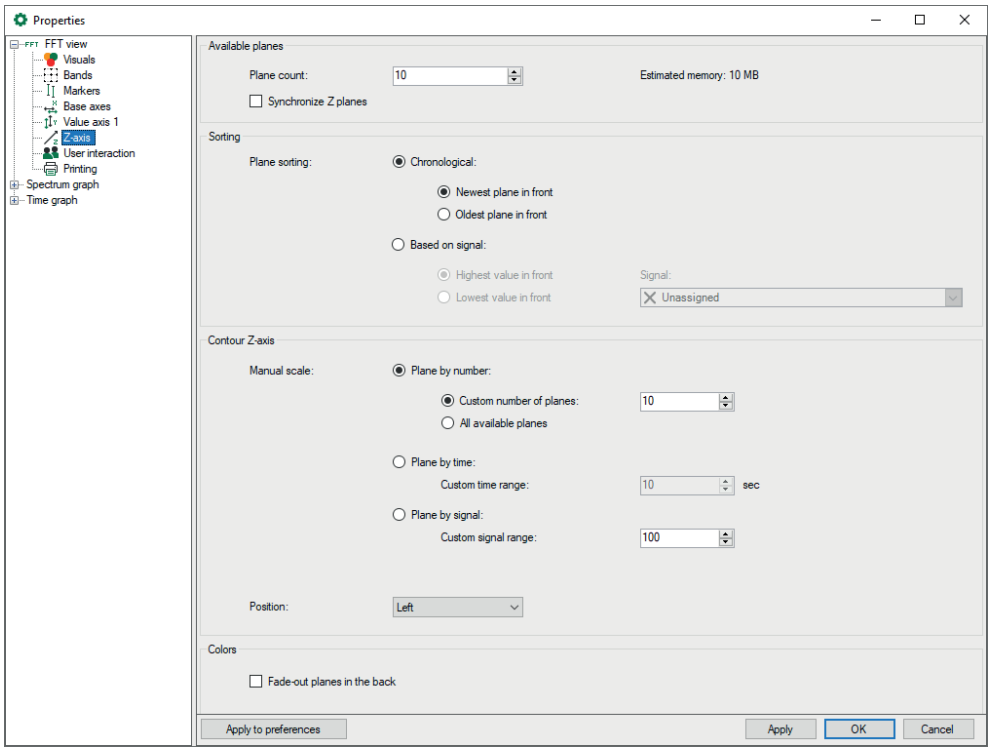
The inner area of a spectrum can be filled with a transparent or opaque color. The figure below shows the four types of spectra visualization, all filled transparently.



The option *Improve isometric visibility* is used to make spectra opaque. This makes some effects in the waterfall view more visible.

7.10.6 Z-axis

In the *Z-axis* node you can define the display options for the waterfall view.



Available planes

- Plane count
  Set the number of planes you wish to be displayed in the Z direction.
- Synchronize Z planes (option in *ibaPDA* only)
  If you use multiple spectra in an FFT view, the spectra move forward at their own pace by default, depending on their sample rate or bin count.
  With this option you can synchronize the advance rates of the Z planes across multiple spectra. With this option enabled, the FFT view will not allow a spectrum to advance over the Z planes until all spectra have generated a new FFT result. While the view is waiting for certain spectra to generate results, the other spectra keep showing their newest results on the front plane.
- Determine plane count automatically (option in *ibaAnalyzer* only)
  Number of spectra is detected automatically (max = 500).

Sorting

Here, you can select how the planes should be sorted:

- When choosing the chronological plane sorting you can select whether the the newest (default) or oldest plane is in the foreground.
- When choosing the sorting by a signal an average value of this process signal is calculated for each FFT. You can select whether the plane with the highest or lowest value should be in front. The sorting happens on the fly: Each time a new FFT comes in, all the planes are sorted again. This applies to the contour plot and the waterfall. You can select the desired signal in the field nearby.

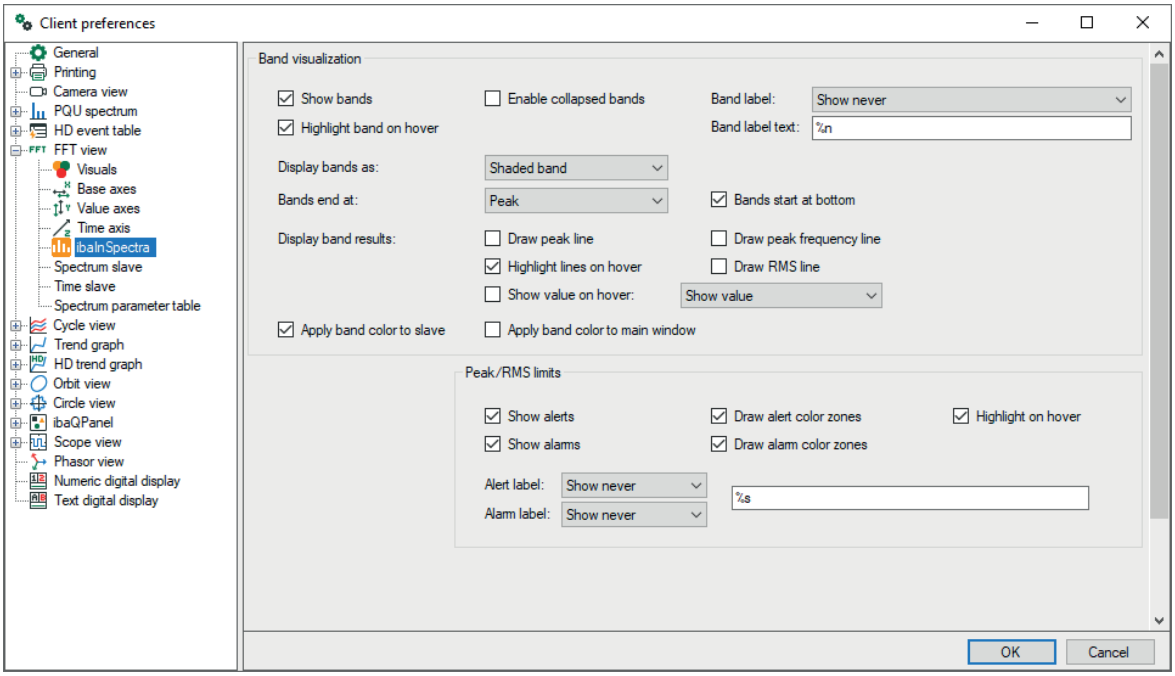
Contour Z-axis
Specify here the distance at which the planes are to be arranged:

- Plane by number
  The distance between the planes in the contour view is fixed. You can choose between a manually defined number of planes or display all available planes.
- Plane by time
  A new plane is displayed after a defined time. The distance corresponds to the time difference between the planes.
- Plane by signal
  This option can only be used if the plane sorting is *Based on signal*. The distance between the planes depends on the value of the process signal. The *Custom signal range* is the default range displayed in the contour plot. In the context menu of the time axis, there is an option *Auto scale* to scale the time axis so that all planes of that moment are shown.

Position
Specify the position (left or right) of the Z-axis.

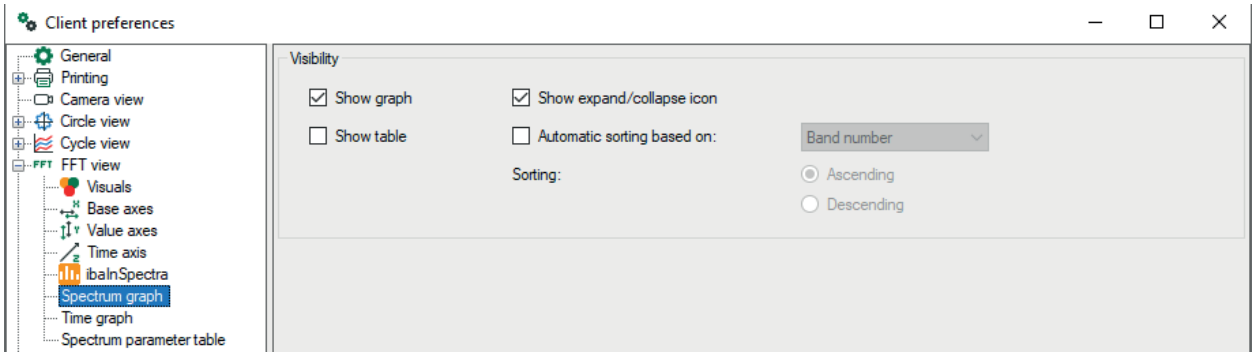
7.10.7 InSpectra

The dialog *Preferences* is the same as the properties dialog for InSpectra bands. For a description, see chapter ↗ *Bands, page 90*



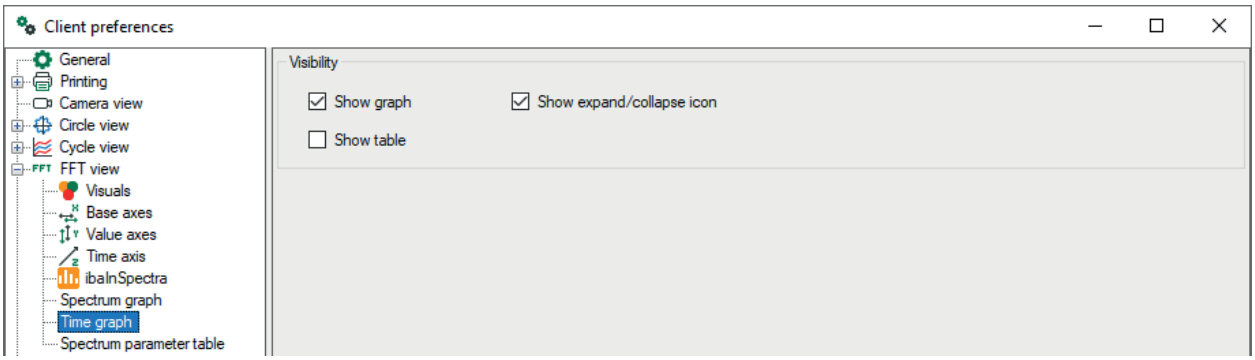
7.10.8 Spectrum graph

The *Preferences* dialog only offers setting options for displaying the chart and the table. Optionally, the table can automatically be sorted according to a selected column.



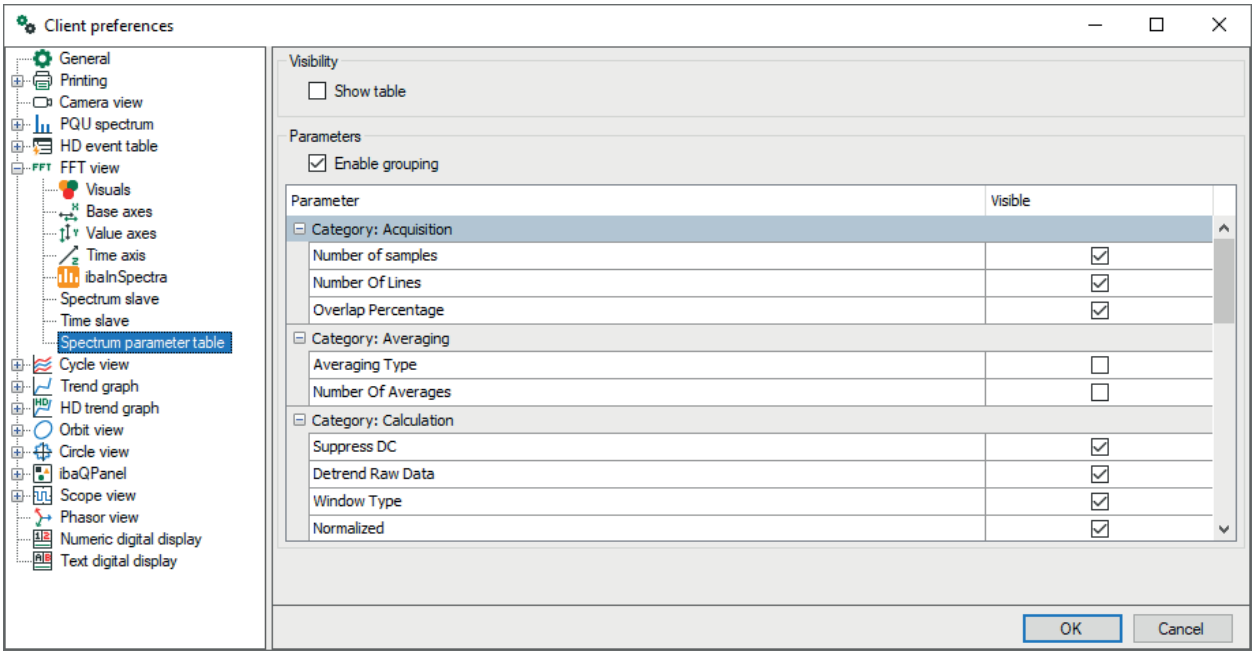
7.10.9 Time graph

The *Preferences* dialog only offers setting options for displaying the chart and the table.



7.10.10 Spectrum parameter table

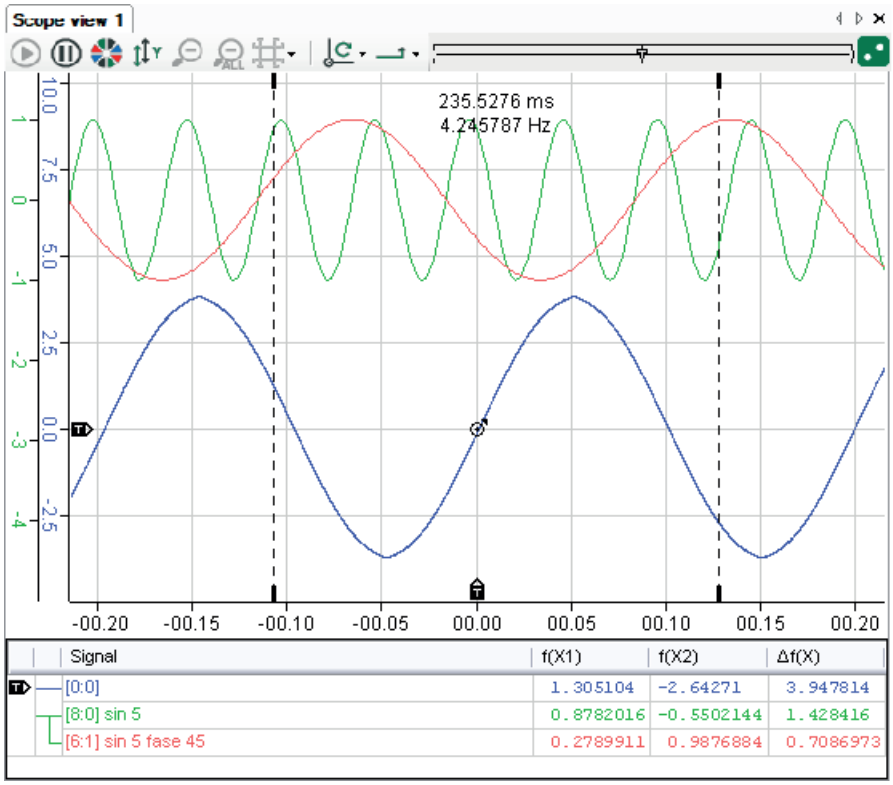
The settings for the spectrum parameter table are the same in the dialogs of the preferences and properties. For a description of the settings, see chapter ↗ *Spectrum parameter table, page 73*



8 Scope view

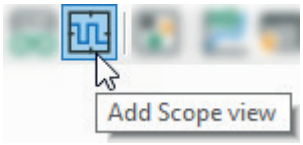
8.1 General

The scope view is an oscilloscope included as a standard view in *ibaPDA*. Each view supports one base axis and multiple signals on multiple value axes.



A new scope view can be added to the view by

- clicking the corresponding toolbar button



- or by using the *View – Add view – Add scope view* menu

Signals can be dragged from the *ibaPDA* signal tree onto the view to add them. If the signal is dropped on the chart, a new value axis will be created. To add a signal to an already existing axis, drop the signal on that axis. Every value axis will have the same color as the first signal it contains.

8.2 Trigger

The view's trigger is visualized on the chart by a little circle with a dot inside. If the trigger can be moved, trigger markers appear at the side of the chart, near the base and value axes. The trigger can be moved by dragging these markers.

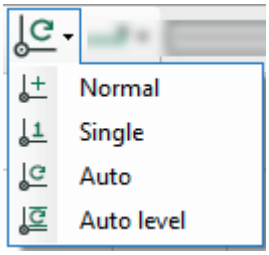
There are four triggering slopes:

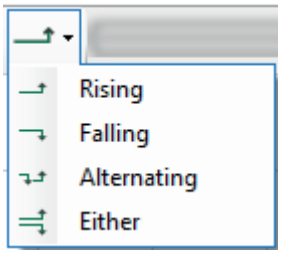
Edge type	Description
Rising	Trigger can only fire on the rising edges of the trigger signal
Falling	Trigger can only fire on the falling edges of the trigger signal
Either	Trigger can fire on both falling and rising edges of the trigger signal.
Alternating	If the trigger previously fired on the rising/falling edge, it will fire the next time on the falling/rising edge.

There are four trigger modes:

Trigger mode	Description
Normal	Trigger fires every time the trigger signal passes the trigger value
Single	Trigger fires when the trigger signal passes the trigger value for the first time.
Auto	Like normal mode, but also fires when a time-out expires.
Auto level	Like auto mode, but the trigger value is set to the trigger signal's center value

The trigger mode and slope can be set in the toolbar or in the "properties" dialog of the view.





To enable the trigger, click the "Play" button in the toolbar. To disable the trigger, click the Pause button. If the trigger is disabled, it will not scan incoming samples.

8.3 State indicator

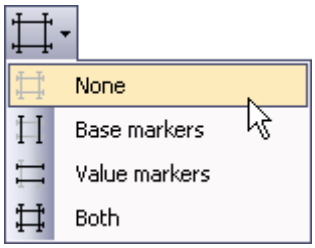
The state indicator shows the state of the trigger. If the trigger is enabled and samples are received, the state graphic will show 2 dots moving in a circle. If the trigger is disabled, the dots do *not* move in a circle. If a trigger fires, a circle runs through the middle of the dots connecting them. If the auto trigger fires, this circle is smaller, only touching the dots.

Scanning	Triggered	Auto triggered

Scope status indicator

8.4 Markers

Markers can be enabled or disabled in the toolbar or in the "Properties" dialog. There are two base markers and two value markers. These two marker types can be activated separately.



If a marker type is active, it is displayed on the chart and can be moved. More information on markers is displayed between the markers or in the signal table below the chart.

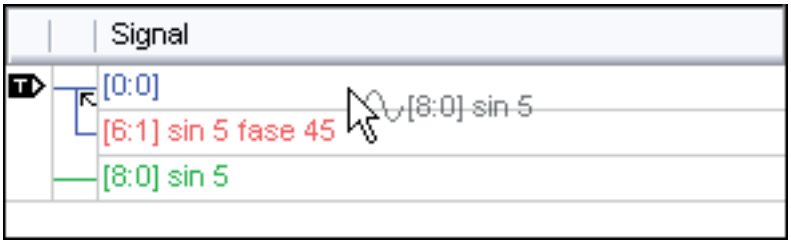
8.5 Signal grid

The signal table can be found at the bottom of the view. It displays the ID and name of each signal that has been added to the view.

Signal	f(X1)	f(X2)	Δf(X)	Y1	Y2	ΔY
[0:0]	2.79194	1.25627	1.53566	7.97447	-1.50742	9.4819
[8:0] sin 5	-0.39137	-0.87570	0.48432	-0.07347	-3.37960	3.30612
[6:1] sin 5 fase 45	0.89100	-0.75011	1.64111	-0.07347	-3.37960	3.30612
[3:0] 50 Hz	-2.79903	0.58778	0.58778	2.48046	-2.55859	5.03906
[3:8] 10 Hz	0.58778	0.12533	0.46245	2.48046	-2.55859	5.03906

To the left beside the signal name column, there are two more permanent columns: The trigger column and the axis column. The trigger column shows which signal is currently being used as trigger signal. You can change the trigger signal by dragging the trigger icon onto another row. The second column visualizes a small list for every value axis in the view distinguished by different colors. Each list item connects the signals contained with the axis it represents. If the base markers are visible and the base axis is in time mode (X-T), the table shows the f *(X1)*, f *(X2)* and Δf *(X)* columns. If the base axis is in "Value" mode (X-Y), the grid shows the *X1*, *X2* and *ΔX* columns.

If the value markers are visible, *Y1*, *Y2* and *ΔY* are shown. You can click inside the "Name" column to shift a signal to another axis. If the signal being dragged is the only one on its axis, this axis will be removed. In order to remove a signal from an axis, drag it to any position, but not into another column. The signal is put on a new axis and added to the bottom of the signal table. In the figure below, the signal [8:0] is added between [0:0] and [6:1] of the blue axis.



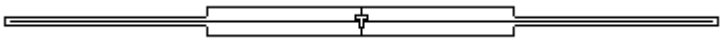
The signal table's height can be adjusted. When moving the mouse to the top of the table, a resize icon appears. To auto-size the table, double-click if the resize icon is visible.

8.6 Memory length

The scope view has a user-defined memory size. It can be set in the view's "Properties" dialog as milliseconds. This determines how many samples are buffered for all of the view's signals. The length is limited, depending on the time bases of the signals currently used in the view. So, by adding a new signal, the memory length might change.

8.7 Overview

This component is a visual representation of the memory length. It shows which part of the memory length is currently visible on the diagram. It also shows the position of the trigger.



The entire width of the overview visually represents the memory length. The large rectangle represents the currently visible part of the memory length. It can be dragged to the left or right to show other parts of the buffered signals on the diagram. If the cursor is above the rectangle, its size can be increased or decreased by scrolling the mouse wheel. The trigger can also be dragged onto a new base position.

8.8 Chart

The chart is the part of the scope view displaying the signals and grid lines. In the X-Y mode, a background image can be placed in the diagram. Furthermore, the chart provides a zoom function. You can draw a zoom rectangle within the diagram, when releasing it, it is zoomed in to the drawn range. When zooming in, the trigger will be disabled. The toolbar buttons can be used to zoom out to the previous level or to zoom out entirely. You can also press the Play button to zoom out and enable the trigger.

If a trigger fires, the chart will be updated with new signal data. The maximum update frequency of the chart is 20 Hz. As the visible part of the memory length gets larger, the update frequency gets smaller. The reduction degree depends on the number of signals, the width of those signals, the base axis mode and its value signal, and the memory length itself.

8.9 Axes

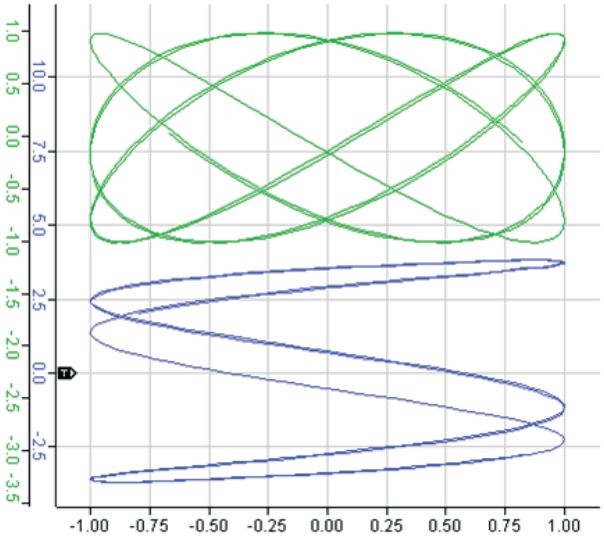
All of the axes on the view can be panned by dragging them with the mouse. You can also perform an axis center zoom (in and out) by scrolling the mouse as soon as the cursor is placed over the axis. If the scaling mode is set to manual, you can auto scale the axis by clicking the center mouse button.

The base axis can operate in two modes:

- time mode (X-T)
- value mode (X-Y)

In time mode, you can choose to center the axis values around the trigger or to use absolute time values.

Value mode requires a signal to be configured in the properties dialog. The base axis then uses the signal values and thus becomes a value axis instead of a time axis. The signal will not be added to the signal grid. The following figure shows a base axis in value mode.

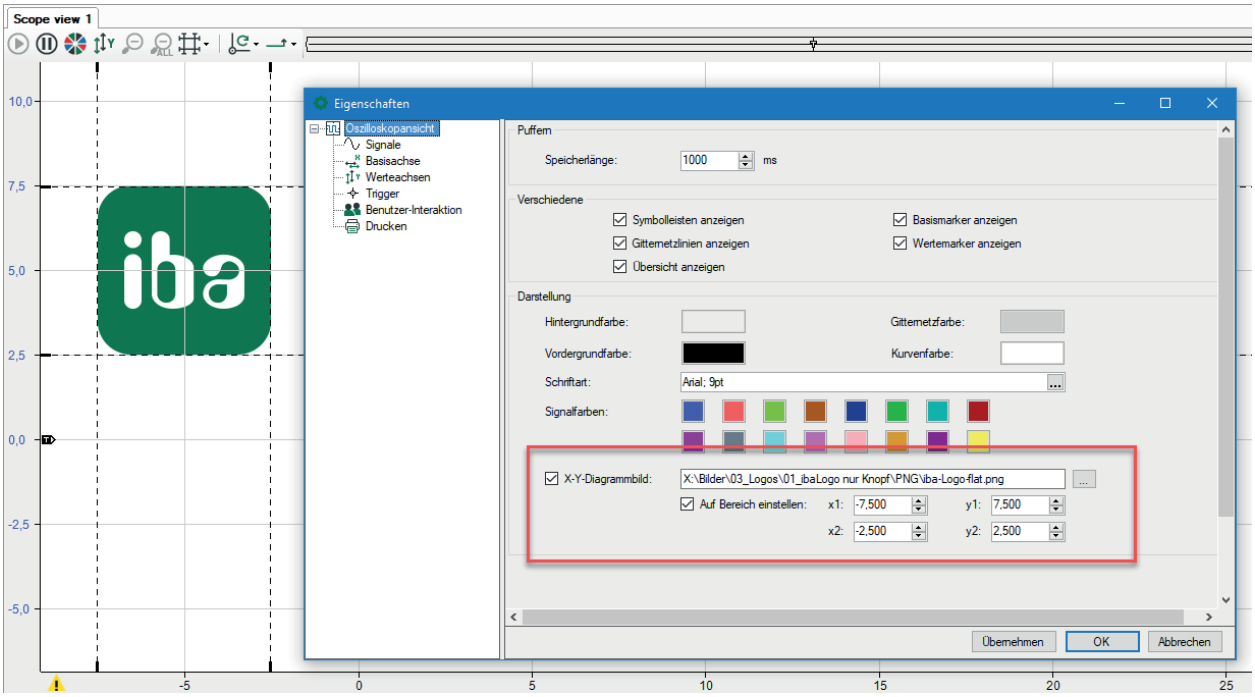


Interacting with the base axis in time mode changes the visible memory length (overview also changes). In value mode, the visible part of the memory length can only be changed by using the overview or by opening the properties dialog.

Note that the axis containing the trigger signal is always placed next to the chart.

8.10 Background image

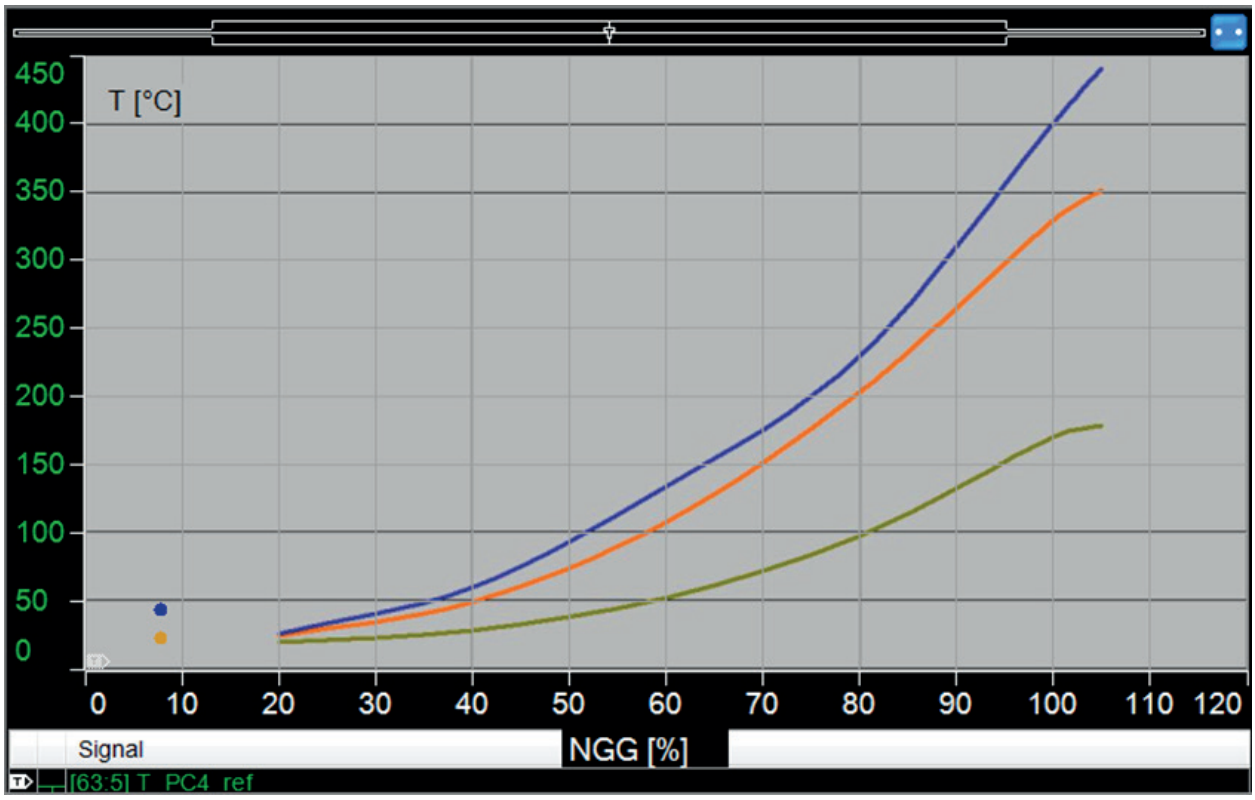
In the X-Y display mode, the oscilloscope view allows to use a graphic file as background image. In order to set up the background image, open the "Properties" dialog of the scope view via the context menu.



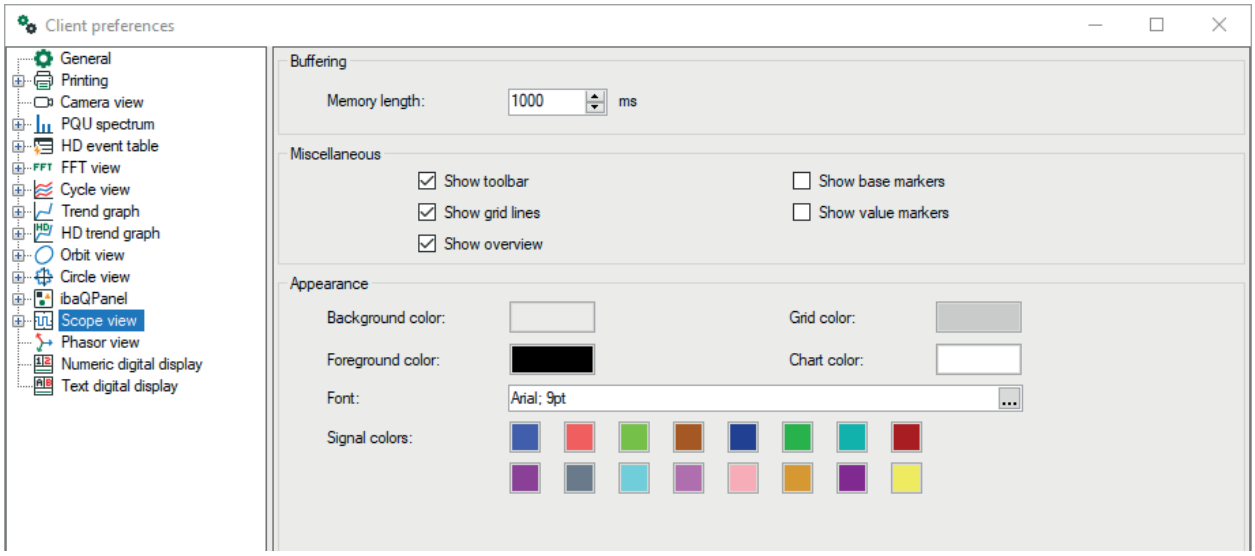
In order to enable/disable the image display, check/uncheck the *X-Y chart image* checkbox. Enter the path and file name of the graphic file in the corresponding field or use the browser button to select the file. The *Map to range* option permits to define size and position of the image in the background. By specifying the coordinates x1/y1 and x2/y2, you can map the 4 corners of the image with regard to the X/Y scales in the view. If the *Map to range* option is disabled, the background image will always be scaled to the size of the scope view.

Example

A field of characteristic curves (image) as a background for the measured data.



8.11 Properties and preferences scope view



Buffering

Memory length

This determines how many samples are buffered for all of the view's signals. The length is limited, depending on the time bases of the signals currently used in the view. So, by adding a new signal, the memory length might change.

Miscellaneous

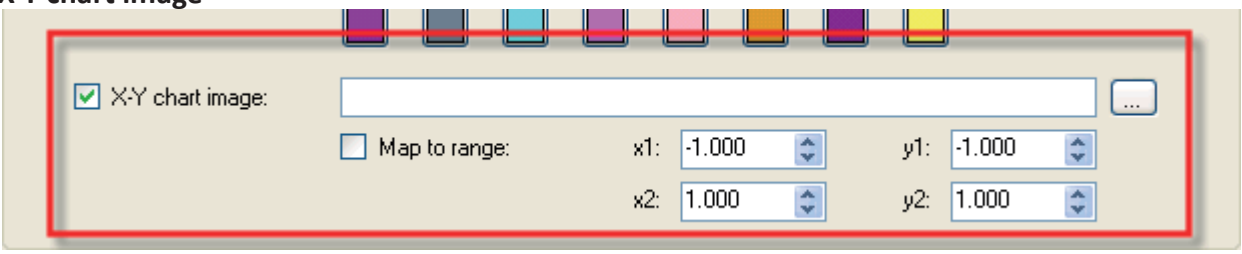
These settings determine whether the named view elements are shown or hidden.

Appearance

Setup your preferred colors for the different parts of the scope view, similar to the trend view.

In the "Properties" dialog, there is another option available among the display settings.

X-Y chart image

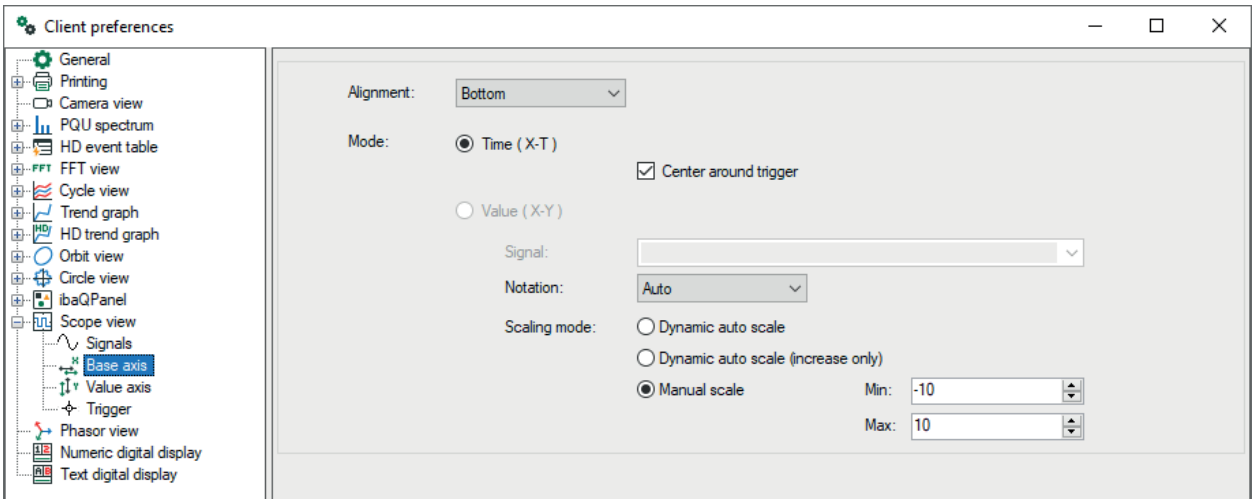


- In order to enable/disable the image display, check/uncheck the "X-Y chart image" checkbox.
- Enter the path and file name of the graphic file in the corresponding field or use the browser button to select the file.
- The "Map to range" option permits to define size and position of the image in the background. By specifying the coordinates x1/y1 and x2/y2, you can map the 4 corners of the image with regard to the X/Y scales in the view.
- If the "Map to range" option is disabled, the background image will always be scaled to the size of the scope view.

8.11.1 Signals

As for trend graphs, you can set up basic properties for displaying analog and digital signals at this stage. On the "Signals" branch in the "Properties" dialog of a scope view, you can also change the color of a signal.

8.11.2 Base axis



The base axis is always the horizontal axis in a scope view.

Alignment
Select your preferred position of the base axis scale from the pick-list.

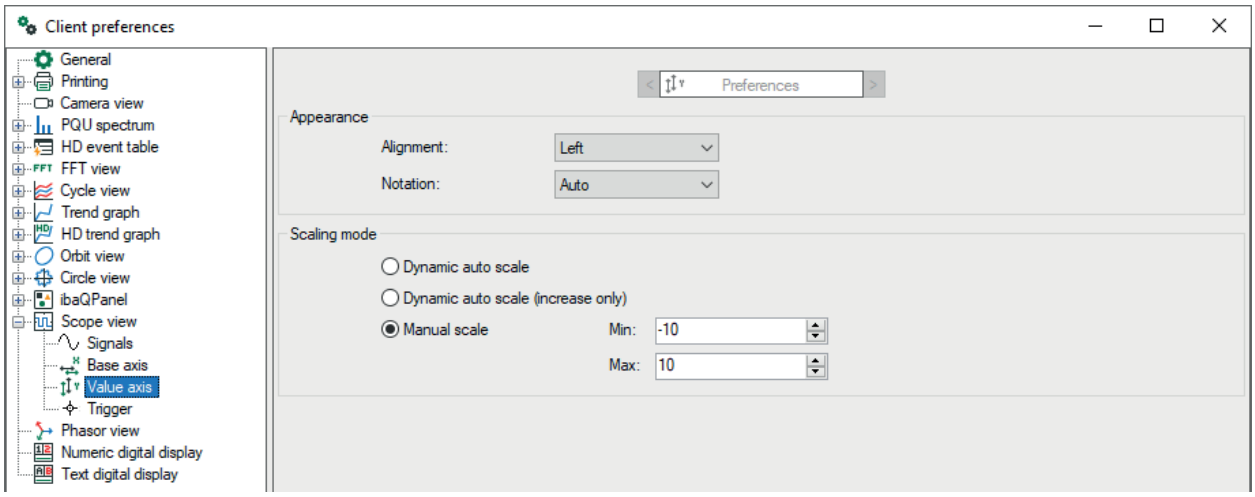
Mode
In the preferences, only the "Time (X-T)" mode can be enabled.

The "Center around trigger" option, if enabled, always keeps the trigger level(s) in the middle of the view.

The "Value (X-Y)" mode is only available in the "Properties" dialog. It can be used for X-Y charts. The "Value" mode requires the selection of analog signals whose values are used for the base axis instead of time.

Like for trend graph view, you can set the notation and scaling mode.

8.11.3 Value axis

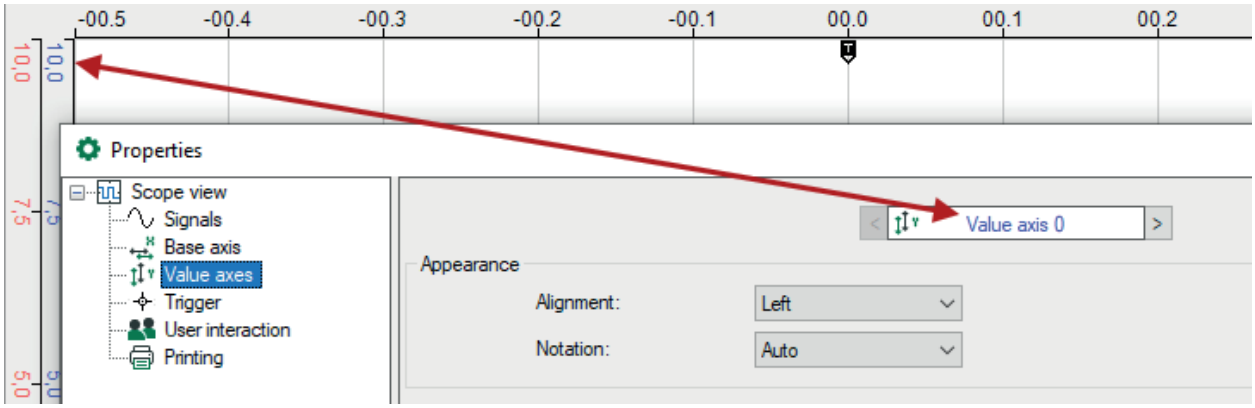


In the "Preferences" dialog, you can set the general options for appearance and scaling mode.

The value axis is always the vertical axis of a scope view.

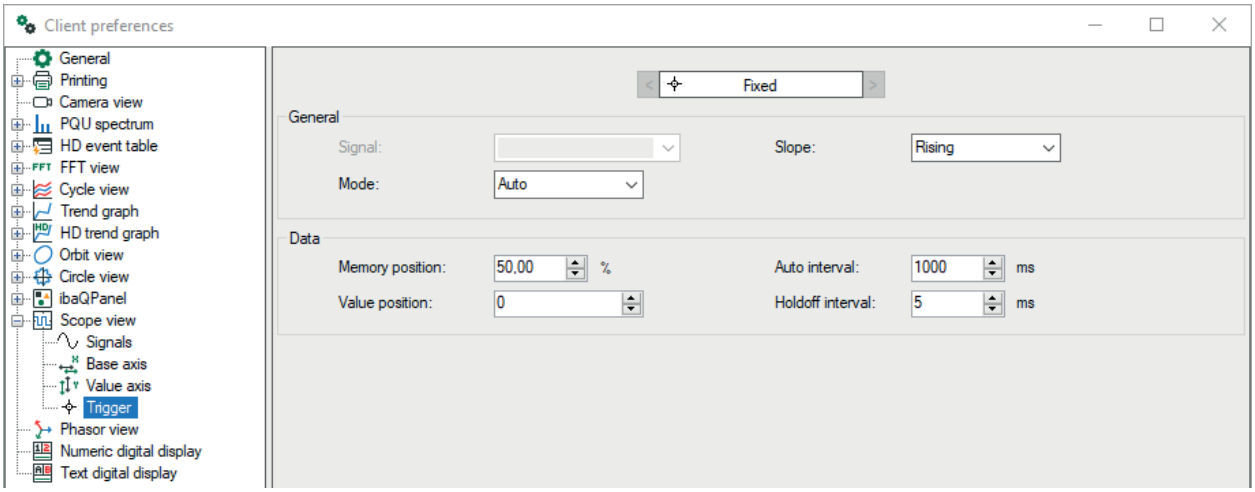
The value axis selection control on top of the dialog is disabled in the preferences. It is only available in the "Properties" dialog.

Value axis selector



The value axis selector shows the number of the value axis the settings below are related to. In case you have multiple value axes in the scope view, you can select the value axis you would like to set up with the arrow buttons on the left and right side of the selector. The color helps to identify the axis.

8.11.4 Trigger



In the "Preferences" dialog, you can set the general options for the trigger, such as the mode (normal, single, auto and auto level) and trigger edge (rising, falling, both and alternating). The signal selection is disabled. The signal for controlling the trigger can be selected in the view's "Properties" dialog.

Data

Memory position
The memory position is a percentage determining the trigger base position, measured from the left side of the view to the right. The default value is 50 % and puts the trigger in the middle.

Auto interval
The auto interval is used in "Auto" and "Auto level" trigger modes. If the trigger is fired in one of these modes, it will take up to the given amount of milliseconds before the next trigger fires. Of course, this trigger can also be fired earlier, as up to this moment the trigger keeps on scanning input samples.

Value position
The value position determines the trigger level. You can set a value by using the spinners (going in integer steps) or manually enter any value in this field.

Holdoff interval
The hold-off interval is the amount of milliseconds after the trigger fired where the trigger does not scan any incoming samples. So, in this period the trigger cannot fire.

9 ibaQPanel

ibaQPanel is a licensed software add-on for *ibaPDA*. However, a mini license for two display objects comes with every *ibaPDA* standard version.

Other documentation

For a full description of *ibaQPanel*, please read the corresponding manual for the software product *ibaQPanel.*

9.1 General

ibaQPanel provides an extended range of objects for visualization of measured values, which have been acquired by *ibaPDA*.

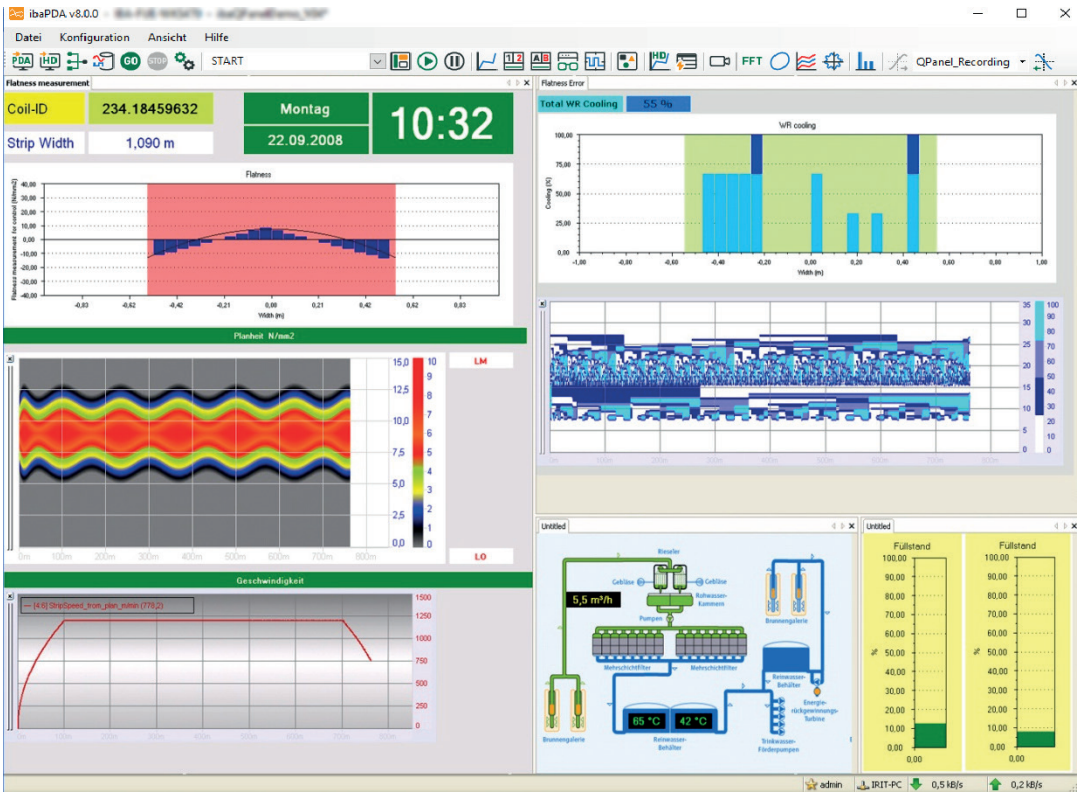
For every computer where the QPanel should be displayed, an *ibaQPanel-Add-on* license and an *ibaPDA* client license are required. Both single and multi-user licenses are available.

The display objects may be arranged in one or more layers and placed anywhere on the screen with a user-defined configuration in terms of size, color, font, etc.

Both data from an *ibaPDA* server and data from an *ibaHD-Server* can be displayed in the view.

Images may be used as background graphics and thus enable the user to create HMI-like screens.

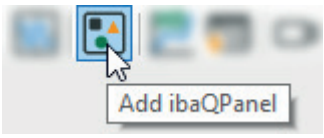
The following figure shows an example with different typical QPanel items.



9.2 The ibaQPanel view

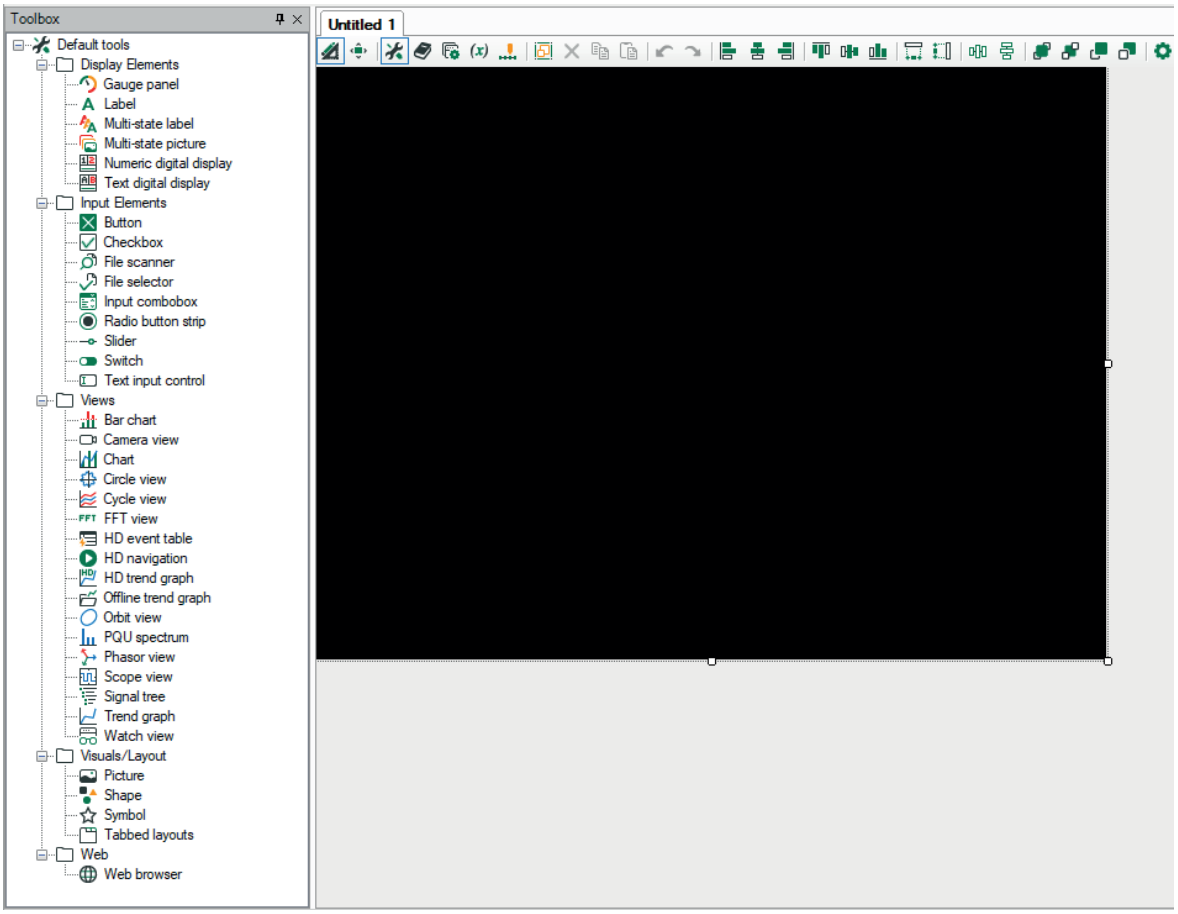
A new *ibaQPanel* view can be added by

- clicking the corresponding toolbar button



- or by using the *View – Add view – Add QPanel* menu

The new QPanel view opens in the design mode and the toolbox containing all types of display objects is displayed.



In a first step, resize the black background according to your needs, as it provides the area where you can place the objects. You may also change the background color or choose a background image.

Then, start to drag the display objects you need out of the tool box into the panel and set them up.

9.3 The display objects

Views		
Object	Function	Application
Bar chart	Display of signal values as single-bar or multi-bar charts Color change for value ranges, dynamic limits, polynomial calculation for profiles	General value display Fill levels Profiles
Chart	Multifunction object for the combined display of trend curves, graphic objects, markers, harmonics and texts as well as several base and value axes	Machine presentation, layered presentation
FFT view	Spectrum slave graph of a signal	Frequency analysis Chatter monitoring
HD event table	Display a list of events based on an event-based HD data store	Error and alarm monitoring; search for process events
HD navigation	Parent toolbar for controlling multiple HD views	Space-saving control of multiple HD trend graphs
HD time periods table	Display a list of predefined time periods based on a time-based HD data store	Facilitate the analysis of recurring process steps or procedures as well as regular periods (day, week, shift)
HD trend graph	Display historical data in a trend view; connection to any HD-Server in the network	Comparing data from the past with current values
Camera view	Display a camera view of an ibaCapture-CAM server on the screen Scenario player screen for event-based camera switching.	Monitoring Process control Troubleshooting
Circle view	Relevant parameters for the circle view are derived from the module settings.	Visualization of the results of the InCycle expert module and the InCycle auto-adapting module for rotating processes
Offline trend graph	Display a trend graph from a data file (no live data)	Comparison of older and current measured values, comparison of current values with reference curve
Orbit view	The orbit itself is the appearance of the X/Y data pair during a (or several) revolution(s) of the shaft	Results of the orbit calculation are given as analog values of the respective InSpectra orbit module
Scope view	Display signals in a digital oscilloscope	Commissioning Troubleshooting

PQU spectrum	Display of a complete PQU spectrum module	Spectrum presentation of electrical values with up to 50 harmonics (power quality)
Signal tree	ibaPDA/ ibaHD signal tree	Source for signals to be added to a view by drag and drop
Trend graph	Display signal values in a trend graph X axis time or length-based; automatic 2D top-down view on vector signals	General value display False color display of profiles (temperature, flatness, etc.)
Watch view	Display window for signal values	Monitoring selected signals regarding change of value or status
Phasor view	Display of a complete PQU phasor module	Phasor presentation of electrical values in a power grid (power quality)
Cycle view	Display of special graphs and tables	Visualization of the results of the InCycle expert module and the InCycle auto-adapting module

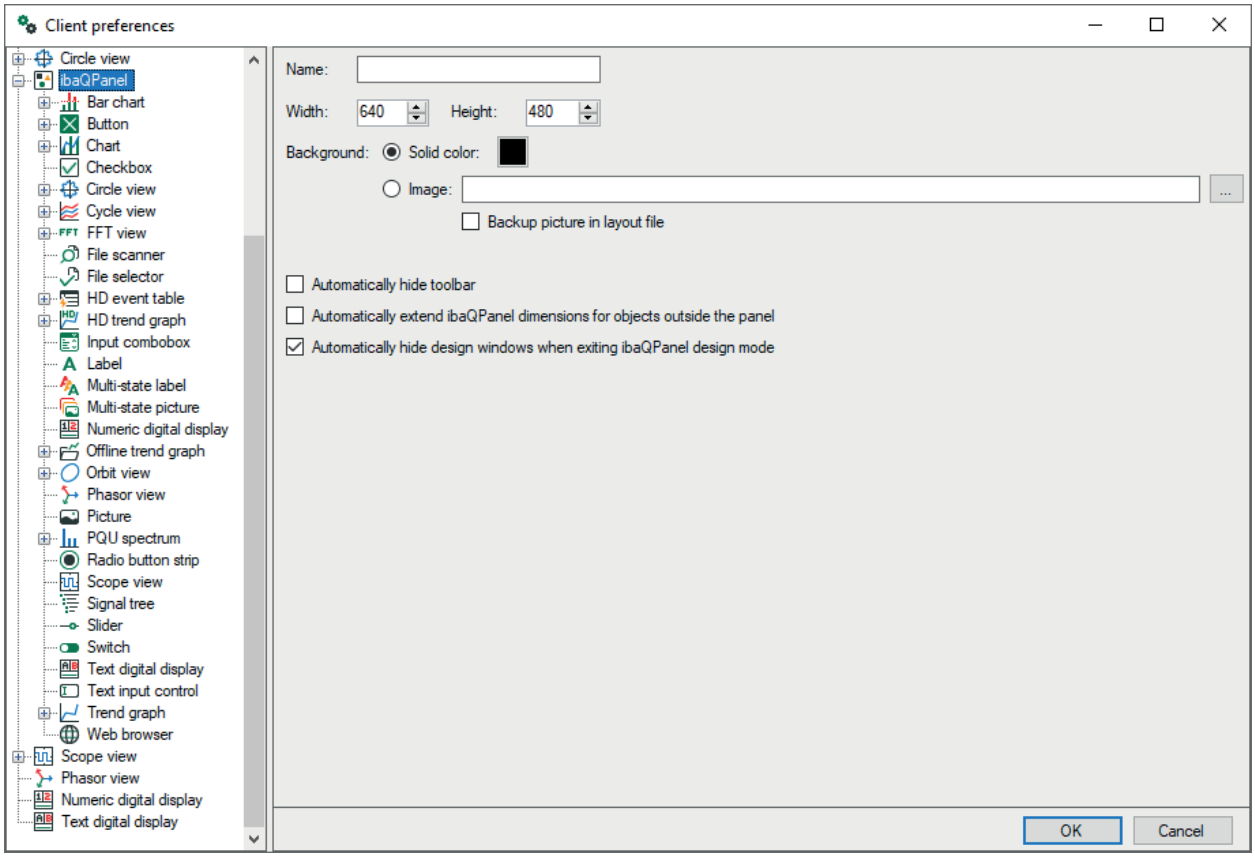
Display elements		
Text digital display	Display of text signals	Large and clear display of text in the style of an LED display
Numeric digital display	Display of signal values as numbers, maintaining the positive and negative peak value (resettable)	Large, clear display of signal values
Gauge panel	Display of signal values on a classic analog-style gauge Circular, linear and numerical gauges available; numerous design options, free set-up of shape, color, scales and pointers.	Highlighted value display Replaces traditional displays
Multi-state picture	Display of various graphics, depending on the signal value	Status display Security warning
Multi-state label	Convert a signal value into different states of a label Color change depending on value range(s), text in label	Status indication
Text field	Display signal values in a trend graph X axis time or length-based; automatic 2D top-down view on vector signals	Labeling of the displays Display of the text signal

Input elements		
Checkbox	Check box with check mark	Enabling / disabling options
Input combobox	Drop-down list with any number of plain text entries. By selecting a list item, a target signal is set to a defined value.	Manual setting of default values
Button	Execute commands using a button: Close ibaPDA client; Toggle view; Run command line; Enable application (*.exe); Print (QPanel screen); Digital pulse signal; Set signal value; Trigger value input control; Pause (stop one or all views); Start (continue one or all views); Load layouts from server; Save layouts on server; Open user management; Connect to ibaPDA server; Connect to ibaHD server; Change language; Log out from ibaPDA server; Log out from ibaHD-Server; Change unit system; Switch ibaPDA user; Switch ibaHD user; Commands can be run manually or via signal edges; button text and graphics can be adjusted.	Control of QPanel display Execution of other applications or scripts
File picker	Shows the contents of a folder in order to select a file. Name of the selected file is written into a text signal.	Interactive load of a data file in an offline trend curve
File scanner	Monitors contents of a folder for automatic selection of the last, second to last, third to last (and so on) file. Name of the selected file is written into a text signal.	Automatic display and update of the last x data file(s) in offline trend curves
Numeric up/down	The numerical value of an analog destination signal can be changed either by clicking buttons (+/- or up/down), turning the mouse wheel or typing in.	Quick, safe and easy way to change a value
Option field	Group of radio buttons	Selection of alternative options, operating mode selection

Switch	On/off switch; switch appearance can be determined by graphics file. Separate graphics for ON, OFF, disabled ON and disabled OFF state possible. Digital target signal.	Switch functions on/off
Slider	Slider's appearance (scale style, orientation, color of background/handle, min/max values etc.) can be configured.	Control element to display values within a scale
Value input control	String or value entry for a text signal or numeric signal. This control replaces the former text input control. There is a conversion function available to convert text input controls in older QPanels into value input controls.	Signal value or additional information on data file, printout, further applications, etc.

Visuals/Layout		
Image	Embed a graphic object, optionally statically, dynamically (image file name with wildcards) or by order (controlled by folder monitoring)	Process or system presentation
Shape	Basic geometric shapes (line, rectangle, ellipse, polygon); filling and line configurable, optional flashing mode	Simple shapes, animation
Tabbed layouts	Summary of multiple ibaQPanel views and layouts with control over tabs.	Clearer and simpler operation of multiple ibaQPanel layouts
Symbol	Integration of vector graphics (.svg) with option to change the line color and thickness, fill color, rotation and scaling	Use of vector graphics
Web		
Web browser	Display web pages, PDFs and other browser-fitting files; download of files	Display data from ibaDaVIS

9.4 Properties and settings Qpanel view



As described in ↗ *ibaQPanel, page 117*, the *ibaQPanel* view is the base for an HMI-like screen design. Beside some special QPanel objects, like multi-state label, multi-state picture, button, label, bar chart, gauge panel and text input control, you can add all other types of views to a QPanel view. Consequently, there are preferences and properties settings for these objects available within the QPanel preferences and properties.

Basically, you can refer to the descriptions of preferences and properties for each view type in the chapters before.

However, some standard views, like the trend graph view, have gained more options and capabilities in the QPanel environment. Only the QPanel-specific features are explained hereinafter.

As for all views, the difference between preferences and properties of a view is mainly the unavailability of signal selection in the preferences.

Name
Choose and enter any name for the Qpanel.

Width and height
This is the size of the QPanel background. All QPanel elements must be placed within this area. The default size is 640 x 480 pixels. The size can be adjusted according to your needs. If you plan to design a full screen QPanel, make sure that width and height match the figures of your monitor.

Background
You can set a solid background color or use an image (bitmap) instead.

Note that the image size should fit the QPanel background size (given in pixels). You cannot resize the image in ibaQPanel.

If the image is smaller than the background, it will be tiled.

If the image is larger than the background, you will just see a part of it.

Automatically hide toolbar
If this option is chosen, the toolbar only appears in design mode when the mouse moves over the upper limit of the window. When fading in, the toolbar slides across the panel; the panel itself, however, will not be moved to the bottom.

Automatically extend QPanel dimensions for objects outside the panel

If you copy elements from one panel to another, they will be displayed in the target panel at the same position as in the source panel. If the target panel is too small, the element will be shifted to the border or is not visible anymore. If this option is enabled, the target panel will be automatically enlarged so that the copied element fits the panel.

Automatically hide design windows when exiting ibaQPanel design mode
This option is only available in the preferences.

If enabled, the window with the toolbox will be closed automatically on exiting the design mode. Enabling this option may be helpful if you have to switch over frequently between design mode and display mode during the creation of a QPanel.

Other documentation

More detailed information about the functions, settings and properties of each QPanel object can be found in the product manual of *ibaQPanel*.

10 ibaCapture camera view

The *ibaCapture* system consists of hardware and software components and is used for the synchronized recording of measured data and video capture. *ibaCapture* works closely with *ibaPDA*.

Note

The basic license for the *ibaCapture* server and the license(s) for the permitted number of cameras and image(s) are provided on the dongle of the *ibaCapture* server PC.

If you have subsequently purchased the *ibaCapture* system for an existing *ibaPDA* system, please note the following:

For ibaCapture >= V4 (ibaCapture v4.x) and ibaPDA >=6.36:
The license for the *ibaCapture* interface in *ibaPDA* is available by default.

The permission for an *ibaPDA* system to access the *ibaCapture* server is licensed in the dongle of the *ibaCapture* server.

If you want to connect multiple *ibaPDA* systems with an *ibaCapture* server, for each connection you will need an *ibaPDA Interface Capture* license in the dongle of the *ibaCapture* server.

Other documentation

For a full description of *ibaCapture*, please read the corresponding manual for the *ibaCapture* software product.

10.1 General

The precise linking of measured data and visual information provides a new quality of process analysis when signal trends and video recording can be watched side-by-side in *ibaAnalyzer* in a synchronized manner.

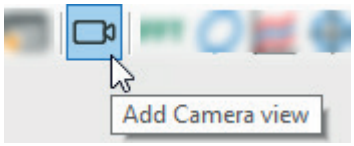
Using the *ibaCapture* player, the camera views of the cameras connected to the *ibaCapture* server can be displayed on the *ibaPDA* client screen allowing you to watch live video streams or video recordings.

In addition, depending on the type of cameras used, control signals for the cameras (e. g. PTZ cameras) can be configured in the I/O-Manager by *ibaPDA*.

10.2 The camera view

A new camera view can be added by

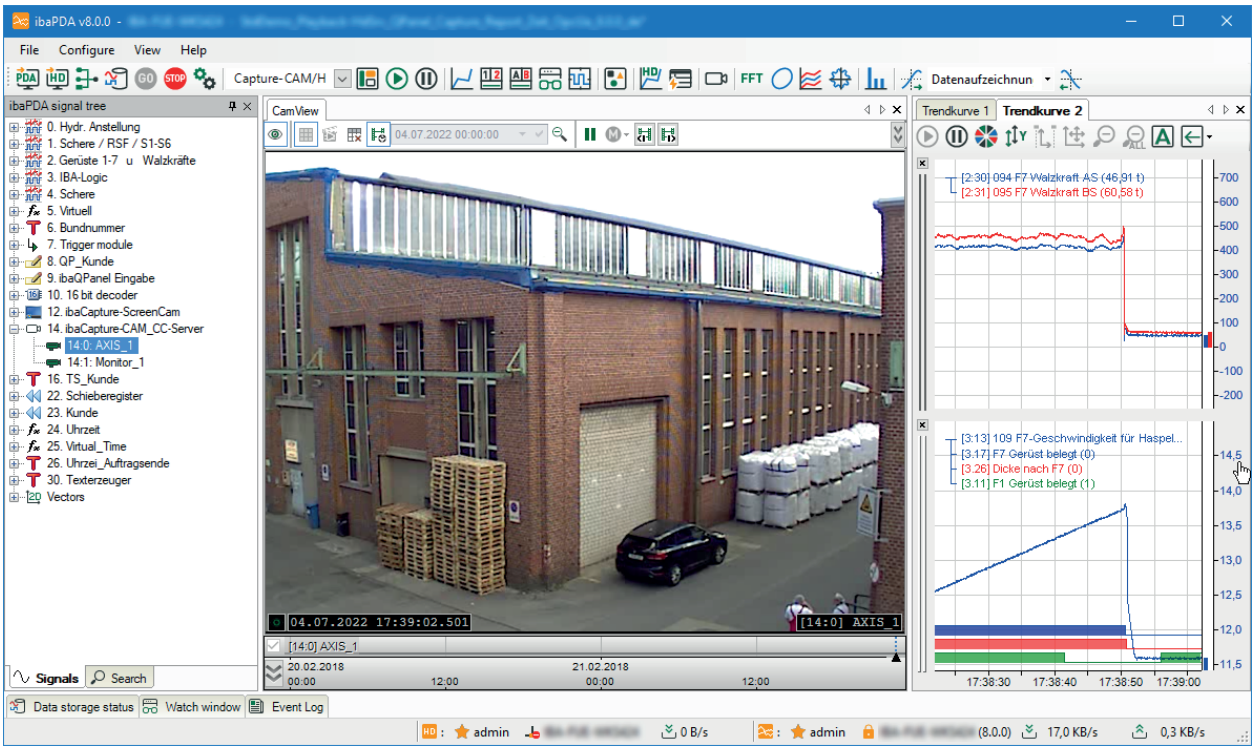
- clicking the corresponding toolbar button



- or by using the menu *View – Add view – Add Camera view*
- or by double clicking on a camera in the signal tree (showing immediately the actual picture of the respective camera).

If you choose one of the first two methods, you should add the cameras to the view by drag and drop them from the signal tree.

In *ibaPDA*, an *ibaCapture* server is assigned to a so called *ibaCapture* module. The *ibaCapture* modules are listed in the signal tree like all other modules. The cameras – up to 64 cameras per module – can take the same position like the measurement signals in other modules.



Generally, the camera view pane can be moved and rearranged like any other view in *ibaPDA* (dockable windows).

The tool used for video display is *ibaCapture* Player. In the camera view, you have different functions such as zooming or panning by using the mouse wheel, showing current images (live) or replaying recordings, etc.

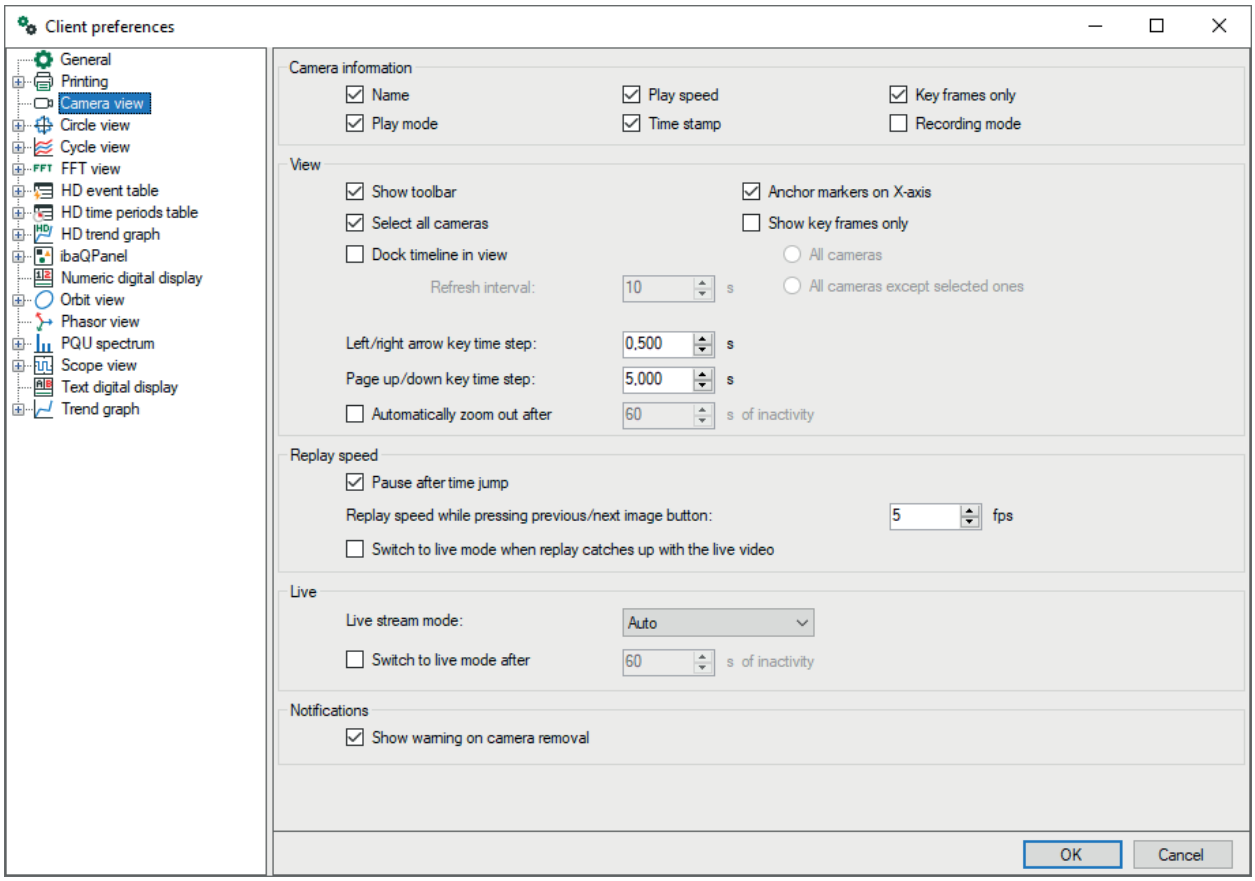
If you use an *ibaCapture* camera view either on your normal *ibaPDA* client or on an *ibaQPanel*, you can use text channels for overlay text displays.

For more information, see manual part 2, Using text signals.

Other documentation

For more information about settings and operation of the camera view, please refer to the *ibaCapture* product manual.

10.3 Properties and preferences camera view



ibaPDA uses the same software components for the camera views like the *ibaCapture* Player. Thus, you can set up the same properties and features in the preferences of the camera view like you do in the properties of a camera in the *ibaCapture* Manager.

There is no difference in the settings between the preferences and properties dialog.

Camera information

By setting/removing a checkmark, you can hide/show the five camera information elements.

- Name
- Play mode
- Play speed
- Time stamp
- Key frames only
- Recording mode

If enabled, the information is displayed at the bottom of the camera view.

View

- Uncheck the "Show toolbar" option to hide the replay toolbar in the camera view.
- Enable the option *Anchor markers on X-axis* in order to place the play marker in the timeline of the camera view at a certain point of time. When zooming or panning the X-axis the marker sticks to the set point of time.
- Check "Select all cameras" to select and lock all camera windows.
  For doing so, use the symbol in the toolbar.
- Show key frames only
  In order to save system resources, you can switch the camera display to reduced pictures in the data flow, e.g. by enabling the "Key frames only" view mode. For that purpose, set a checkmark in the "Show key frames only" option field. This activates the two option fields "All cameras" and "All cameras except selected ones".
  Selecting the "All Cameras" option enables you to set the "Key frames only" view mode for all cameras.
  With the "All cameras except selected ones" option, all cameras except for those selected are set to "Key frames only" view mode, i.e. only the selected cameras show fluid motion.
- Check "Dock timeline in view" to display the video timeline on the bottom of the camera view window. Specify in the "Refresh interval" input field the update time of the timeline. Do not set the update time too low in order to avoid unnecessary network load. For more details about the time line, please refer to the *ibaCapture* manual.
- Navigation key time step settings
  By entering values given in seconds for the <←/<→> and <Page up>/<Page down> keys, you can define short time step and long time step setting when using these keys for navigation on the timeline.
- Automatically zoom out after X s of inactivity
  If you enable this option, the camera view will return from a zoomed image back to the full image automatically, unless you operate mouse or keyboard within the specified time range (given in seconds).

Replay speed

In the section “Replay speed” you can enable the option *Pause after time jump* in order to switch the display in pause mode after performing a time jump. The playback then will not start immediately but wait for your start command.

In addition, for confirming the (Next image) and (Previous image) buttons, you can set additional replay speed in images per second (fps).

If you enable the option *Switch to live mode when replay catches up with the live video*, the display continuous scrolling with the actual time at normal speed, after the playback has reached the current point of time. This can only be the case if the replay speed is higher than the original speed.

Live

The live streaming settings apply to the streaming protocols between the *ibaCapture* server and client. Clients can be *ibaCapture* Manager (camera view), *ibaPDA* camera view or *ibaQPanel* camera view.

For setting up the streaming protocol, you can choose between the following options:

- Auto (default)
  In auto mode, the system starts using the unicast UDP protocol, usually the best choice. If, however, too many frames get lost, the system automatically switches to TCP mode. This could happen, for instance, if the server is running in a Gigabit network while the client is connected over a 100 Mbit network.
- Unicast UDP
  Recommended streaming protocol
- TCP
  Recommended streaming protocol if UDP does not work satisfactorily.
- TCP Active
  Like TCP, if *ibaPDA* should establish the connection.

Furthermore, you can enable an option for the camera view switching to live mode automatically, unless you operate mouse or keyboard within the specified time range (given in seconds).

Notifications

You can optionally enable a warning message in the view, which appears when a camera has been removed.

11 HD trend graph

The HD trend graph is a special trend graph for HD data that lines up from a time in the past until the current moment. It has its own properties, feed and navigation controls and refers to a dedicated HD signal tree.

Only signals supplied by an *ibaHD server* and available in an HD signal tree can be drawn into an HD trend graph. The signal selection of the HD signal tree depends on the HD data storage configuration. The HD data storage is a special type of data storage for recording and representation of historical data. The optional "*ibaHD* server" add-on is required for this feature.

Other documentation

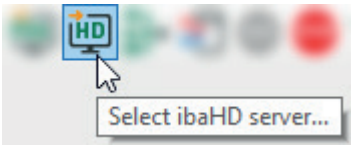
For more information about configuration of HD server and HD data storage, please refer to the *ibaHD-Server* manual.

11.1 Connecting to an HD server

Like the *ibaPDA* client must be connected to an *ibaPDA* server for displaying the normal signals, it must be connected to an *ibaHD-Server* for displaying historical data. The *ibaPDA* client can connect to one HD server at a time which can be installed on the same *ibaPDA* computer or on a remote computer in the network.

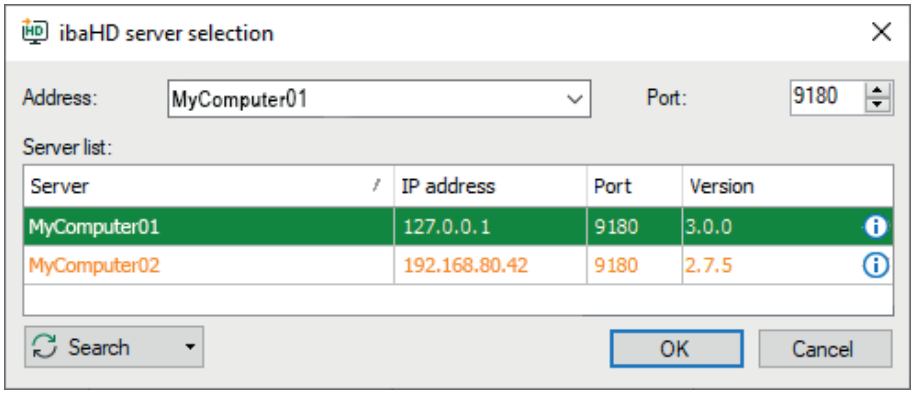
Before you can display historical data, you have to connect to the *ibaHD-Server* server as follows:

- Open the server selection screen either by clicking on the corresponding toolbar button



- or by using the *Configuration – Select HD server…* menu

The form looks like the *ibaPDA* server selection form. By default, the server is set to "Not connected". This means that *ibaPDA* will not connect to an *ibaHD-Server*.



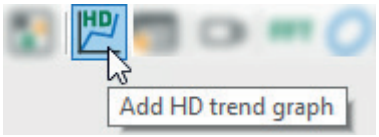
Use the <Search> button to scan the network for active HD servers and select a server from the table or type in a server address. Click the <OK> button to connect to the *ibaHD-Server*. If the HD signal tree has not been visible yet, it will be shown automatically in the *ibaPDA* client.

By the way, the version of the available HD servers will be checked. Version conflicts are color-coded in the table.

11.2 Adding an HD trend graph

A new HD trend graph can be added by

- clicking the corresponding toolbar button



- or by using the *View – Add view – Add HD trend graph* menu
- or by double-clicking on a signal in the HD signal tree (if no HD trend graph has been added so far)

If the HD signal tree is not visible, you have to open it by using the *View – HD signal tree* menu, provided the *ibaPDA* client is connected to the HD server. The HD signal tree provides the same functions like the normal signal tree, including a search function.

11.3 Operation of the HD trend graph

The historical data trend graph (HD trend graph) almost complies with the normal trend graph. The only difference is the data coming from the *ibaHD-Server* instead of the *ibaPDA* server. This means that the scrolling of the X axis is controlled by the *ibaHD server*.

For the HD trend graph you can only select signals out of the *ibaHD* signal tree. Make sure that the *ibaHD* signal tree is open (menu *View- ibaHD signal tree*).

When adding an HD trend graph, no distinction is made between time- and length-based HD store yet. Depending on whether you pull the first signal in an HD trend graph from a time-based or length-based HD recording, the X-axis automatically converts to time or length units. Only signals with the same base (time or length) can then be added.

Whenever you add a new signal to the HD trend graph, you will see all the historical data for this signal. In the normal trend graph, only the data stored in the signal buffer of the client is displayed.

Along with the introduction of the HD server, new control elements were added to the trend view, e.g. panning and zoom buttons as well as a signal table. While these elements can be hidden or activated for the normal trend view, they are always available in the HD trend view.

For more information on controlling zoom and pan buttons, please refer to ↗ *X-axis buttons*, page 23.

If numerical and/or text fields have been configured for the event text along with the definition of HD events, these numerical values and texts can be displayed in the HD trend graph.

Link to ibaAnalyzer
When the HD trend graph is in pause mode, then it is possible to open the displayed time range in *ibaAnalyzer*. Just right-click on an HD trend graph and select *Open with ibaAnalyzer* in the context menu.

11.4 Properties and preferences HD trend graph

Preferences and properties basically equal those of the normal trend graph.

Refer to ↗ *Properties and preferences trend graph*, page 34

12 HD event table

The event table is a special display linked to an event-based HD data storage on an HD server. Event-based data recording is a special kind of (HD) data storage for process events and alarms.

Events must be configured in the event-based HD data storage configuration dialog before they are available in a separate branch in the HD signal tree.

Events can be drawn from the HD signal tree into the event table.

In addition, events can be mapped as text channels and therefore also used in an HD trend graph.

Other documentation.

For more information about configuration of the HD server and event-based HD data storage, please refer to the *ibaHD-Server* manual.

12.1 Connecting to HD server

Before events can be displayed in an event table, a connection between the *ibaPDA* client and the HD server where the events are recorded must be established.

See chapter ↗ *Connecting to an HD server*, page 130.

12.2 Configuration of events

The events to be recorded in the event-based HD data storage must be configured in a special dialog.

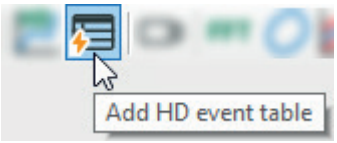
Other documentation

For more information about configuration of events, please refer to the *ibaHD-Server* manual.

12.3 Adding an event table

A new event table can be added by

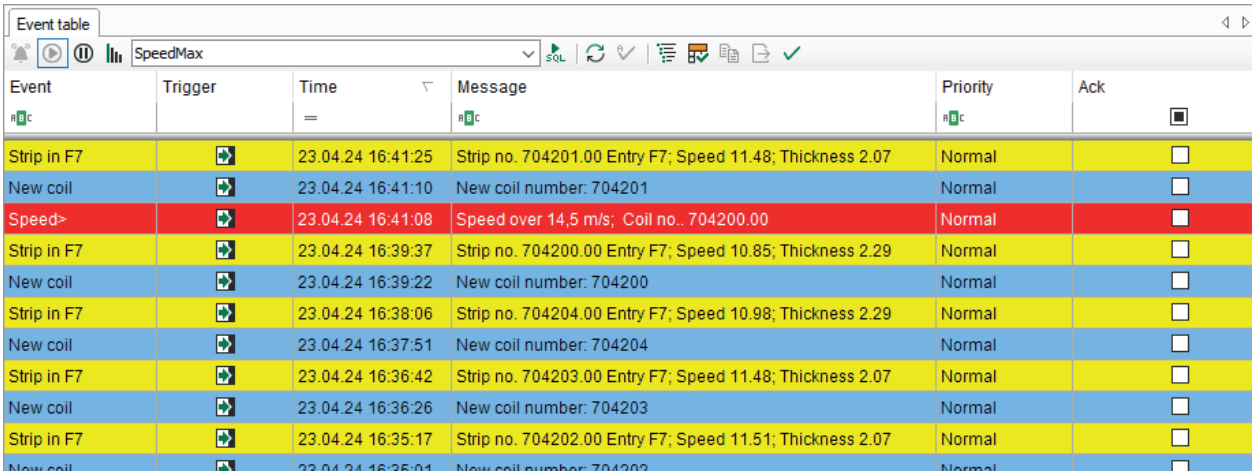
- clicking the corresponding toolbar button



- or by using the *View – Add view – Add HD event table* menu
- or by double-clicking on an event in the HD signal tree (if no event table has been added so far)

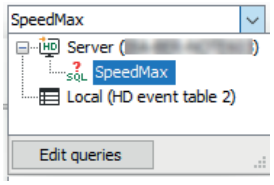
If the HD signal tree is not visible, you have to open it by using the *View – HD signal tree* menu, provided the *ibaPDA* client is connected to the HD server. The HD signal tree provides the same functions like the normal signal tree, including a search function.

12.4 Operation of the event table



Toolbar

The toolbar of the event table contains the following control elements:

	Starts live alarm stream.
	Starts the live display of the event table. All triggered live events are immediately displayed in the event table.
	Stops the live display of the event table. It is now possible to mark individual signals in the event table.
	Displays the number of events in a time range. Select the events of the event table and the time range in the dialog window. The table that appears contains a list with the number of events.
	Selection list for selecting the desired event query. If you expand the list field, you can see all existing event queries and select the one you want. If you click on the <Edit queries> button, the configuration dialog for queries opens. You can then edit existing queries or create new ones.
SQL	Executes the event query that was selected in the selection list.
	Updates the display according to the query executed.
	Automatic acknowledgement of all incoming events
	Display/hide signal tree
	Automatically marks the last line added.
	Copies the line(s) selected in the event list to the clipboard. Only possible in pause mode.

	Exports the line(s) selected in the event list to an Excel or text file. Only possible in pause mode.
	Acknowledges all events displayed in the event table.

Add and query signals

All events configured in the HD event-based data storage and enabled for display appear in the table at the time of their occurrence (trigger). An event can be added from the HD signal tree either by:

- drag and drop from the signal tree into the event table or
- enabling the corresponding checkbox in the properties of the event table. To open the properties, right-click on the event table and chose *Properties*.

In order to search for historic occurrences of events, you can edit queries.

Other documentation

For more information about configuration of event queries, please refer to the *ibaHD-Server* manual.

After a query had been executed, the event table automatically switches from live mode to pause mode. If you want to return to live mode, you have to press the corresponding button in the toolbar.

For easier control, there is a filter row with some functions below the toolbar:

- Sorting by mouse click
  Click on a column header to sort the events by this column in alphabetical/numerical order. Every click on the header toggles between ascending/descending order.
- Filtering
  You can filter the table by entering the entire name or parts of it in the input field or by choosing from available options. Press <Enter> after your entry and the table will be sorted. To return to unfiltered view, clean the filter row fields. The filter options are determined by the data type of the event fields.

For detailed information about the filter functions, please refer to part 7 of the manual, chapter *Advanced filters in tables*.

Tip

If you double-click an event in the list, the HD trend graph will jump to the time of the event and enter Pause mode.

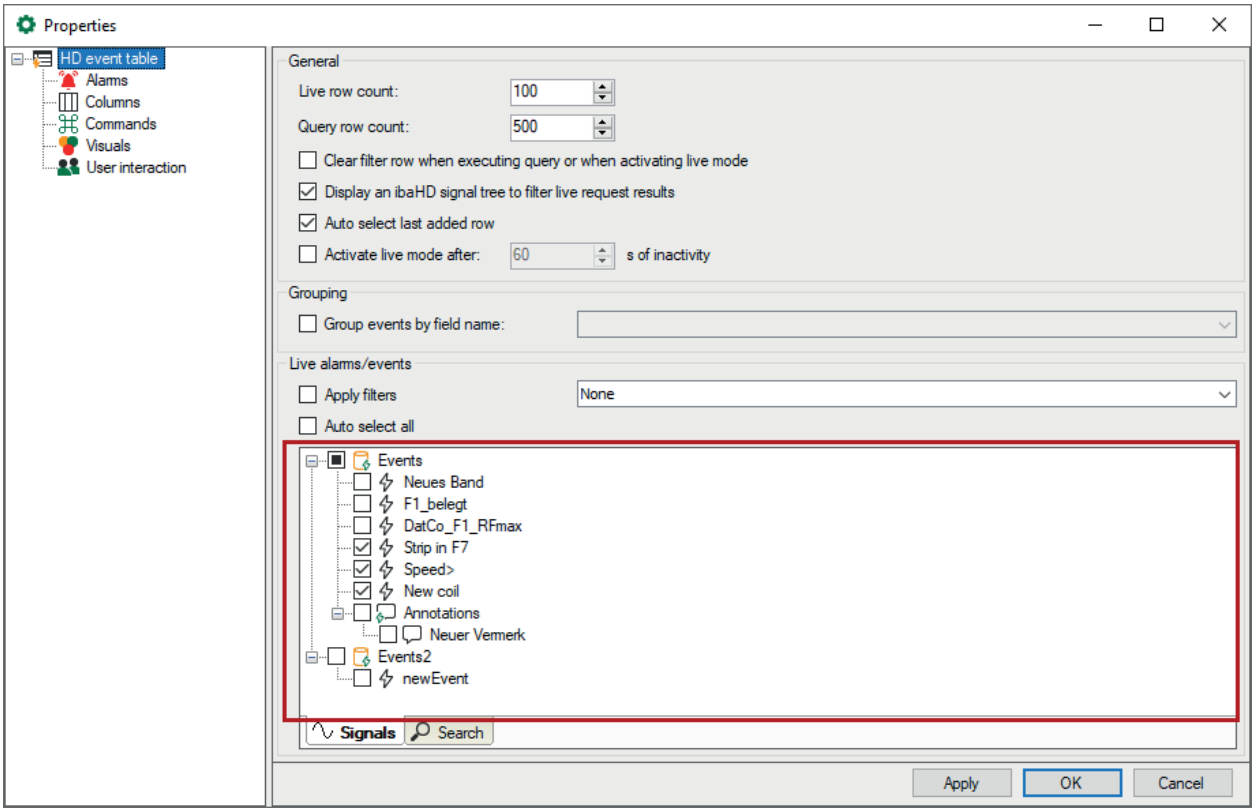
Special features in pause mode

When the view is in pause mode you can select multiple rows. If you then open the context menu with a right-click, you have more features available:

- Copy selected rows to clipboard: This will copy the displayed text of the selected rows to the clipboard. Columns are separated by tabs and rows are separated by new lines.
- Export selected rows to file: This allows you to export the values of the selected rows to an Excel file (.xlsx or .xls) or a text file.

12.5 Properties of the HD event table

In the following, you'll find a description of the properties of the HD event table.



The event tree (see red frame in figure above) is only available in the properties of an event table.

12.5.1 Basic settings

General

Live row count
Here, you can set the number of lines to be displayed in the event table with actual (live) events. A maximum of 1000 rows can be displayed.

Query row count
Here, you set the number of lines shall be displayed in the event table after a query had been performed. A maximum of 1000 lines can be displayed.

Clear filter row when executing query or when activating live mode
When you enable this option, then the search patterns that you may have entered in the filter row are automatically deleted as soon as you execute a query or switch to the live mode of the event table. You therefore do not have to remember to manually remove the filter terms again.

Display an HD signal tree to filter live request results
Enabling this option displays the tree structure for the ibaHD time periods.

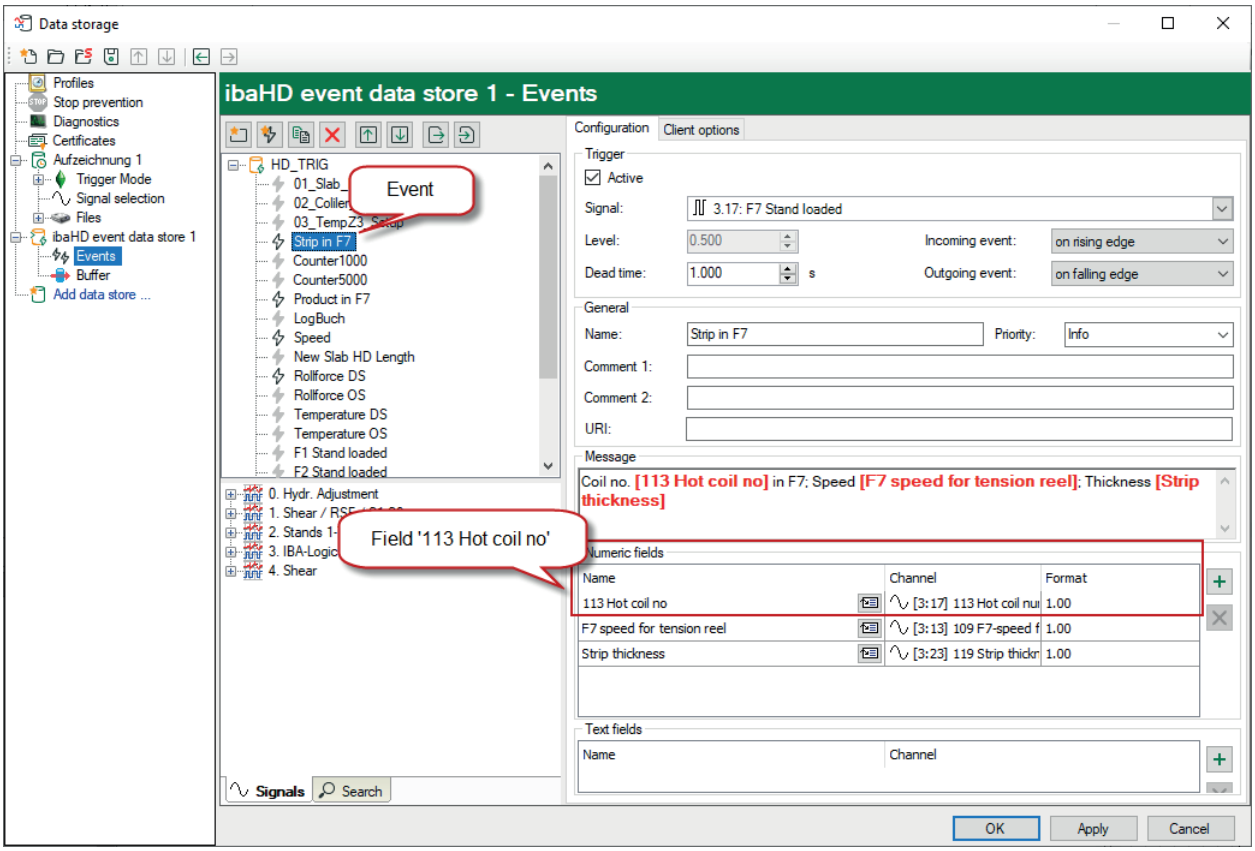
Auto select last added row
Enabling this option will cause the newest row being selected automatically. Thus, the most recent event will always have the focus.

Activate live mode after ... s of inactivity
Enabling this option will cause the view to return automatically from pause mode to live mode if no mouse or keyboard operation occurred over the set time span (given in seconds).

Grouping

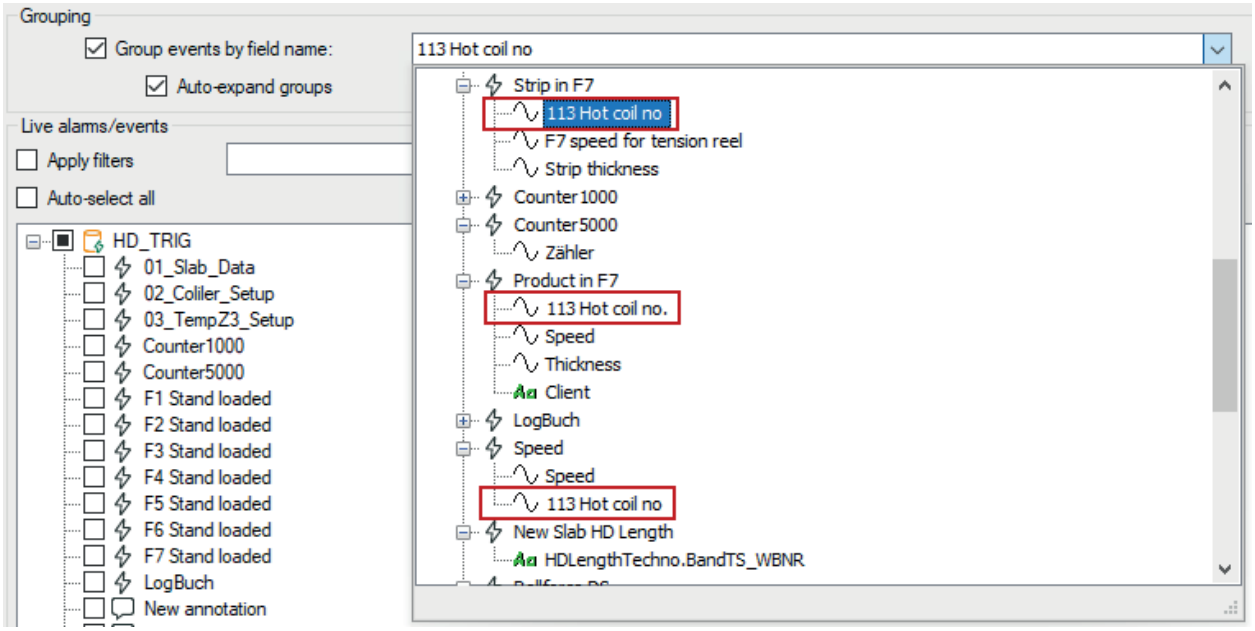
By activating this option you can group events in the table. The grouping is made with a (dynamic) field which is used in the events (see event configuration).

Example for the use of a field during the event configuration:



A grouping can only be carried out reasonably if the field has the same name in all events.

To select a field for the grouping, open the drop-down list in the settings of the event table.



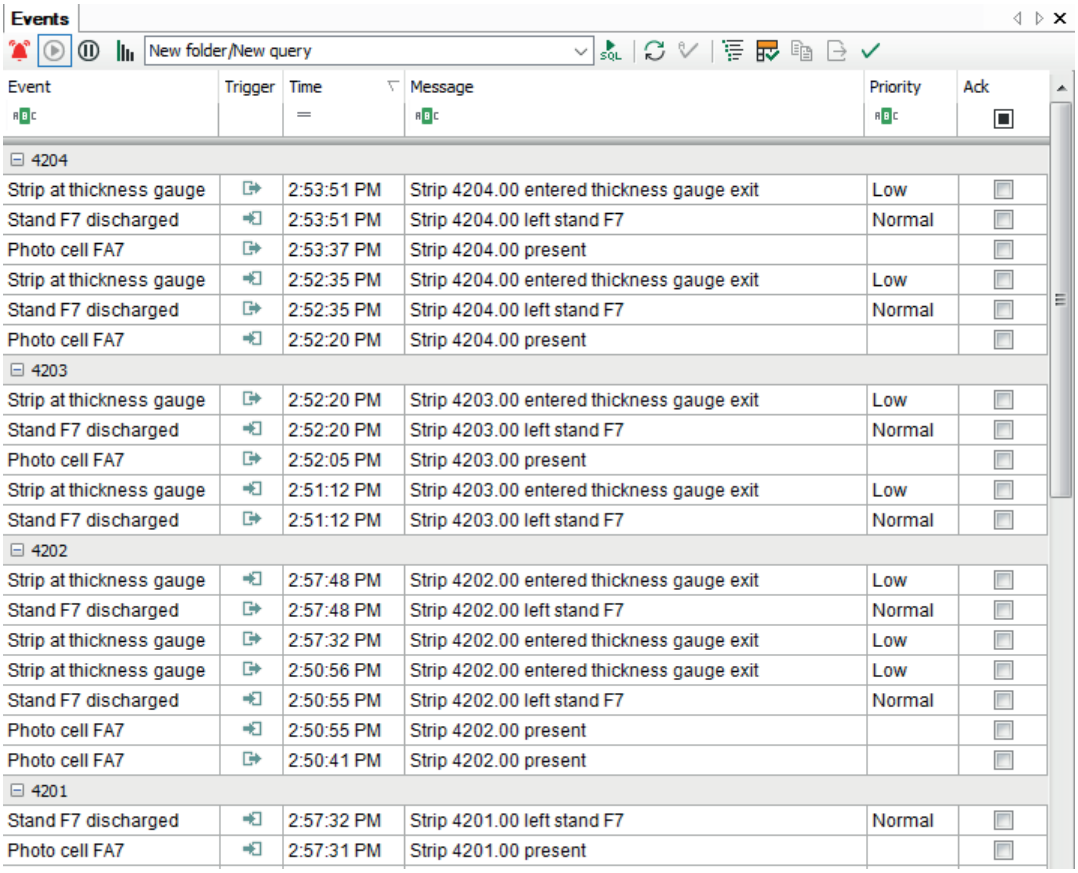
All configured events with their fields are listed there. Select the required field for the grouping and click on <Apply> or <OK>.

All events contained in this field are displayed in groupings now.

In this example this is the field “Coil number”.

In the table, a grouping header is set for every value of the field “113 Hot coil number” and the corresponding events will be listed below.

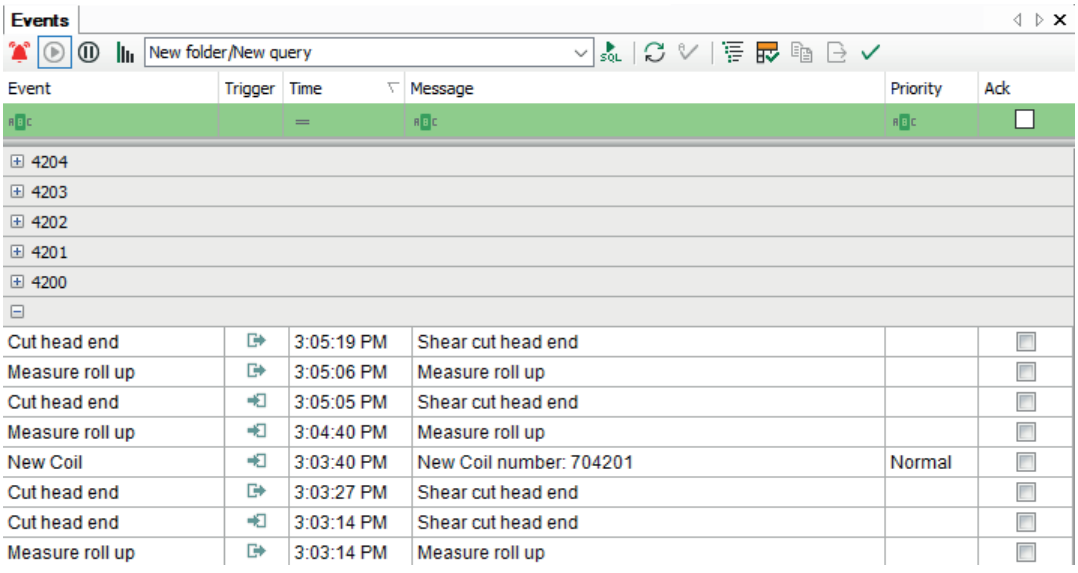
Events grouped according to coil numbers:



Event	Trigger	Time	Message	Priority	Ack
4204					
Strip at thickness gauge		2:53:51 PM	Strip 4204.00 entered thickness gauge exit	Low	
Stand F7 discharged		2:53:51 PM	Strip 4204.00 left stand F7	Normal	
Photo cell FA7		2:53:37 PM	Strip 4204.00 present		
Strip at thickness gauge		2:52:35 PM	Strip 4204.00 entered thickness gauge exit	Low	
Stand F7 discharged		2:52:35 PM	Strip 4204.00 left stand F7	Normal	
Photo cell FA7		2:52:20 PM	Strip 4204.00 present		
4203					
Strip at thickness gauge		2:52:20 PM	Strip 4203.00 entered thickness gauge exit	Low	
Stand F7 discharged		2:52:20 PM	Strip 4203.00 left stand F7	Normal	
Photo cell FA7		2:52:05 PM	Strip 4203.00 present		
Strip at thickness gauge		2:51:12 PM	Strip 4203.00 entered thickness gauge exit	Low	
Stand F7 discharged		2:51:12 PM	Strip 4203.00 left stand F7	Normal	
4202					
Strip at thickness gauge		2:57:48 PM	Strip 4202.00 entered thickness gauge exit	Low	
Stand F7 discharged		2:57:48 PM	Strip 4202.00 left stand F7	Normal	
Strip at thickness gauge		2:57:32 PM	Strip 4202.00 entered thickness gauge exit	Low	
Strip at thickness gauge		2:50:56 PM	Strip 4202.00 entered thickness gauge exit	Low	
Stand F7 discharged		2:50:55 PM	Strip 4202.00 left stand F7	Normal	
Photo cell FA7		2:50:55 PM	Strip 4202.00 present		
Photo cell FA7		2:50:41 PM	Strip 4202.00 present		
4201					
Stand F7 discharged		2:57:32 PM	Strip 4201.00 left stand F7	Normal	
Photo cell FA7		2:57:31 PM	Strip 4201.00 present		

You can minimize or extend the groups separately, if required. In the context menu of the event table you can minimize or extend all groups. In the properties dialog you can configure if the group should be extended automatically or not.

Any other events which do not contain the field for grouping are listed under an empty header.



Event	Trigger	Time	Message	Priority	Ack
4204					
4203					
4202					
4201					
4200					
Cut head end		3:05:19 PM	Shear cut head end		
Measure roll up		3:05:06 PM	Measure roll up		
Cut head end		3:05:05 PM	Shear cut head end		
Measure roll up		3:04:40 PM	Measure roll up		
New Coil		3:03:40 PM	New Coil number: 704201	Normal	
Cut head end		3:03:27 PM	Shear cut head end		
Cut head end		3:03:14 PM	Shear cut head end		
Measure roll up		3:03:14 PM	Measure roll up		

As expected, the sorting of the lines in the groups is made according to the selected sorting column (e.g. column "Time", descending). The sorting of the groups will be made after the comparison of values in the sorting column in every first line of the groups.

The first line of a group always contains the maximum or minimum value of the sorting column at ascending or descending sorting.

Live events

Here, you can see a tree structure of all event-based HD stores of the associated *ibaHD-Server* with the events stored there.

Select here the events which have to be displayed in live mode in the event table.

Selecting the setting "Auto-select all," all events of all HD stores of the associated *ibaHD-Server* are always displayed in the event table.

Note

Events which you select by drag and drop from the HD signal tree are automatically addressed in the tree structure of the live events.

12.5.2 Alarms

Acknowledgment

Local acknowledgement Do not send the acknowledgements to the HD-Server but handle them within this Event Table
If you enable this option, the manual acknowledgements of alarms will only be logged in the local event table. The same alarm remains unacknowledged on other clients and/or event lists. If you disable this option, the acknowledgement of an alarm will be performed by the ibaHD-Server in all event lists.

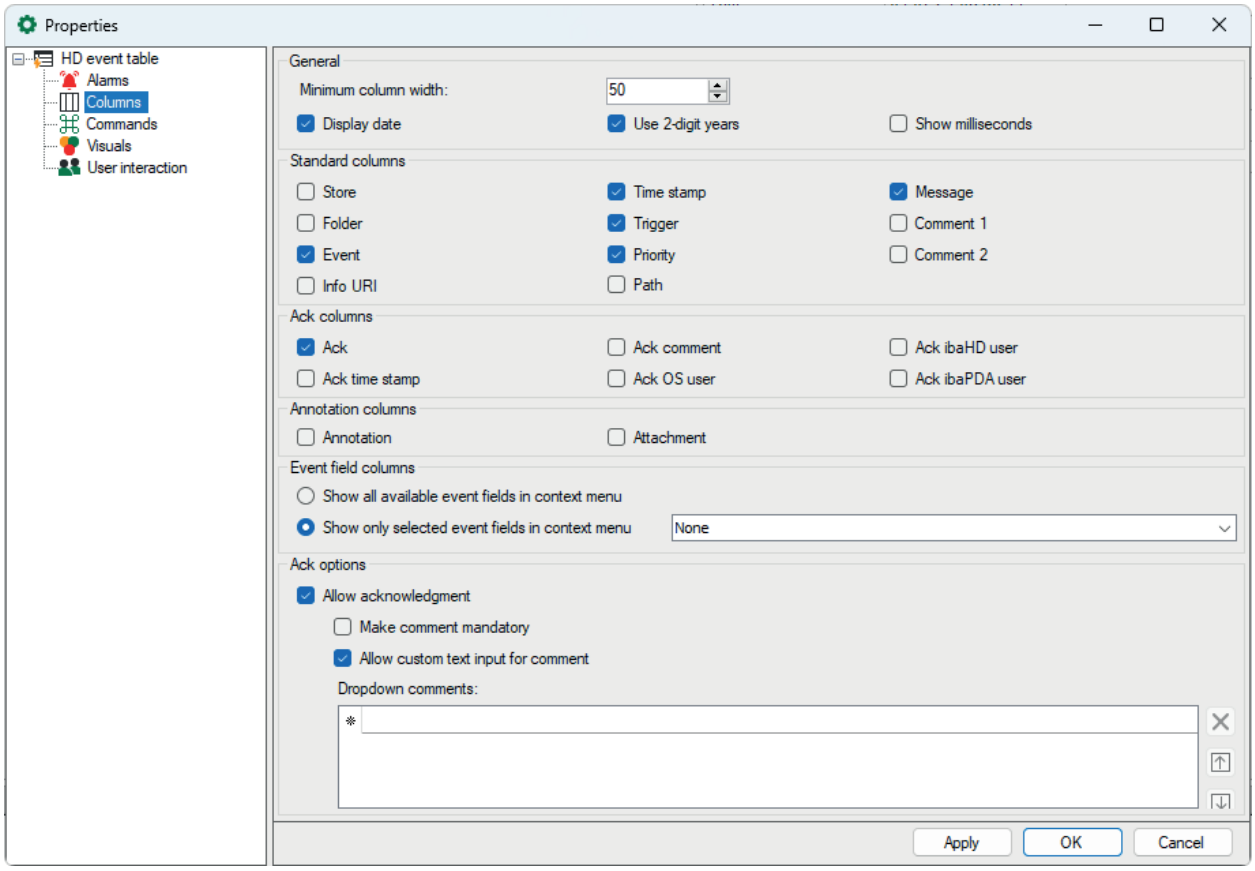
Additionally, you may activate an automatic acknowledgement of local events, if this option is enabled.

Style

If you enable this option, the alarms will always be displayed in a certain format. You can determine how this format looks like by defining the font as well as the foreground and background color.

12.5.3 Columns

In the *Columns* node, you define the display and behavior of the columns in the event list.



General

Here you can specify the values for the *Minimum column width* in pixels and define the date display.

Standard, acknowledge and note columns

Here you can select which columns should be shown in the event table and which should not. In addition to the standard columns, you can enable or disable special columns for acknowledgment and notes.

Column Info URI

If you click on the URI displayed in this column, the URI will be opened in the computer's default browser.
If this column is not displayed, you can also open the URI by <Alt>+mouse-clicking on any area of the event entry.

Tip

In the event table, you can always enable and disable columns via the context menu.

Event fields

Show all available event fields in context menu
You can show and hide all fields defined in the various events as columns via the context menu of the event table.

Show only selected event fields in context menu
From the selection list, select the event fields that are to be available as additional columns in the event table. You can show and hide these via the table's context menu.

Meaning of the acknowledgment columns

Acknowledgment
This column is enabled by default and is used to acknowledge an event message. The acknowledgment occurs by clicking on the box. Once the message has been acknowledged, a check mark is shown in the box. If the column in the event table is not enabled, then an acknowledgment is not possible.

Acknowledgment of the time stamp
The time of the acknowledgment is shown in this column.

Acknowledgment of a comment
This column shows the comment that was entered or selected during the acknowledgment (see *acknowledgment options*, further below).

Acknowledgment system user, ibaHD-User and ibaPDA-User
In these columns, the respective user name of the user is shown who actuated the acknowledgment.

- *System-User* is the user name who is registered under Windows on the computer or the user account under which the *ibaPDA* client is running.
- *ibaHD-User* is the user name under which the client is registered on the *ibaHD-Server* according to the user management.
- *ibaPDA-User* is the user name under which the client is registered on the *ibaPDA* server according to the user management.

Acknowledgment options

You can adjust the display and handling of the acknowledgment of event messages to your needs in the acknowledgment options settings.

Allow acknowledgment
Enable this option to be able to acknowledge messages in the event table. If you disable this option, then a mouse click in the column *Acknowledgment* does nothing.

Comment required
If you enable this option, then a dialog for entering or selecting a comment opens automatically as soon as a message has been acknowledged. If you disable this option, then an acknowledgment is completed without comment. An acknowledgment comment can also be added or changed later in the event list.
The comment is displayed in the *Ack comment* column in the event list.

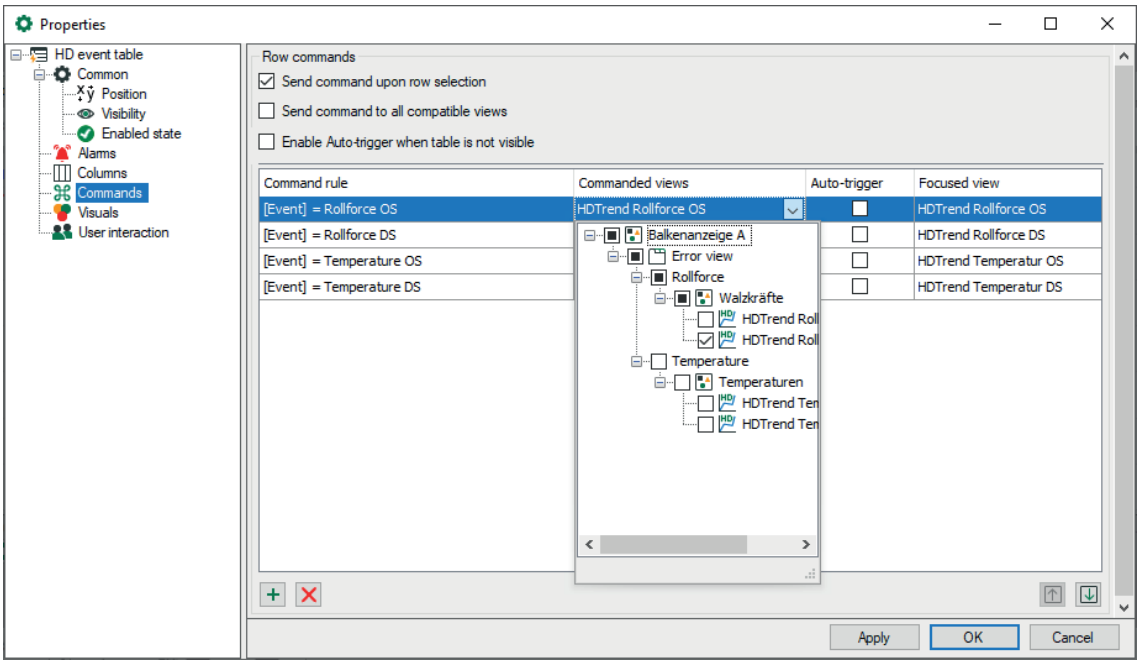
Allow user text input for comments
If you enable this option, then any plaint text can be entered as a comment during the acknowledgment. This is possible as an alternative to selecting prepared comments in the drop-down list. If you disable this option, only the entries from the drop-down list can be used as an acknowledgment comment.

Drop-down comments
In this list, you can enter acknowledgment comments that are then available to choose from in the drop-down list during acknowledgment. Use the arrow symbols to the right of the list to change the sequence of the comments. Use the delete icon to remove the comment in which the cursor is located.

12.5.4 Commands

Event tables can be coupled with HD trend graphs so that when double-clicking on an event line the desired trend graph stops and is placed on the right position of the time or length axis.

In the branch "Commands" in the properties of the event table you can set such line commands.



If you do not enable both options for row commands, then double clicking on an event in the event table does nothing.

Row commands

If you want all HD trend graphs to jump to the time of the event that you have double clicked, then enable the option *Send command to all compatible views*. The command rules will be disabled in this case.

If you want only certain trend graphs to jump to the time of an event and others to continue to run as normal, then only enable the option *Send command during row selection* and configure the appropriate command rules.

The third option determines whether a configured auto-trigger executes a jump or not, if the event table is not visible. This option is disabled by default, i.e. an auto-trigger executes the jump only if the event table is visible.

Command rules

Using the command rules, you can specify exactly for which event and under which conditions which trend views jump to the time of the event.

In the column *Relevant views,* you select the trend graphs which have to be stopped and positioned when the conditions are fulfilled.

If you make a check mark for a command rule in the 'auto-trigger' column, then the view is automatically switched when the configured condition occurs.

You can define as many command rules as you want.

Note

The list of command rules will be checked from the top down when double clicking on a row in the event table. At the first applied rule, the check will be finished and the command is forwarded to the assigned views.

You can modify the sequence of the rules: Use the button to move the marked rule up and the button to move it down.

Note

When using the command rules, double clicking in the event table is only successful for the events that are included in the command rules.

Configure command rule

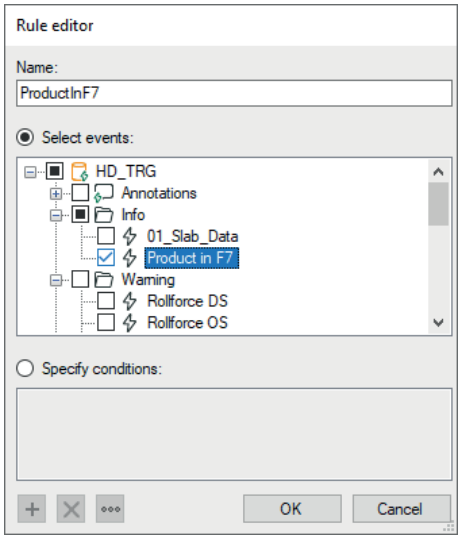
Adds a new rule.

Deletes the marked rule.

Opens the editor for editing the rule

Add a new command rule.

The dialog *Edit rule* opens.



First give the rule a meaningful name.

A rule offers two mutually exclusive methods.

If you enable the “Select events” option, then you can mark one or more events in the tree below that trigger a switching of the trend view, either by the status change "incoming" and/or "outgoing" (auto-trigger) or by double clicking on an event row. For events that are not marked here, the trend view will continue to run.

Tip

This method is suitable if you only want to switch certain trend views when double clicking on any event. To do this, mark all events in the tree.

If you enable the"Specify conditions” option, then you can formulate check conditions in the field below that are executed when double clicking on an event row. In the auto-trigger mode, the conditions are checked constantly.

If a condition is met, the trend view is switched.

Configure conditions

A condition can consist of one or more expressions that are linked with each other.

+ Adds a new expression.

✕ Deletes the marked expression.

Opens the editor for editing the expression.

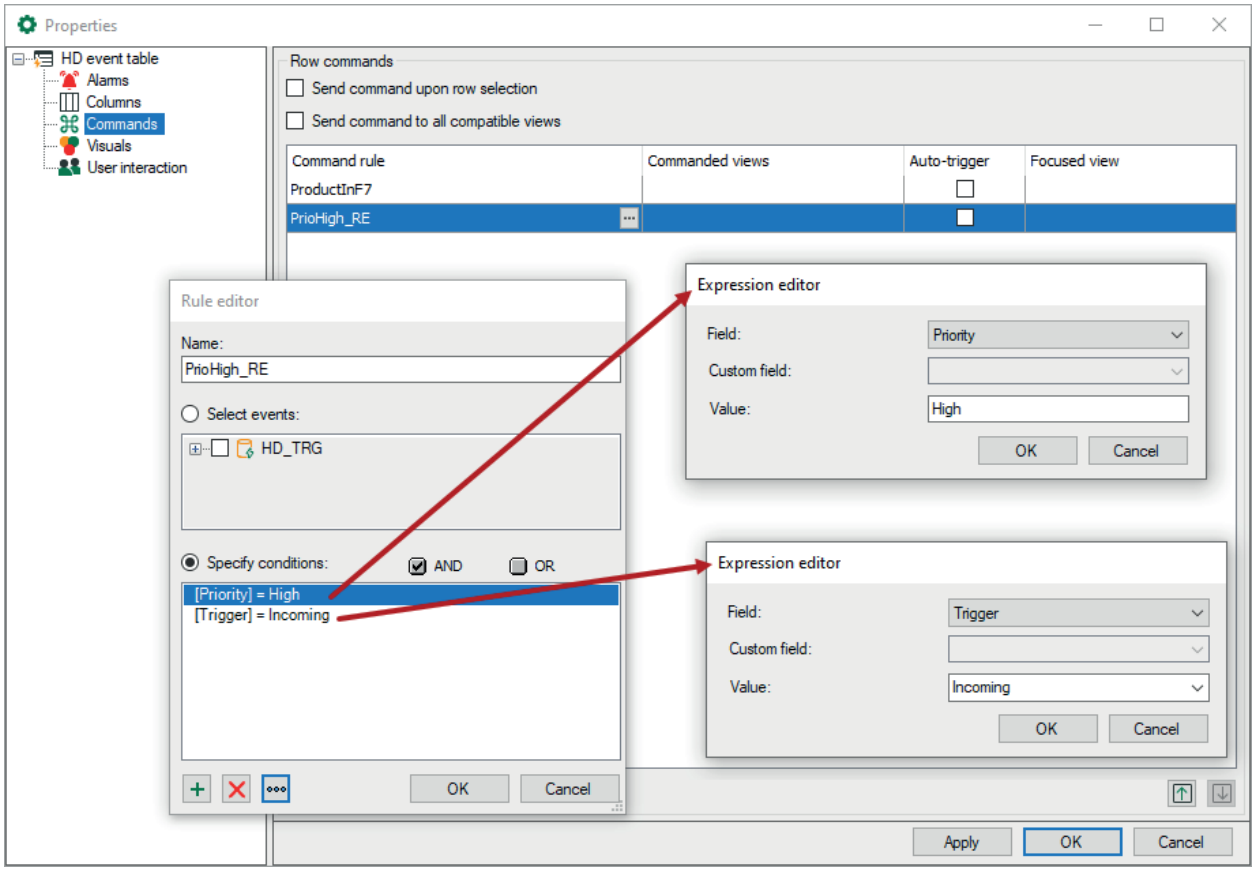
If a condition contains several expressions, then these will be linked with a logical AND or OR.

Click the button + in order to add an expression or the button to edit an existing expression. In the *Edit expression* dialog, first select a field (e.g. event, comment, trigger, etc.) and then the value (e.g. event name, plaint text incoming/outgoing, etc.)

In the following example, double-clicking on events stops the HD trend graph named "Rolling forces" and sets it to the time stamp of the clicked event. The events meet the following requirements:

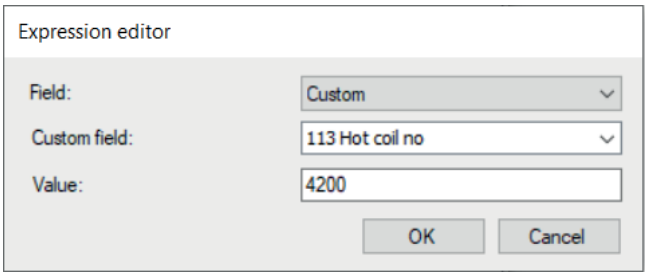
- Priority = "High"
- Trigger = "Incoming"

Command rule with two conditions that must be simultaneously met:

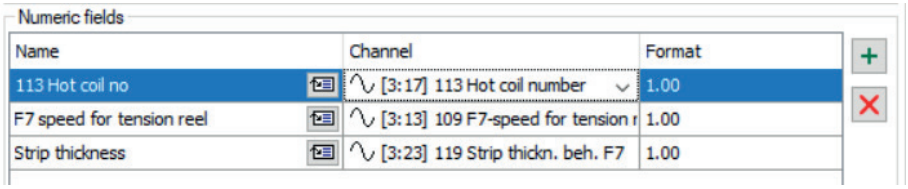


Note

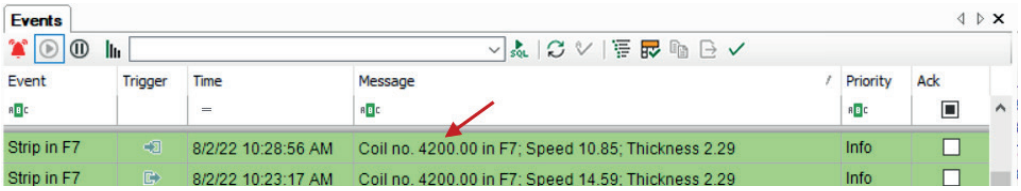
In an expression you can select the field type "Custom". Therefore commands can be written in order to request number and text fields from the event definition.



With this expression, for example, you could check by double clicking on the event whether it has a number field or label called "coil number" and if it is equal to the value "4200".

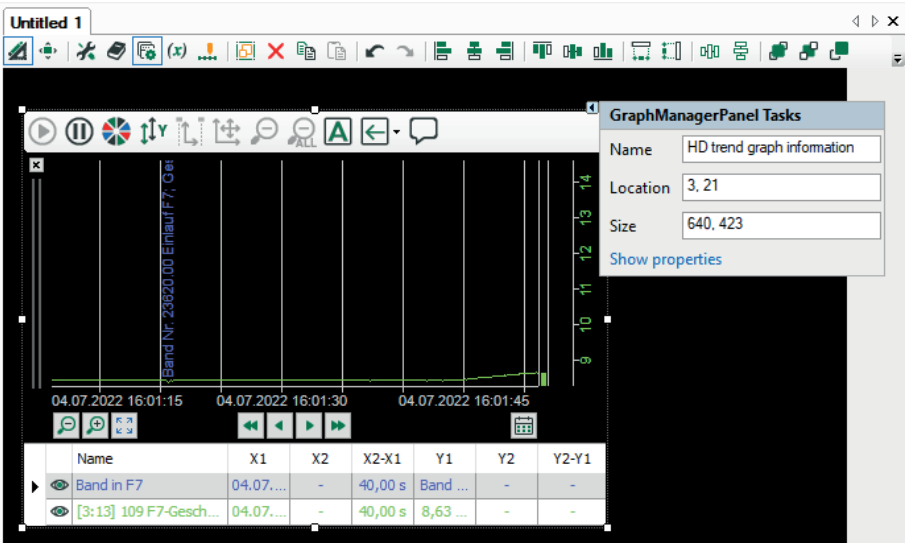


This argument would be met, for example, for the following event:



Note

You can also send commands to HD trend graphs which are placed in *ibaQPanel*. To name an HD trend graph in *ibaQPanel*, in the design mode click on the triangle icon in the upper right corner of the HD trend graph. Enter the desired designation in the text field "Name".



12.5.5 Visuals

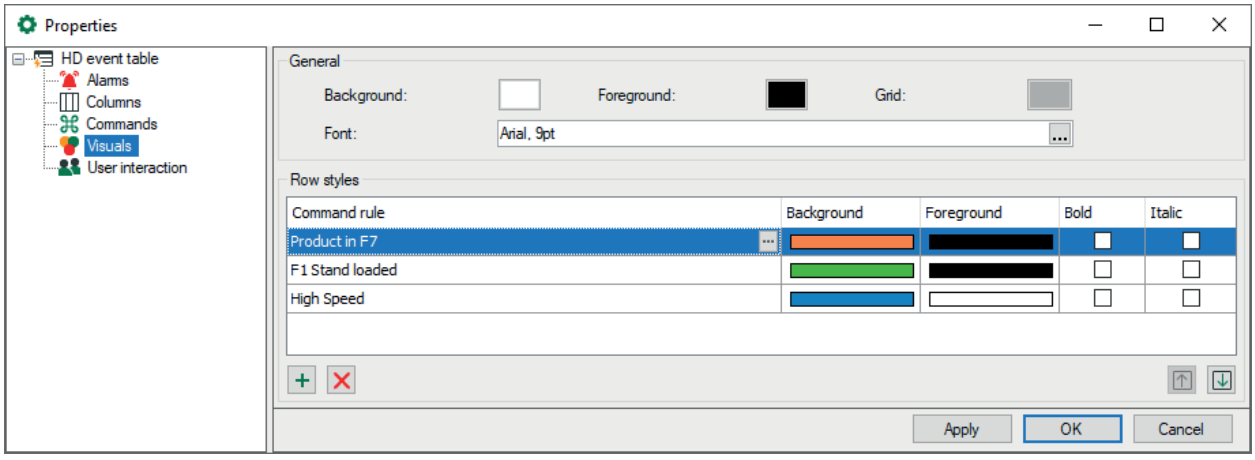
Similar to the commands (see above), the style of the rows in the event table can be controlled, depending on the properties of the occurred event.

In the "Pane" heading in the properties of the event table, you can set such row styles locally on the *ibaPDA* client.

Note

For every event, you can specify the row style globally already during the event definition (configuration of the HD record, tab *Client options*).

These global row styles are overwritten by the local settings that you make in the properties of the event table.



General

Here, you set the overall performance of the event table, background and foreground color, the color of the grid and the font.

Row styles

In the "Row styles" list, you create rules and corresponding style modifications (background and foreground color, bold and italic print of the font).

+ Adds a new rule at the end of the list.

✕ Deletes the marked rule.

··· Opens the editor for editing the rule

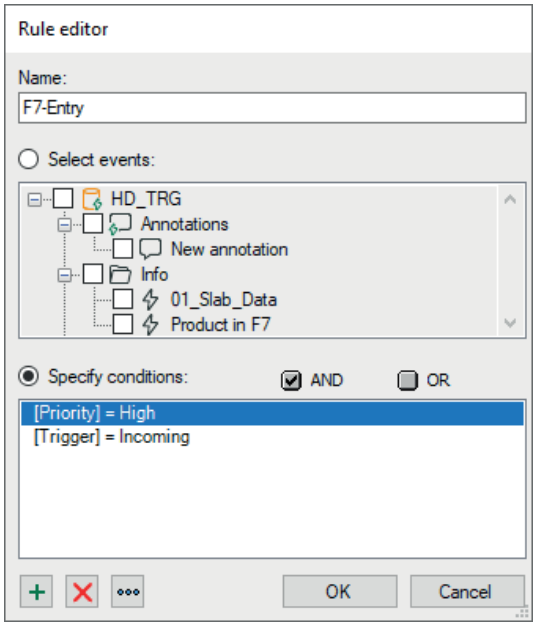
Note

By double-clicking, the list of rules is checked from the top to the bottom. At the first applied rule, the check will be finished and the style modification will be made.

You can modify the sequence of the rules: Use the button ↑ to move the marked rule up and the button ↓ to move it down.

Add a new command rule.

The dialog *Rule editor* opens.



First give the rule a meaningful name.

A rule offers two mutually exclusive methods.

If you enable the "Select events" option, then you can mark one or more events in the tree below that are shown with the row style corresponding to the rule.

Tip

This method is suitable if you want to assign a row style to certain events.

If you enable the "Specify conditions" option, then you can formulate check conditions in the field below that are checked constantly during acquisition.

If a condition is met, the corresponding event is provided with the row style.

A condition can consist of one or more expressions that are linked with each other.

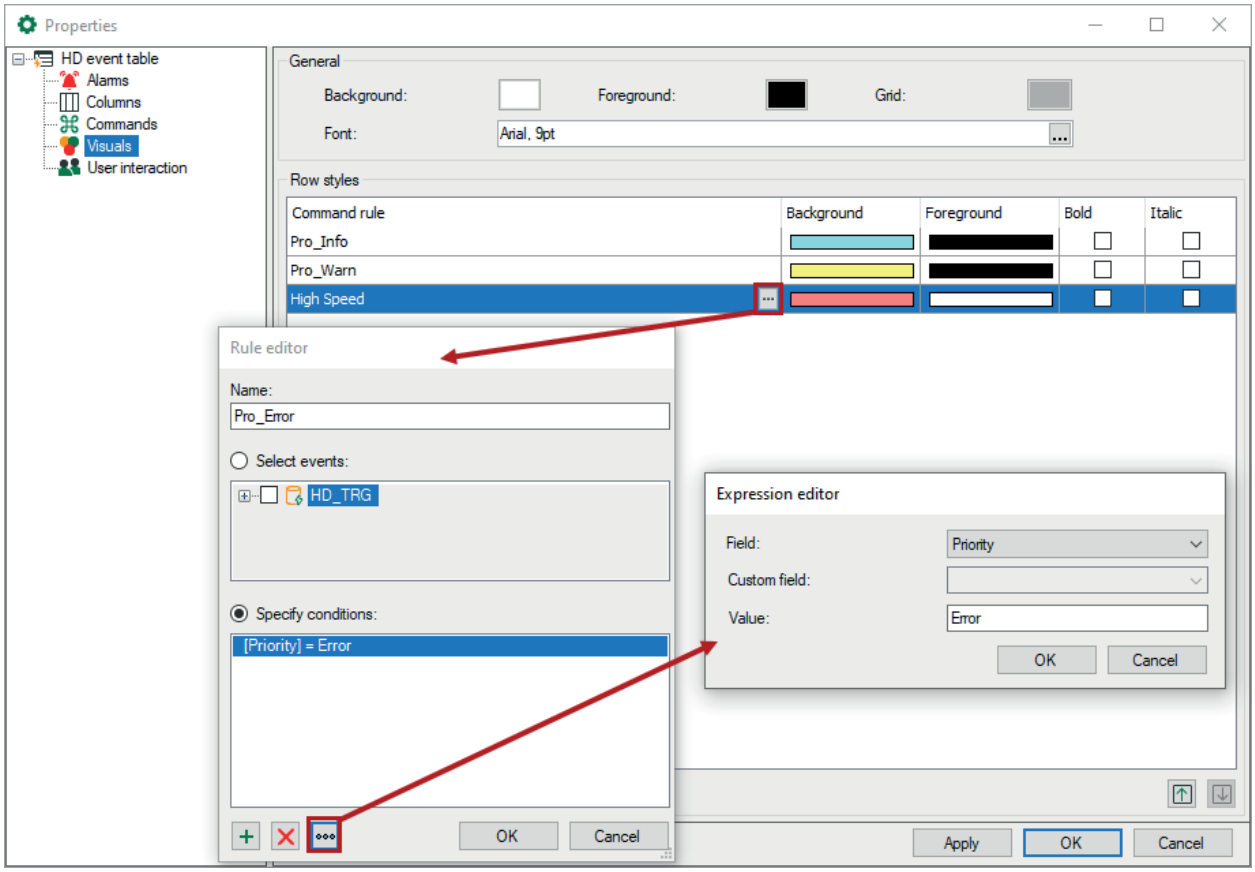
+ Adds a new expression.

✕ Deletes the marked expression.

... Opens the editor for editing the expression.

If a condition contains several expressions, then these will be linked with a logical AND or OR. In the *Edit expression* dialog, first select a field (e.g. event, comment, trigger, etc.) and then a value (e.g. event name, plaint text incoming/outgoing, etc.)

In the example below, for all events the background is colored red and the font is shown in bold. The events meet the requirement priority = "Error".



The row styles shown above as an example change the appearance of an event table as follows:

Event	Tri...	Time	Message	Priority	Ack
Tension Uncoiler 1		**25.04.12 12:00:46**	**Tension Uncoiler #1 50% above SP level (SP: 35.0 [kN], AV: 35.8 [kN]**	**Error**	☐
Tension Uncoiler 1		25.04.12 12:00:46	Tension Uncoiler #1 30% above SP level (SP: 35.0 [kN], AV: 35.8 [kN]	Warning	☐
Tension Uncoiler 1		**25.04.12 12:00:39**	**Tension Uncoiler #1 50% above SP level (SP: 35.0 [kN], AV: 62.1 [kN]**	**Error**	☐
Tension Uncoiler 1		25.04.12 12:00:39	Tension Uncoiler #1 30% above SP level (SP: 35.0 [kN], AV: 62.1 [kN]	Warning	☐
Coil on Uncoiler 2		*25.04.12 12:00:25*	*New Coil '4AB12346 ' on Uncoiler #2 (Diam.: 1675 mm, Thickn.: 1.01 mm)*	*Information*	☐
Tension Uncoiler 1		25.04.12 12:00:14	Tension Uncoiler #1 30% above SP level (SP: 35.0 [kN], AV: 37.0 [kN]	Warning	☑
Tension Uncoiler 1		25.04.12 12:00:09	Tension Uncoiler #1 30% above SP level (SP: 35.0 [kN], AV: 47.8 [kN]	Warning	☑
New Coil welded		*25.04.12 11:59:17*	*Coil '4AB12345 ' welded (Diam.: 1870 mm, Thickn.: 1.02 mm)*	*Information*	☐
Coil on Uncoiler 1		*25.04.12 11:58:24*	*New Coil '4AB12345 ' on Uncoiler #1 (Diam.: 1870 mm, Thickn.: 1.02 mm)*	*Information*	☑

13 HD time periods table

The HD time periods table is a special view linked to the so called "time periods", which is an extended feature of the time based HD data store. The time periods need to be defined in the configuration dialog of the timebased HD datastore before they can be displayed in the time periods table. Furthermore, the configured time periods must have been recorded at least once from the beginning to the end in order to display their data in the table.

By means of the time periods, ranges of time which are delimited by a start event and a stop event (trigger condition, signals etc.) can be selected from an HD time based data store and put on display. Temporal limited time ranges (hours, days, months, shifts etc.) are possible as well.

Moreover, KPIs or other statistical quantities can be evaluated over these time periods and stored, like info fields in a data file.

TheHD Time period table supports the unit system, i.e. the numeric values can be converted to another unit system for display.

For more information about the unit system see ibaPDA manual, part 2, chapter Unit table.

Other documentation

For more information about configuration of the HD server, the time-based HD data storage and HD time periods, please refer to the *ibaHD-Server* manual.

13.1 Connecting to an HD server

Before HD time periods can be displayed in an HD time periods table, a connection between the *ibaPDA* client and the HD server where the time periods are recorded must be established.

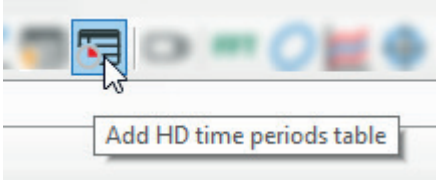
See chapter ↗ *Connecting to HD server*, page 133

The feature of time periods is supported by *ibaHD-Server* version v3.1 or higher.

13.2 Adding an HD time periods table

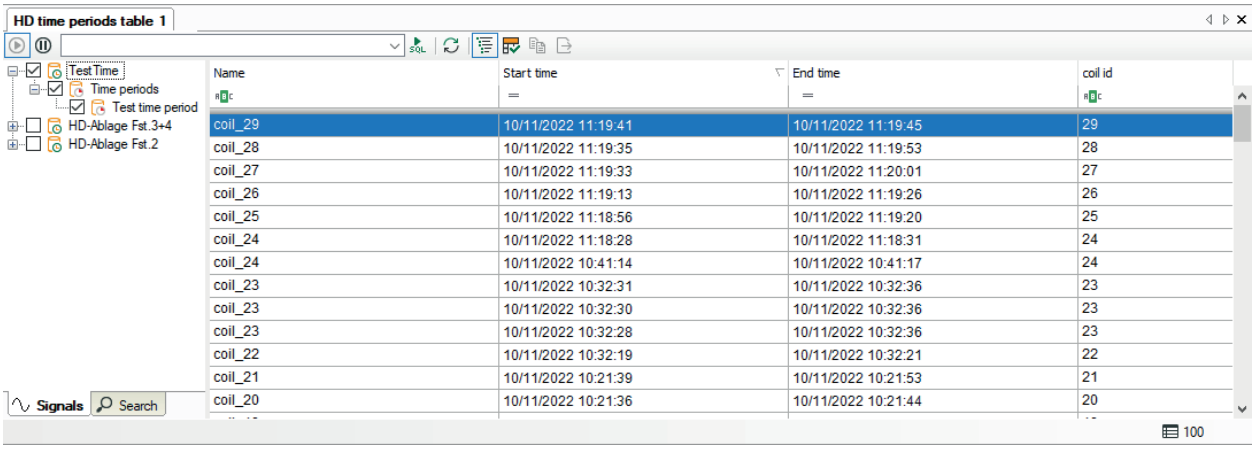
An HD time periods table can be added by

- clicking the corresponding toolbar button



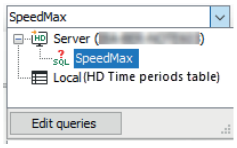
- or by using the *View – Add view- Add HD time periods table* menu

13.3 Operation of the time periods table



Toolbar

The tool bar of the time periods table contains the following control elements:

	Starts the live display of the time periods.
	Stops the live display of the time periods (pause). It is now possible to mark individual lines in the table.
	Drop-down list for selection of time period query. If you open the drop-down list you can see all available queries and select the one you want. A mouse click on the button <Edit queries> opens the editor for creating time period queries.
SQL	Carries out the SQL-query which has been selected in the drop-down list. If the query takes some time, a progress bar will be displayed in the status bar at the bottom of the table. Furthermore, there will be a link which you can use to abort the query in case it takes too long.
	Refreshes the display according to the executed query.
	Show/hide signal tree
	Highlights automatically the most recently added line.
	Copy the time periods which are selected in the table to the clipboard. Only possible in pause mode.
	Exports the time periods which are selected in the table to an Excel or text file. Only possible in pause mode.

You can display time periods from different time period stores of an HD-Server in the table.

Select the time period store which should be displayed. You have the following options:

- Open the tree by clicking the signal tree button in the toolbar and select there the time period store which should be displayed.
- Or drag and drop the time period store out of the HD signal tree into the table.
- Or open the properties by a right mouseclick in the time periods table and select the time period store there.

In live mode new lines will be added when new time periods are generated. The lines are updated when the end time or other info fields are updated.

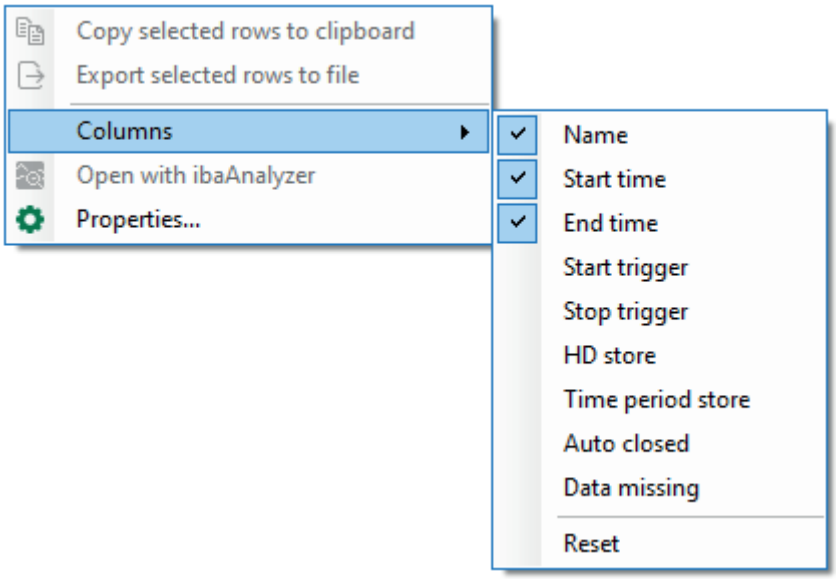
Status bar
The status bar is on the lower border of the table. It gives information about the SQL-queries carried out (e.g. status of the running query, number of the events, error messages) and if a filter is currently active. Furthermore, while the query is running there will be a link which you can use to abort the query in case it takes too long.

Sorting
The table can be sorted ascending or descending by every column. Click on the header of the column which you want to use as sorting criterion. The little arrow (up or down) indicates the sorting order (descending or ascending). By default, the descending order is selected for the "Start time" column. Hence, the most recent event is always on top.

Filtering
Right below the column headers there is a filter row. You can filter the table by entering the entire name or parts of it in the input field or by choosing from available options. Press <Enter> after your entry and the table will be sorted. To return to unfiltered view, clean the filter row fields. The filter options are determined by the data type of the info fields.

For detailed information about the filter functions, please refer to part 7 of the manual, chapter *Advanced filters in tables*.

Show/hide columns
If you right-click on the table, you can select the visible columns or hide columns in the context menu.



In the upper section of the context menu, you can select the standard columns, and in the lower section the columns for the info fields. The info field columns that appear in the context menu can be configured in the properties of the time periods table, see next section.

View a time period in ibaAnalyzer
In order to load a time period out of the table into *ibaAnalyzer* and open it, make a right-click on the desired row and select *Open with ibaAnalyzer* in the context menu. If *ibaAnalyzer* v8.1 or higher is installed on your system, the time period will be opened showing the info fields of the time period. Moreover, the name of the time period is applied as file name. If an older version of *ibaAnalyzer* is installed, then just the time range will be opened and no time period info fields will be queried.

Special features in pause mode
When the view is in pause mode you can select multiple time period rows. If you then click on *Open with ibaAnalyzer* in the context menu, all selected time periods are opened as separate files in *ibaAnalyzer*.

There are some more features available in the context menu while in pause mode:

- Copy selected rows to clipboard: This will copy the displayed text of the selected rows to the clipboard. Columns are separated by tabs and rows are separated by new lines.
- Export selected rows to file: This allows you to export the values of the selected rows to an Excel file (.xlsx or .xls) or a text file.

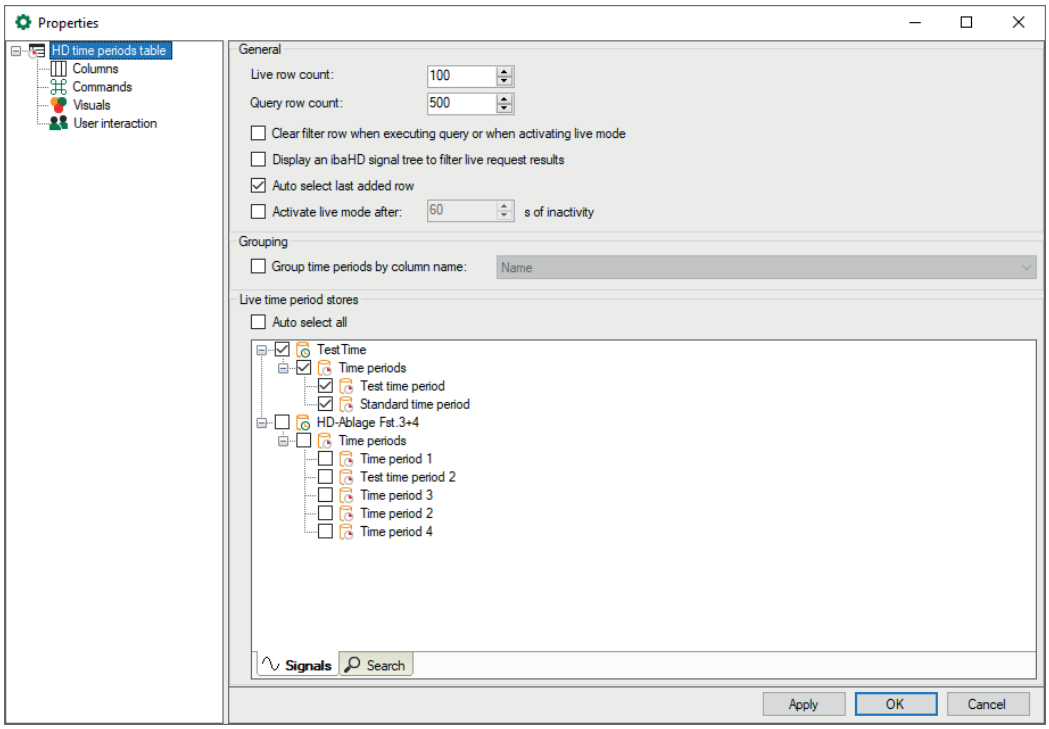
These commands can also be triggered by the last two buttons in the toolbar.

13.4 Properties of the HD time periods table

To open the properties dialog, right-click in the HD time periods table and select *Properties* from the context menu.

13.4.1 General Settings

In the *Properties* dialog, determine the display of the HD time period table and select the time periods to be displayed.



General

Live row count / Query row count

Here, you can set how many rows are to be displayed in the HD time periods table with current time periods and/or how many rows are to be displayed after a query. A maximum of 1000 rows can be displayed.

Clear filter row when executing a query or when activating live mode

Enabling this option automatically deletes any search patterns you have entered in the filter row as soon as you execute a query or switch to live mode in the table. This means that you do not have to remember to manually remove the filter terms.

Display an HD signal tree to filter live request results

Enabling this option displays the tree structure for the ibaHD time periods.

Auto select last added row

Enabling this option will cause the newest row being selected automatically. Thus, the most recent event will always have the focus.

Activate live mode after ... s of inactivity

Enabling this option will cause the view to return automatically from pause mode to live mode if no mouse or keyboard operation occurred over the set time span (given in seconds).

Grouping

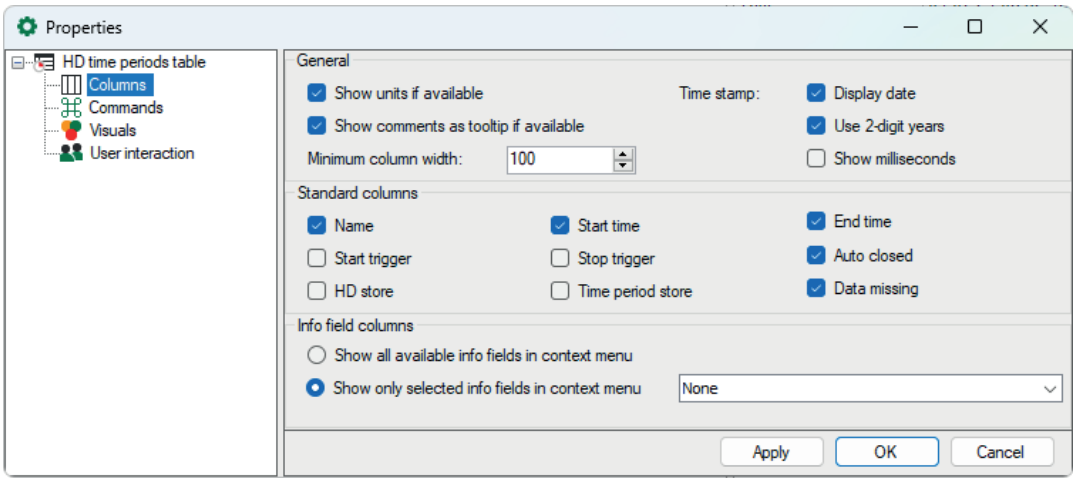
Enabling this option allows you to group time periods in the table using columns. Select a column in the adjacent field.

Live time periods

In the tree structure, you can select the time period store from which the time periods are to be displayed. Select the corresponding stores and the time periods. If you enable the option *Auto select all*, then all stores and time periods will be selected for live display.

13.4.2 Columns

In the *Columns* dialog, you define the formatting of the columns and the info fields.



General
Use the options to choose how the data will be formated in the columns.

Standard columns
The selected columns are displayed in the HD time periods table.

In addition to the *Name* of the time period, you can choose from the following columns:

- *Start time* and *End time*
- *Start trigger* and *Stop trigger*
- *HD store* und *Time period store*
- *Auto closed*:
  This value is "0" if the time period was closed by a stop trigger and "1" if *ibaPDA* has closed it automatically. *ibaPDA* closes a time period automatically if the maximum time period duration is exceeded. *ibaPDA* also automatically closes all open time periods if the acquisition is stopped in standard mode.
- *Data missing*:
  This value is "0" if the signal data is available in the HD store for the entire duration of the time period and "1" if some data is missing. *ibaPDA* sets the value for all open time periods in tracking mode to "1" when the acquisition is stopped.

Info field columns
You can choose from the following options:

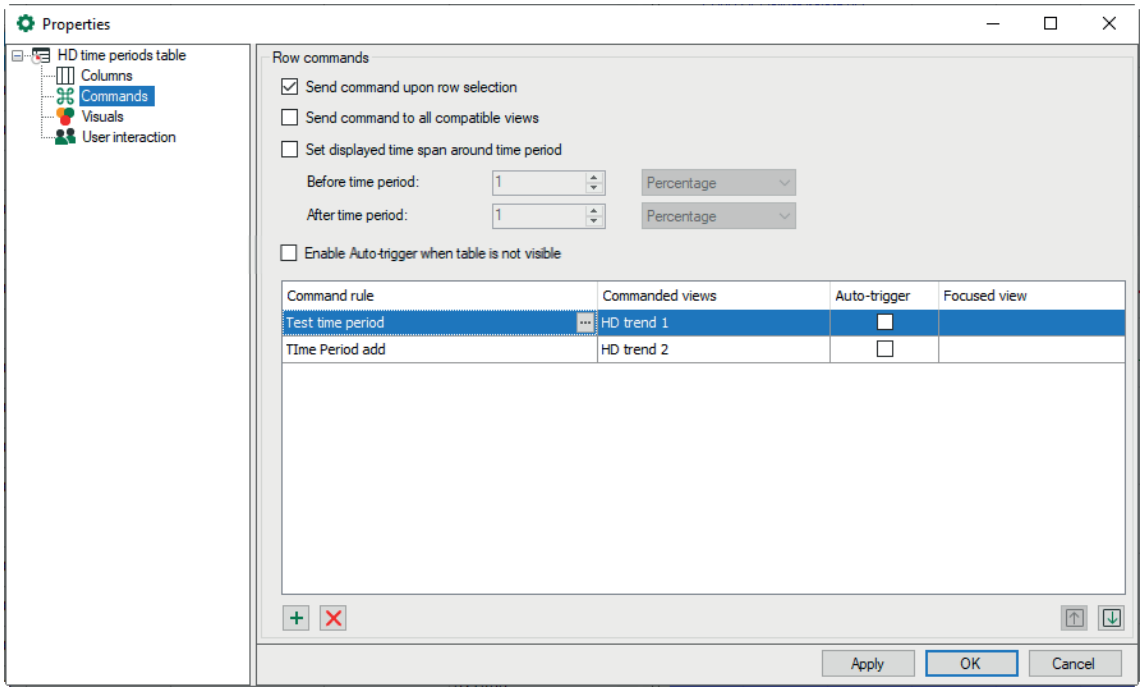
- Show all available info fields in context menu

  If enabled, you can show and hide all info fields defined in the various events as columns via the context menu of the HD time segment table.
- Show only selected info fields in the context menu

  In the list, select the info fields that are to be available as additional columns in the HD time period table. You can show and hide these via the context menu.

13.4.3 Commands

Time period tables can be linked to HD trend graphs, which means that when you double-click on a row in the table the HD trend graphs show the time period selected in the row from start to end.



The *Send command to all compatible views* option is enabled by default. This means that when you double-click on a row in the table, all HD trend graphs show the selected time period. In this case, the command rules are disabled.

If you enable *Send command upon row selection*, you only have to select a row to execute the command, instead of double-clicking on the row.

The *Set displayed time span around time period* option enables you to add an additional time span before and/or after the period in which the signals are shown in an HD trend graph.

If you only want certain HD trend graphs to show the selected time period, disable the *Send command to all compatible views* option and configure appropriate command rules.

A command has the following properties

- Command rule: The rule defines the time period to which the command applies. You can select time period stores or define conditions for columns.
- Commanded views: Here, you can select the views that you want to switch to the selected time period.
- Auto-trigger: If Auto-trigger is enabled, the command is triggered when a row that meets the rule is added to the table. If Auto-trigger is not enabled, you have to select or double-click on the row, depending on the *Send command upon row selection* option, to trigger the command.

You can configure an auto-trigger for three different situations:

- Start time: The related trends jump to the start time of a new time period.
- End time: The related trends jump to the end time of a new time period.
- start time/end time: The related trends show the complete time span from start time to end time after a time period has ended.

■ Focused view: Here, you can select whether you want to enable a different view after the command has been triggered.

Note

The list of command rules is checked from the top down when you double-click on a row in the table. When the first applicable rule is found, the check is finished and the command is forwarded to the assigned views.

You can change the order using the arrow buttons to move the selected row up or down.

Configure command rule

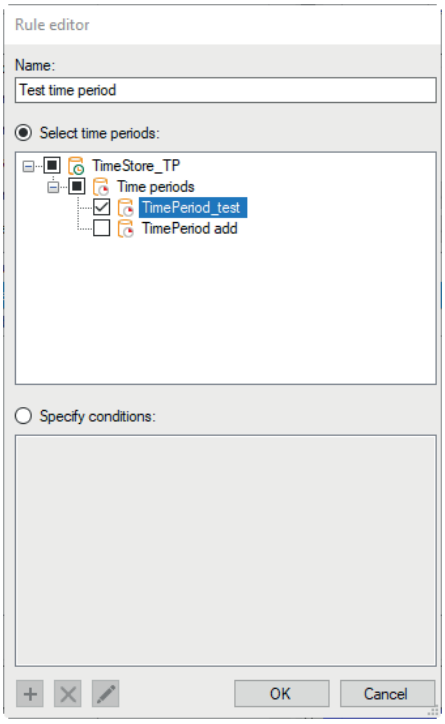
You can change the order using the arrow buttons to move the selected row up or down.

+ Adds a new rule.

✕ Deletes the selected rule.

Opens the editor for editing the rule

Add a new command rule. The *Rule editor* dialog opens.



First give the rule a meaningful name. A rule provides two mutually exclusive methods.

If you enable the *Select time periods* option, you can select one or more time periods in the tree below that trigger a switching of the HD trend view when a row is either selected or double-clicked.

If you enable the *Specify conditions* option, then you can enter check conditions in the field below that are executed when double clicking on a row. In auto-trigger mode, the conditions are checked continuously.

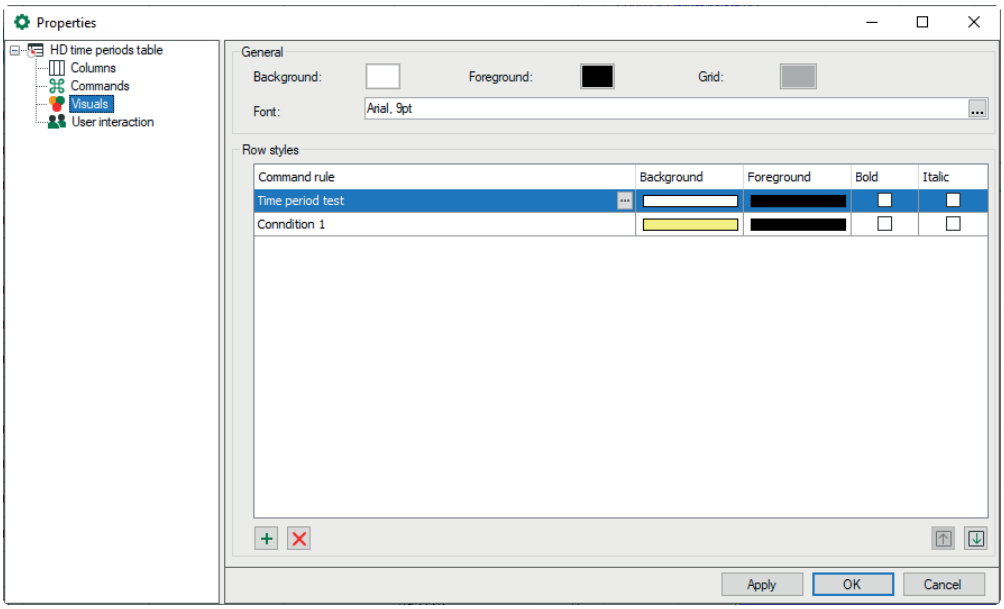
If a condition is met, the trend view is switched.

Configuring conditions

The procedure for configuring a condition is the same as that for event tables. See chapter ↗ *Commands*, page 144.

13.4.4 Visuals

Set up the appearance of the time periods table here.



General

Here, you can set the overall appearance of the table, i.e. the background and foreground color, the color of the grid and the font.

Row styles

The row style can be controlled depending on the properties of the time periods. In the *Row styles* list, you can create rules and associated style modifications.

+ Adds a new rule at the end of the list.

✕ Deletes the selected rule.

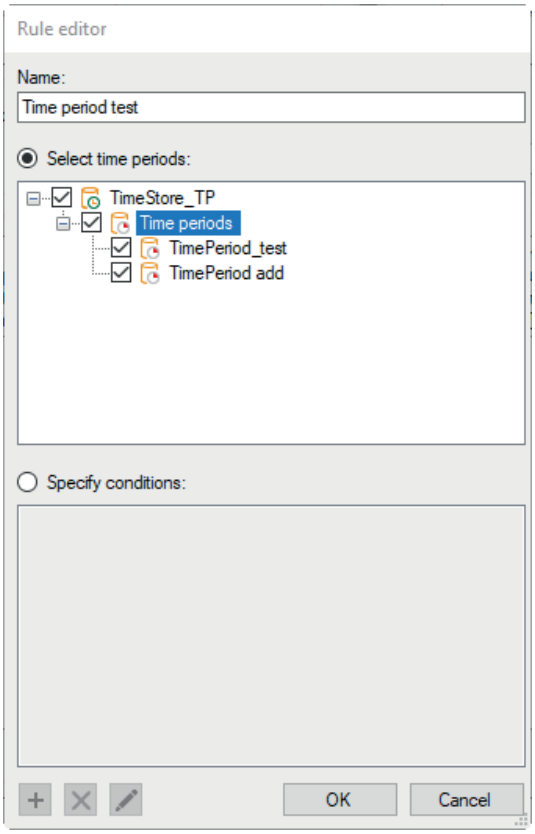
… Opens the editor for editing the rule

Note

Double-clicking checks the list of rules from top to bottom. When the first applicable rule is found, the check is finished and the style modification is made to the row.

You can modify the sequence of the rules: Use the button to move the selected rule up and the button to move it down.

Add a new command rule. The *Rule editor* dialog opens.



First give the rule a meaningful name. A rule provides two mutually exclusive methods.

If you enable the “Select time periods” option, you can select one or more time periods in the tree below that will be displayed using the row style corresponding to the rule.

Tip

This method is suitable if you want to assign a row style to certain time periods.

If you enable the *Specify conditions* option, then you can enter check conditions in the field below that are checked continuously during acquisition. If a condition is met, the corresponding time period is assigned the relevant row style. A condition can consist of one or more expressions that are linked with each other.

Adds a new expression.

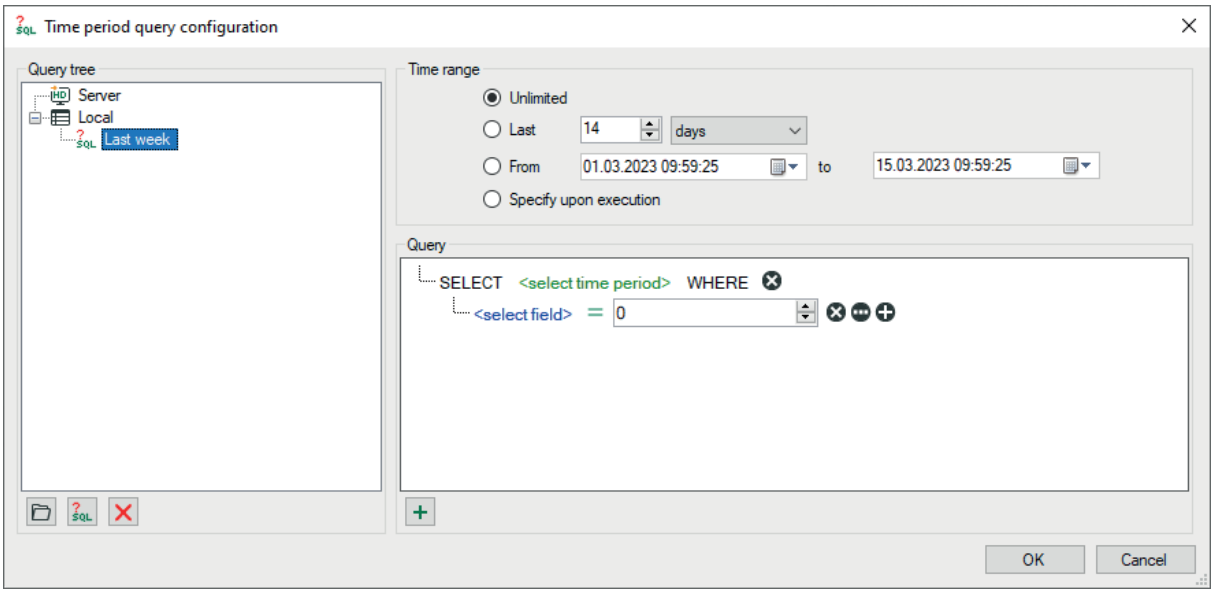
Deletes the selected expression.

Opens the editor for editing the expression.

13.5 Configuration of time period queries

In the HD time periods table toolbar, you can filter the rows displayed by particular criteria. Enter a value or configure a query by clicking on the icon. If you remove a value, the filter is deleted.

To configure time period queries, open the query drop-down menu and click on the <Edit queries> button.



Note

The event queries are stored on the *ibaHD server*. This means that the queries are available to all ibaHD clients (*ibaPDA*, *ibaAnalyzer*) if they are connected to the *ibaHD-Server*.

Note

The event queries are saved on the *ibaHD-Server* or locally, depending on the configuration.

Configuring queries

All existing queries are shown in the query tree. You can arrange queries by using a directory structure.

	Adds a new folder at the selected position
	Adds a new query at the selected position
	Deletes the selected node and any secondary nodes

There are two types of queries:

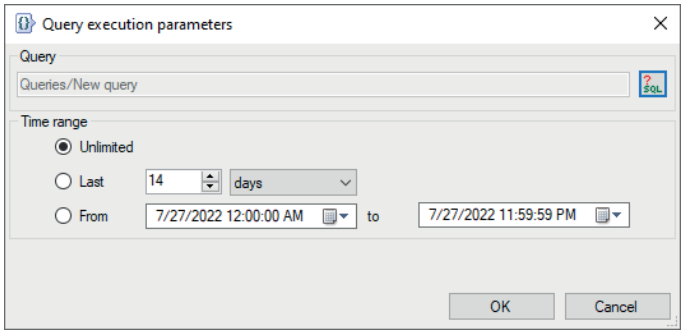
- *Server queries* are saved on the *ibaHD-Server*, which means that all connected clients can use the same server queries.
  Note: Only users who have the right *Edit ibaHD queries* can save or edit server queries.
- *Local queries* are saved in the layout. Other connected clients cannot see these.

  Local queries are saved in the query. Other connected clients cannot see these.

Query range

Every query is executed over a defined time range. There are several options for restricting the query range.

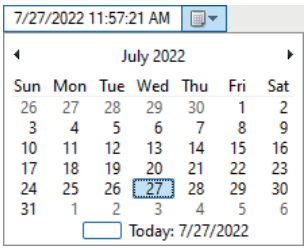
- Unrestricted: The query covers the entire HD recording
- Specify upon execution: After clicking on the start button for the query, a dialog appears in which you have to specify the query range



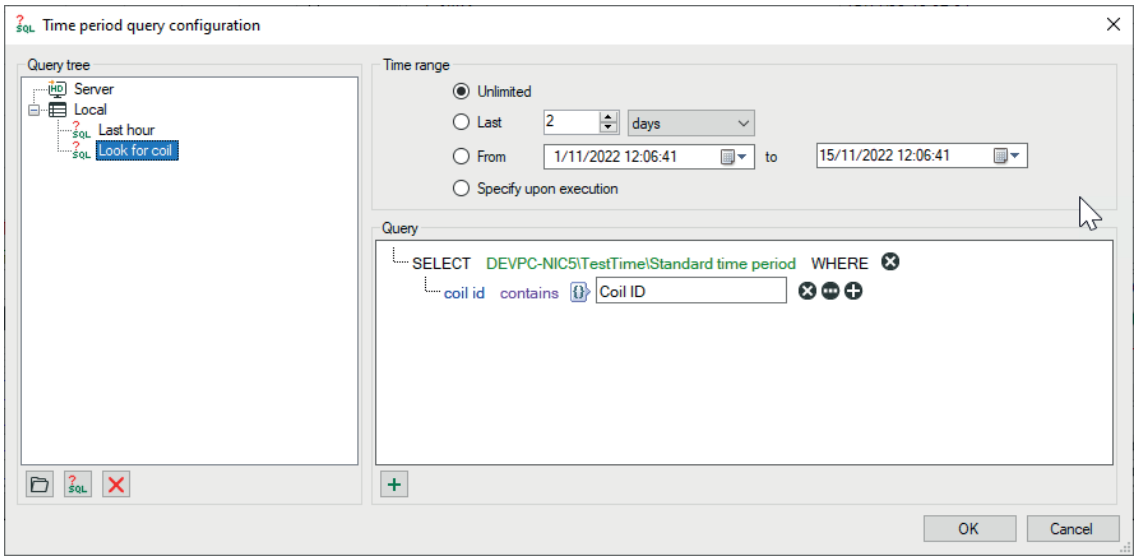
  Clicking on <OK> executes the query.
- The last xx minutes/hours/days/weeks/months: The query only covers a fixed period of time from the start of the query, e.g. the last 8 hours for a shift overview.
- From *Date* to *Date*: The query covers the time range between the set days (inclusive).

Tip

To set the current date, open the Date & Time Picker by clicking on the down arrow. Here, you can select the entry under *Today…*.



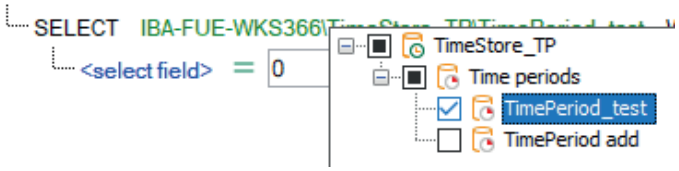
Query details



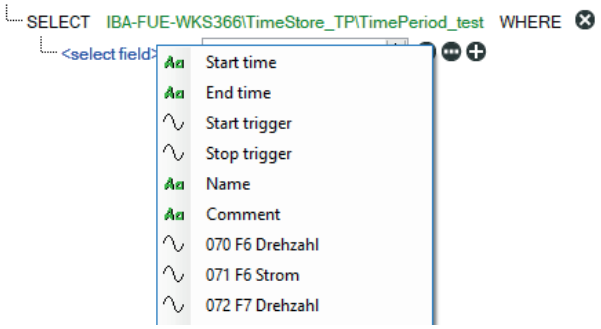
Here, you create the actual query using notation similar to SQL.

The query begins with a SELECT statement.

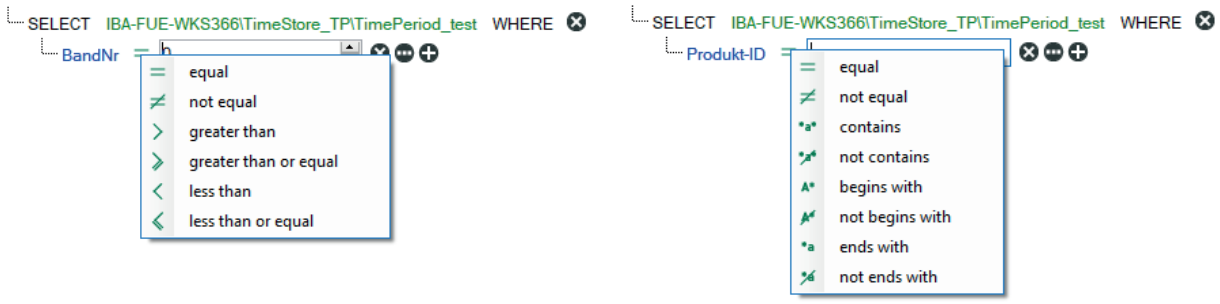
Clicking on the green text opens the tree structure for the configured time periods. Select the time period you want to search for.



Click on the blue text to select a column in the table or any existing information fields.



Depending on the selected field (numeric or text field), clicking on the green equals symbol displays a list of operators. Select the required operator.



Behind the elements in the query, there are different buttons which you can use to extend or modify the query:

Deletes the expression

Switches the right side of an expression between a fixed value and a field

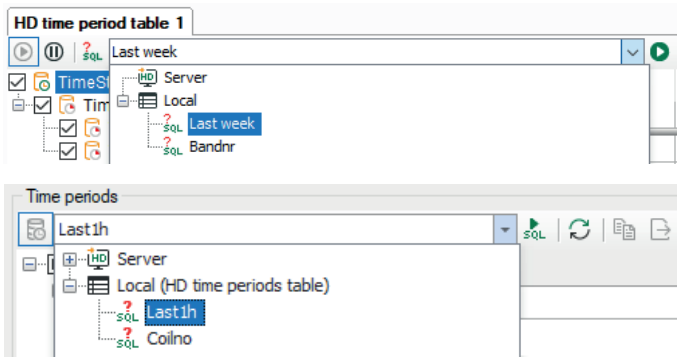
Adds a new expression to the instruction

You can change the logical operator that is linked with a group of expressions by clicking on the operator.

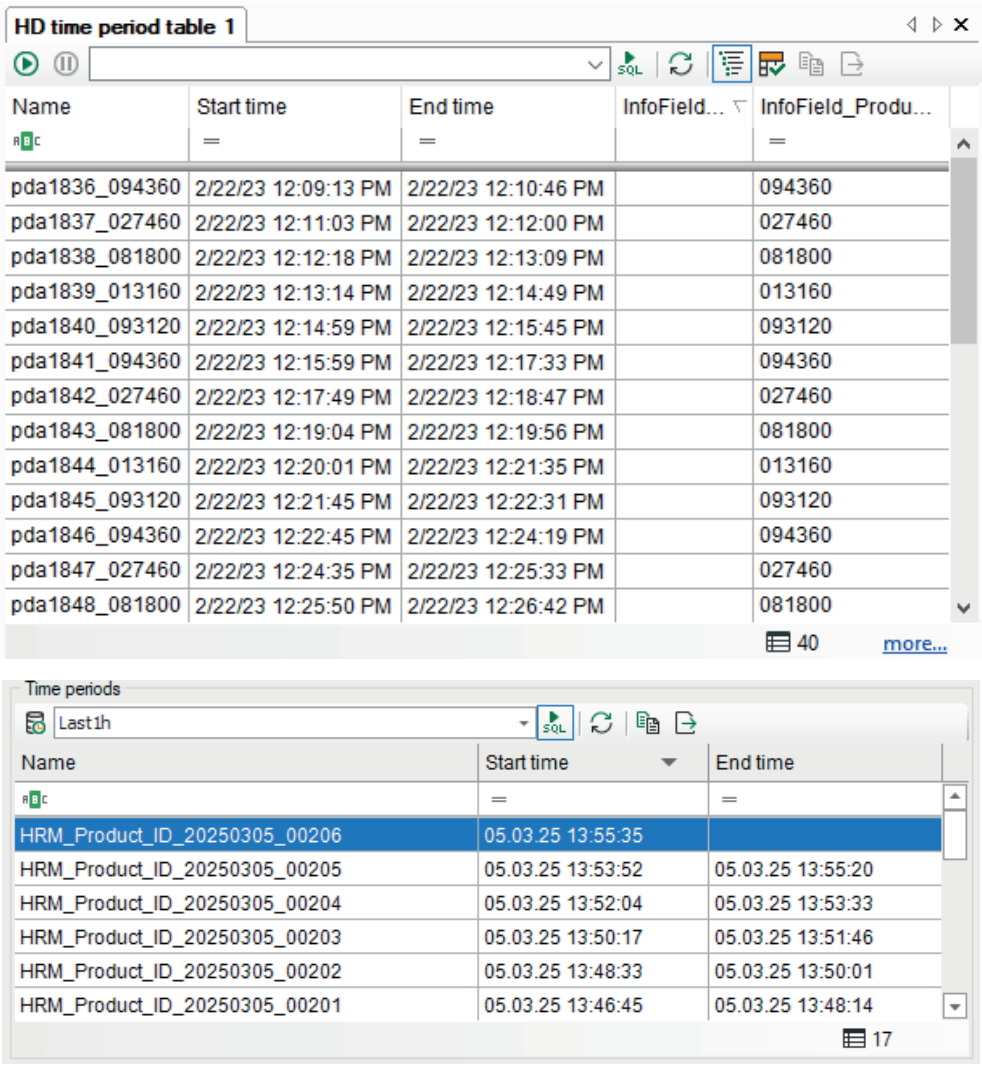
Clicking on <OK> closes the editor and saves the created queries.

Executing the query

First select the relevant query from the list in the toolbar and then click on the button to the right.



When the query has been completed, the number of results found is displayed in the status bar on the right.

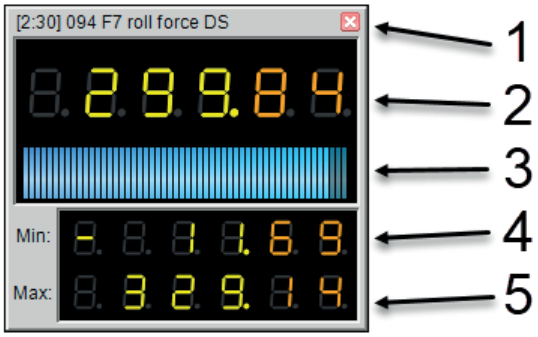


A query can return more results than the number of rows configured in the table. To display any other results, click on the blue "more..." link.

You can return to live mode by clicking on the button.

With the button you can return to the display of the latest available time periods.

14 Numeric digital display

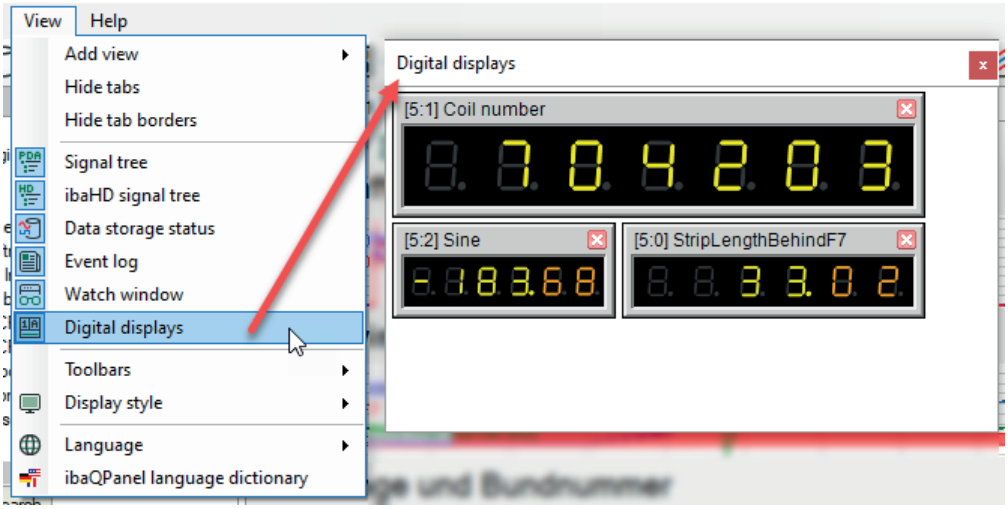


1	Signal name, dock and close button
2	Actual value
3	Range bar
4	Minimum
5	Maximum

A numeric digital display shows the actual value of an analog signal. Optionally, it can be selected the minimum and maximum values of a signal for display (resettable). It can also show a range bar showing the position of the current value between the minimum and maximum signal value. Minimum and maximum value as well as the range bar can be enabled or disabled in the context menu. The digital displays can be positioned freely floating or docked to a digital meter window. Size, number of digits and color of the value indicator can be configured.

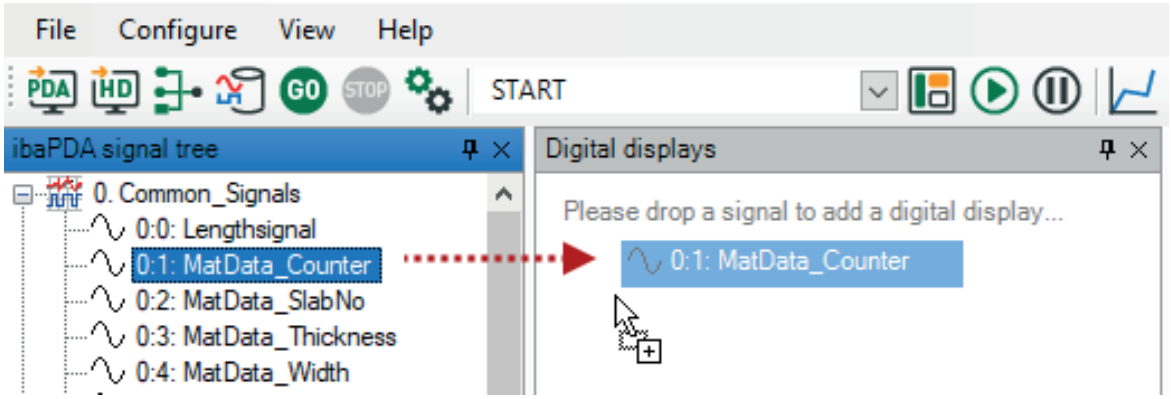
Adding a digital display to the view

Usually, the digital displays are attached to a special window. The digital display panel can be opened using the *View – Digital displays* menu. If no digital meter has been added yet, the panel will be empty.

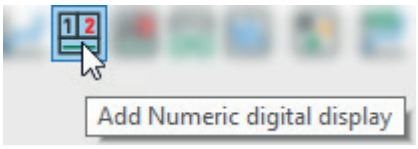


You have several possibilities to add a digital display to the panel:

- Drag a signal from the signal tree into the digital meter panel and let it drop.
  A digital display displaying the actual value is automatically added.

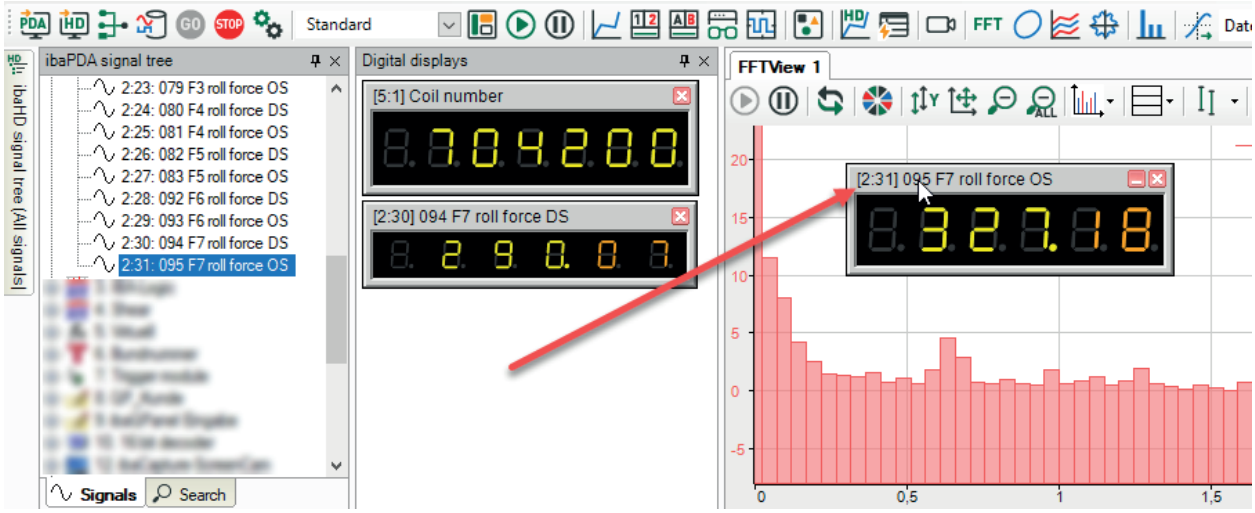


■ Click on the corresponding tool button in the toolbar.



The digital display panel opens and a new digital meter is added to it, yet without a signal assigned. The easiest way to transfer a signal to the digital display is to drag and drop the signal out of the signal tree.
Alternatively, you can enter the signal name in the "Properties" dialog of the display.

→ If you want the digital display to be positioned somewhere else as a floating window instead of being attached to the digital display panel, drag it out of the digital display panel. This allows you to move the digital display, e. g.over the trend views.



Changing the size of the digital display

You can enlarge a digital display in its size by dragging the edges or corners of the display using the mouse. Basically, the size is limited by the screen. If a digital display is docked to a corresponding window, you can reset it back to the default size using the context menu of the digital display window.

A right mouse click in the digital display pane opens the context menu containing 3 functions:

- Clear
  This function deletes all digital displays, docked as well as floating displays.
- Reset size
  This function resets all digital displays docked in the window back to default size.
- Dock all
  This function draws back all floating digital displays and attaches them to the window without resizing.

Setting up the digital display

For setting up the properties of a digital display, open the context menu.

In the *Properties* dialog, you can specify the following:

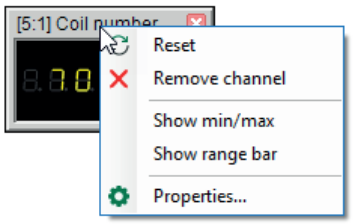
- The data source signal and reset signal
- Display update interval
- Display colors
- Style and format of the value display
- Range bar color

All settings are also available in the "Preferences" dialog except the data source and reset signals.

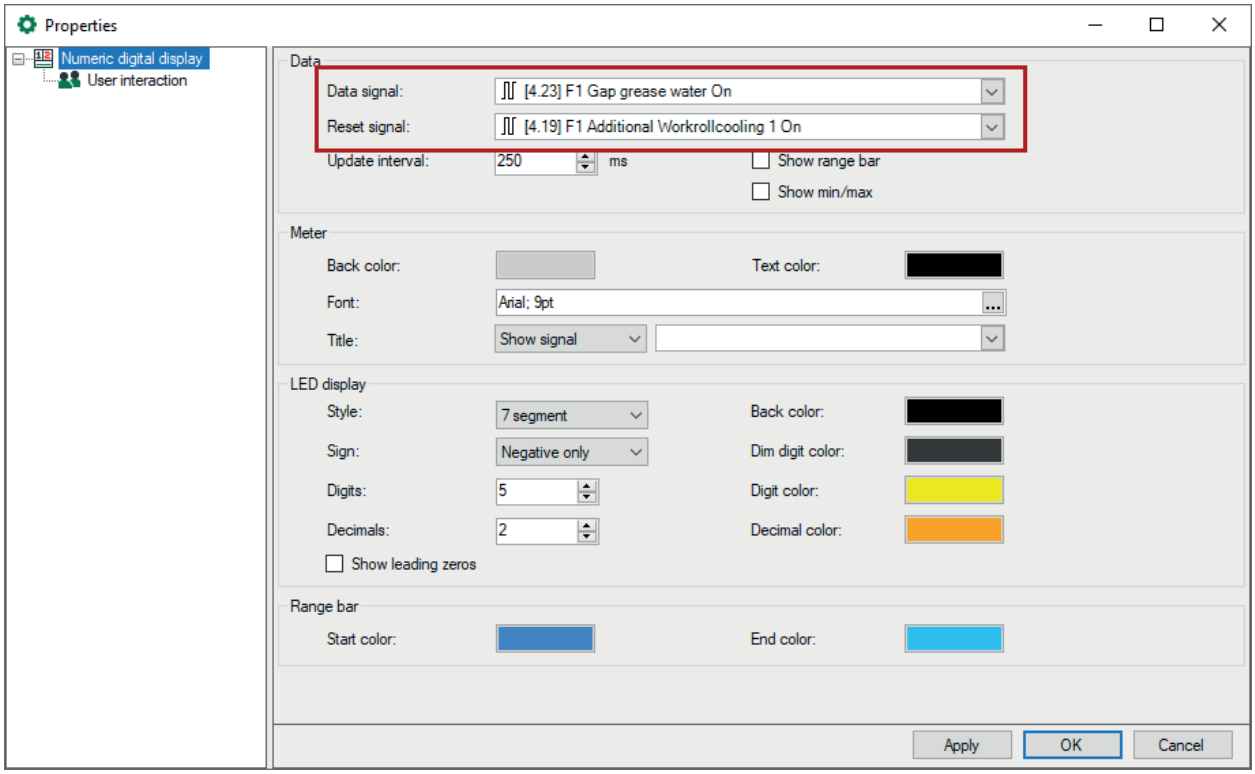
For further information, see chapter "↗ *Properties Numeric digital display, page 171*."

Context menu

For individual setup of a digital display, for enabling/disabling the range bar and resetting the min/max values, right-click on a display to open the context menu.



14.1 Properties Numeric digital display



Preferences and properties of digital displays are basically the same except for the signal selection. Data signal and reset signal selection is available in the "Properties" dialog only (red frame in the picture above).

Data

Data signal

Select here the analog signal, whose value is to be displayed in the digital meter display.

Reset signal

Select here a digital signal, which is to reset the min/max values of the digital meter.

Update interval

The update time determines the refresh rate for the value display. It is recommended setting a reasonable value (between 250 and 1000 ms) with regard to the human eye.

Show range bar, show min/max (Numeric digital display only)

Enabling this option will add a range bar or the minimum and maximum indicators to the display.

Frame and labeling

Here, you can set up the color scheme of the digital display's frame. By default, the digital display is labeled with the name of the related data signal. Alternatively, you can enter a title instead of the signal name. The font setting applies to both the signal name and the title.

LED display

Here, you can set up the value display.

Style
Choose between a style containing 7 or 14 segments.

Sign
Decide how to deal with positive/negative signs.

- Never: Neither "+" nor "–" are displayed
- Negative only: "+" is not displayed
- Always: Both "+" and "-" are displayed

Digits and decimals
The number of digits determines the total number of digits in the display, including the integer part and decimal part. If you select "Negative only" or "Always" for sign, an additional digit will be added automatically.

If you set decimals to 0, the value on display will be rounded to the next integer.

Colors
Set up your preferred colors for the different parts of the display.

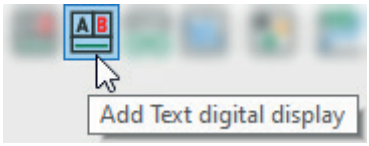
Range Bar

Set up the start color and end color of the range bar in order to get a color gradient from green to red.

15 Text digital display

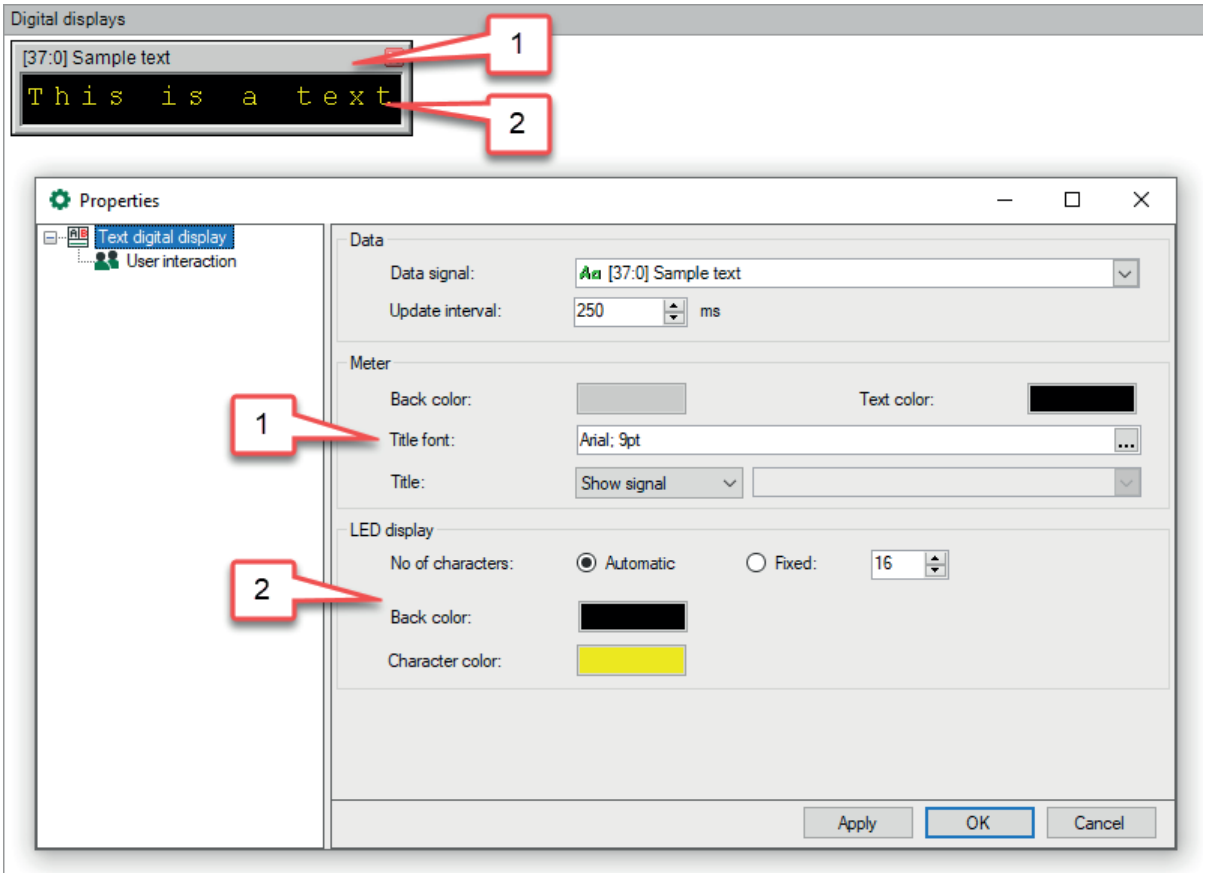
Text digital displays are used to display text signals. Up to 256 characters are allowed.

In terms of basic operation, this element is similar to the digital numeric display (see ↗ *Numeric digital display*, page 168). You can also add the digital text display via the view menu or the toolbar and use it in *ibaQPanel*.



The fundamental difference is that the source signal for the display is text signal. The text can be both received from the outside as well as entered via a text input element in *ibaQPanel*.

The following figure shows the relationship between the display and the settings:

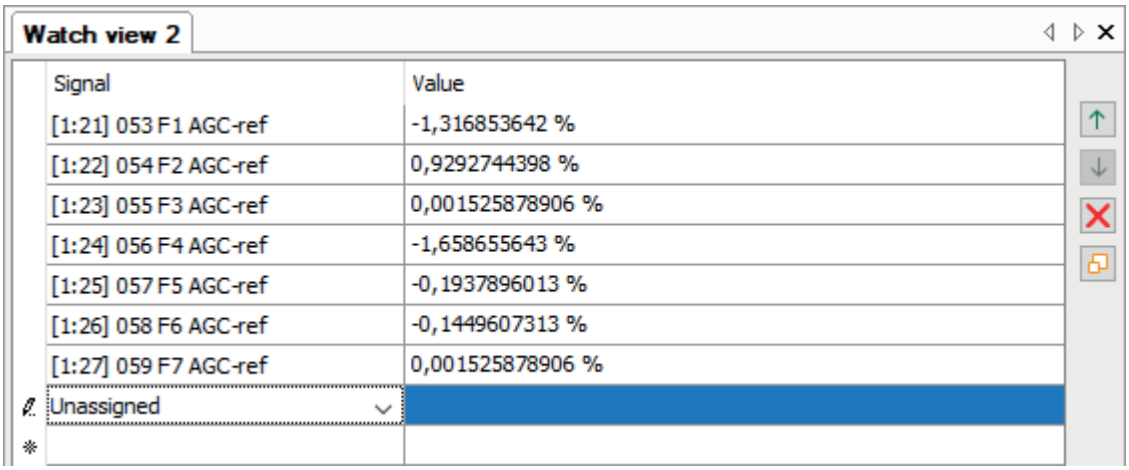


1. Properties of frame and caption, headline optional
2. Properties of the text display; the font size of the display text is calculated automatically and can only be influenced by resizing the text display (using the mouse).

16 Watch view

You can use the watch view to monitor the values of any signal in a tabular manner. Numeric signals as well as text signals are supported. Signals of an *ibaPDA* server and of an *ibaHD-Server* can be displayed. You may open and position as many watch views as you like. The watch view is also available in *ibaQPanel*.

You can open the watch view either via the toolbar button or by using the *View – Add view – Add Watch view* menu.



You can drag one or more signals out of the *ibaPDA* or *ibaHD-Server* signal tree and drop them into the watch view. The number of rows is adapted automatically. Alternatively, you may add signals to the view by clicking on the drop-down icon in the *Signal* column and select the signals in the respective signal tree there.

With the buttons on the right margin of the view you can move or delete the rows. You can select multiple lines by mouseclick + <Ctrl> or mouseclick + <Shift>.

	Move the selected row(s) up by one position.
	Move the selected row(s) down by one position.
	Remove the selected row(s) from the view.
	Group the selected row(s) in a separate section.

Table 1: Control buttons for the watch view

With the lowest button *Assign rows to a section* you can organize the view into two or more sections.

Select one or more rows and click on the button .

A dialog pops up where you can either enter a name for a new section or select an existing section to assign the signals.

Moving the rows with the control buttons only work within the section. If you want to assign a row to another section, use the button *Assign rows to a section* .

If you select a section, the control buttons apply to the entire section. Thus, you can move or delete a section. If you delete a section, all rows assigned to this section will be deleted too. If you want to avoid this, you should first select the rows inside the section and assign them to another section.

You can rename a section by making a right-click on the section title and selecting *Rename...* in the context menu.

Properties

You can open the properties of the watch window by making a right-click anywhere in the watch window and choosing *Properties...* in the context menu.

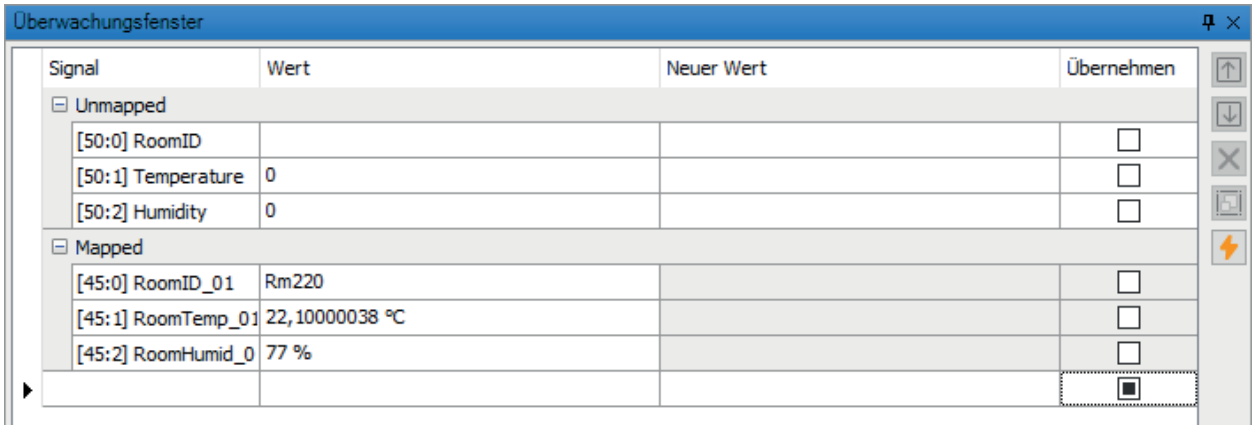
The following properties are available:

- Numeric value formatting: Beside the automatic formatting you have the possibility to customize the formatting of numeric values in terms of number of digits before and after the decimal point as well as the display of positive values.
- User interaction: If you enable this option, users can change signals in the view (add, replace, delete) even if they don't have the right to change layouts but the right to interact with views. If you disable this option, users must have the right to change layouts in order to change signals in the view.

Special function for signal mapper signals

Basically, signal mapper signals behave like all other signals. However, if no source signals are mapped to these signals, they are handled slightly different in the watch view.

If the watch view contains signal mapper signals without having assigned source signals, another button with a lightning symbol and two more columns will appear automatically.



Thereby, you can manually assign values to these unmapped signals. For signals which do not apply to this function, the table cells are grey.

First, check the checkbox in the Apply column for the signals you want to assign a value to. Then, enter your preferred value in the New value column.

In order to apply the values to the signals, click on the lightning button at the right side or press <F6> on the keyboard. Make sure that you don't edit a cell value at this time, otherwise, the hotkey won't work. Digital signals can easily be flipped by pressing <F8> .

The New value and Apply columns support header clicking. It means that when clicking the column header, the contents of the selected cell for that column will be copied to the subsequent rows.

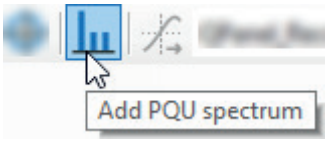
Note, that ibaQPanel signals can also be written using the watch view.

17 PQU spectrum

If you collect the measurement data from an *ibaPQU-S* device to measure the electrical power quality, you can define spectrum submodules in the configuration of the device module. These modules calculate the harmonics and interharmonics of voltages and currents up to 50th order.

Using the *PQU spectrum* view, the harmonics and interharmonics can be clearly illustrated as a bar chart. The limit values for individual harmonics are displayed as horizontal marks on the bars.

You can also add the PQU spectrum using the *View* menu or the toolbar:



You can only assign one PQU spectrum module to one *PQU spectrum* view.

Other documentation

You can find more information on the PQU phasor view in the manual for the product *ibaPQU-S*.

18 Phasor view

If you collect the measurement data from an *ibaPQU-S* device to measure the electrical power quality, you can define phasor submodules in the configuration of the device module. This module computes the amplitudes and phase angles for voltages and currents for the three phases.

By using the PQU phasor diagram, the voltage and current values are presented clearly. You can also add the PQU phasor diagram using the *View* menu or the toolbar:

You can only assign one PQU phasor module to one PQU phasor diagram.

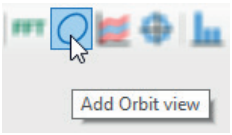
Other documentation

You can find more information on the PQU phasor diagram in the manual for the product, *ibaPQU-S*.

19 Orbit view

The orbit view represents the position of a shaft inside a sleeve bearing. It is supplied with data by an ibaInSpectra Orbit module.

You can open a new orbit view by clicking on the respective button in the tool bar:



Alternatively, you can use the menu *View - Add view... - Add Orbit view*.

Subsequently, drag the respective ibaInSpectra Orbit module from the signal tree into the orbit view.

Other documentation

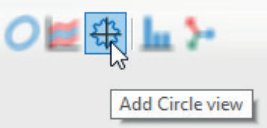
Detailed information about the configuration and The methods of Orbit view signals can be found in the *ibaInSpectra* product manual.

20 Circle view

The Circle view represents the results of an *ibaInCycle Expert* module and of an *ibaInCycle Auto-Adapting* module in *ibaPDA* when applied to rotating processes.

A circle view may include one or more *ibaInCycle* modules.

You can open a new Circle view by clicking on the respective button in the tool bar:



Alternatively, you can use the menu *View - Add view... - Add Circle view*.

Subsequently, drag the respective *ibaInCycle* module from the signal tree into the Circle view.

Other documentation

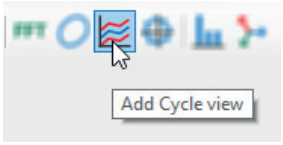
Detailed information about the configuration and operation of the Circle view can be found in the *ibaInCycle* product manual.

21 Cycle view

The Cycle view represents the results of an *ibaInCycle Expert* module and an *ibaInCycle Auto-Adapting* module in *ibaPDA* when applied to cyclic or periodic processes.

A Cycle view may include one or more *ibaInCycle* modules. The charts can have individual value axes or lie on a common value axis

You can open a new Cycle view by clicking on the respective button in the tool bar:



Alternatively, you can use the menu *View - Add view... - Add Cycle view*.

Subsequently, drag the respective *ibaInCycle* module from the signal tree into the Cycle view.

Other documentation

Detailed information about the configuration and operation of the Cycle view can be found in the *ibaInCycle* product manual.

22 Zoom function

22.1 Zooming in a trend graph view

The zoom function works on both X and Y direction. Place the left mouse cursor in the graph, hold the mouse key depressed and drag it until the desired area is surrounded by a frame. When releasing the mouse key, the designated area will be zoomed in. If you zoom in a graph, the X axis parts of all graphs located in the same view also get zoomed. A view can always have only one time base for all graphs contained. Only parts of the Y axis can be zoomed individually. In the scrolling display, zooming stretches the time base and enlarges the display. The signals, however, move faster, as the same geometrical length of the X axis now refers to a shorter time slice.

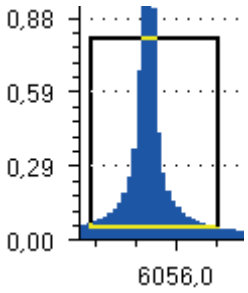
Zooming first applies to the X and Y direction. In the zoomed-in condition, the scale in the Y direction can be changed at any time without affecting the zoomed section of the X axis. Auto-scaling in the Y direction applies to the values in the zoomed area (= visible). Zooming with the mouse keeping the <SHIFT> key pressed will only zoom the X axis.

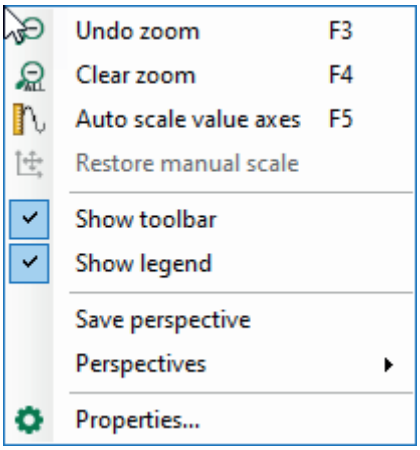
Zooming out can be carried out in steps using the button or the key <F3>. Every click successively reverses all previous zoom steps.

Use the button or the key <F4> to reactivate the original, non-zoomed display.

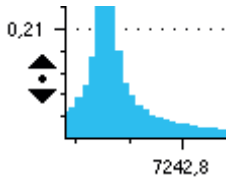
22.2 Zooming in the FFT view

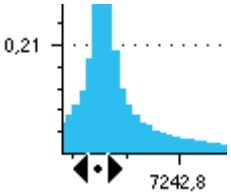
It is possible to zoom in to a certain area of the spectrum by drawing a zoom rectangle (keep left mouse key depressed) or by using the context menu (right mouseclick).





Once zoomed, you can pan the spectrum by dragging one of the visible axes in the desired direction. The pan cursor appears when moving over an axis.





You can also use the mouse wheel to zoom in or out around the center of the axis. Note that you can never move outside the scale extremes of an axis.

23 Managing layouts

Note

The layout management has been remodeled completely by release of *ibaPDA* v8.0.0.

If you want to migrate from a former *ibaPDA* version to a version v8.0.0 or higher, please refer to the NewFeatures document of version v8.0.0!

23.1 Definition

The term “layout” denominates the arrangement of the ibaPDA client’s screen. Parts of a layout are all visible elements of the ibaPDA client, which are used for visualization and operation such as

- Type, number, size and position of signal views
- Signal assignment of the views incl. color, font size, feed direction etc.
- Type, size, position and visibility of standard windows such as signal tree, data storage status etc.
- Structure and visibility of menue and tool bars

Moreover, the layouts have the following properties

- A layout can be designed and saved individually by the user.
- Any number of layouts can be created.
- Layouts can be switched over while acquisition is running.
- Layouts can be assigned to one or more users.
- The Layout Manager provides for clearly arranged administration of layouts.

23.2 Internal structure and file storage

Each layout is stored internally in a file (*.layout).

All single layout files will be stored together in a ZIP-file (*.layouts), the so called layout set.

An index file (LayoutConfiguration.config) manages all corresponding layouts.

The images from *ibaQPanel* views are stored in a backup file (BackupStorage.storage).

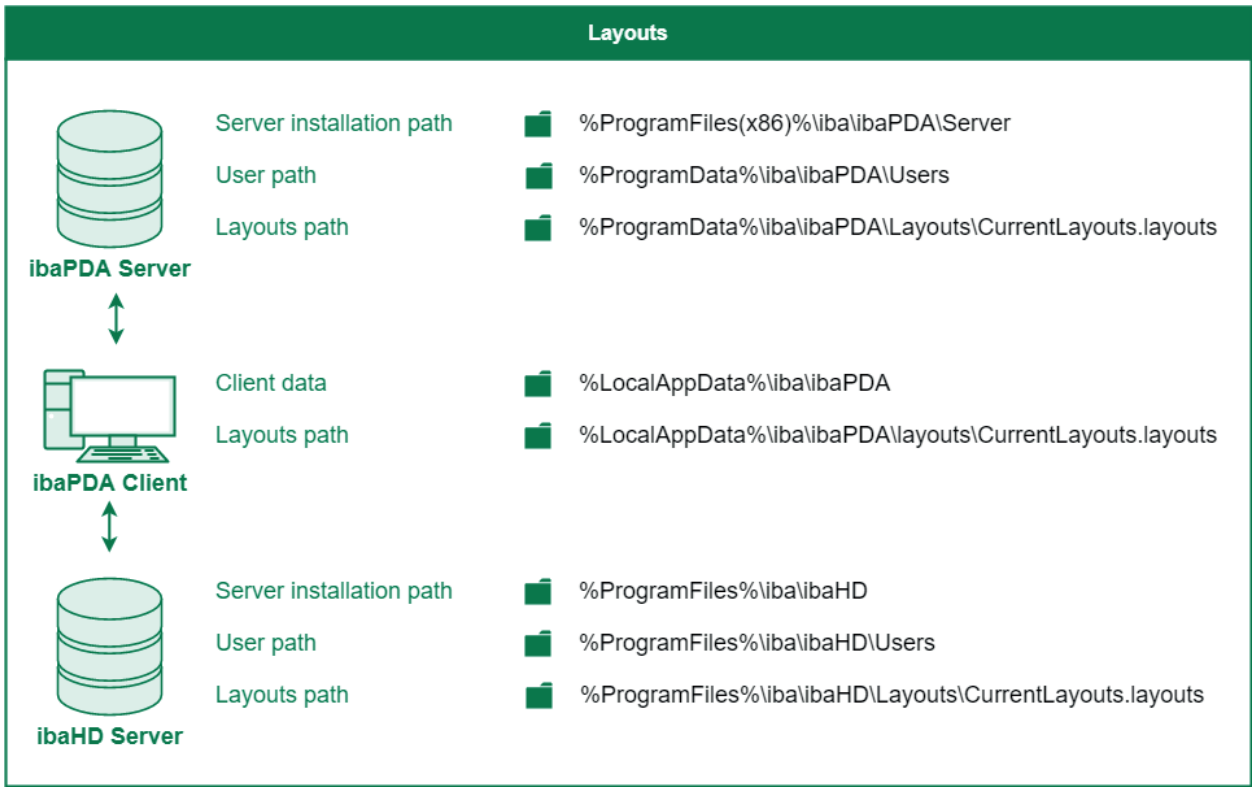
The *ibaQPanel* text translations are stored in a dictionary file (Dictionary.dict).

All these files are XML-files.

The user decides where the files should be stored:

- locally on the client
- on the *ibaPDA* server or
- on the *ibaHD-Server*

The following picture shows an outline of the storage locations



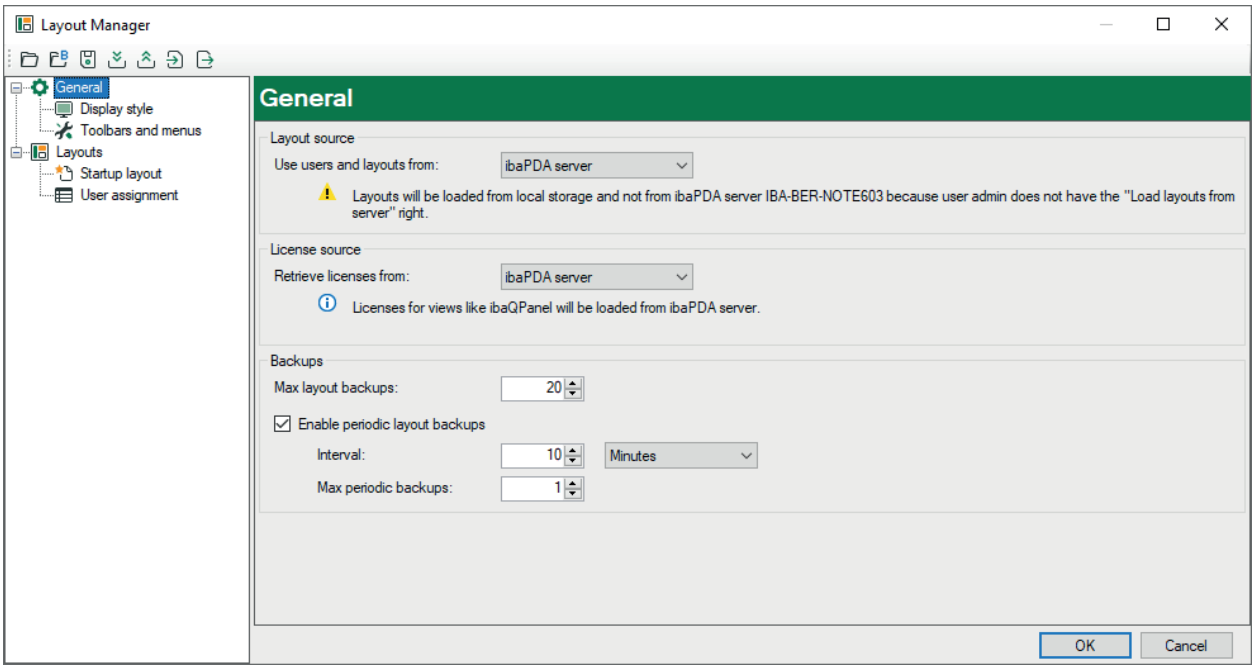
23.3 Layout Manager

The Layout Manager is the central dialog to administer all layouts in a clear arrangement. The following functions are available:

- Add, remove and duplicate layouts
- Load, save, import and export layouts
- Arrange layouts in a folder structure
- Assign layouts to different users
- Define startup layouts for users and computers
- Focus general properties of layouts at one place, e.g. layout source (ibaPDA server or ibaHD-Server), display style, backup strategy, user-depending menus and tool bars etc.

You can open the Layout Manager as follows:

- via the symbol in the tool bar
- via the menu *Configuration - Layout Manager...*
- via the shortcut <Ctrl>+<L>



The user interface of the Layout Manager is divided in three sections:

On top there is the toolbar with the buttons for loading, saving, importing and exporting of layouts.

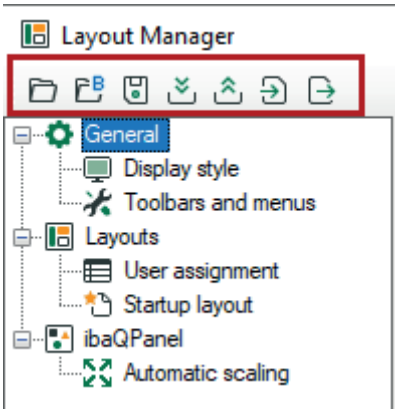
On the left side you'll find the window with the navigation tree, consisting of the following nodes:

- *General* for general settings like layout source, license source and backup strategy
  - *Display style* for setting up the layout display (full screen etc.)
  - *Toolbars and menus* for setting up the visibility of symbols and menus
- *Layouts* for creating and organizing layouts
  - *Startup layout* for setting up user-specific layouts at start of *ibaPDA* client
  - *User assignment* for assigning layouts and users

The right and more prominent section consists of the settings with regard to the selected node in the tree.

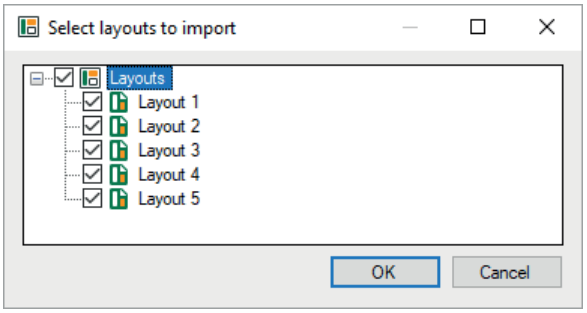
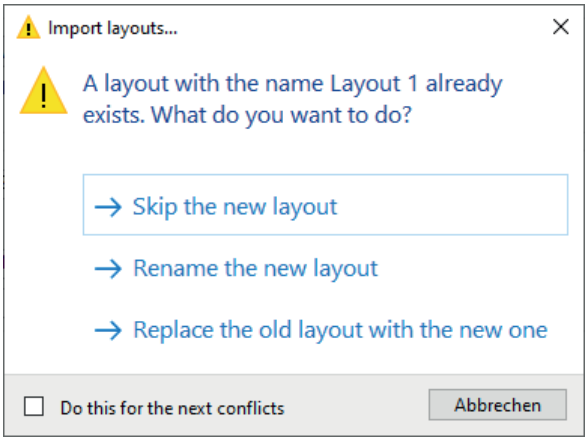
23.4 Toolbar for basic functions

In the topmost toolbar of the Layout Manager you'll find the buttons for loading and importing of stored layouts as well as saving and exporting of new or changed layouts.



The buttons provide functions as follows:

Symbol	Function	Description
	Load layouts	Opens a set of layouts from the file system. You can load layouts from new layout files (*.layouts) or old layout files (*.lay). All loaded layouts at this time will be discarded and replaced by the new layouts.
	Load layouts from backup directory	Opens the directory where the backups are stored. Here, the user can select one of the automatically created backups. See also information about setup of the maximum number of backups ↗ *General settings, page 189*.
	Save layouts	Saves the current layout set in the file system.
	Load layouts from server	Loads all layouts from the server which is entitled as layout source on the *General* node. All loaded layouts at this time will be discarded and replaced by the new layouts. This is only permitted if the logged in user has the right "Load layouts from server" in the user rights or is user "admin".
	Save layouts on server	All currently available layouts will be saved on the server which is entitled as layout source on the *General* node. The entire set of layouts will be saved. It is not possible to save single layouts The previous layout set stored on the server will be replaced by this. The function is only permitted if the logged in user has the right "Save layouts on server" in the user rights or is user "admin". If there are other logged in users with the right "Load layouts from server", they will be notified, that there is a new layout set available. They can then choose to load the layouts from the server.

Symbol	Function	Description
	Import layouts	Layouts will be imported from a layout file in the file system and added to the currently available layouts. These can be old layout files (*.lay) or layout files of the current version (*.layout, *.layouts). The layouts to be imported can be selected from the layout collection in a dialog.  If a layout to be imported already exists (same name) a dialog opens to resolve the conflict.  ■ *Skip the new layout*: The layout will not be imported. The original layout with the same name will remain. ■ *Rename the new layout*: The new layout will be imported and the name will be extended by the suffix “-copy”. ■ *Replace the old layout with the new one*: The new layout will be imported and overwrites the original layout.
	Export layouts	The current layouts will be exported into a layout file either as a complete layout set or part of it. A dialog opens for selecting the desired layouts. Tip: If there are layouts to be exported, which are already selected under *Layouts*, then you can use the command *Export layouts* in the context menu of the layout pool. In the subsequent export dialog the corresponding layouts are already selected.

Note

It is also possible to open layout files of previous *ibaPDA* versions (v7.x). They will be automatically converted.

For more information about migrating from ibaPDA-v7 to v8 see the document "NewFeatures v.8.0.0".

23.5 General settings

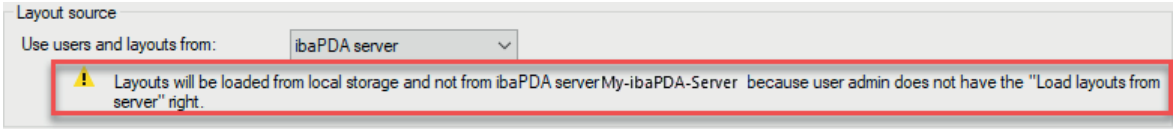
Source for users and layouts

You should determine the server the layouts will be loaded from and saved to with regard to the *ibaPDA* client you are working on. The following settings are available for selection:

- *Local storage*: Layouts will be loaded from or saved to a local drive of the client only.
  - ZIP file containing all layouts: `C:\Users\MyUsername\AppData\Local\iba\ibaPDA\CurrentLayouts.layouts`
  - Single layout files: `C:\Users\MyUsername\AppData\Local\iba\ibaPDA\layouts`
- *ibaPDA-Server*: Layouts are available on the currently connected *ibaPDA* server and will be loaded from or saved to this server. Regarding the user rights, the user management of *ibaPDA* applies.
- *ibaHD-Server*: Layouts are available on the currently connected *ibaHD* server and will be loaded from or saved to this server. Regarding the user rights, the user management of *ibaHD-Server* applies.

This setting is important with regard to the user rights (Save layouts on server/Load layouts from server) in the user management of the corresponding server. Only if the logged in user has these rights on the selected server he can manually store or load layouts to/from this server.

In case the logged in user has not the rights for saving or loading according to the user management of the selected server, he will be notified by a message.



If you choose *Local storage*, the user management settings will be ignored.

License source

If license-depending views are used in a layout, e.g. *ibaQPanel*, the client must fetch the required licenses in order to get the view working.

Setup here whether the licenses should be fetched from the currently connected *ibaPDA* server or *ibaHD-Server*. If you select “Nowhere”, no licenses will be fetched. Choose this setting, if you are sure that the layout does not contain license-depending views.

Backups

The feature of backups helps to prevent unintended loss of sumptuous designed layouts or to support switching back to a previous configuration step in the course of layout configuration.

You can configure two kinds of backups in the Layout Manager. Both kinds will always store the entire layout set in a layout file with date and time added to the filename.

- Automatic backups: `CurrentLayout_YYYY_MM_DD_HH_MM_SS.layouts`
- Periodic backups: `P_CurrentLayout_YYYY_MM_DD_HH_MM_SS.layouts`

Storage location is the backup path in the folder of the logged in user on the client computer:

`c:\User\Username\AppData\Local\iba\ibaPDA\backup`

Automatic backups

The setting of *Max layout backups* determines the maximum number of automatic backups. Values between 5 and 100 are possible, default is 20. If the number has been reached, the oldest backups will be gradually deleted.

A new backup will be created automatically by each of the following events:

- Closing *ibaPDA* client
- Opening a project
- Generating a support file
- Closing the Layout Manager after applying changes
- Saving layouts on the server
- Connecting an *ibaPDA* server
- Connecting an *ibaHD-Server*

Periodic backups

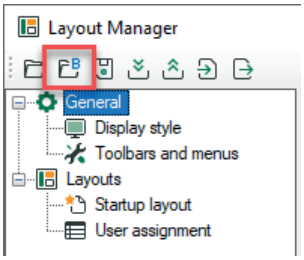
You must enable the periodic backups by a checkmark in order to use them. Periodic backups are particularly suited to prevent loss of already invested work in layouts, e. g. in case the client crashes while you are configuring layouts.

Setup the desired backup interval. Backups will then be continuously created based on this interval in the background. Values from 1 minute up to 100 days are possible.

Here you can also setup a maximum number. When reached, the oldest backups will be deleted.

Accessing the backups

To load a layout backup, use the appropriate button in the toolbar of the Layout Manager:



23.6 Display style

By setting up the *Initial display style* you determine the display style which will always be applied on startup of *ibaPDA* client. The same styles are available for selection as available in the *View* menu or by using the corresponding short keys.

Note

If you have already set a display style for the user in the user management, then this style will be applied.

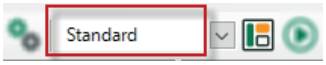
You will find more detailed information about the display styles in Part 1 of the *ibaPDA* manual, in chapter *The view menu*.

For the special display style “Fixed size” you can setup width and height in pixel units and select some other options.

23.7 Toolbars and menus

The toolbars and menus to operate *ibaPDA* client can be adapted for different users in the Layout Manager.

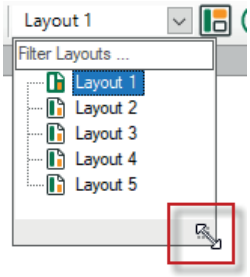
Width of layout selection in toolbar
This setting determines basically how wide (in pixels) the field for the display of the current layout in the toolbar is.



You can see immediately the result in the toolbar when you change the width value in the entry field.

Note

If needed, you can change the size of the expanded drop-down window with the layouts easily by grabbing the lower right corner with the mouse.



This change will be stored only temporarily for the current session. With restart of *ibaPDA* client the size will be reset to default setting.

Profiles
The visibility of toolbars and menus is independent from the different layouts. Thus, the visibility of toolbars and layouts cannot be configured differently per layout. However, it is possible to customize their visibility with regard to the user.

So called profiles, which can be assigned to the users, provide for distinction. Thereby, you can control which user can see which toolbar or menu and thus his or her scope of action.

As soon as a user logs in the *ibaPDA* server with his or her name, he or she can only see the toolbars and menus according to the assigned profile. Of course, the user management must be enabled in *ibaPDA*.

Which toollbars and menus are visible or not is stored in a profile. The basic structure of the toolbar, i.e. the arrangement of the single tools and tool groups is fixed. Only single elements can be hidden. The same applies to the menus.

There is a basic setting (default), which show all toolbars and menus.

Configure profiles
This window lists all available profiles. On the right side of the list you will find buttons for the configuration.

Button	Function
+	Create and add a new profile You can create as many profiles as you like. Give a profile a comprehensible name. You can rename the profiles at any time.
✕	Delete the selected profile
⚙	Configure the profile Select *Toolbars* or *Menus* and follow the sub-menus. In the first sub-menu you can hide or show entire toolbars or menus. In the further sub-menus you can hide or show single functions. With the command *Reset* in the first menu level you can reset the toolbars and menus for the related profile back to default settings.

While configuring, you can check the results of your selection in the toolbar and menus. Each time you select a profile in the list and click on the configuration button, the toolbar and menus will be displayed accordingly.

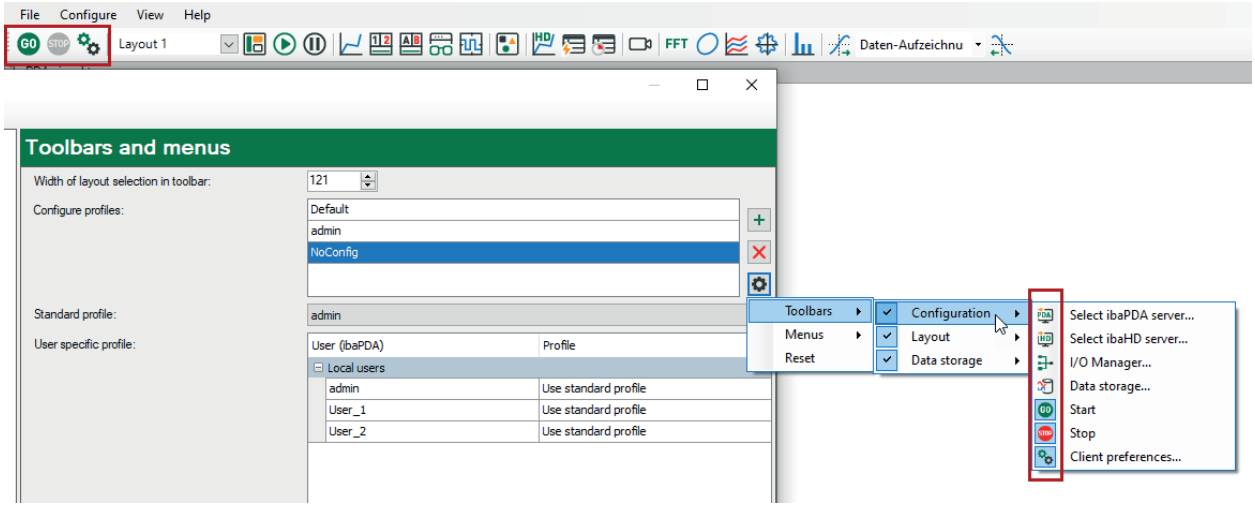
Standard profile
Here, select a profile which should be applied when *ibaPDA* gets started. If you have not configured your own profiles yet, only the profile "Default" is available. If there are already different profiles available they will be shown in the drop-down list. With the user management enabled, all users see the toolbars and menus according to this profile, unless they personally have a different profile assigned.

User specific profile
In this area, all configured local users and domain users are listed.

You can assign each user the desired profile in the *Profile* column:

- Use standard profile: The user will be assigned the profile which has been set under *Standard profile*.
- Default: The user will be assigned the default profile (everything visible).

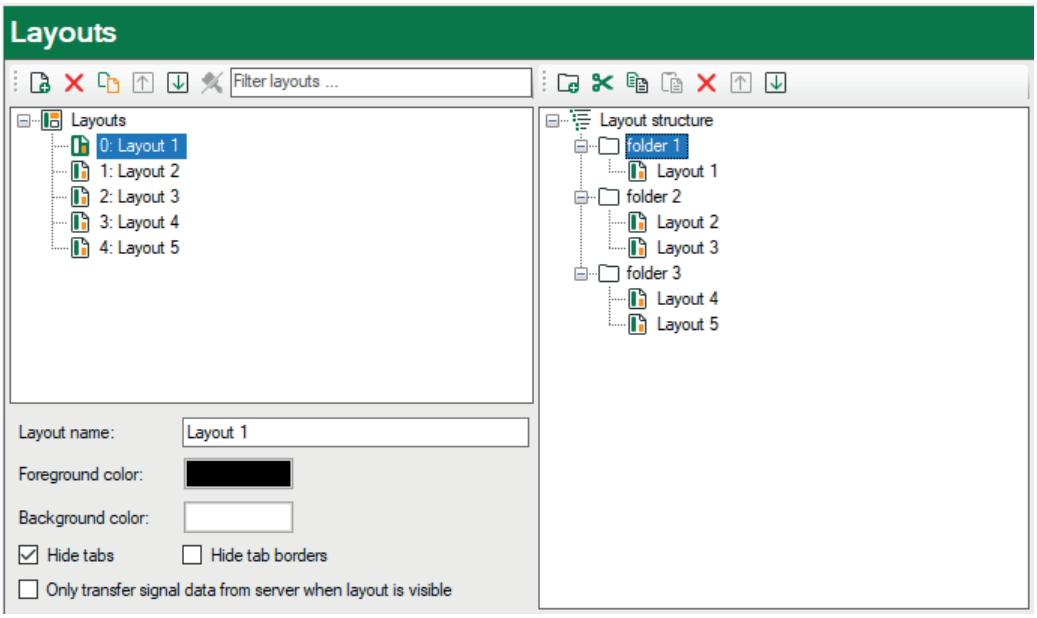
The following figure depicts for example a reduced toolbar, where the buttons for configuration of server, I/O Manager and data storage have been hidden. This setting has been saved as profile "NoConfig" and assigned to *User_1* and *User_2*.



In the configuration toolbar section in the upper left corner, the buttons for server selection, I/O Manger and data storage configuration are missing.

23.8 Layouts

On the node *Layouts* you define the layouts and arrange them in a structured manner by using folders and/or colors.



The dialog is divided in two parts:

- In the left part, there is the *layout pool*. Here, all currently available layouts are listed and you can create, rename or remove layouts as well as apply some properties to the layouts.
- In the right part, there is the *layout structure*. Here, you can create folders in order to organize the layouts. In the left part, each layout can only exist once, while layouts can appear multiple times in the right part.

Basic procedure
If you want to create a new layout, proceed as follows:

1. Add a new layout in the Layout Manager and rename it.
2. If needed, you may apply a color to the new layout and/or move it to a folder.
3. Close the Layout Manager
4. Open the drop-down box in the toolbar and select the new layout.
5. Now, change the *ibaPDA* client user interface according to your needs. All changes will be saved immediately in the local layout file in order to prevent loss when closing the *ibaPDA* client.

23.8.1 Layout pool

The left part of the dialog is the window with all available layouts. In this area you create new layouts and rename or remove the layouts. Moreover, you can assign each layout some basic properties.

On top of the window there is a toolbar for following actions:

Symbol	Function	Description
	Add new layout	The new layout will be placed below the selected layout in the tree. If the root node *Layouts* is selected the new layout will be placed at lowest position. The name of the new layout is generated by the pattern "layout *index*". The new layout will also be added to the layout structure at the lowest position of the first structure level.
	Delete selected layouts	Multi-selection is possible. If you delete a layout here, it will also be deleted in the structure on the right side.
	Duplicate selected layouts	Multi-selection is possible. The selected layout(s) will be pasted as a copy. The name will be extended by the suffix “-copy”.

Symbol	Function	Description
	Move selected layouts up	Multi-selection is possible. You may change the order by drag and drop as well or by using the hotkeys <Ctrl>+<↑>. If you select layouts which are not adjacent to each other, the gap between them remains when using the button or the hotkeys. When using drag and drop the selected layouts will be placed adjacent to each other at the new position.
	Move selected layouts down	Multi-selection is possible. You may change the order by drag and drop as well or by using the hotkeys <Ctrl>+<↓>. If you select layouts which are not adjacent to each other, the gap between them remains when using the button or the hotkeys. When using drag and drop the selected layouts will be placed adjacent to each other at the new position.
	Remove color settings	Only available if colors are used. Function applies on the selected layout. Multi-selection is possible. If the root node *Layouts* is selected, the color settings of all layouts will be removed.
Filter	Filter out layouts by their names	You can use the filter function to filter the layout list by name. Enter one or more characters or an entire name and the list of layouts will be adapted immediately. As soon as characters have been entered, a green filter icon indicates that the layout list is filtered. In order to remove the filter just click on this filter symbol.

The same functions (except filter) are available in the context menu of the window too.

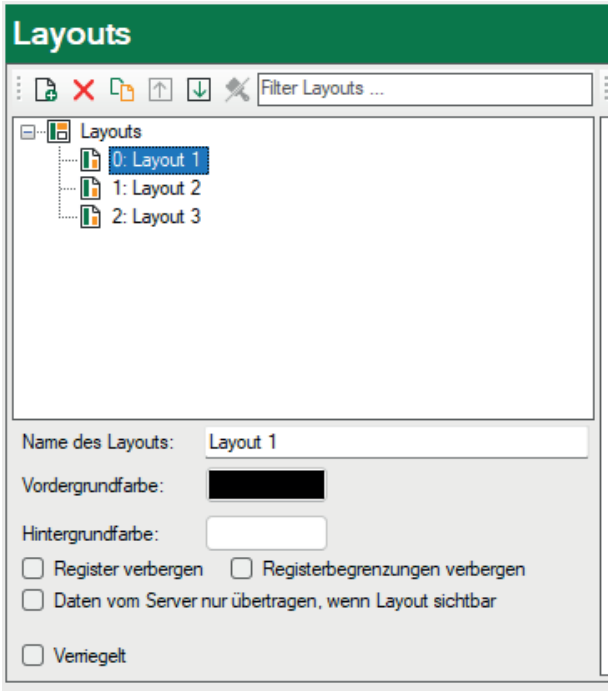
More commands in the context menu
If you open the context menu on the root node of the layout pool, you will find the command *Reset all docking configurations*.

This menu function allows you to reset all free floating windows (e.g. signal tree, watch window, data storage status etc.) in all layouts to their default position, size and visibility state.

If you open the context menu on a layout node you can choose between resetting the docking configuration of this layout or applying the docking configuration of this layout to other layouts. In the latter case, a subsequent dialog will open for selecting the layouts you want the docking configuration be applied to.

If you select multiple layout nodes, you can only reset the the docking configuration of the selected layouts.

Layout properties
Below the layout list you can setup some properties of the layouts.



Layout name
If you have selected a layout in the list you can rename it. Any name is permitted. Duplicate use of the same name is possible, due to backward compatibility with former program versions. When closing the Layout Manager you will be informed if duplicate names exist and you can correct it manually if needed.

In order to rename a layout you can also use the context menu on a selected layout in the tree or the key <F2>.

Foreground and background color
The settings of foreground color and background color only apply to the representation of the layout names inside the Layout Manager (pool and structure).

Hide tabs, Hide tab borders
With the options *Hide tabs* and *Hide tab borders* you determine the representation of all views which are part of the layout. These are the same functions, which can be executed in the *View* menu. *Hide tabs* only works if the views are not arranged in tab groups. Temporarily, these settings can be changed at any time in an active layout via the *View* menu.

Only transfer signal data...
With the option *Only transfer signal data from server when layout is visible* you can reduce the data traffic between server and client. This setting refers to the measurement data (signals) in the views of a layout.

By default this option is disabled. Hence, all data which is contained in any layout will be refreshed continuously, no matter whether the layout is active (visible) or not. This ensures that the historical signal values of all signals are visible after a switch-over from one layout to another, no matter whether the signals had been visible in the previous layout or not.

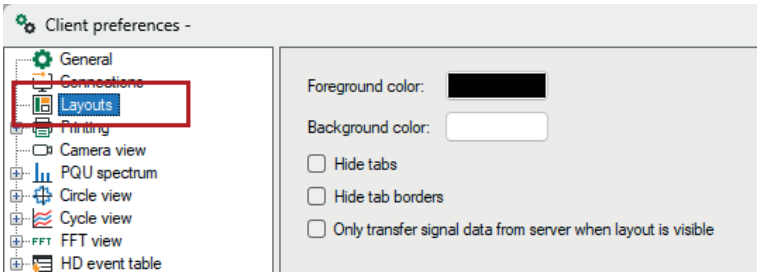
If you enable this option, then the values of the signals in the new layout will be loaded after the switch-over. Thus, trend graphs, for example, will be written not before the time of switch-over. For signals which had not been visible in the previous layout, no historical data is available.

Locked

You can use this option to lock a layout to prevent unintended changes. Only unlocked layouts can be changed. No specific user rights are required to lock or unlock a layout.

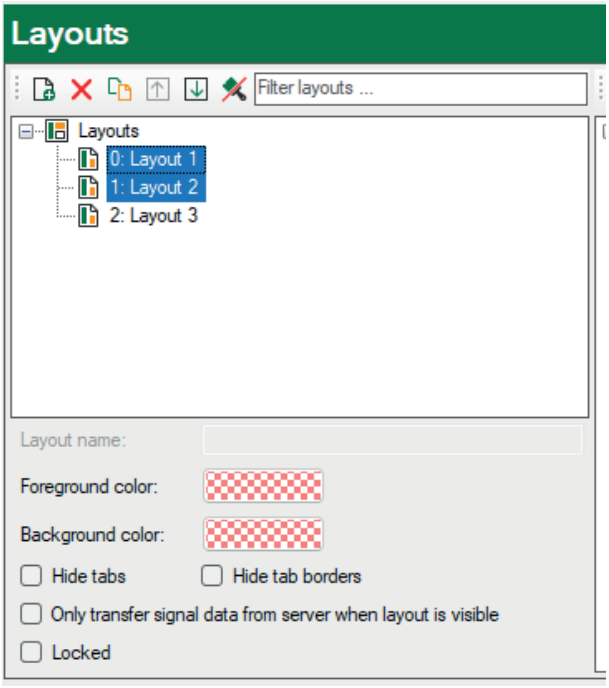
Note

Except for the *Locked* property, you can also configure the above mentioned properties in the *Client preferences*. There, click on the *Layouts* node.



These settings will be applied automatically to all new layouts.

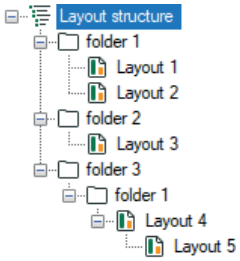
If several layouts with different properties are selected, it looks like as follows:



23.8.2 Layout structure

The right part of the dialog shows the layouts listed in a structured manner. By default it is just a flat list like on the left side. You don't have to do nothing if this form is suitable for you. But if you want to arrange or structure your layouts by certain criteria, you can do this here. Particularly, if you have many layouts, a proper arrangement can provide for better clarity.

In the layout structure you can create folders and sub-folders which contain the layouts. You can even subordinate layouts to other layouts. The latter can be helpful, e.g. if you have layouts configured as tabbed layouts in *ibaQPanel*.



On top of the window there is a toolbar for following actions:

Symbol	Function	Description
	Add new folder	This function is only available when the root node *Layout structure* or another existing folder is selected. If the root node *Layout structure* is selected, the new folder will be created on the first structure level. If a folder is selected, the new folder will be created as a sub-folder. The name of the new folder is generated by the pattern "folder *index*".
	Cut selected elements	Multi-selection is possible. The selected elements are copied to the Windows clipboard.
	Copy selected elements	Multi-selection is possible. The selected elements are copied to the Windows clipboard.
	Paste copied elements	Folders cannot be subordinated to a layout node.
	Delete selected layout or folder	Multi-selection is possible. If you delete a folder, the contained layouts will be deleted in the structure too. If you have deleted a layout in the structure you can easily add it again by drag and drop from the layout pool.

Symbol	Function	Description
	Move selected layout or folder up	Multi-select on folders of the same level or on layouts within one folder is possible. You may change the order by drag and drop as well or by using the hotkeys <Ctrl>+<↑>. If you select multiple folders or layouts which are not adjacent to each other, the gap between them remains when using the button or the hotkeys. When using drag and drop the selected layouts will be placed adjacent to each other at the new position.
	Move selected layout or folder down	Multi-select on folders of the same level or on layouts within one folder is possible. You may change the order by drag and drop as well or by using the hotkeys <Ctrl>+<↓>. If you select multiple folders or layouts which are not adjacent to each other, the gap between them remains when using the button or the hotkeys. When using drag and drop the selected layouts will be placed adjacent to each other at the new position.

The same functions are available in the context menu of the window.

If you want to subordinate one or more layouts to another layout, you have different possibilities:

- Drag the respective layout with <Ctrl> key depressed out of the layout pool onto the layout in the layout structure where it should be subordinated.
- Drag the respective layout with <Ctrl> key depressed within the layout structure onto the layout where it should be subordinated.
- Copy or cut the respective layout, select the layout where it should be subordinated and paste it.

Rename folders
You can rename the folders. Any name is permitted. Duplicate names are possible. When closing the Layout Manager you will be informed if duplicate names exist and you can correct it manually if needed.

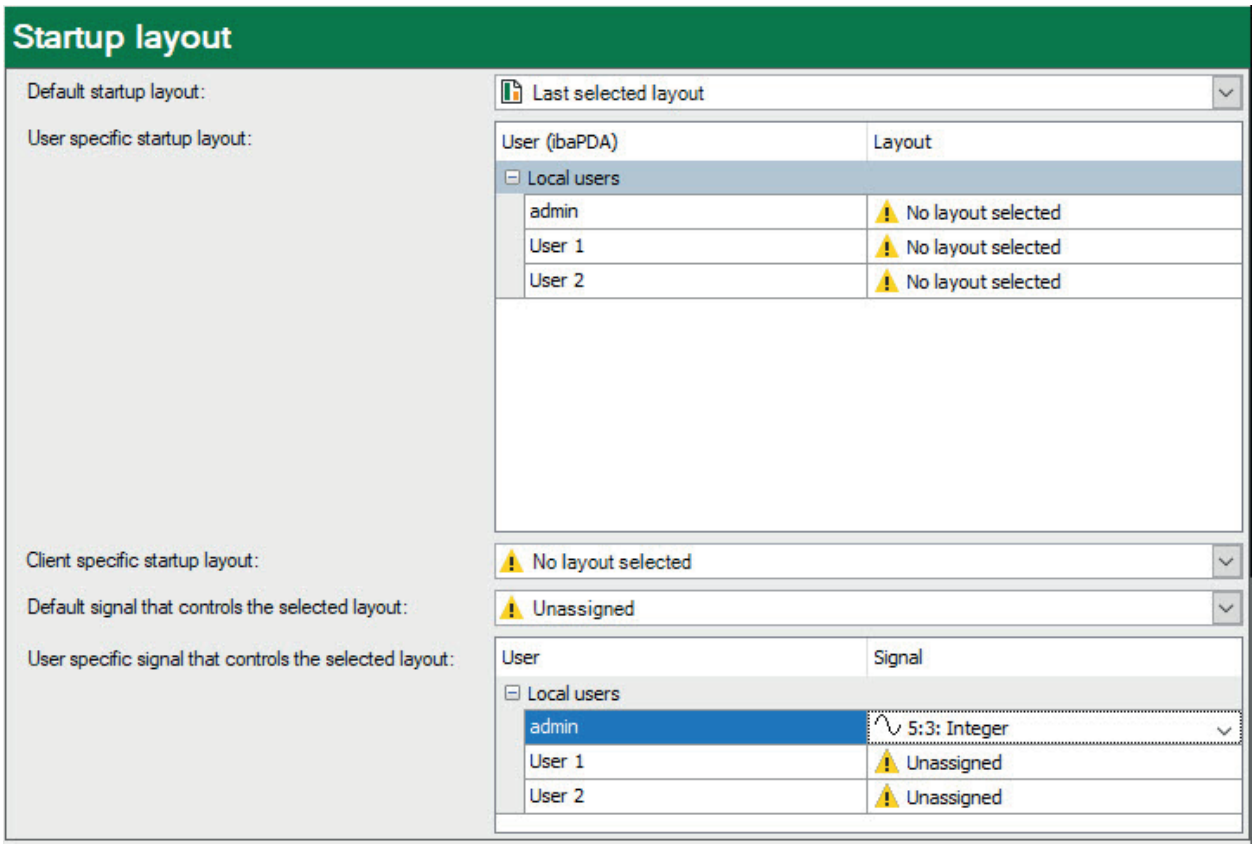
Other structuring rules

- Folders are always displayed on top before the layout nodes.
- A layout node can only exist once inside a folder or as a sub-node below another layout.

23.8.3 Startup layout

The settings on this node determine which layout will be shown when you open *ibaPDA* (client). You can choose between several methods to select the startup layout. Fixed, user-specific, client-specific or controlled by a signal

The priority of the methods increases from top to bottom. This means that, for example, the user-specific setting dominates the default startup layout setting.



You can make the following settings.

Default startup layout
By default, the last layout of the previous session will be the startup layout again. Alternatively, you may select a layout here, which will always be shown when *ibaPDA* client is opened, no matter which layout was active at the end of the previous session. The precondition is that no other method is used.

User specific startup layout
Here, you can assign each user a startup layout. The indicated users are derived from the server which is entitled as *layout source* on the *General* node. If a user loggs in, the assigned layout will be displayed. If you select here a layout, which is not assigned to this user on the node *User assignment* and hence, would not be visible to this user, a warning pops up.

Client specific startup layout
This setting determines the startup layout for the client computer you are working on. Thereby, you can make sure that this computer always shows the selected layout on startup, no matter which user has logged in.

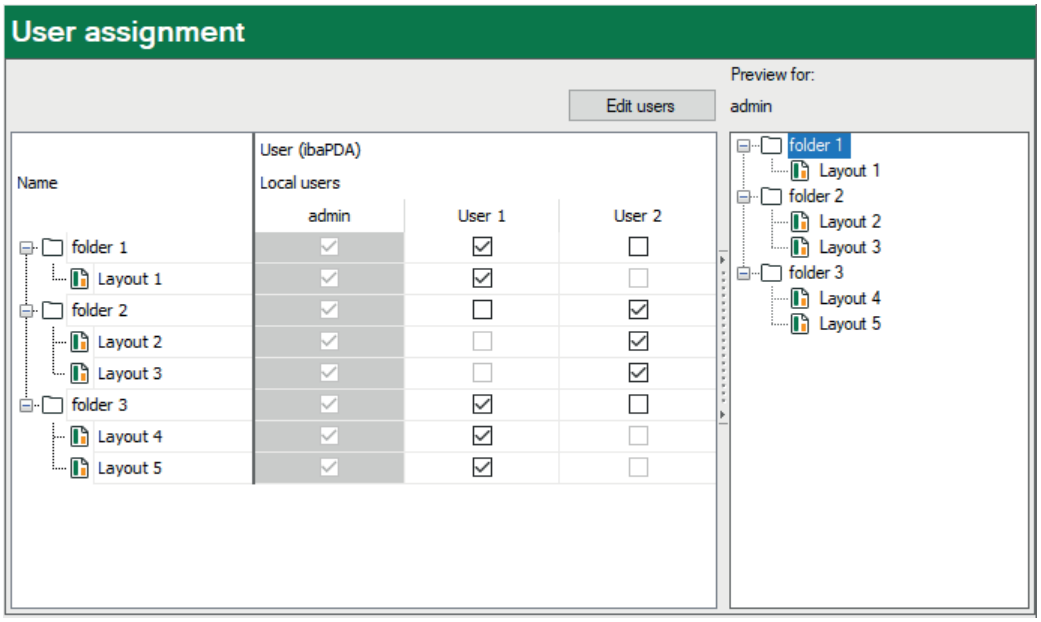
Signal that controls the selected layout
Here you can setup an analog signal, whose value 0 ...n will be used to select a corresponding layout. Best choice is an analog signal with integer values (INTEGER). The link between signal value and layout is the layout index, which you can find in front of every layout in the left window.

User specific signal that controls the selected layout
Here you can setup an analog signal for each user, whose value 0 ...n will be used to select a corresponding layout. Best choice is an analog signal with integer values (INTEGER). The value of the signal should match the index of the requested layout. The layout index is the number in front of each layout in the layout overview (*Layouts* node, left pane).

In contrast to the fixed user specific startup layout (see above), you can setup a user specific layout selection, which additionally depends on a signal.

23.8.4 User assignment

On this node you determine which layouts a user is allowed to see or not. The indicated users are derived from the server which is entitled as *layout source* on the *General* node.



The dialog is divided in two sections. On the left side there is a table with all the layouts from the layout structure in the first column and the available users in the columns aside.

On the right side there is a preview window, showing what the selected user would see in the layout selection box in the toolbar of *ibaPDA* client. You can hide the preview window in order to have more space for user assignment. Simply click on the control on the border between table and preview window.

Making assignments
You can assign each user the layouts individually by checkmarks. When assigning layouts in folders, the folder must be assigned too. Even if you have assigned a folder, the desired layouts in the folder must be assigned too.

If you have to assign a larger number of layouts, you can select multiple table cells (mouse click + <Ctrl> or <Shift>) and make the checkmark in any of the selected cells. The same applies to unchecking.

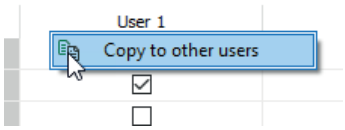
Tip

If you want to suspend temporarily the assignment of layouts in a folder for a user you just need to uncheck the folder. Then the user will not see any of the contained layouts. However, the assignment of the layouts sustains and after checking the folder again, all layouts are visible as before.

Copying assignments

If you have configured the assignments for one user and want the same for other users, you can copy them. Proceed as follows:

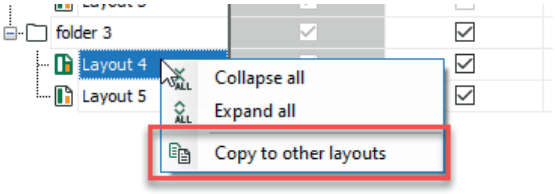
1. Make a right mouseclick on the column header of the user whose assignments you want to copy.
2. Select *Copy to other users* in the context menu.



3. All users will be shown in the subsequent dialog. Select the user(s) here who should get the assignments. The user *admin* ist not provided because it has assigned all layouts anyway.

The same method applies to layouts and folders accordingly. For instance, if you have configured a comprehensive user assignment for a layout and you want to transfer it to other layouts alike or similar, proceed as follows:

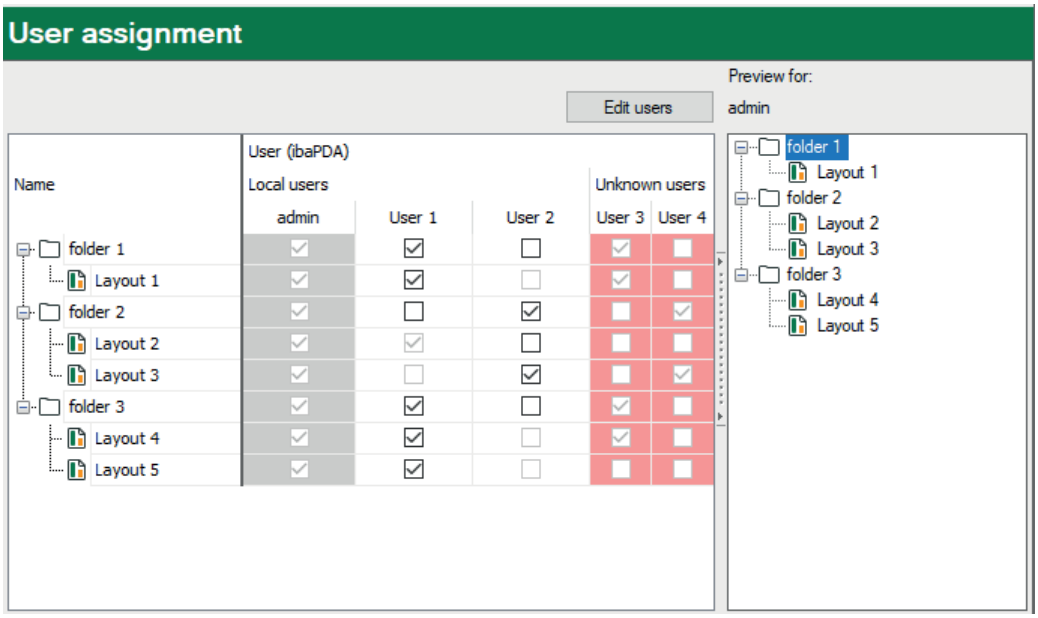
1. In the column with the layout structure select the layout or the folder whose assignments are to be copied.
2. Open the context menu (right mouseclick) and select *Copy to other layouts*.



3. All other layouts/folders will be shown in the subsequent dialog. Select the destinations (layouts or folders) here.

Unknown users

Some situation may occur when user assignments are stored in the Layout Manager while these users do not exist. The columns of the respective users are colored red.



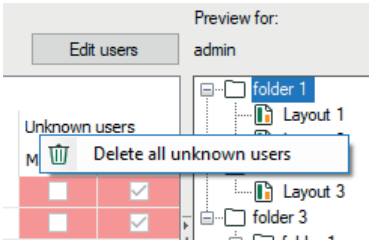
The following scenarios are possible:

- Deleting users in the user management who have layouts assigned
- Switching the *layout source* (ibaPDA, ibaHD-Server) to another server
- Loading a layout set (*.layouts) with layouts which are assigned to users who are not configured in the user management of the actual server

You have the following options:

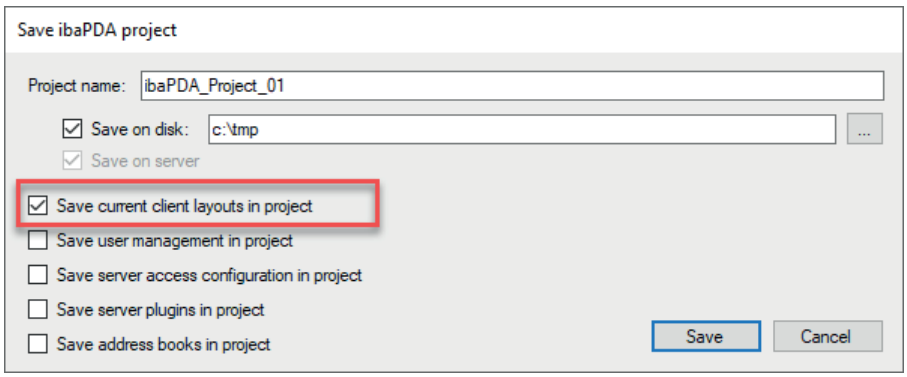
If an unknown user is required, you can create it in the user management of the connected server. Thus, the user is not unknown anymore. Via the button <Edit users> you can easily enter the user management without leaving the Layout Manager.

If an unknown user is not required, you simply can delete it in the Layout Manager. Use the context menu on the column header of the respective user for that. If there are several unknown, not required users make a right mouseclick on the common column header *Unkown users* and you can delete all unknown users at once.



23.9 Save layout in project

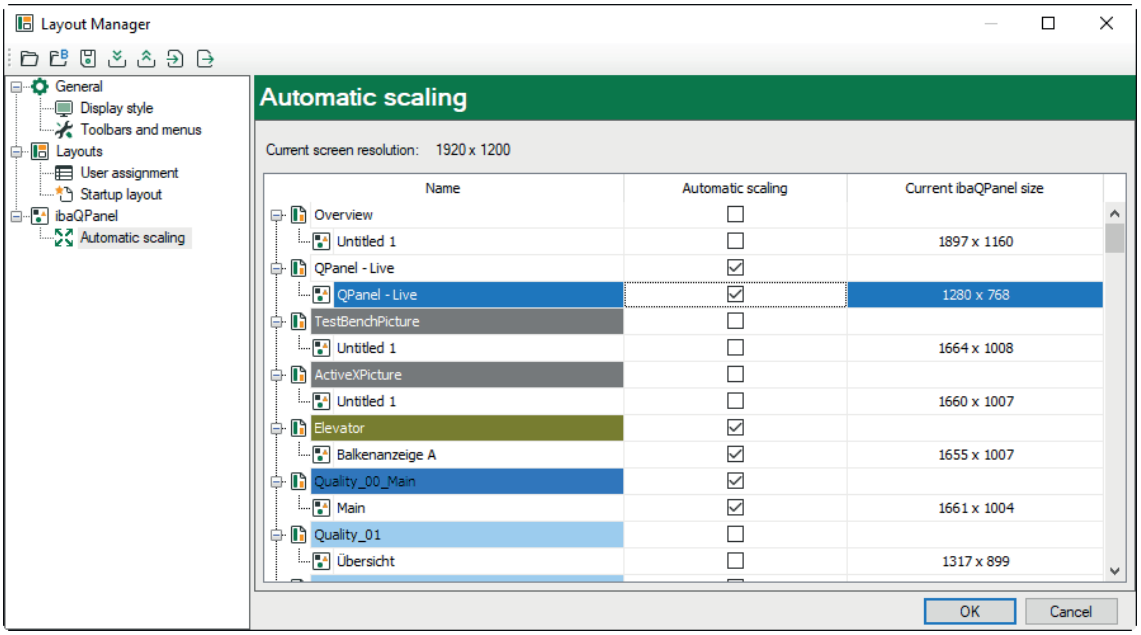
If you want to save the current layouts in a project, make sure that the option *Save current client layouts in project* is enabled in the dialog for saving the project (*File - Save project...*).



23.10 Automatic scaling of the QPanel window

An *ibaQPanel* window can be configured to fill the entire screen. If you then open it on a monitor with a different screen resolution, the size of the window changes. You can use the *Automatic scaling* option to automatically adapt the size of the *ibaQPanel* window and all the objects it contains if it is used on a monitor with a different resolution. All text sizes are also rescaled.

Automatic scaling can be enabled in the Layout Manager. Open the Layout Manager by selecting *Configuration - Layout Manager* in the menu or using the icon.

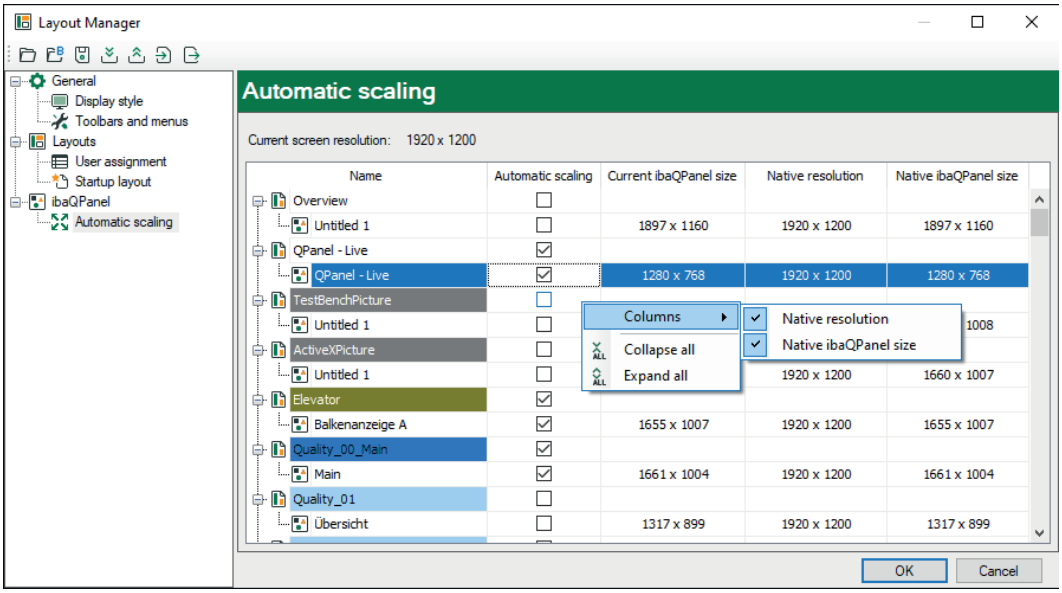


The scaling is carried out when opening *ibaPDA* on a screen with a different resolution. It also works on the fly if you drag the window to another screen with a different resolution.

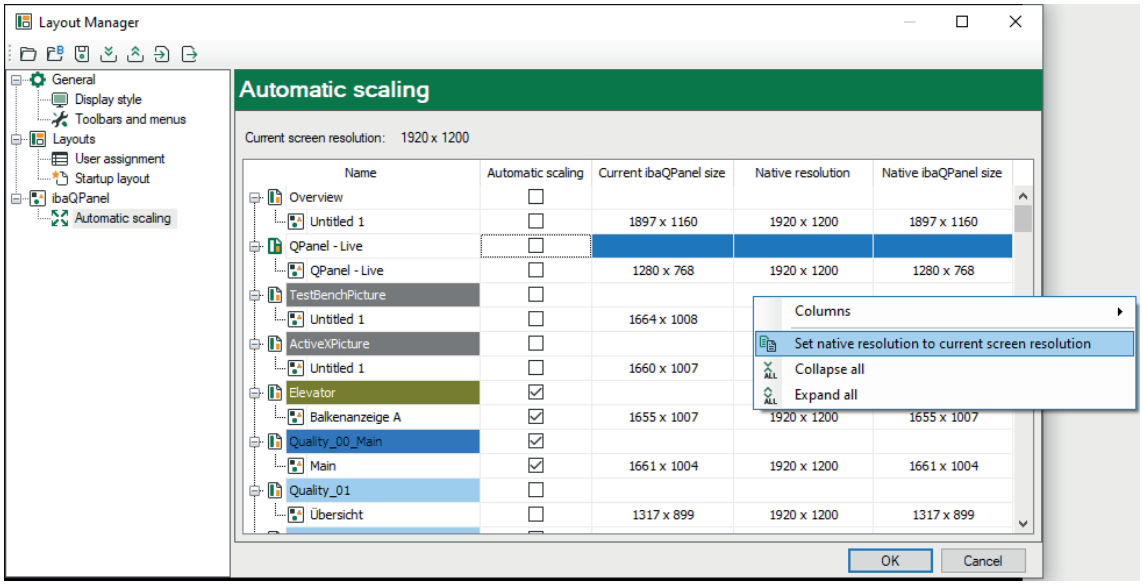
You can enable automatic scaling separately for each window by clicking on the checkbox for the relevant window.

You can use the context menu to show or hide the columns in the table:

- Automatic scaling: Enable/disable automatic scaling
- Current *ibaQPanel* size (in pixels)
- Native resolution: Resolution of the screen on which the window was created
- Native *ibaQPanel* size: Size corresponding to the system resolution.



You can adjust the native resolution to the current screen resolution. To do this, right-click to open the context menu in the *Native resolution* column and select *Set native resolution to current screen resolution*.



When you open an existing layout with an *ibaQPanel* window for the first time in the new version, the native resolution is initialized to the current screen resolution and the native size to the current size.

24 Support and contact

Support

Phone: +49 911 97282-14

Email: support@iba-ag.com

Note

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

Contact

Headquarters

iba AG
Gebhardtstrasse 10-20
90762 Fuerth
Germany

Phone: +49 911 97282-0

Email: iba@iba-ag.com

Mailing address

iba AG
Postbox 1828
D-90708 Fuerth, Germany

Delivery address

iba AG
Gebhardtstrasse 10
90762 Fuerth, Germany

Regional and Worldwide

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

www.iba-ag.com

ibaPDA

Appendix

Manual Part 7

Issue 8.14

Measurement Systems for Industry and Energy

www.iba-ag.com

Manufacturer

iba AG

Gebhardtstrasse 10-20

90762 Fuerth

Germany

Contacts

Main office	+49 911 97282-0
Support	+49 911 97282-14
Engineering	+49 911 97282-13
E-mail	iba@iba-ag.com
Web	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 2026, 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 and can be found in the iba help center docs.iba-ag.com.

Version	Date	Revision	Author	Version SW
8.14	06-2026	Chapter about I/O configuration file moved to Part 2	rm	8.14.0

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

1 About this documentation 4

1.1 Target group and previous knowledge 4

1.2 Notations 4

1.3 Used symbols 5

1.4 Documentation structure 6

2 ASCII watchdog telegram structure 7

3 Binary watchdog telegram 8

4 Setting up the ibaPDA client as an ActiveX control 9

4.1 Methods and properties 11

4.2 ibaPDA ActiveX control in SIMATIC WinCC 17

4.3 ibaPDA ActiveX control in SIMATIC WinCC flexible 21

4.4 Sample VB Script for printing out of an ActiveX Control 24

5 ibaPDA-S7-Xplorer Proxy for PC/CP interface 25

6 Advanced filter in tables 26

6.1 Simple filter function 26

6.2 Advanced filter functions 28

6.3 Search function 32

7 Uninstalling ibaPDA 33

8 Support file generation via the command line 35

8.1 Application 35

8.2 Exit codes 36

8.3 Error messages 36

9 Support and contact 37

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 they are capable of assessing safety and recognizing possible consequences and risks on the basis of their specialist training, knowledge and experience and knowledge of the standard regulations.

1.2 Notations

In this manual, the following notations are used:

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

1.3 Used symbols

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

Danger!

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

Observe the specified measures.

Warning!

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

Observe the specified measures.

Caution!

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

Observe the specified measures.

Note

A note specifies special requirements or actions to be observed.

Tip

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

Other documentation

Reference to additional documentation or further reading.

1.4 Documentation structure

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.

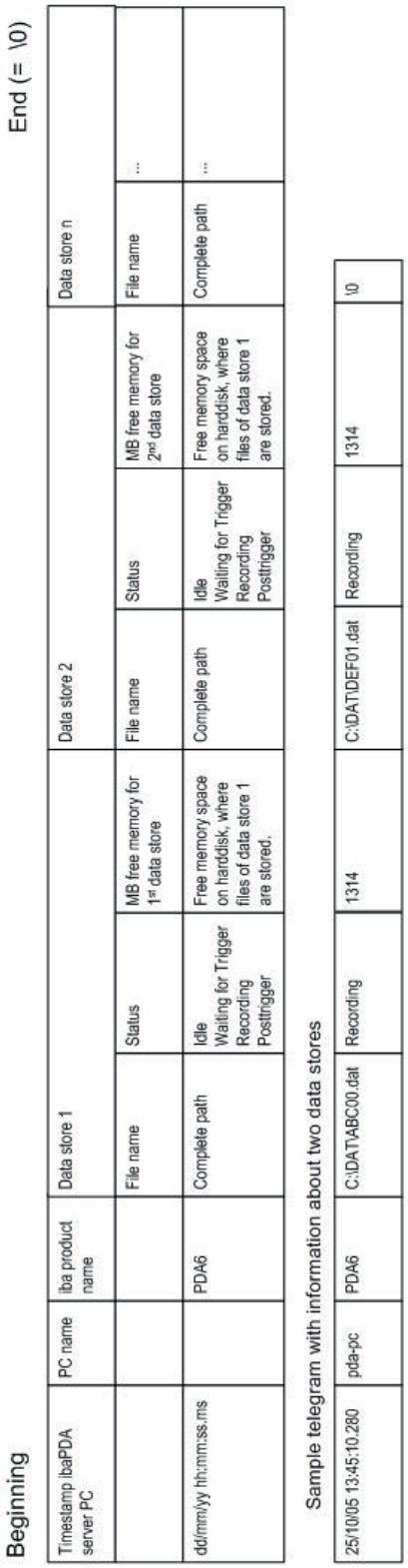
In addition, special "NewFeatures..." documentation comes with any software update that includes significant new features, which provides a more detailed description of the new features.

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.

Part 1	Introduction and installation	General notes, license policy and add-ons Installation and program start User interface, system architecture, client server User management, printing
Part 2	I/O Manager	Basic information about the I/O Manager, general settings Groups and vector signals, text signals, outputs, configuration files
Part 3	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.
Part 4	Expression builder	All functions for calculating virtual signals
Part 5	Data storage	Types of data recording, recording profiles, signal selection
Part 6	Data visualization	All display modes for live data, their operation and settings
Part 7	Appendix	Various additions, error listings, etc.

2 ASCII watchdog telegram structure



3 Binary watchdog telegram

```
#ifndef PDA_WATCHDOG_H
#define PDA_WATCHDOG_H

#define N_MAX_DATASTORES      16

typedef struct _PDA_DATASTORE_INFO
{
    unsigned short status;       // (0) Status of the datastore:
                                 //       =0: Inaktiv:
                                 //       =1: Waiting for trigger
                                 //       =2: Recording
                                 //       =3: Post-trigger
    unsigned short directory;    // (2) Current directory:
                                 //       =0: Pda is writing to the base directory
                                 //       =1: Pda is writing to the backup directory
    unsigned int   freeSpace;    // (4) Free space on the harddisk in MB
                                 // (only available if datastore is active)
    unsigned char  reserved[4];  // (8)

} PDA_DATASTORE_INFO, *PPDA_DATASTORE_INFO;
typedef struct _QDR_DATASTORE_INFO
{
    unsigned short status;       // (0) Status of the QDR datastore:
                                 //       =0: Inaktiv:
                                 //       =1: Not synchronized
                                 //       =2: Synchronized
    unsigned short directory;    // (2) Current directory:
                                 //       =0: Pda is writing to the base directory
                                 //       =1: Pda is writing to the backup directory
    unsigned int   freeSpace;    // (4) Free space on the harddisk in MB
                                 // (only available if datastore is active)
    unsigned char  reserved[4];  // (8)

} QDR_DATASTORE_INFO, *PQDR_DATASTORE_INFO;
typedef struct _PDA_WATCHDOG
{
    unsigned int  msgCounter;   // (0) Message counter is incremented after each message
    unsigned int  msgVersion;   // (4) Version number (currently = 1)

    unsigned int  isMeasuring;  // (8)  =1: Pda is measuring
    unsigned char allOk           : 1;  // (12.0) =1: Everything is OK (measuring und no disabled signals)
    unsigned char disabledSignals : 1;  // (12.1) =1: There are disabled signals
    unsigned char reserved[3];          // (13)
    unsigned int  captureCamStatus[2];  // (16) Bit per connection
                                        // to ibaCapture-CAM server
                                        // bit = 1: connection ok
    unsigned int  captureHmiStatus[2];  // (24) Bit per connection
                                        // to ibaCapture-HMI server
                                        // bit = 1: connection ok

    QDR_DATASTORE_INFO qdrDatastore;  // (32)   Information about the QDR datastore
    PDA_DATASTORE_INFO datastores[N_MAX_DATASTORES];  // (44) Information
                                                      // about all datastores

} PDA_WATCHDOG, *PPDA_WATCHDOG;
#endif
```

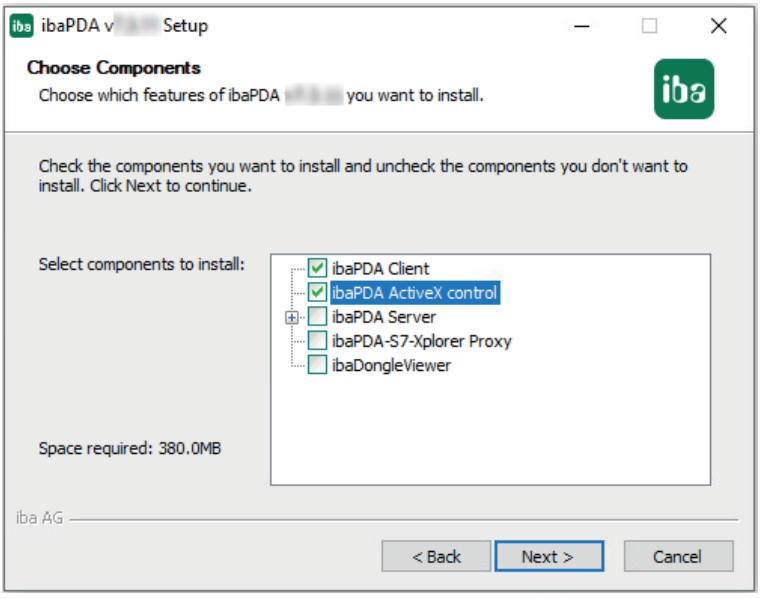
4 Setting up the ibaPDA client as an ActiveX control

Requirements and licenses

As a prerequisite for using the *ibaPDA* client as an ActiveX control, the following software components must be installed on the computer on which the application is to run with ActiveX:

- ibaPDA Client
- ibaPDA ActiveX control

For this purpose, start the *ibaPDA* installation on the corresponding computer and select the above-mentioned components in the selection dialog.



Each ActiveX control establishes a client connection to the configured *ibaPDA* server using the configured user name and password. If there are simultaneously several ActiveX controls in a host process (ActiveX container), which connect to the same *ibaPDA* server using the same user, these ActiveX controls will share a common connection.

An example would be two ActiveX controls located side by side on one image in an HMI application like SIMATIC WinCC, or even on two different images, connecting to the same *ibaPDA* server using the same user. The runtime process for the HMI application is the ActiveX container.

If two ActiveX controls with different users connect to the same *ibaPDA* server, then two connections are created.

An *ibaPDA* client license is required for each connection. If *ibaQPanel* is also used in the views shown in the *ibaPDA* ActiveX control, an *ibaQPanel* license is additionally required.

For workstations on which no images can be called up with the *ibaPDA* ActiveX Control (e.g., due to a corresponding configuration in the HMI application), no license is required.

If you wish to make sure that a computer is always guaranteed to get the license it needs, you can reserve client and QPanel licenses for selected computers using the server access control feature in *ibaPDA* (see *ibaPDA* manual part 1).

The connection is only established when the ActiveX control is called for the first time in the host process. An example of this is when an image with a projected ActiveX control is displayed for the first time in the HMI application, such as SIMATIC WinCC.

From the moment the connection is established, the license is also activated. When switching to an image without ActiveX control, the connection to the *ibaPDA* server is maintained in the background, consequently the license is still in use. The data for the signals visualized in the views continues to be queried in the background via the maintained connection, so that the signal trends can be displayed immediately when the image is displayed again with ActiveX Control.

The *ibaPDA* ActiveX control does not establish a connection to the *ibaPDA* server in design mode (e.g., during configuration in SIMATIC WinCC Graphics Designer). In runtime mode the ActiveX control will automatically try to connect to the configured *ibaPDA* server.

Note

The ActiveX control can only be used for visualization. This means you cannot start/stop the acquisition or configure the I/O manager and data storage. Layout changes and interaction with the signal displays are possible with the corresponding permissions.

The ActiveX Control creates log files and saves them, e.g.,in Windows 10 at `LocalAppData%\iba\ibaPdaActiveX`.

This creates a subdirectory with the name of the ActiveX host process (e.g.,"PdlRt" for the SIMATIC WinCC Runtime process). All ActiveX control instances within an ActiveX host process use the same log file.

4.1 Methods and properties

The *ibaPDA* ActiveX Control according to its definition provides some object properties which are listed in the table below.

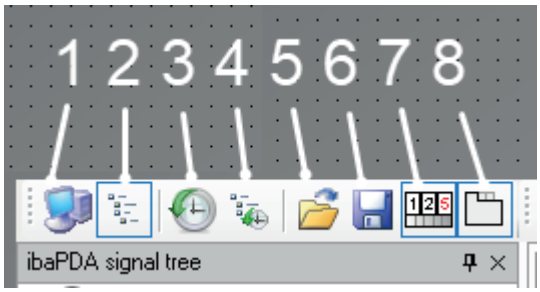
The properties can be manipulated by using scripts which enable you to change the ActiveX Control dynamically. Therefor, some methods are available which are described further below.

Object properties of ibaPDA ActiveX Control

Property	Select, enter	Note
LayoutFile	Path specification [string]	Enter the path and file name of a layout file here. The *ibaPDA* client will always load this file when the image is called up (even after an image change). If *AutoSaveLayout* is enabled, layout changes will be saved to this file. Even if you click the Save button in the ActiveX toolbar in runtime mode, the layout is written to this file. Changing this property at runtime will **not** trigger a reload of the layouts.
ServerAddress	IP address or computer name [string]	Enter the IP address or the computer name of the *ibaPDA* server to which the *ibaPDA* client should connect in this ActiveX control.
PortNr	Port number [int32]	Port through which the client connects to the server Default: 9170
UserName	User name [string]	Enter a user name with which the *ibaPDA* client should log on to the *ibaPDA* server in this ActiveX Control. The user name must already exist in the user aanagement dministration of the *ibaPDA* server. You must enter a user name here, unless you are using the *UseCurrentwindowsUser* login (see below).
Password	Password [string]	Enter the password matching the *ibaPDA* user here.
ShowToolbars	Yes/No [bool]	Lets you choose whether the ActiveX- and/or Layout toolbars of are displayed (Yes) or not (No). Changing this property via a script at runtime will immediately take effect.

Property	Select, enter	Note
ShowTabs	Yes/No [bool]	Lets you choose whether the view tabs are displayed (Yes) or not (No). This does the same as the button in the Active-X toolbar. It overwrites the settings in the layout configuration. Changing this property via a script at runtime will immediately take effect.
AutoSaveLayout	Yes/No [bool]	Allows you to specify whether changes to the layout are automatically saved to the *CurrentLayout.lay* file or not. If you have specified a file for the *LayoutFile* property (see above), then the changes will be saved to that file.
DisableLayoutChange	Yes/No [bool]	Allows you to specify whether the layout may be changed (Yes or No). If layout changes are disabled here, then no *ibaPDA* user – even users with the corresponding rights – can change the layout. If layout changes are permitted here, then the user's *Change layout* right is also taken into account.
DisableInteraction	Yes/No [bool]	Allows you to specify whether the signal displays can be operated interactively (Yes or No). If such interactions are disabled here, then no *ibaPDA* user – even users with the corresponding rights – can interact with the signal displays. If such interactions are permitted here, then the user's *Interact with views* right is also taken into account.
DisableMessageboxes	Yes/No [bool]	Allows you to specify whether notification boxes from *ibaPDA* may be displayed here (Yes or No). These are internal system notifications, e.g., if the I/O configuration has been changed on another client.
HdServerAddress	IP address or computer name [string]	Enter the IP address or the computer name of the *ibaHD* server to which the *ibaPDA* client should connect in this ActiveX control.
HdPortNr	Port number [int32]	Port through which the client connects to the server Default: 9180

Property	Select, enter	Note
ShowClientDisconnect	Yes/No [bool]	This lets you define how to proceed if no more free *ibaPDA* client licenses are available for the ActiveX control. ■ Yes: In the ibaPDA client, a dialog appears in which you can select another client and log off in order to free up the license for the ActiveX control. ■ No: An error message is merely displayed notifying you that no more client licenses are available.
HdUserName	User name [string]	Lets you enter the user name with which the *ibaPDA* client should log on to the *ibaHD-Server* in this ActiveX control. The user name must already exist in the user management of the *ibaHD* server. You must enter a user name here, unless you are using the *HdUseCurrentwindowsUser* login (see below).
HdPassword	Password [string]	Enter the password matching the *ibaHD-Server* user here.

Property	Select, enter	Note
ShowActiveXToolbar	Yes/No [bool]	This defines whether the special ActiveX toolbar is displayed (Yes or No). The functions of this toolbar can only be used in runtime mode.  1 Select *ibaPDA*-Server 2 Show/hide *ibaPDA* signal tree 3 Select *ibaHD-Server* 4 Show/hide *ibaHD-Server* signal tree 5 Open layout, loads a layout from a file 6 Save layout, saves the current layout in the file specified above 7 Show/hide window for digital displays 8 Show/hide view tabs Changing this property via a script at runtime will immediately take effect.
ShowLayoutToolbar	Yes/No [bool]	This defines whether to show (Yes) or hide (No) the Layout toolbar. Changing this property via a script at runtime will immediately take effect.
UseCurrentWindowsUser	Yes/No [bool]	Lets you define whether the current Windows user on the SIMATIC WinCC computer can log on to the *ibaPDA* server (Yes or No). This permission can be granted in addition to the entering a local *ibaPDA* user (see *UserName* above). The prerequisite for this is that an Active Directory is set up and domain users or groups to which the Windows user also belongs are configured in the user management of the *ibaPDA* server.

Property	Select, enter	Note
HdUseCurrentWindow-sUser	Yes/No [bool]	Lets you define whether the current Windows user on the SIMATIC WinCC computer can log on to the *ibaHD server* (Yes or No). This permission can be granted in addition to the entering a local *ibaHD-Server* user (see *HdUserName* above). The prerequisite for this is that an Active Directory is set up and domain users or groups to which the Windows user also belongs are configured in the user management of the *ibaHD* server.

Table 1: Object properties of ibaPDA ActiveX control

Methods to control the ibaPDA ActiveX Control

When using scripts you can dynamically control the ActiveX control by using the following methods in the script.

Method name	Parameter	Result	Description
Print	Title [string] ShowPrintSetup [bool]		This prints the currently visible layout. The title will be shown at the top of the page. If *ShowPrintSetup* is true then the print setup dialog is shown. You can select the printer there and select page size, margins, scaling etc. If *ShowPrintSetup* is false the last saved settings are used.
ConnectToPdaServer			This will attempt to connect to an *ibaPDA* server using the current values of the ServerAddress, PortNr, UserName and Password properties (see table before). So in the script you can change the properties and then call *ConnectToPdaServer* to switch to a different server or log in as a different user.
ConnectToHdServer			This will attempt to connect to an *ibaHD-Server* using the current values of the HdServerAddress, HdPortNr, HdUserName and HdPassword properties (see table before). So in the script you can change the properties and then call *ConnectToHdServer* to switch to a different server or log in as a different user.
LoadLayoutFromFile	LayoutFile [string]	bool	This will try to load the layouts file *LayoutFile*. It basically does the same thing as when you manually click the open layout button on the Active-X toolbar and then select the layout file. The return value is true if the layouts were loaded from the file.
SaveLayoutToFile	LayoutFile [string]	bool	This will try to save the current layouts to the file *LayoutFile*. The return value is true if the save was successful.

Table 2: Methods to control the ibaPDA ActiveX Control

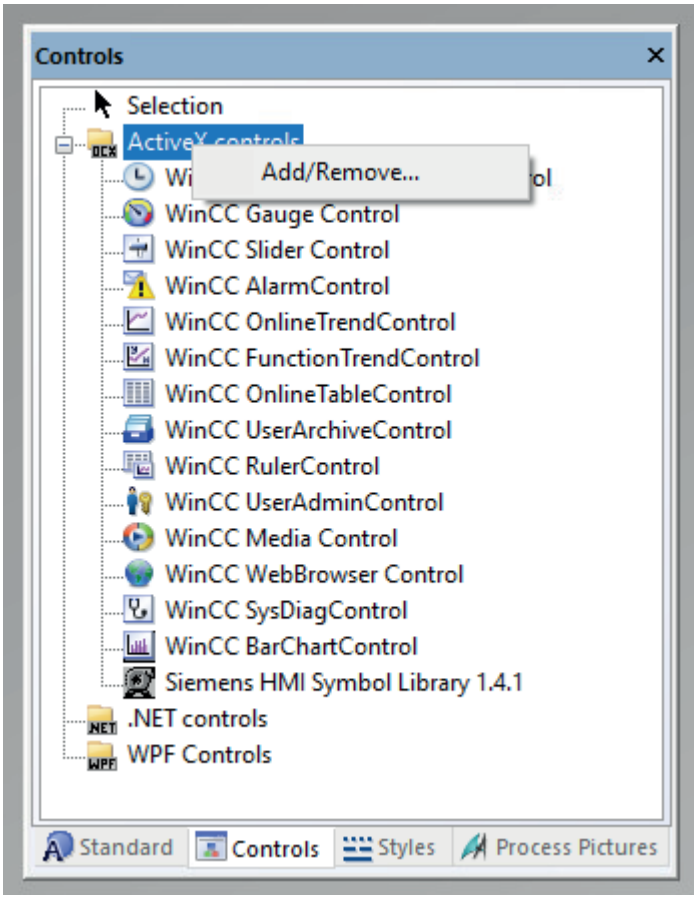
4.2 ibaPDA ActiveX control in SIMATIC WinCC

To set up an ibaPDA *ActiveX Control* on a SIMATIC WinCC system, proceed as follows.

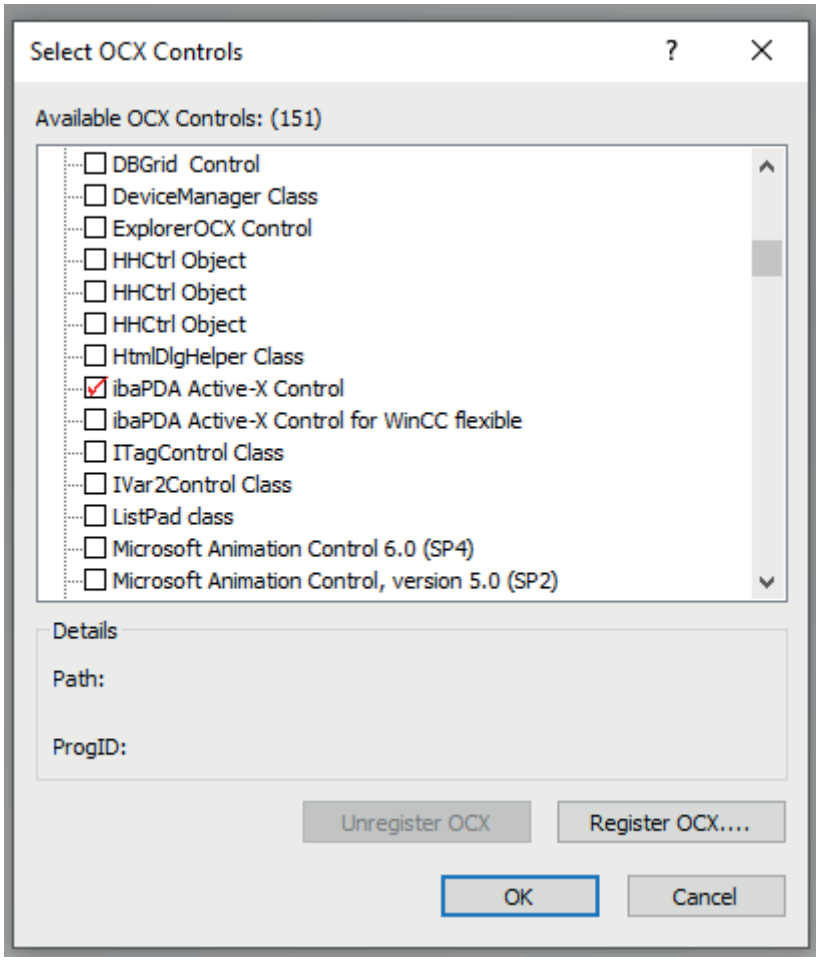
Please note that as a prerequisite, the components *ibaPDA-Client* and *ibaPDA ActiveX-Control* must be installed on the SIMATIC WinCC computer.

Create ActiveX control

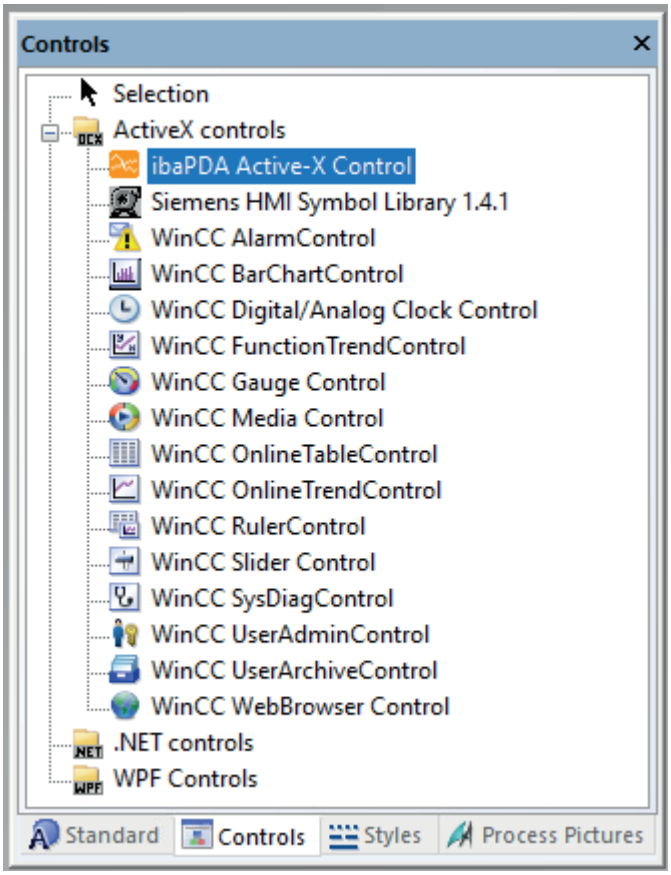
1. Create a new image in the Graphics Designer of SIMATIC WinCC.
2. Open the window with the controls ("Controls"). If *ibaPDA ActiveX Control* is not yet present, you will need to add this first. If the control is already present, proceed to step 5).
3. Right-click in the "Controls" window and select *Add/Remove...* in the context menu.



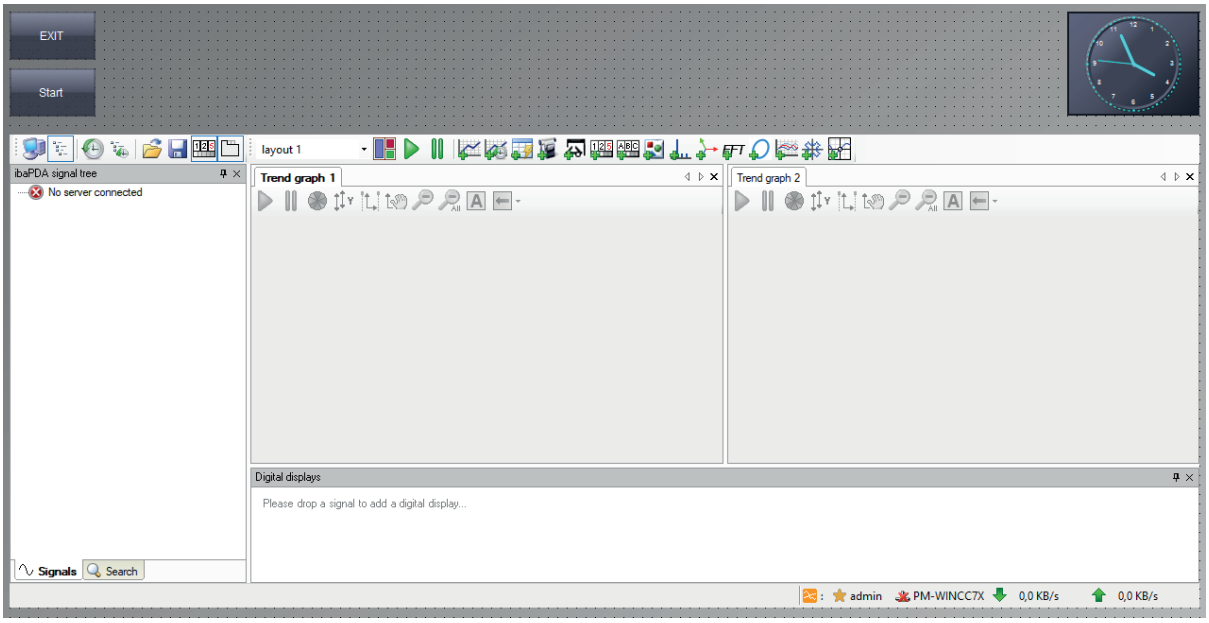
4. In the "Select OCX Controls" dialog, set a check mark next to *ibaPDA ActiveX Control* and close the dialog again with <OK>.



5. Now select the *ibaPDA ActiveX Control* in the "Controls" dialog and insert a new instance into the image.



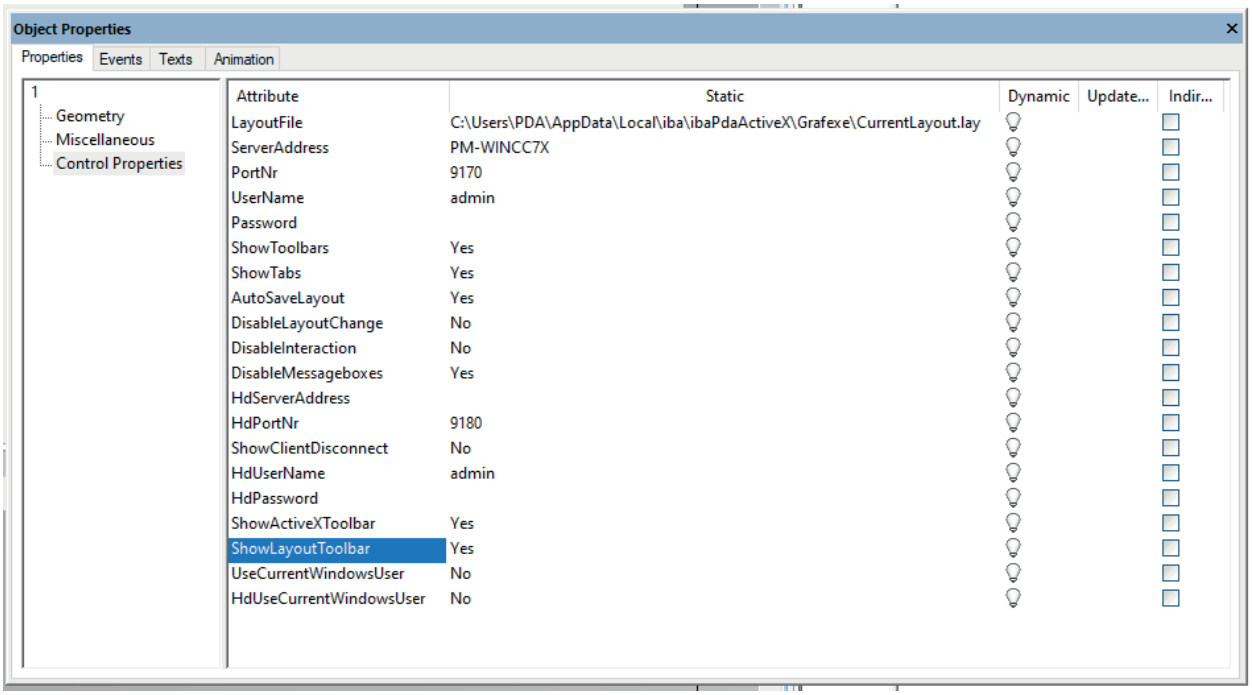
6. The result is a typical *ibaPDA* client interface, whose appearance may differ in design mode and in the subsequent runtime mode, because it is initially a placeholder.



Object properties

The *ibaPDA ActiveX Control* has some properties that you can configure.

In the Graphics Designer, double-click the ActiveX control to access its properties.



A listing of the properties and their description can be found under ↗ *Methods and properties, page 11*.

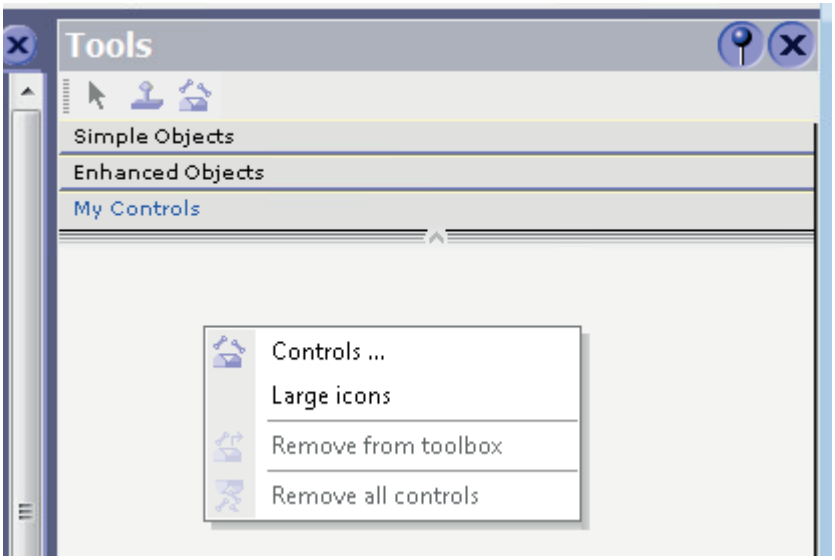
4.3 ibaPDA ActiveX control in SIMATIC WinCC flexible

To set up an ibaPDA *ActiveX Control* on a SIMATIC WinCC flexible system, proceed as follows.

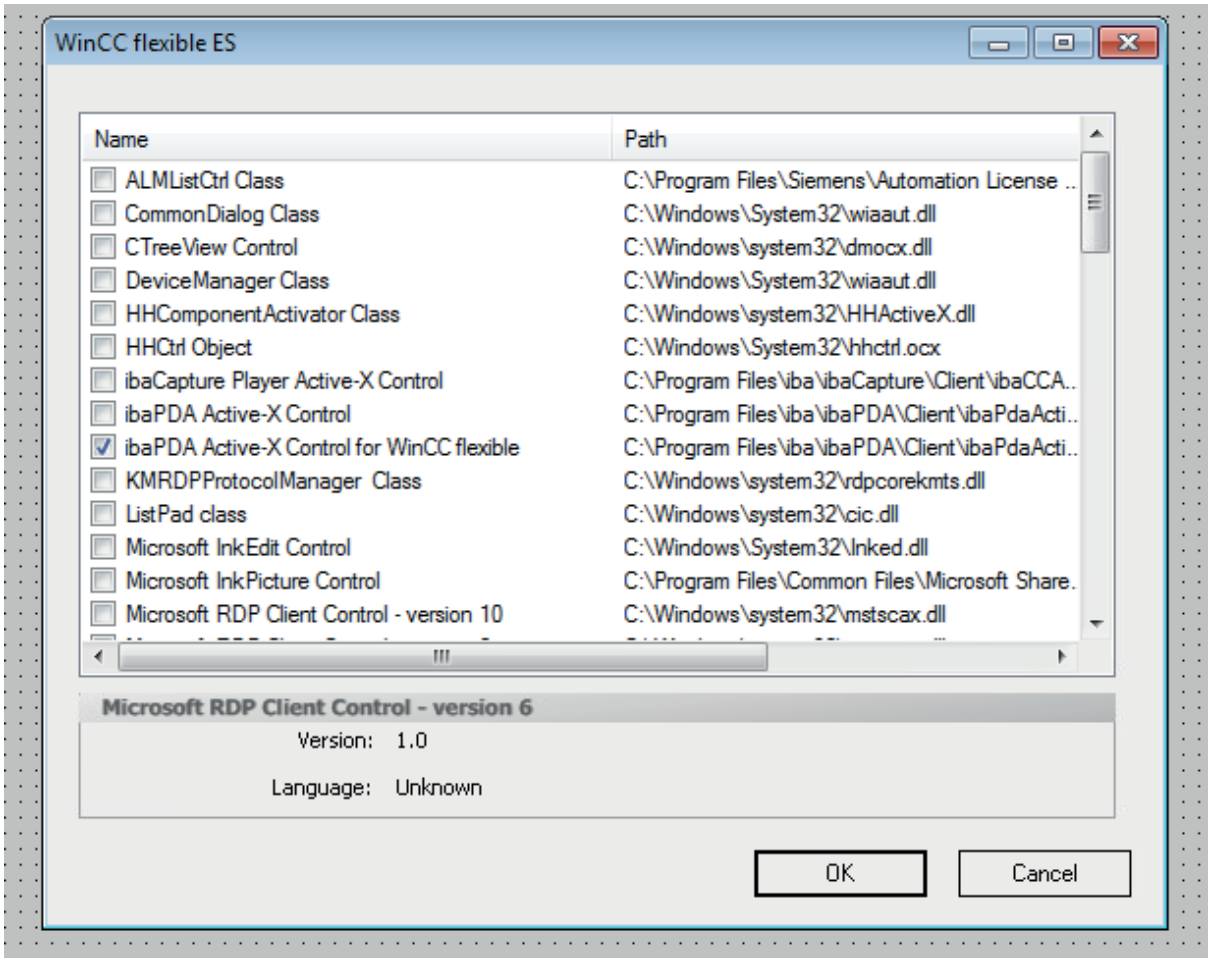
Please note that as a prerequisite, the components *ibaPDA-Client* and *ibaPDA ActiveX-Control* must be installed on the SIMATIC WinCC computer.

Create ActiveX control

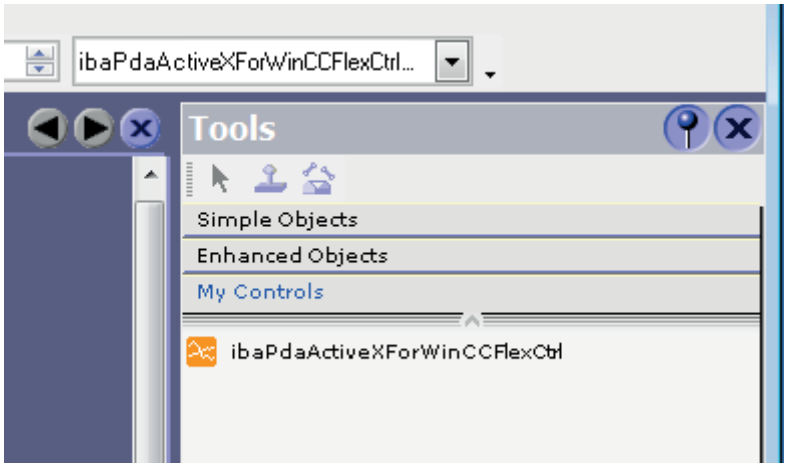
1. Create a new image in SIMATIC WinCC flexible.
2. Open the "Tools" window. If *ibaPdaActiveXForWinCCFlex Control* is not yet listed under *My Controls*, you will need to add this first. If the control is already present, proceed to step 5).
3. Right-click in the "Tools" window and select *Controls...* in the context menu.



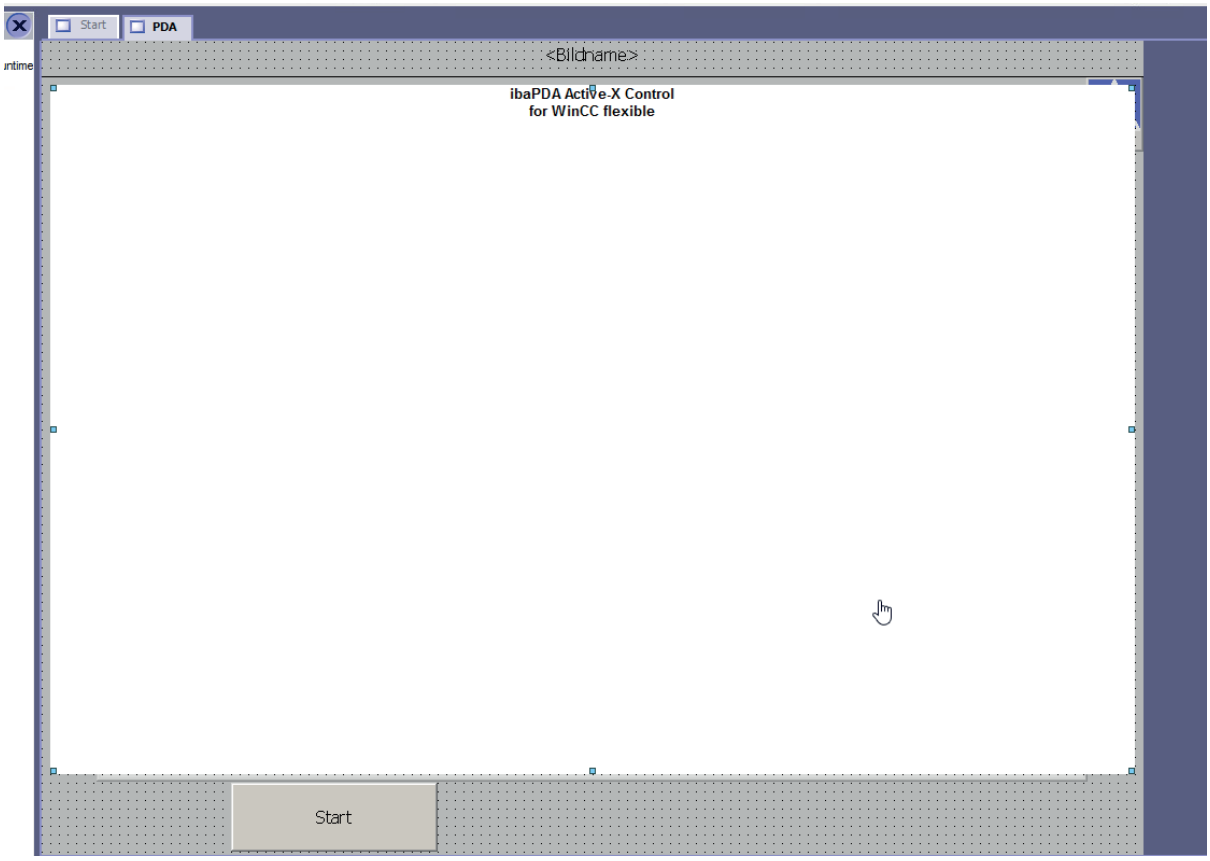
4. In the "WinCC flexible ES" dialog, set a check mark next to *ibaPDA ActiveX Control for WinCC flexible* and close the dialog again with <OK>.



5. Now select the control *ibaPdaActiveXForWinCCFlexCtrl* in the "Tools" window under *My Controls* and insert a new instance into the image. To do this, drag a rectangular frame with your mouse.



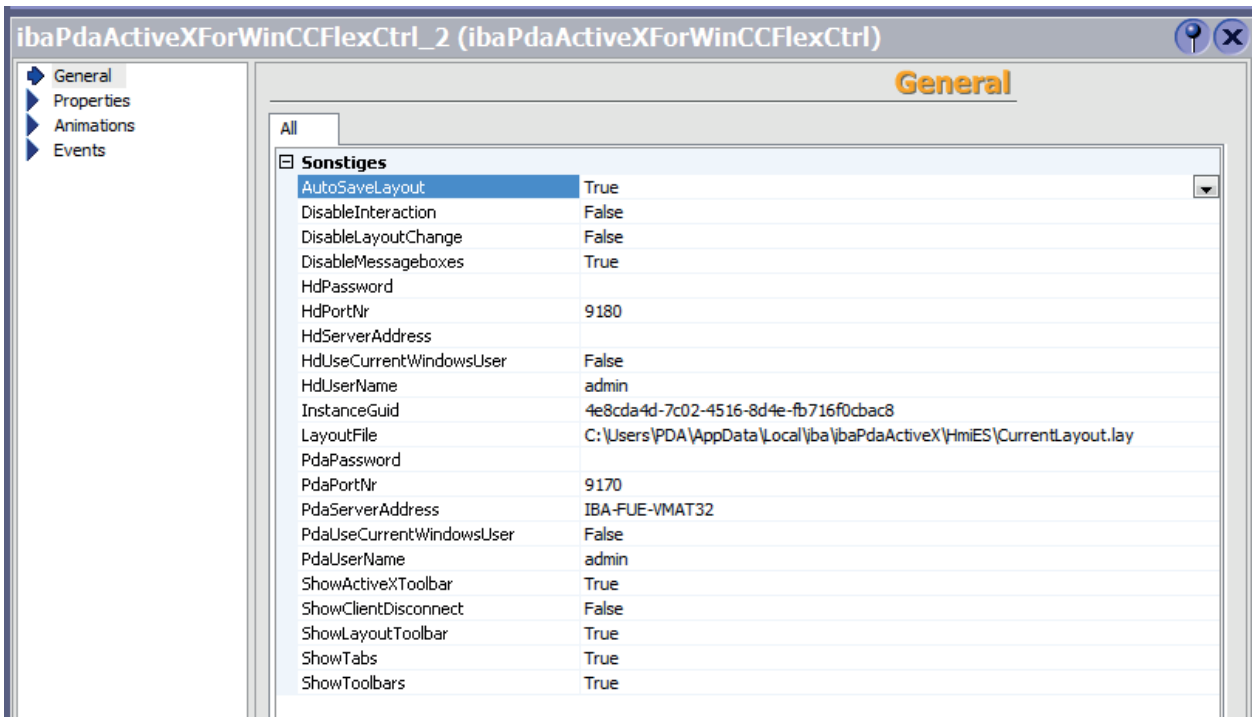
6. The result is a white rectangular box that serves as a placeholder in design mode.



Object properties

The *ibaPDA ActiveX Control* has some properties that you can configure.

Double-click on the ActiveX control to get to its properties.



The properties are similar to those of the ActiveX control for SIMATIC WinCC. They are just listed in a different order, and there is also the Guid of the instance (*InstanceGuid*).

A listing of the properties and their description can be found under ↗ *Methods and properties, page 11*.

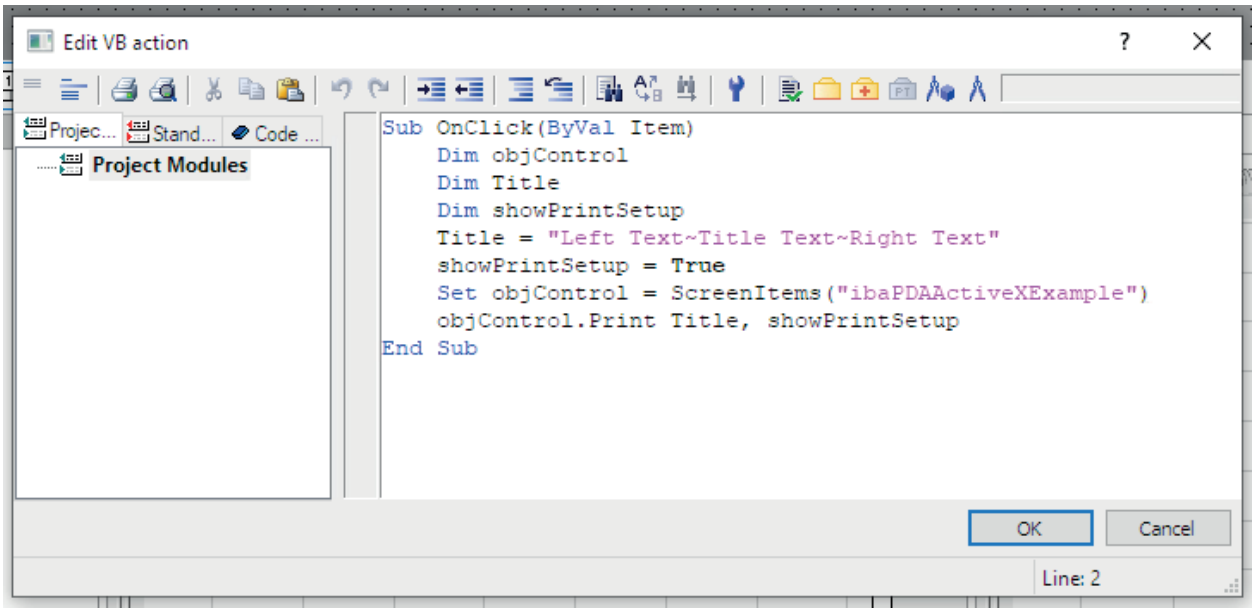
4.4 Sample VB Script for printing out of an ActiveX Control

The *ibaPDA* ActiveX control provides a print function that can be accessed from the ActiveX container (e.g., via VBScript using a button in SIMATIC WinCC).

The print function has 2 arguments:

- STRING title: The header of the printed document. It is possible to use the tilde character ("~") to separately specify three texts for the left, middle and right areas of the header.
- BOOL showPrintSetup: Set this option to TRUE to run the print settings form before printing. The print settings are saved to the layout file configured in the ActiveX control properties.

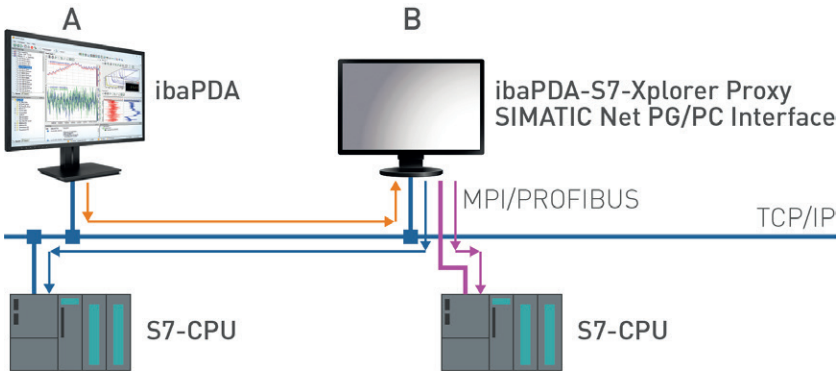
The following is an example of a VBScript in SIMATIC WinCC that can be programmed as a Mouse Click Event using a button, for example. In this case, "ibaPDAActiveXExample" is the object name of the *ibaPDA* ActiveX control in SIMATIC WinCC



The applied method *print* is called by *objControl* in the last line.

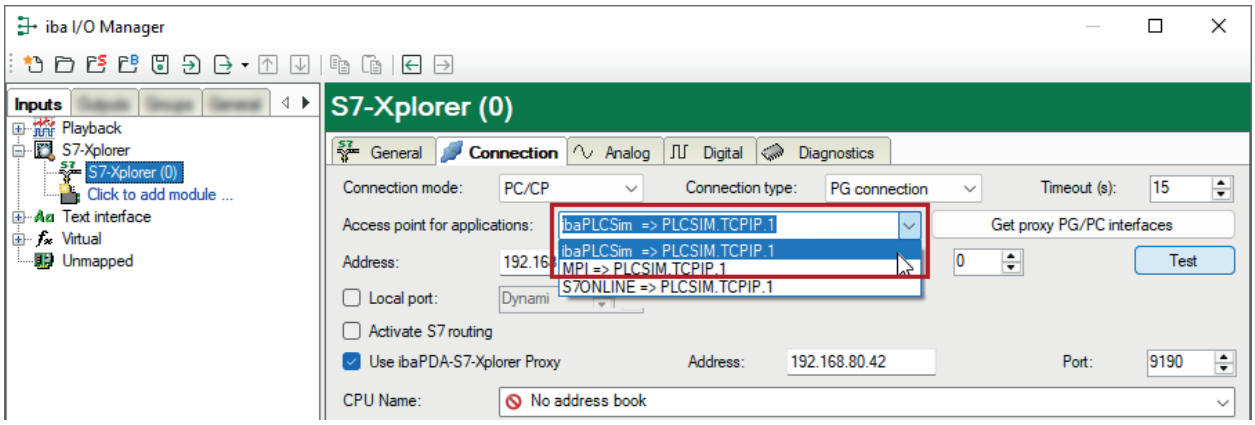
5 ibaPDA-S7-Xplorer Proxy for PC/CP interface

You can also use the *ibaPDA-S7-Xplorer Proxy* when STEP 7 or SIMATIC Net is not installed on the *ibaPDA* computer (computer A), but the CPU still must be accessed via an *S7-Xplorer* module with connection mode PC/CP.



For this purpose, install ant start he *ibaPDA-S7-Xplorer Proxy* on the computer with the STEP 7/ SIMATIC NET installation (computer B).

Unlike the configuration with S7-PLCSIM, you must set a real interface (e.g. the network interface card) as access point for applications. Install the access point on computer B.

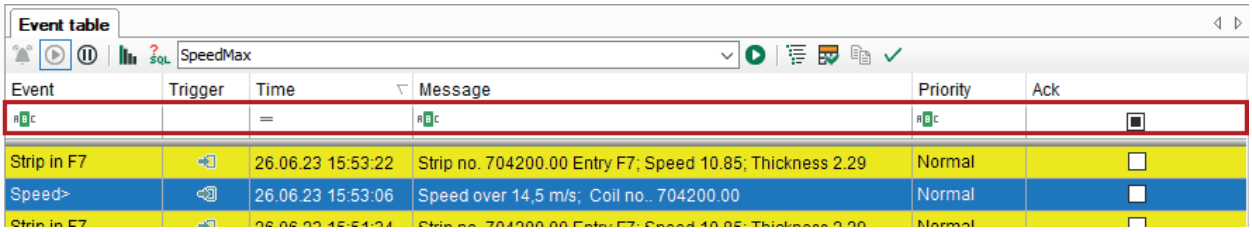


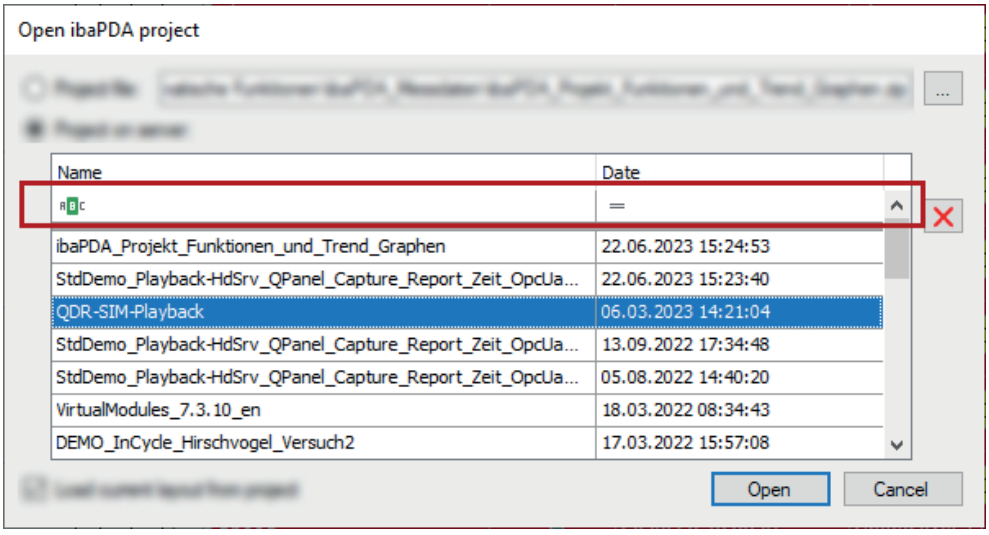
S7-CPU can be accessed via all installed access points.

6 Advanced filter in tables

Many tables used in *ibaPDA* provide a filter function. Right below the column headers there is a filter row.

Tables providing such a filter function are for example the HD Event table or the table listing the *ibaPDA* projects in the *Open ibaPDA project* dialog.

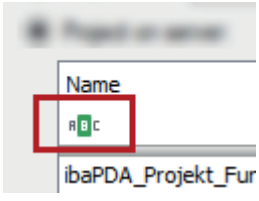




In some of the tables you may immediately see the filter icons, in others you first have to click in the filter row, in order to see the filter icons. Both simple and advanced filter functions are provided.

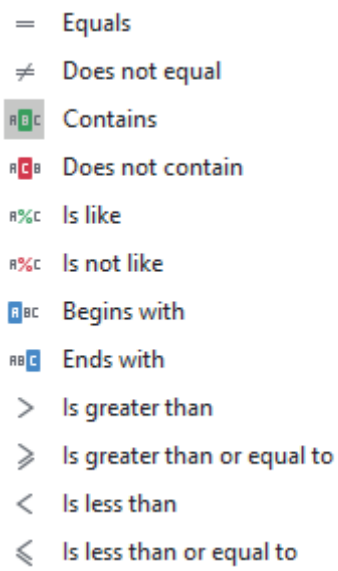
6.1 Simple filter function

Click in a column of the filter row and you will see the icon of the current filter method.



The default method is *Contains* on columns with text as content and *Equals* on columns with numerical content. If the method has already been altered by a user, the latest selected method is shown.

Clicking on the icon displays a list of the available methods for this column.

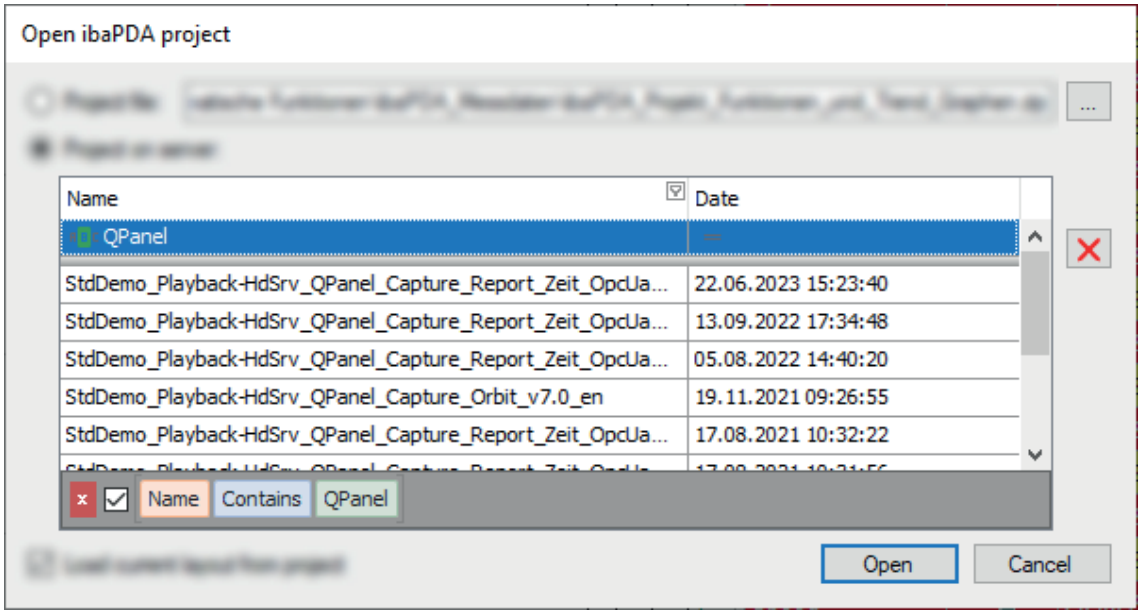


Each method leads to a different filter behavior.

If another filter method suits more your needs, just select it here and enter the filter pattern right into the filter row.

After you have entered a filter pattern the filter is also visualized on the bottom of the table in a human-readable way, showing three fields:

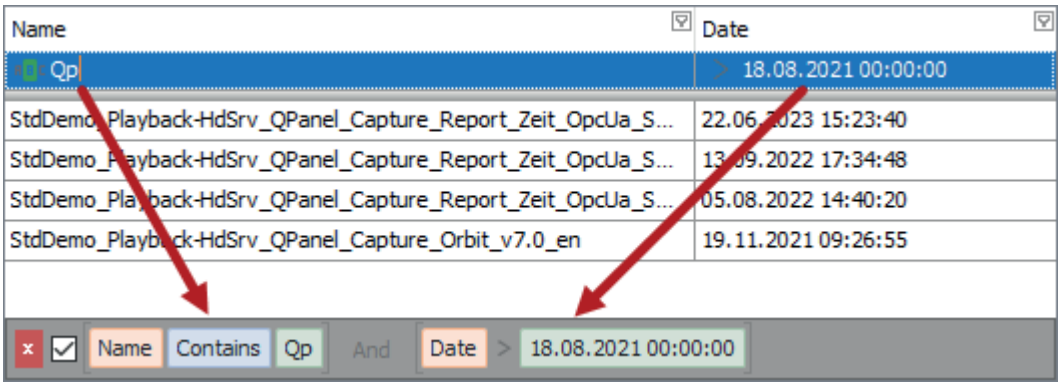
Column name (here *Name*) – filter method (here *Contains*) – filter pattern (here *QPanel*)



The table then only shows the rows which match the filter pattern.

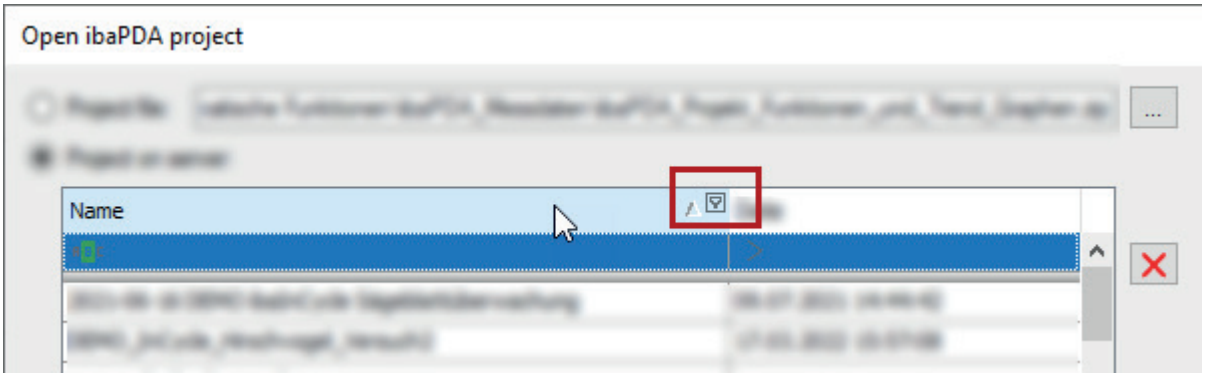
By using the little checkbox you can disable and enable the filter. With the red delete button you remove the filter finally.

If you filter on multiple columns at the same time, the filters are shown as a logical AND combination.



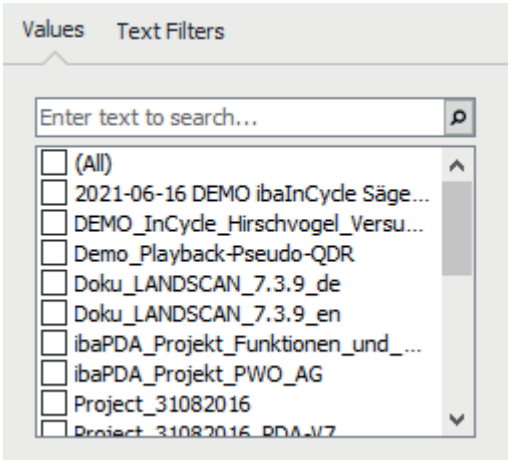
6.2 Advanced filter functions

If you move the mouse cursor over a column header a little filter icon appears at the right end of the column header.



This filter icon, by the way, is always displayed as soon as a filter is enabled.

Clicking on the filter icon opens the advanced filter dialog.

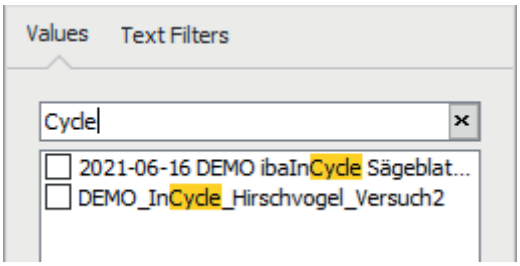


The advanced filter has two tabs:

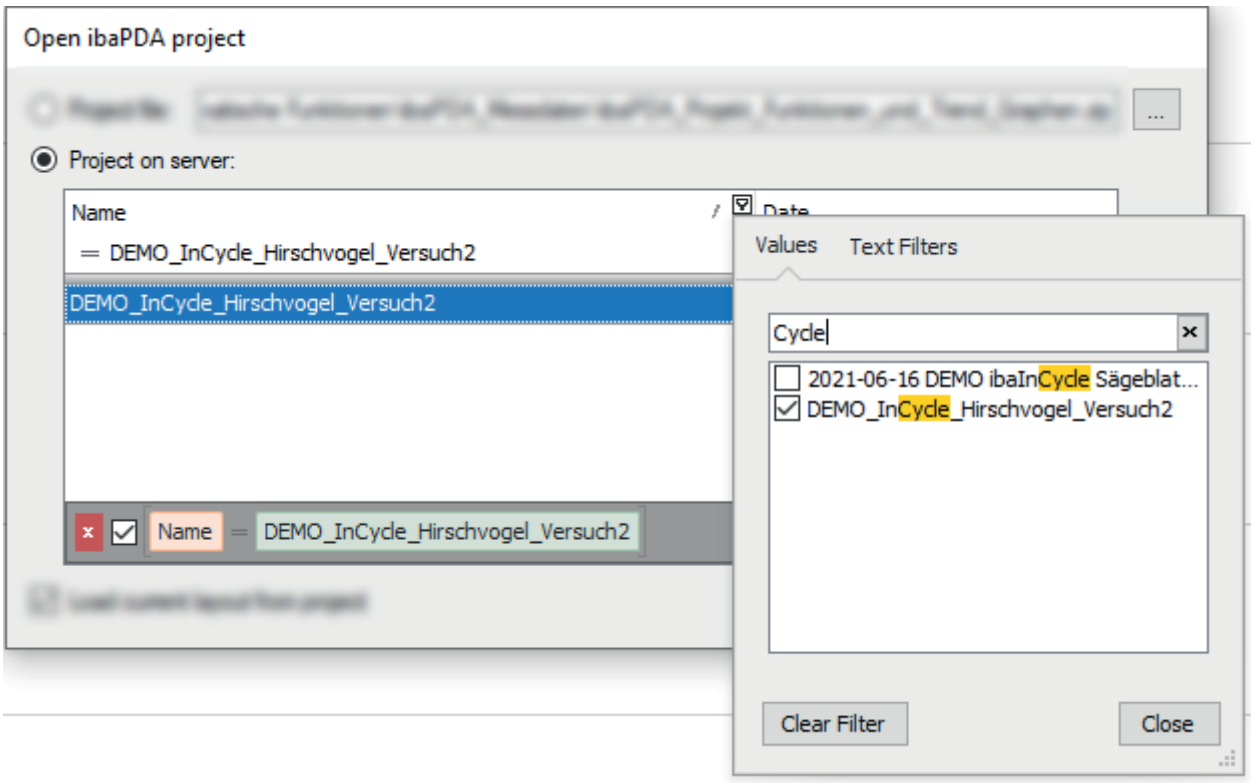
- *Values*, to search for a value or to select one or more values (similar to MS Excel)
- *Advanced filters*, to create a filter based on the value type
  In the example above, the value in the column *Name*, is a text and therefore the second tab is *Text Filters*. There would be a *Date Filters* tab if the value is a date and a *Numeric Filters* tab if the value is a number.

Values

The *Values* tab contains a list of all values in the selected column. Above the list, there is a field where you can enter a search pattern. Rows which don't match the pattern are removed from the table. The search pattern is highlighted yellow in every matching row.



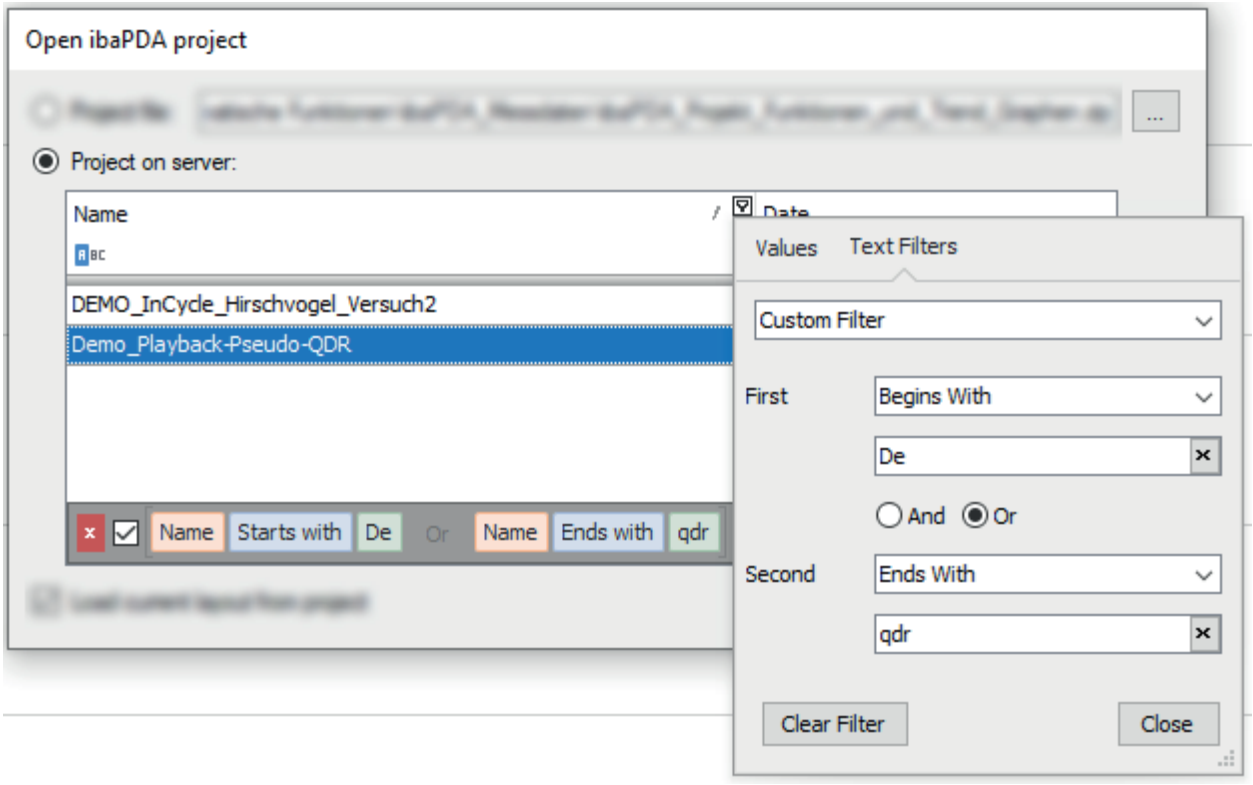
If you select one or multiple search results by a checkmark, only these rows will be displayed in the table.



Advanced filters

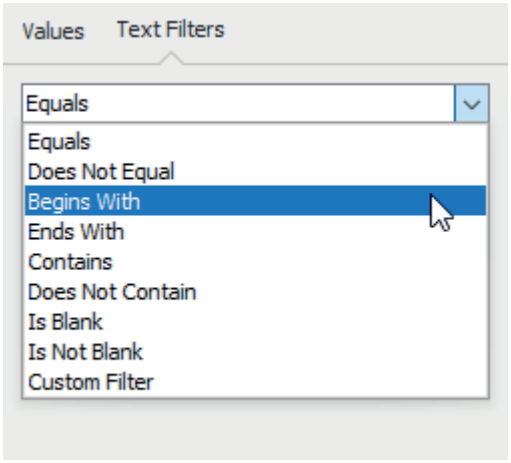
The filter options of the advanced filters depend on the column's value type. A column with numeric value, for example, has different filter options than a column with date values.

Moreover, each filter type provides a custom filter option. In case the standard filters are not suffcient for your needs you can use the custom filter option to create additional filters.



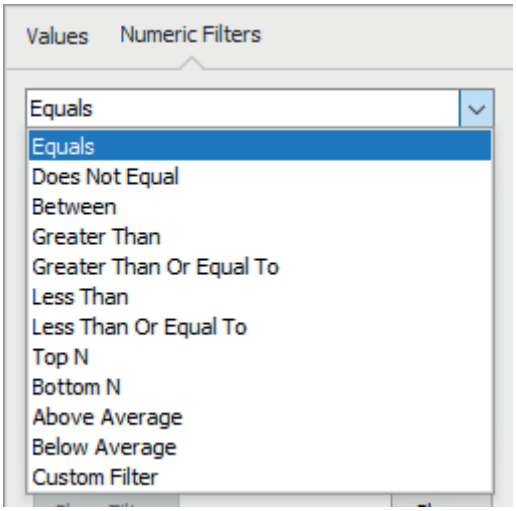
Text filters

Special text filters can be used in columns with text values.



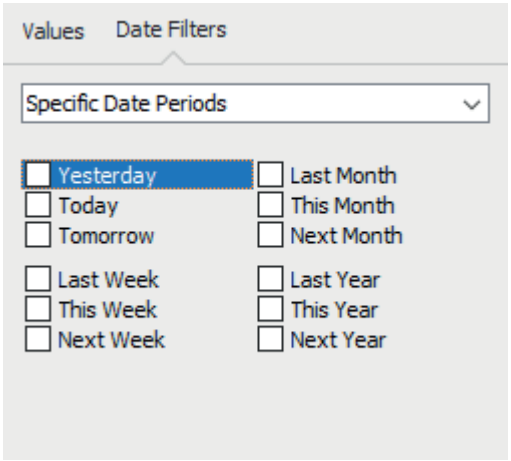
Numeric filters

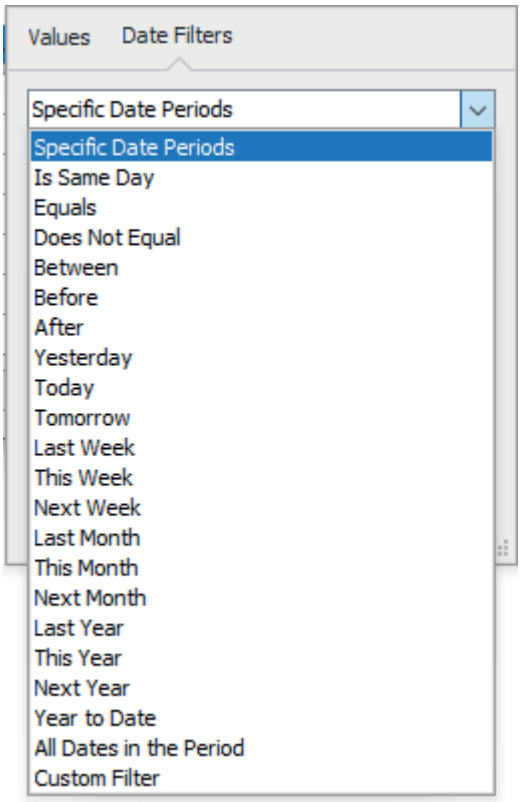
Numeric filters are only available in columns with numbers.



Date filters

Plenty of filters for time ranges and days are available in columns with date values.



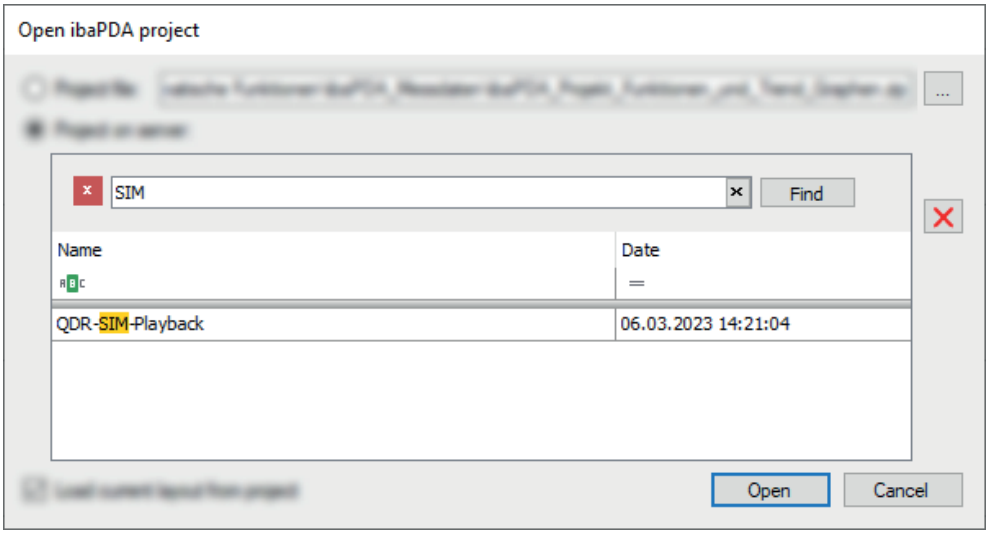


6.3 Search function

Beside the filters you can also use a general search function.

Click into the table and then press <CTRL>+<F>.

A search row will open on top of the table where you can enter a search pattern.



7 Uninstalling ibaPDA

In case you want to uninstall *ibaPDA*, you have different options as follows:

1. Make a right mouseclick on *ibaPDA* Client in the Windows start menu under *ibaPDA* and select *Uninstall* (you will then be transferred to the Windows control panel where you can find the *ibaPDA* installation).
2. Look for *ibaPDA* entry in the Windows settings under Apps and start uninstalling there.

A dialog for uninstalling opens, providing the option to remove all user data along with the program.

You should not enable this option unless you want to remove *ibaPDA* completely and don't want to use it again.

If you enable the option *Delete all ibaPDA user data* the following items will be deleted:

1. All server data in `%programdata%\iba\ibaPDA`. This folder contains:
   - I/O configuration
   - Data store configuration
   - User management configuration
   - Server access configuration
   - Backups of the configurations
   - Saved projects
   - Address books
   - Certificates
   - Log files
   - File buffer
   - Plugins
2. All user data in `%localappdata%\iba\ibaPDA`. This applies to all users of the system. This folder contains:
   - Layouts
   - Backups of layouts
   - Log files
   - Plugins
   - Cached address books
   - Last connected servers
   - Last main window status

3. The user registry key `HKCU\Software\iba\ibaPDA`. This applies to all users of the system. The registry key contains:
   - Preferences
   - Position and size of the *ibaPDA* client window
   - Client settings
4. *ibaPDA* device driver will be uninstalled.

Data files will not be removed.

8 Support file generation via the command line

Generating a support file is usually done interactively in the *Help* menu in *ibaPDA* client with the command *Save information for iba support...*

In addition to the interactive support file generation the program `ibaPdaGenerateSupportFile.exe` facilitates the automatic generation.

Note

The execution of the program requires administrator rights.

8.1 Application

Syntax

```
ibaPdaGenerateSupportFile.exe [/t timeout] <destination>
```

Arguments

- /t timeout (optional)
  Timeout in minutes for the generation of the support file. The process stops if the support file could not be generated within the specified time. Default setting: 2 min
- <destination> (required)
  Destination where the support file should be saved. May be a path or path + filename. If you only specify a folder, the filename will be generated automatically by the program.
- /help or /?
  This will display the help.

```
> .\ibaPdaGenerateSupportFile.exe /help
Usage:
        ibaPdaGenerateSupportFile.exe [/t timeout] <destination>
Arguments:
        /t timeout      Optional: Timeout in minutes for the creation of the support file. Default: 2
        <destination>   Required: The destination where the support file should be created. Can be a file or folder.
Examples:
        ibaPdaGenerateSupportFile.exe "C:\temp"
        ibaPdaGenerateSupportFile.exe /t 5 "C:\temp"
        ibaPdaGenerateSupportFile.exe "C:\temp\supportFile.zip"
```

Examples
Define a folder as destination

```
> ibaPdaGenerateSupportFile.exe "C:\supportFiles"
```

In this case the filename will be generated automatically in the format `ibaPDA_support_yyyyMMdd_HHmmss.zip` in order to prevent duplicates.

Define a filename as destination

```
> ibaPdaGenerateSupportFile.exe "C:\supportFiles\report.zip"
```

Existing files with the same name will be overwritten.

Define an extended timeout (e. g. for larger projects)

In this example 5 minutes.

```
> ibaPdaGenerateSupportFile.exe /t 5 "C:\supportFiles"
```

8.2 Exit codes

If the generation of the support file was successful, the process exits with code 0.

If the generation of the support file failed, the process exits with code -1. In this case the output or the *ibaPDA* server log file should contain more information.

8.3 Error messages

The following errors may occur while generating a support file.

Message	Explanation
Missing destination	The destination argument is missing. Please ensure that a valid <destination> argument is specified.
ibaPDA service is not running	The support file is generated by an active server instance. If the *ibaPDA* service is not running the support file cannot be generated. Please ensure that the service is running.
Registry key <key> cannot be opened	Please ensure that the provided registry key can be accessed.
Timeout of <timeout> minutes exceeded	The file could not be generated within the specified timeout. Please specify a longer timeout and try again.
Invalid zip files	The generated support file is invalid. Please refer to the *ibaPDA* server log files for more information.

9 Support and contact

Support

Phone: +49 911 97282-14

Email: support@iba-ag.com

Note

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

Product Security Incident Response Team (PSIRT) at iba

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

Email: psirt@iba-ag.com

Contact

Headquarters

iba AG
Gebhardtstrasse 10–20
90762 Fuerth
Germany

Phone: +49 911 97282-0

Email: iba@iba-ag.com

Mailing address

iba AG
Postbox 1828
D-90708 Fuerth, Germany

Delivery address

iba AG
Gebhardtstrasse 10
90762 Fuerth, Germany

Regional and worldwide

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

www.iba-ag.com