ibaPDA-Interface-TDC-TCP/UDP

Data Interface TCP/UDP to SIMATIC TDC

Manual

Issue 3.0

Measurement Systems for Industry and Energy

www.iba-ag.com

Manufacturer

iba AG
Koenigswarterstrasse 44
90762 Fuerth
Germany

Contacts

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

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.

Version	Date	Revision	Author	Version SW
3.0	10-2024	New version ibaPDA v8	nm	8.6.0

Contents

1 About this documentation

This documentation describes the function and application of the software interface *ibaPDA-Interface-TDC-TCP/UDP*.

Other documentation

This documentation is a supplement to the *ibaPDA* manual. Information about all the other characteristics and functions of *ibaPDA* can be found in the *ibaPDA* manual or in the online help.

1.1 Target group and previous knowledge

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

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

- Windows operating system
- Basic knowledge of *ibaPDA*
- Knowledge of configuration and operation of the relevant control system

1.2 Notations

In this manual, the following notations are used:

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

1.3 Used symbols

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

Danger!

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

- Observe the specified measures.

Warning!

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

- Observe the specified measures.

Caution!

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

- Observe the specified measures

Note

ℹ A note specifies special requirements or actions to be observed.

Tip

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

Other documentation

📖 Reference to additional documentation or further reading.

2 System requirements

The following system requirements are necessary for the use of the data interface TDC-TCP/UDP:

- *ibaPDA* v8.0.0 or higher
- License for *ibaPDA-Interface-TDC-TCP/UDP*
- Network connection 10/100 Mbit/s
- SIMATIC TDC CPU with integrated PN port or communication processor CP51M1

For further requirements for the used computer hardware and the supported operating systems, please refer to the *ibaPDA* documentation.

Note

It is recommended carrying out the TCP/IP or UDP communication on a separate network segment to exclude a mutual influence by other network components.

System Restrictions

- For different ways of handling the TCP/IP acknowledge, see ↗ *TCP performance problems caused by Delayed Acknowledge*, page 34 (all *ibaPDA* versions).

License information

Order no.	Product name	Description
31.001056	ibaPDA-Interface-TDC-TCP/UDP	Extension license for an *ibaPDA* system by one TCP/IP and UDP/IP interface Number of connections: 64
31.101056	one-step-up-Interface-TDC-TCP/UDP	Extension license for an existing interface *ibaPDA-Interface-TDC-TCP/UDP* by other 64 TCP/UDP connections, max. 3 permitted

3 Data interface TCP/UDP to SIMATIC TDC

3.1 General information

ibaPDA-Interface-TDC-TCP/UDP is able to record measurement data from the SIMATIC TDC control via the standard network card of the *ibaPDA*-PC by means of the protocol TCP/IP or UDP. This requires the data transmission to be programmed in the controller.

The signals to be measured are selected by mapping the values in the telegram buffer whose data blocks are defined by the module types of *ibaPDA*. The telegrams are sent to the *ibaPDA*-PC as standard transmitter block.

Three module types are defined in *ibaPDA-Interface-TDC-TCP/UDP*:

- Integer: 32 analog values (Integer) and 32 binary signal
- Real: 8, 16 or 32 analog values (Real) and 32 binary signals
- Generic data structure with a maximum length of 4096 bytes.

Every module is assigned to a connection. You can create up to 256 connections on the side of *ibaPDA*. The max. number of connections depends on the performance of CPU and CP on the TDC side.

The main advantage for this kind of measurement is that no specific hardware is needed when an Ethernet connection to the control already exists.

Note

The outdated communication processor CP5100 is not able to establish several TCP or UDP connections within the same IP address.

Therefore, iba AG recommends the use of the interface *ibaPDA-Interface-Sisteam-TCP-Generic* (license no. 31.001055).

TCP and UDP

The TCP, Transmission Control Protocol, is a connection-oriented protocol and shall prevent significant data losses, split data and data streams and assign data packages to applications.

The UDP, user datagram protocol, is a connectionless transportation protocol. Its function is similar to that of the connection-oriented TCP. However, it works connectionless and therefore insecure. This means that the sender does not know whether the data packets it has sent have actually arrived. TCP sends confirmations upon receiving data, UDP does not. This method has the advantage that the packet header is much smaller and no acknowledgments have to be sent over the link. In principle, this enables a slightly higher data rate.

Both work with the Internet Protocol IP, the layer 4 (transportation layer) of the OSI-layer model.

Note

When in the following examples connections are referred to in UDP, these connections are not to be established nor to be terminated; only the communication channel from sender to recipient is labeled.

3.2 SIMATIC TDC configuration & engineering

3.2.1 General settings

This section describes the establishment of the TCP/IP or UDP connection, the necessary data blocks and the parameterizing of the send blocks.

There are two versions of the standard transmitter block:

- CTV: The data to be transmitted are saved as “virtual connections” in the telegram buffer. The telegram length is a result of the number and size of the data stored in the telegram buffer. Therefore, all data must be loaded, gaps in the telegram are not possible.
- CTV_P: The telegram buffer will be created by the send block with the indicated length. By means of a pointer the telegram buffer address is handed over to the write blocks in order to store the data sent in the telegram buffer. Because every write block indicates an offset, not all data have to be filled in.

The following chapters only describe the request with the transmitter block CTV_P.

Please observe the following in all connections described:

- SIMATIC TDC TDC has to be configured as TCP/IP client, i. e. the TDC side establishes the connection, therefore in the AT connector of the send block the address stage 2 has to be configured.
- The data are swapt on the TDC side, because they are expected to be in the same format as S7; this means all WRITE blocks have to be set on SWP=1.
- The remote port must match the setting in *ibaPDA* (interface TDC TCP UDP) (default setting in *ibaPDA*: 4171).
- This port has to be enabled in the *ibaPDA* PC in the Windows firewall.
- This port cannot be allocated elsewhere.
- When creating additional conditions, please observe:
    - Always assign new channel names
    - Always issue a new local port number
    - Always use the same remote IP address
    - Always use the same remote port number

3.2.2 Data structures

According to the ibaPDA module structure, the data for each module are transmitted with a telegram. The telegrams have an unified header and a data structure matched to the module type.

3.2.2.1 Header

The header consists of 3 Integer values.

- message_length
  total size (in bytes) of the data package. The value cannot be changed during data transmission. This value also has to be specified when connecting to the send block (CTV_P). The value depends on the module type:
  - with module type Integer: 74
  - with module type Real: 42, 74 or 138 (with 8, 16 or 32 Reals),
  - with module type Generic 8…4096
- module_index
  Identifier for assigning the data record to the interface module in *ibaPDA.* In this index, the module type is encrypted as well .The index is created by a serial number 00....63 and an offset that corresponds to the module type and the license.

  The module index complies with the *ibaPDA* module settings. The value has to be unique and cannot be changed during the data transmission.

Module type	1st License	2nd License	3rd License	4th License
Integer	0-63	1000-1063	2000-2063	3000-3063
Real	100-163	1100-1163	2100-2163	3100-3163
Generic	200-263	1200-1263	2200-2263	3200-3263

- sequence_counter
  With every successful send job the value will be incremented by one. This has to be programmed on TDC side. If the counter value does not change by +1, *ibaPDA* displays a sequence error in the connection list. In the event of an overflow, the counter must jump from 32767 to -32768 (0x7FFF -> 0x8000)

3.2.2.2 Data range

The structure of the data range depends on the module type.

Module type Integer

After the header, starting at offset 6, follow the 32 integer analog values and subsequently, starting at offset 70, the 4 bytes of binary values.

Offset	Data type	Name	Meaning
00	INT	message_length	Telegram length = 74
02	INT	module_index	Module index, i000 – i063
04	INT	sequence_counter	Message counter
06	INT[32]	Analog values 0-31	32 Values in 16 bit Integer format
70	DWORD	Digital values 0-31	32 Digital values

Module type Real

After the header, starting at offset 6, follow the 4 bytes of binary values and subsequently, starting at offset 10, either 8, 16 or 32 analog values in the Real format.

Offset	Data type	Name	Meaning
00	INT	message_length	Telegram length 42, 74 or 128
02	INT	module_index	Module index i100-i163
04	INT	sequence_counter	Message counter
06	DWORD	Digital values 0-31	32 Digital values
10	FLOAT[n]	Analog values 0-n	n values in IEEE float format n=8, 16 or 32

Module type Generic

Any order of data with different data types can follow after the header starting at offset 6. *ibaPDA* supports the following data formats for analog signals:

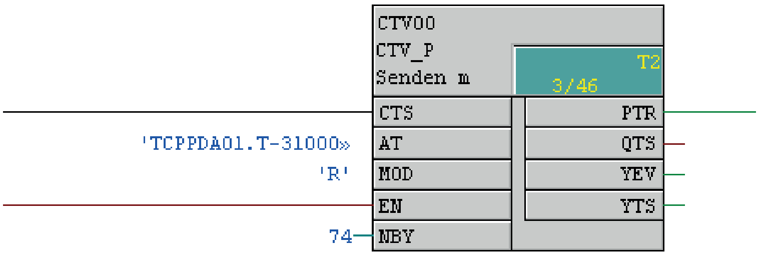
SINT, BYTE, INT, WORD, DINT, DWORD, FLOAT, DOUBLE, STRING[32]

In *ibaPDA* the data structure outlined here have to be recreated. The BYTE, WORD and DWORD variables may also be interpreted as 8, 16 or 32 bits (and vice versa).

Offset	Data type	Name	Meaning
00	INT	message_length	Telegram length max. 4102
02	INT	module_index	Module index i200-i263
04	INT	sequence_counter	Message counter
06	BYTE[n]	data	Generic data buffer n <= 4096

3.2.3 Configuration of the connection

The connection setup is done via the transmitter block CTV or CTV_P. All communication parameter will be encrypted in the address connector AT.



Connection	Meaning	Meaning
CTS	HW connection	Connected to CP51M1
AT	Address	See below

MOD	Channel mode	R = Refresh (recommended)
		H = Handshake

EN	Enable	Transmit trigger
NBY	No of Bytes	Telegram buffer length
PTR	Buffer indicator	Indicator on telegram buffer
QTS	Block status 1	1 = OK
YEV	Block status 2	Coupling status: 0 = OK
YTS	Block status 3	Additional information

Structure address connector AT:

'aaaaaaaa.b-ccccc.dddddddddddd-eeeee' with

aaaaaa: unique channel name, max. 8 characters

b: 'T' stands for TCP, 'U' stands for UDP

ccccc: local port number, freely selectable, unique, 5 digit with leading zeros

dddddddddddd: remote IP address, decimal representation but without full stop, with leading zeros

eeeee: remote port number, has to match *ibaPDA* interface 5 digit with leading zeros

Example: "TCPPDA01.T-31000.192168050203-04171" means

TCPPDA01	Channel name
T-	TCP connection
31000	local port number
192168050203	Remote IP address 192.168.50.203
04171	remote port number 4171

Example: "UDPPDA014.U-31003.192168050203-04171" means

UDPPDA01	Channel name
U-	UDP connection
31003	local port number
192168050203	Remote IP address 192.168.50.203
04171	remote port number 4171

Other documentation

Further information can be found in the manual "SIMATIC TDC - System and Communication Configuration D7-SYS"(Siemens AG).

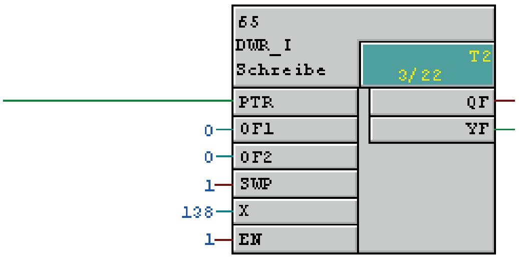
3.2.4 Sending data

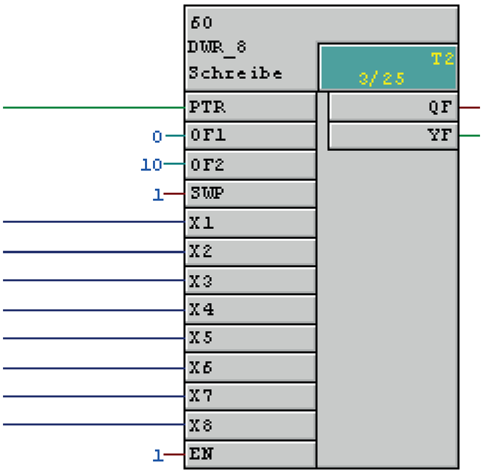
The data to be transmitted are written with the block DWR in the telegram buffer There are several variants of this block for the data types BYTE, INT and DINT and the number of signals.

Note

On the TDC side, the bytes have to be reversed, since they are expected to be in the same order as on S7.

Please put the connector SWP on 1 with every DWR block.





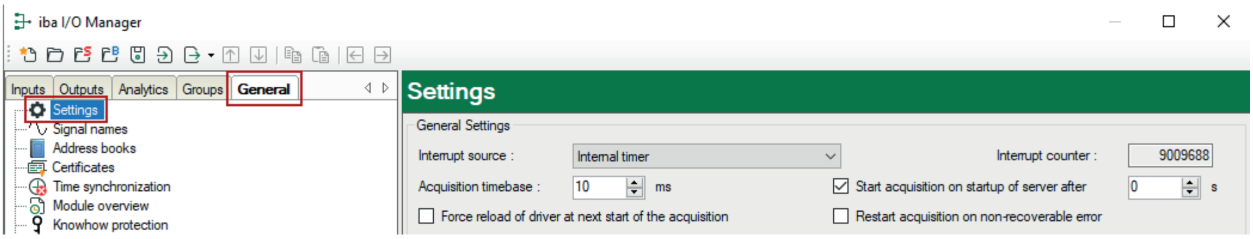
Connection	Meaning	Meaning
PTR	Buffer indicator	Connection to PTR of the transmitter block CTV_P
OF1	Offset 1	The offset in the telegram buffer is formed from OF1 + OF2
OF2	Offset 2	
SWP	SWAP	Byte rotation. For *ibaPDA* telegrams it must be set on 1
Xn	Data transmitted	
EN	Enable	
QF	Block status 1	0 = OK, 1= Error
YF	Block status 2	Error information

3.3 Configuration and engineering ibaPDA

The engineering for *ibaPDA* is described in the following. If all system requirements are fulfilled, *ibaPDA* displays the *TDC TCP/UDP* interface in the interface tree of the I/O Manager.

3.3.1 General settings

The "Alive timeout" is configured jointly for all TCP/IP and UDP protocols supported by *ibaPDA*.



Settings

General Settings

Interrupt source : Internal timer

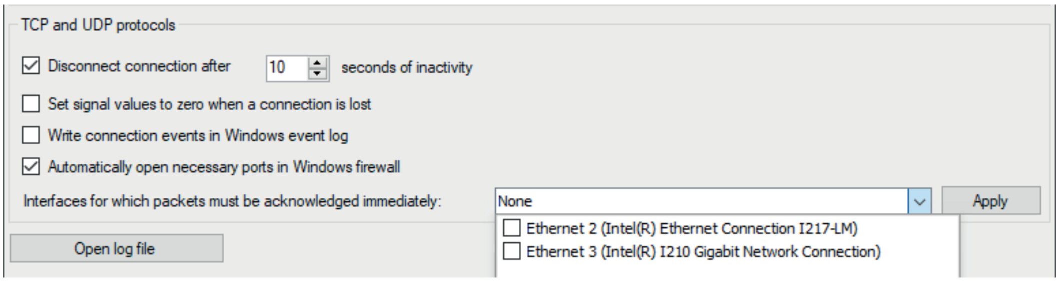
Interrupt counter : 9009688

Acquisition timebase : 10 ms

☑ Start acquisition on startup of server after 0 s

□ Force reload of driver at next start of the acquisition

□ Restart acquisition on non-recoverable error



TCP and UDP protocols

☑ Disconnect connection after 10 seconds of inactivity

□ Set signal values to zero when a connection is lost

□ Write connection events in Windows event log

☑ Automatically open necessary ports in Windows firewall

Interfaces for which packets must be acknowledged immediately: None

□ Ethernet 2 (Intel(R) Ethernet Connection I217-LM)

□ Ethernet 3 (Intel(R) I210 Gigabit Network Connection)

Disconnect connection after … seconds of inactivity
Behavior and timeout duration can be specified.

Set signal values to zero when a connection is lost
If this option is disabled, the value read last will be kept.

Write connection events in Windows event log
Current events are logged in Windows.

Automatically open necessary ports in Windows firewall
If this option is enabled, all ports required for the currently licensed interfaces are automatically opened in the firewall by the *ibaPDA* server service.

If this option is disabled, the required ports can be opened manually in the I/O Manager of the licensed interfaces via <Allow port through firewall>.

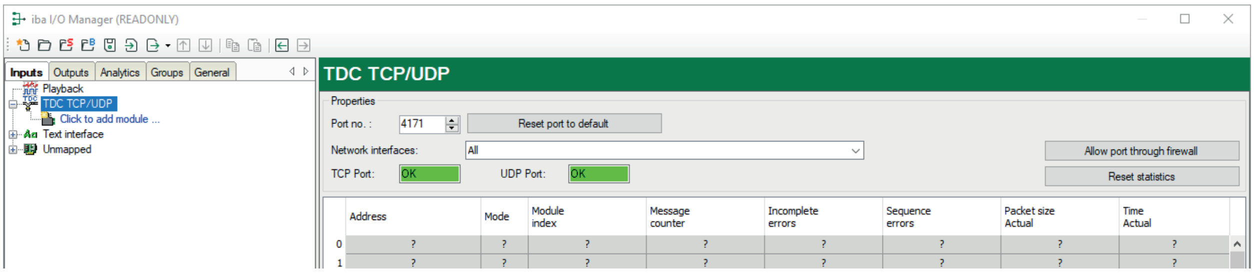
Interfaces for which packets must be acknowledged immediately
Selection of required interfaces.

Note

In case *ibaPDA*is the active partner (Client), *ibaPDA*reestablishes the connection after only a few seconds. Thus, it gives to the passive partner the possibility to send data again.

3.3.2 General interface settings

The interface itself has the following functions and configuration options:



Port no. : 4171

Network interfaces: All

TCP Port: OK

UDP Port: OK

	Address	Mode	Module index	Message counter	Incomplete errors	Sequence errors	Packet size Actual	Time Actual
0	?	?	?	?	?	?	?	?
1	?	?	?	?	?	?	?	?

Port no.
Used port PC side. The port number must be used identically in the SIMATIC TDC connection configuration (see chapter ➚ *SIMATIC TDC configuration & engineering*, page 10).

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.

Reset port to default>
The port number 4171 is set.

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.

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 occupied.

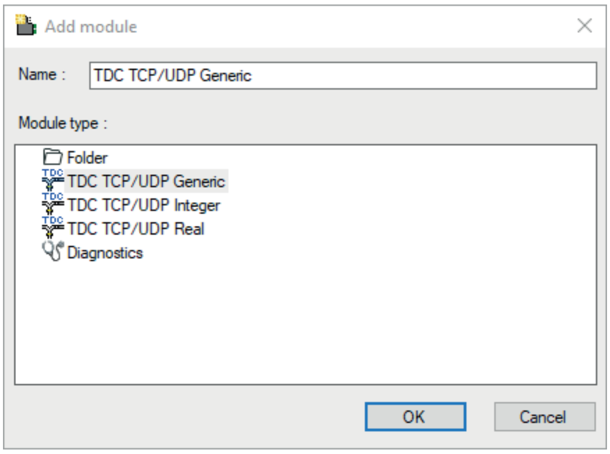
<Reset statistics>
Click this button to reset the calculated times and error counters in the table to 0.

Connection table
See ➚ *Connection table*, page 26

3.3.3 Adding a module

Procedure

1. Click on the blue link *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 with <OK>.



Name : TDC TCP/UDP Generic

Module types

You can add the following module types to the interface:

- TDC TCP/UDP Generic
- TDC TCP/UDP Integer
- TDC TCP/UDP Real

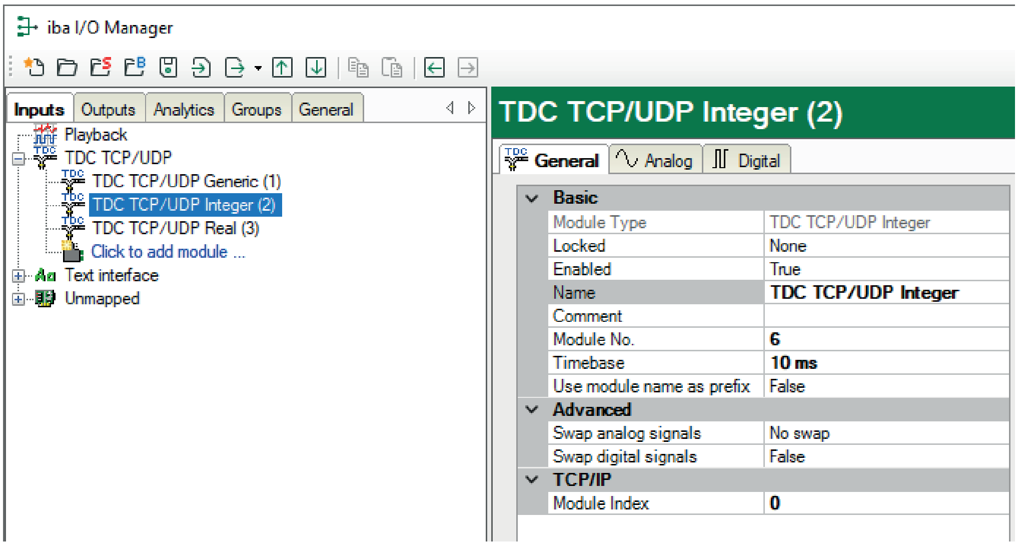
Note

When a TCP/IP or UDP connection to SIMATIC TDC is established, right-click on the interface and choose *Autodetect*. Then the modules are automatically created for all available connections

3.3.4 General module settings

To configure a module, select it in the tree structure.

All modules have the following setting options.



Basic	
Module Type	TDC TCP/UDP Integer
Locked	None
Enabled	True
Name	**TDC TCP/UDP Integer**
Comment	
Module No.	**6**
Timebase	**10 ms**
Use module name as prefix	False
Advanced	
Swap analog signals	No swap
Swap digital signals	False
TCP/IP	
Module Index	**0**

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.

Advanced

Swap analog signals/Swap digital signals
Option to change the order of the byte evaluation

TCP/IP

Module Index

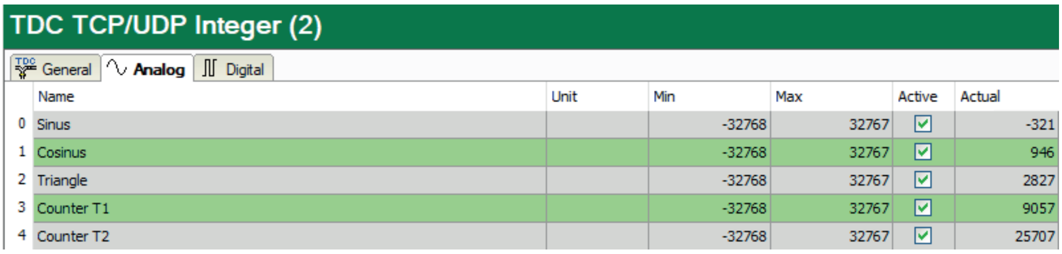
The module indices are created by a serial number 00....63 and an offset that corresponds to the module type and the license. See ➚ *Header*, page 11

3.3.5 Signal configuration

The data to be measured are selected on the SIMATIC TDC side by mapping the signals in the telegram buffer.

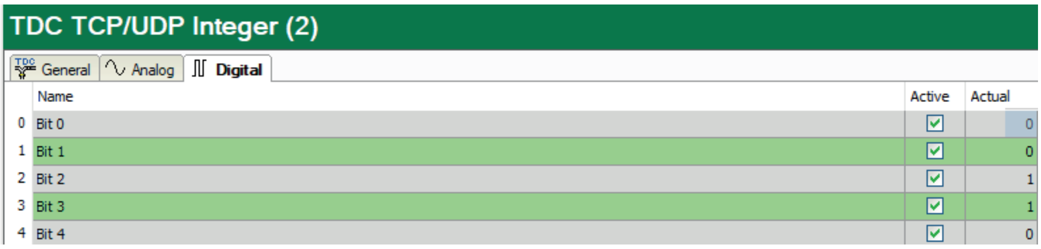
Analog and Digital tab

In the I/O Manager, the signals can be given name and unit (only analog signals) and can be set as active and inactive.



TDC TCP/UDP Integer (2)

	Name	Unit	Min	Max	Active	Actual
0	Sinus		-32768	32767	☑	-321
1	Cosinus		-32768	32767	☑	946
2	Triangle		-32768	32767	☑	2827
3	Counter T1		-32768	32767	☑	9057
4	Counter T2		-32768	32767	☑	25707



TDC TCP/UDP Integer (2)

	Name	Active	Actual
0	Bit 0	☑	0
1	Bit 1	☑	0
2	Bit 2	☑	1
3	Bit 3	☑	1
4	Bit 4	☑	0

Name

Enter a meaningful plain text name for the signal.

Unit (analog signals only)

Assignment of a physical unit for the signal

You can enter a maximum of 11 characters, the field is only considered a comment field. The unit is always displayed in conjunction with a numerical display of the values.

Active

Activation or deactivation of the respective signal

Actual

Display of the current actual value of the signal

Other documentation

Detailed descriptions of the columns and how to fill in the signal tables can be found in the documentation for *ibaPDA*.

3.3.6 Module type Integer

The *TDC TCP/UDP Integer* module allows up to 32 analog values (Integer) and 32 binary signals to be acquired.

For this module, no module specific settings are required.

3.3.7 Module type Real

The *TDC TCP/UDP Real* module allows up to 32 analog values (Real) and 32 binary signals to be acquired.

The following module settings are module-specific:

Number of analog signals
The number of analog signals is configurable in level 8 ,16 ,32 (number of digital signals is fixed at 32).

3.3.8 Module type Generic

Any data block with max. length of 4096 bytes can be measured by means of the *TDC TCP/UDP Generic* module.

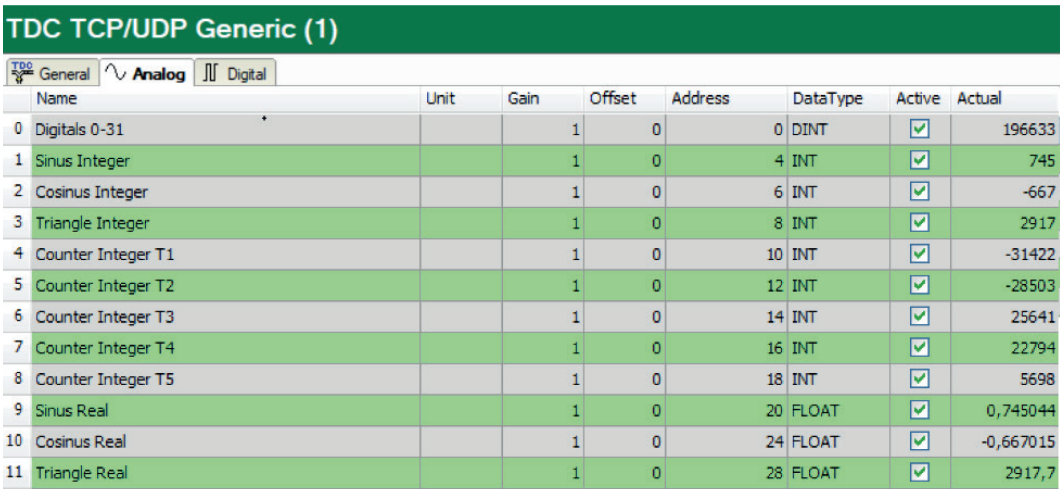
The following module settings are module-specific:

Text encoding
Here you can specify the codepage which should be applied for interpreting the received text data.

Number of analog/digital signals
Maximum number of analog and digital signals that can be configured. Presetting is 32 each. The maximum value is 1000.

For signal configuration, enter the address, i.e. the offset in the telegram buffer and the data type for each variable. Bear in mind that counting starts from the beginning of user data without header.

Various data types are supported for the analog signals, including texts: SINT, BYTE, INT, WORD, DINT, DWORD, FLOAT, DOUBLE, STRING[32].



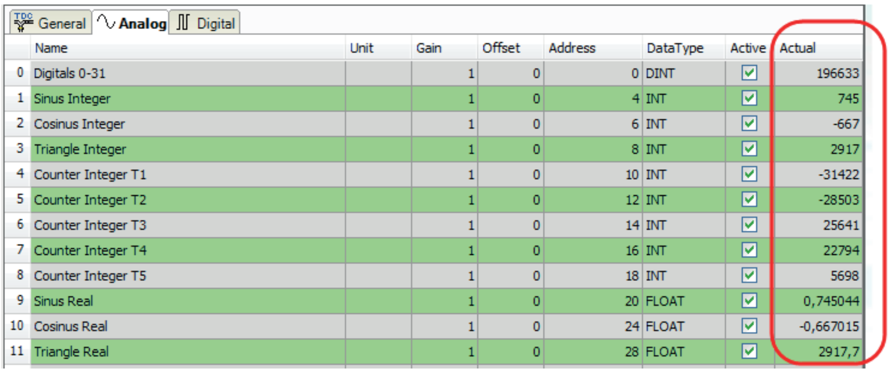
	Name	Unit	Gain	Offset	Address	DataType	Active	Actual
0	Digitals 0-31		1	0	0	DINT	☑	196633
1	Sinus Integer		1	0	4	INT	☑	745
2	Cosinus Integer		1	0	6	INT	☑	-667
3	Triangle Integer		1	0	8	INT	☑	2917
4	Counter Integer T1		1	0	10	INT	☑	-31422
5	Counter Integer T2		1	0	12	INT	☑	-28503
6	Counter Integer T3		1	0	14	INT	☑	25641
7	Counter Integer T4		1	0	16	INT	☑	22794
8	Counter Integer T5		1	0	18	INT	☑	5698
9	Sinus Real		1	0	20	FLOAT	☑	0,745044
10	Cosinus Real		1	0	24	FLOAT	☑	-0,667015
11	Triangle Real		1	0	28	FLOAT	☑	2917,7

Note

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

3.3.9 Module diagnostics

The tables *Analog* and *Digital* of the TDC-TCP/UDP modules show the telegram contents.



	Name	Unit	Gain	Offset	Address	DataType	Active	Actual
0	Digitals 0-31		1	0	0	DINT	☑	196633
1	Sinus Integer		1	0	4	INT	☑	745
2	Cosinus Integer		1	0	6	INT	☑	-667
3	Triangle Integer		1	0	8	INT	☑	2917
4	Counter Integer T1		1	0	10	INT	☑	-31422
5	Counter Integer T2		1	0	12	INT	☑	-28503
6	Counter Integer T3		1	0	14	INT	☑	25641
7	Counter Integer T4		1	0	16	INT	☑	22794
8	Counter Integer T5		1	0	18	INT	☑	5698
9	Sinus Real		1	0	20	FLOAT	☑	0,745044
10	Cosinus Real		1	0	24	FLOAT	☑	-0,667015
11	Triangle Real		1	0	28	FLOAT	☑	2917,7

The following errors may occur:

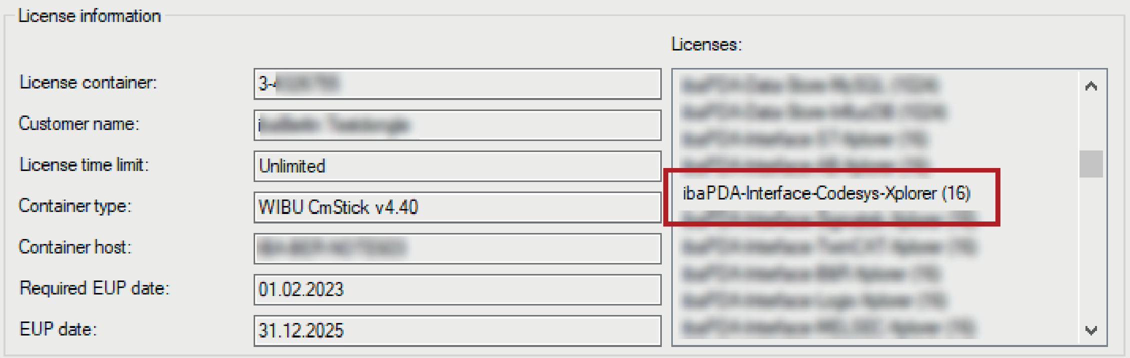
Error	Cause/Remedy
No data is displayed.	The telegram buffer on the TDC side is not filled correctly.
	The connectors of the transmitter block are wired incorrectly.
Incorrect values are displayed.	The telegram buffer on the TDC side is not filled correctly (offset error).
	The byte order is set incorrectly, see ➚ *General module settings*, page 19 and ➚ *Sending data*, page 14.
	There are multiple modules with the same module index.
The digital signals are sorted incorrectly.	The byte order is set incorrectly, see ➚ *General module settings*, page 19 and ➚ *Sending data*, page 14
The telegrams arrive not faster than ca. 200 ms with sequence error.	Problem with "Delayed Acknowledge", see ➚ *TCP performance problems caused by Delayed Acknowledge*, page 34
	Problem caused by "Nagle's Algorithm", see ➚ *TCP data corruption resulting from the Nagle's Algorithm*, page 36

4 Diagnostics

4.1 License

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

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



License information

License container:	3-[redacted]
Customer name:	i[redacted]
License time limit:	Unlimited
Container type:	WIBU CmStick v4.40
Container host:	[redacted]
Required EUP date:	01.02.2023
EUP date:	31.12.2025

Licenses:

[redacted]

ibaPDA-Interface-Codesys-Xplorer (16)

[redacted]

4.2 Visibility of the interface

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

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

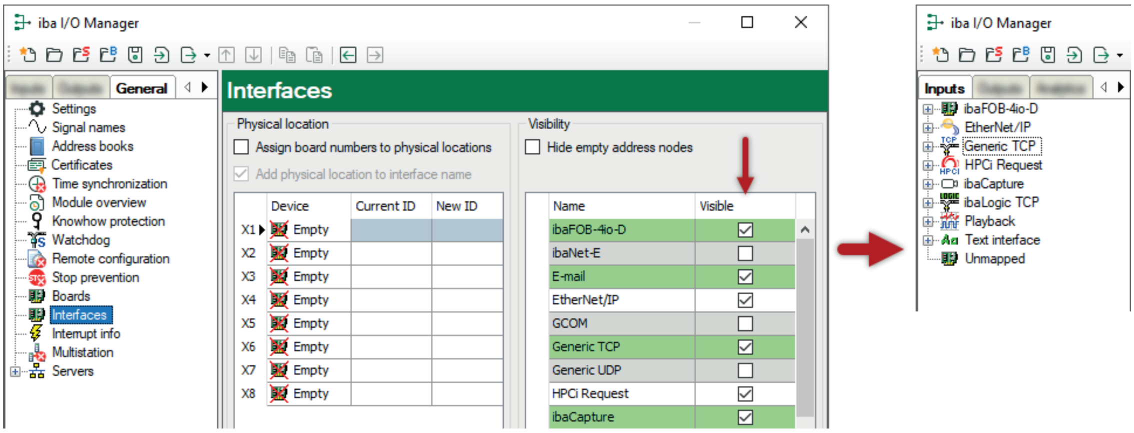
Visibility

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

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

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

Selected interfaces are visible, the others are hidden:



Interfaces

Physical location

☐ Assign board numbers to physical locations

☑ Add physical location to interface name

Visibility

☐ Hide empty address nodes

	Device	Current ID	New ID
X1	Empty		
X2	Empty		
X3	Empty		
X4	Empty		
X5	Empty		
X6	Empty		
X7	Empty		
X8	Empty		

Name	Visible
ibaFOB-4io-D	☑
ibaNet-E	☐
E-mail	☑
EtherNet/IP	☑
GCOM	☐
Generic TCP	☑
Generic UDP	☐
HPCi Request	☑
ibaCapture	☑

4.3 Log files

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

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

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

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

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

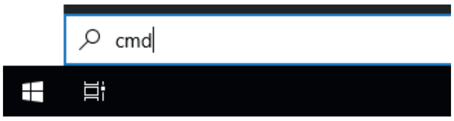
Examples:

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

4.4 Connection diagnostics with PING

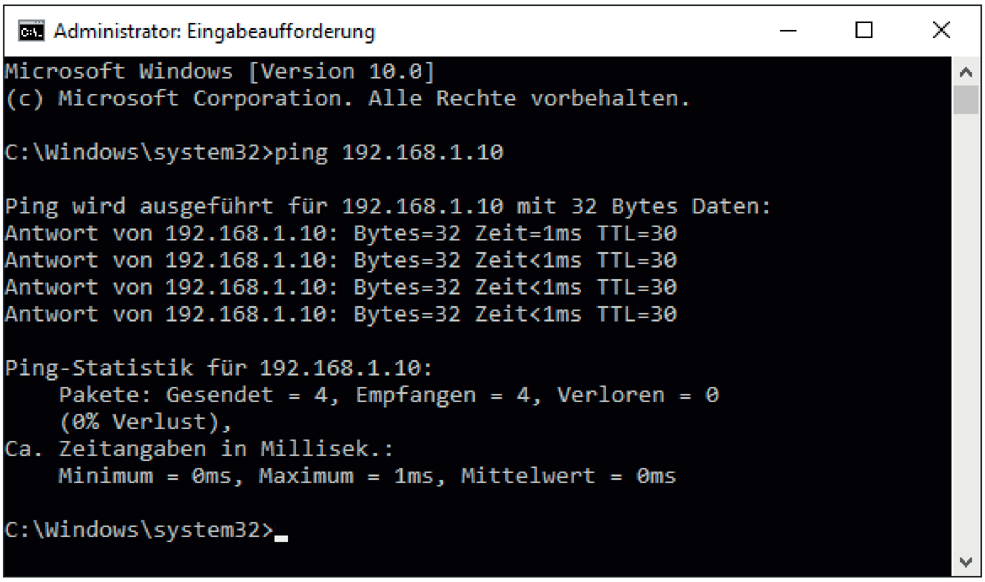
PING is a system command with which you can check if a certain communication partner can be reached in an IP network.

1. Open a Windows command prompt.



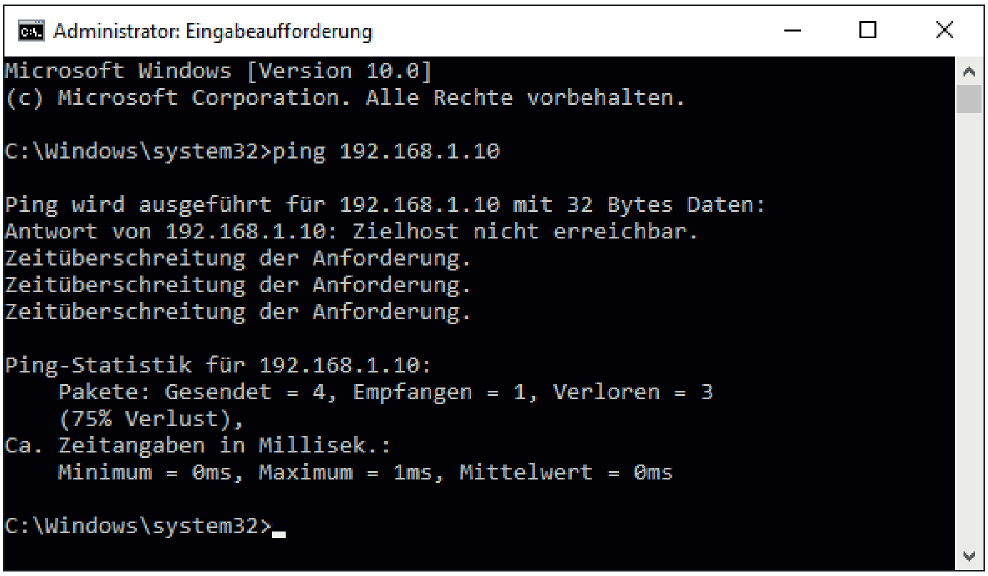
2. Enter the command "ping" followed by the IP address of the communication partner and press <ENTER>.

→ With an existing connection you receive several replies.



```
Microsoft Windows [Version 10.0]
(c) Microsoft Corporation. Alle Rechte vorbehalten.

C:\Windows\system32>ping 192.168.1.10

Ping wird ausgeführt für 192.168.1.10 mit 32 Bytes Daten:
Antwort von 192.168.1.10: Bytes=32 Zeit=1ms TTL=30
Antwort von 192.168.1.10: Bytes=32 Zeit<1ms TTL=30
Antwort von 192.168.1.10: Bytes=32 Zeit<1ms TTL=30
Antwort von 192.168.1.10: Bytes=32 Zeit<1ms TTL=30

Ping-Statistik für 192.168.1.10:
    Pakete: Gesendet = 4, Empfangen = 4, Verloren = 0
    (0% Verlust),
Ca. Zeitangaben in Millisek.:
    Minimum = 0ms, Maximum = 1ms, Mittelwert = 0ms

C:\Windows\system32>_
```

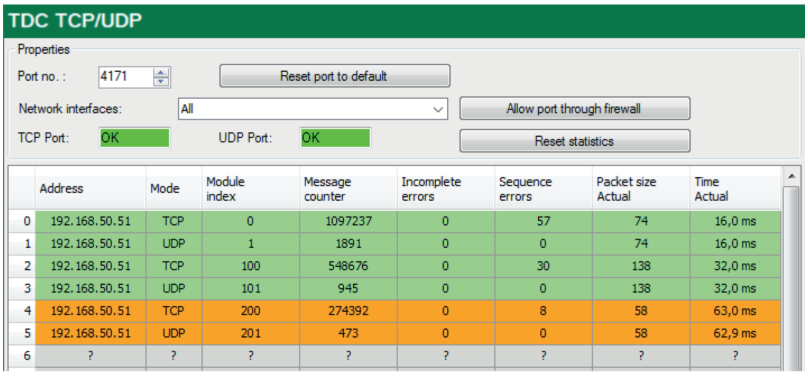
→ With no existing connection you receive error messages.



```
Microsoft Windows [Version 10.0]
(c) Microsoft Corporation. Alle Rechte vorbehalten.

C:\Windows\system32>ping 192.168.1.10

Ping wird ausgeführt für 192.168.1.10 mit 32 Bytes Daten:
Antwort von 192.168.1.10: Zielhost nicht erreichbar.
Zeitüberschreitung der Anforderung.
Zeitüberschreitung der Anforderung.
Zeitüberschreitung der Anforderung.

Ping-Statistik für 192.168.1.10:
    Pakete: Gesendet = 4, Empfangen = 1, Verloren = 3
    (75% Verlust),
Ca. Zeitangaben in Millisek.:
    Minimum = 0ms, Maximum = 1ms, Mittelwert = 0ms

C:\Windows\system32>_
```

4.5 Connection table

After the configuration was accepted, all connections will be shown in the connections overview sorted according to their module index.



TDC TCP/UDP

Properties

Port no. : 4171

Network interfaces: All

TCP Port: OK

UDP Port: OK

	Address	Mode	Module index	Message counter	Incomplete errors	Sequence errors	Packet size Actual	Time Actual
0	192.168.50.51	TCP	0	1097237	0	57	74	16,0 ms
1	192.168.50.51	UDP	1	1891	0	0	74	16,0 ms
2	192.168.50.51	TCP	100	548676	0	30	138	32,0 ms
3	192.168.50.51	UDP	101	945	0	0	138	32,0 ms
4	192.168.50.51	TCP	200	274392	0	8	58	63,0 ms
5	192.168.50.51	UDP	201	473	0	0	58	62,9 ms
6	?	?	?	?	?	?	?	?

The background color of the lines has the following meaning:

Color	Meaning
Green	The connection is OK. The *ibaPDA* module timebase is equally quick or slower than the telegram cycle. The current telegram cycle is shown in the column “Time Actual”.
Orange	The connection is OK, but the telegram cycle is significantly slower than the *ibaPDA* module timebase. It is recommended to adjust the module timebase to the telegram cycle.
Gray	No connection configured.

If the connections are not displayed or only partially, this may have the following causes:

- SIMATIC TDC in stop mode
- No Ethernet connection between *ibaPDA* PC and SIMATIC CP51M1
- Error in configuration:
  - The local port number is unique.
  - The remote IP address is incorrect.
  - The configured port number does not comply with *ibaPDA* port.
  - The port number is blocked by the firewall.
- Wrong module index specified in the telegram header

Other errors:

- If the telegram counters do not increment continuously, the transmitter blocks are not called cyclically on the TDC side.
- If values in the columns *Incomplete errors* and/or *Sequence errors* are incremented, this points to one of the following errors:
  - The “message_length” in the telegram does not meet the expected value.
  - The “sequence_counter” in the telegram is not incremented correctly.
  - The "Delayed Ackowledge” problem occurs (see ➚ *TCP performance problems caused by Delayed Acknowledge*, page 34)

4.6 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.

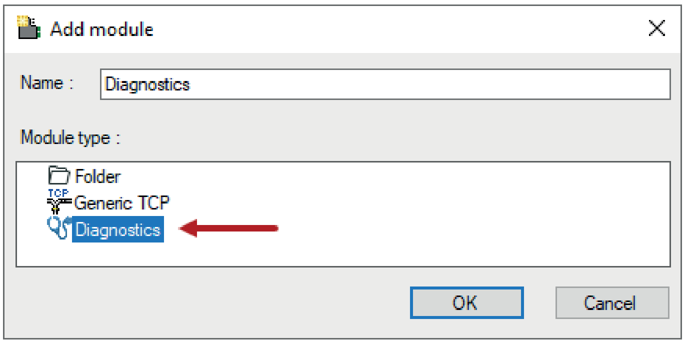
A diagnostic module is always assigned to a data acquisition module of the same interface and supplies its connection information. By using a diagnostic module, you can record and analyze the diagnostic information continuously in the *ibaPDA* system.

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).

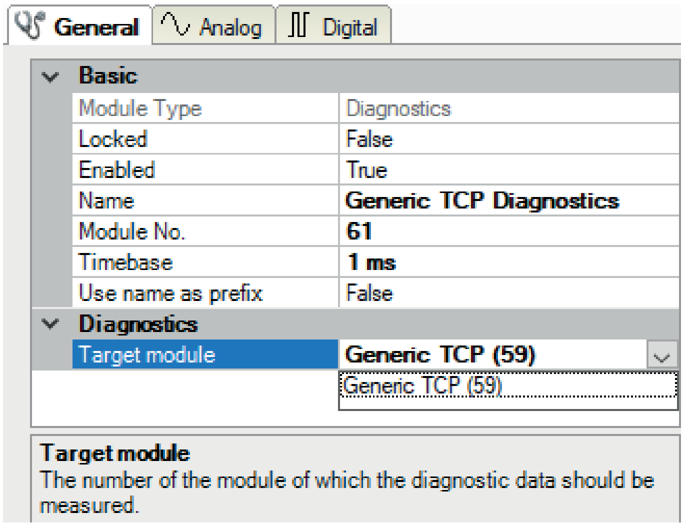


Name : Diagnostics

Module type : Diagnostics

Module settings diagnostic module

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



Basic

Module Type	Diagnostics
Locked	False
Enabled	True
Name	**Generic TCP Diagnostics**
Module No.	**61**
Timebase	**1 ms**
Use name as prefix	False

Diagnostics

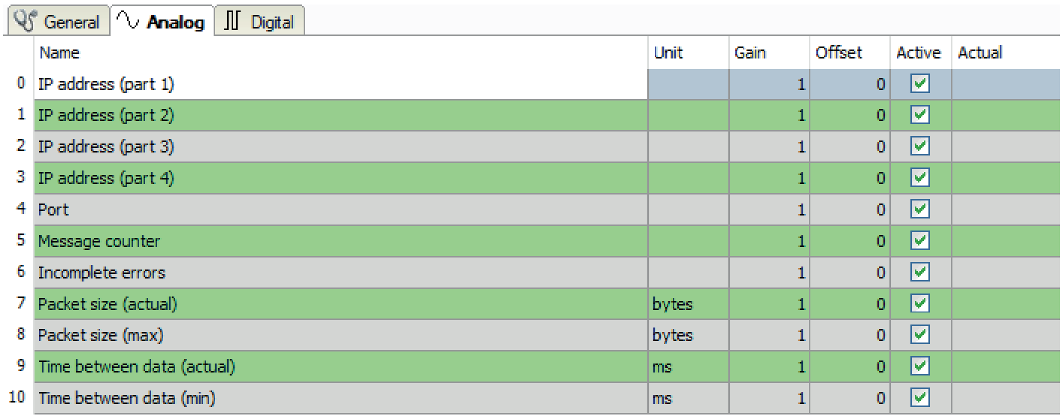
Target module	**Generic TCP (59)**
	Generic TCP (59)

Target module
The number of the module of which the diagnostic data should be measured.

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

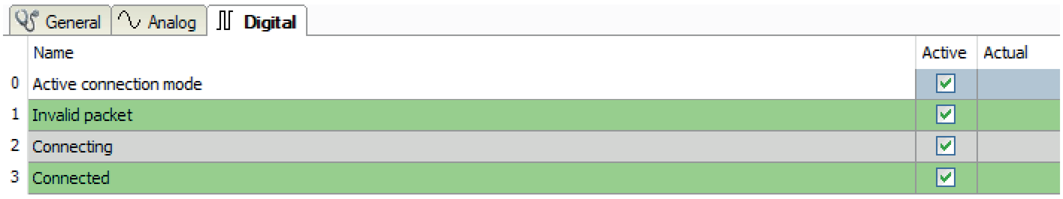
There is only one setting which is specific for the diagnostic module: the target module.

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 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 (actual) can be higher than the set value, if the PLC needs more time to transfer the data. How fast the data is really updated, you can check in the connection table. The minimum achievable update time is influenced by the number of signals. The more signals are acquired, the greater the update time becomes. Average/max/min: static values of the update time since the last start of the acquisition or reset of the counters.
Write counter	Number of successful write accesses
Write lost counter	Number of failed write accesses

4.7 Performance

Please note that the transmission via TCP and UDP cannot guarantee a reliable deterministic. The maximum reliable data rate strongly depends on the quality of the network used. If quicker data cycles (smaller than 20 ms) are required, we recommend an own network.

Our performance measurements were executed in the iba company network.

4.7.1 TCP telegrams

- Data volume: 8 module type 32-Real (256 analog values and 256 digital values)
- TDC transmission cycle: 8 ms
- *ibaPDA* basic sampling speed: 1 ms
- Loss rate: approx. 10 sequence errors with 100000 telegrams
- Gross transmission rate: 138000 bytes/second
- User data transmission rate: 32000 real values + 32000 digital values/second

4.7.2 UDP telegrams

- Data volume: 1 module type Generic UDP, length 1086 bytes (256 analog values and 256 digital values)
- TDC transmission cycle: 4 ms
- *ibaPDA* basic sampling speed: 1 ms
- Loss rate: approx. 8 telegram losses with 45000 telegrams (3 min)
- Gross transmission rate: 271500 bytes/second
- User data transmission rate is the same as the gross transmission rate

Sporadic telegram losses are caused by sporadic network load. With point-to-point connections (crossover cables) no telegram losses are expected.

5 Appendix

5.1 Troubleshooting

In the following you will find help on possible errors when using *ibaPDA-Interface-TDC-TCP/UDP*. If you have any further questions, please contact the iba support.

5.1.1 TCP performance problems caused by Delayed Acknowledge

ibaPDA measurements of automation devices using TCP/IP sometimes do not work with cycle times < 200 ms.

Errors shown in ibaPDA

Incomplete telegrams and/or spikes in data values (depending on the sending controller type)

Cause

There are different variants of handling "acknowledge" in the TCP/IP protocol.

The standard WinSocket works in accordance with RFC1122 using the "delayed acknowledge" mechanism (Delayed ACK). It specifies that the "acknowledge" is delayed until other telegrams arrive in order to acknowledge them jointly. If no other telegrams arrive, the ACK telegram is sent after 200 ms at the latest (depending on the socket).

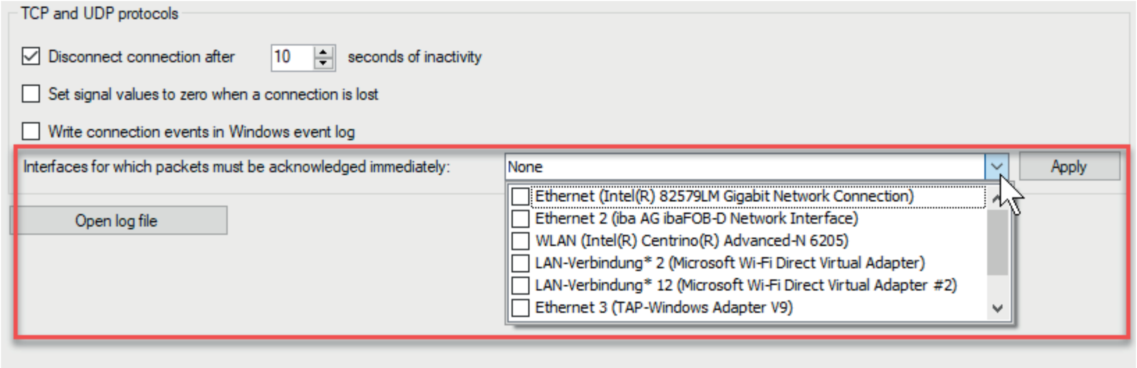
The data flow is controlled by a "sliding window" (parameter Win=nnnn). The recipient specifies how many bytes it can receive without sending an acknowledgment.

Some controllers do not accept this response, but instead, wait for an acknowledgment after each data telegram. If it does not arrive within a certain period of time (200 ms), it will repeat the telegram and include any new data to be sent, causing an error with the recipient, because the previous telegram was received correctly.

Remedy

The "delayed acknowledge" can be switched off individually for each network adapter via an entry in the Windows Registry. For easy modification, *ibaPDA* offers a corresponding dialog in the I/O Manager under *General* in the tab *Settings*.

In the list of network adapters, select those for which you want to disable "delayed acknowledge" and click <Apply>.



☑ Disconnect connection after 10 seconds of inactivity

□ Set signal values to zero when a connection is lost

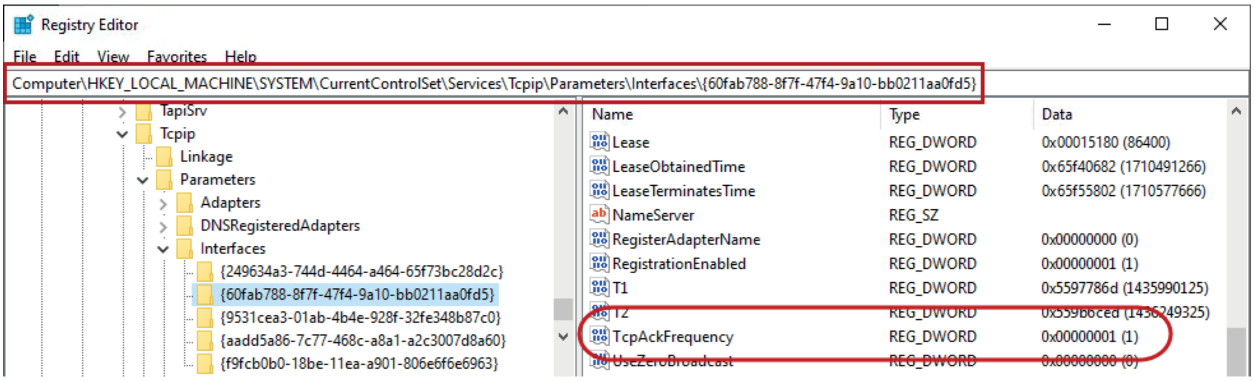
□ Write connection events in Windows event log

Interfaces for which packets must be acknowledged immediately: None

- □ Ethernet (Intel(R) 82579LM Gigabit Network Connection)
- □ Ethernet 2 (iba AG ibaFOB-D Network Interface)
- □ WLAN (Intel(R) Centrino(R) Advanced-N 6205)
- □ LAN-Verbindung* 2 (Microsoft Wi-Fi Direct Virtual Adapter)
- □ LAN-Verbindung* 12 (Microsoft Wi-Fi Direct Virtual Adapter #2)
- □ Ethernet 3 (TAP-Windows Adapter V9)

Thus, the parameter "TcpAckFrequency" (REG_DWORD = 1) is created in the registry path of the selected network adapters:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\
{InterfaceGUID}
```


Registry Editor
File
Edit
View
Favorites
Help
TapiSrv
Tcpip
Linkage
Parameters
Adapters
DNSRegisteredAdapters
Interfaces
{249634a3-744d-4464-a464-65f73bc28d2c}
{60fab788-8f7f-47f4-9a10-bb0211aa0fd5}
{9531cea3-01ab-4b4e-928f-32fe348b87c0}
{aadd5a86-7c77-468c-a8a1-a2c3007d8a60}
{f9fcb0b0-18be-11ea-a901-806e6f6e6963}

Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\{60fab788-8f7f-47f4-9a10-bb0211aa0fd5}

Name	Type	Data
Lease	REG_DWORD	0x00015180 (86400)
LeaseObtainedTime	REG_DWORD	0x65f40682 (1710491266)
LeaseTerminatesTime	REG_DWORD	0x65f55802 (1710577666)
NameServer	REG_SZ	
RegisterAdapterName	REG_DWORD	0x00000000 (0)
RegistrationEnabled	REG_DWORD	0x00000001 (1)
T1	REG_DWORD	0x5597786d (1435990125)
T2	REG_DWORD	0x559b6ced (1436249325)
TcpAckFrequency	REG_DWORD	0x00000001 (1)
UseZeroBroadcast	REG_DWORD	0x00000000 (0)

Note

Basically, you can avoid such TCP-specific problems by using *UDP* instead of *TCP*.

The User Datagram Protocol (UDP) is a minimal network protocol that is not connection-oriented and is unsecured against telegram loss. Among other things, reception acknowledgement of the sent data is dispensed with. In stable and high-performance networks, however, this is not of significant importance and can be neglected due to the cyclic data transmission common with *ibaPDA*.

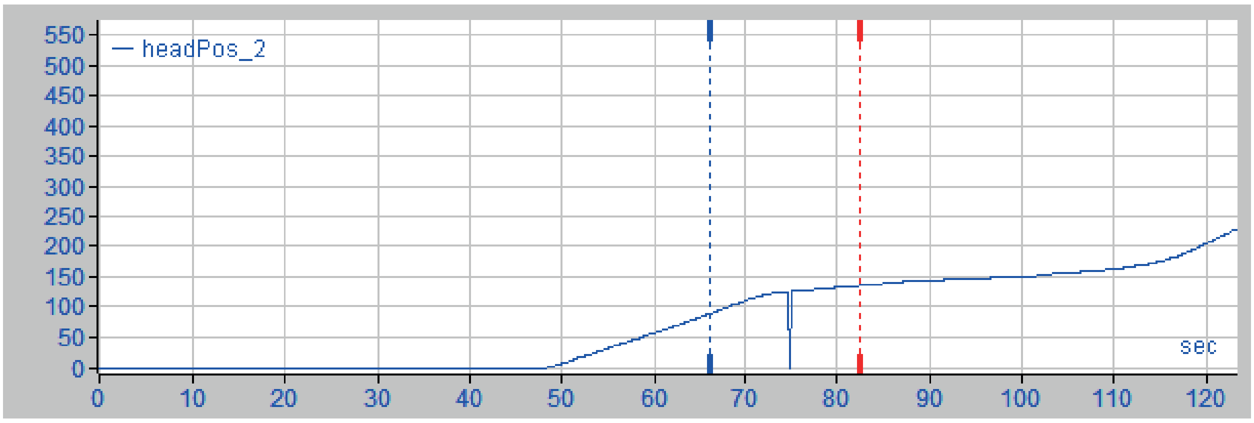
5.1.2 TCP data corruption resulting from the Nagle’s Algorithm

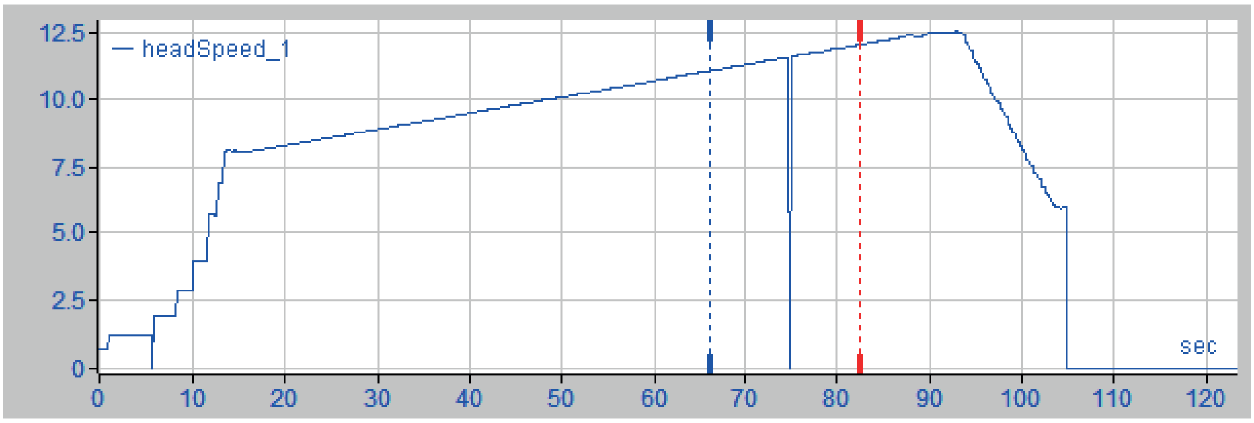
Symptoms

ibaPDA measurements of automation devices using TCP/IP show spikes in the data.

Errors shown in ibaPDA

Incomplete telegrams and/or spikes in the data values (see examples in the following figures)





Cause

Nagle's algorithm is one mechanism for improving TCP efficiency by reducing the number of small packets sent over the network and collecting several data blocks before sending the data over the network.

Because the Generic TCP interface does not use an application level protocol, the receiver *ibaPDA* cannot handle these merged messages correctly. The Generic TCP interface expects only 1 datagram per TCP message with always the same layout and length.

But the Nagle's Algorithm and the option *Delayed ACK* (Delayed Acknowledge, see 5.1.1, page 34) do not play well together in a TCP/IP Network:

The Delayed ACK mechanism tries to send more data per segment if it can. But part of Nagle's algorithm depends on an ACK to send data. So Delayed ACKs are waiting to send the ACK while Nagle's algorithm is waiting to receive the ACK.

This creates random stalls of 200 ms to 500 ms on segments that could otherwise be sent immediately and delivered to the receive-side stack of *ibaPDA* as application.

Remedy

It is recommended to start with disabling the *Delayed ACK* mechanism, see chapter 5.1.1, page 34. In a typical real-time application, the transmitter will then send the new data to *ibaPDA* with a certain cycle time because the previous data has been acknowledged immediately. Depending on the implementation of the TCP/IP stack on the sender's side, the Nagle's algorithm can still become active and automatically aggregate a number of small buffer messages, causing the algorithm to purposely slow down the transmission.

This can also happen sporadically due to a momentary overload on the sender side that causes the stack to merge some messages.

To disable Nagle's buffering algorithm, use the *TCP_NODELAY* socket option. The *TCP_NODELAY* socket option allows the network to bypass Nagle's-induced Delays by disabling Nagle's algorithm, and sending the data as soon as it is available.

Enabling *TCP_NODELAY* forces a socket to send the data in its buffer, whatever the packet size. The *TCP_NODELAY* flag is an option that can be enabled on a per-socket basis and is applied when a TCP socket is created.

(See *Socket.NoDelay* property in .NET applications in the *System.Net.Sockets* namespace.)

Note

Basically, you can avoid such TCP-specific problems by using *UDP* instead of *TCP*.

The User Datagram Protocol (UDP) is a minimal network protocol that is not connection-oriented and is unsecured against telegram loss. Among other things, reception acknowledgement of the sent data is dispensed with. In stable and high-performance networks, however, this is not of significant importance and can be neglected due to the cyclic data transmission common with *ibaPDA*.

5.2 Configuration example SIMATIC TDC

You find the configuration example on the DVD “iba Software & Manuals” under:

`….\04_Libraries_and_Examples\51_ibaPDA-Interface-TDC-TCP_UDP\FixedTelegrams`

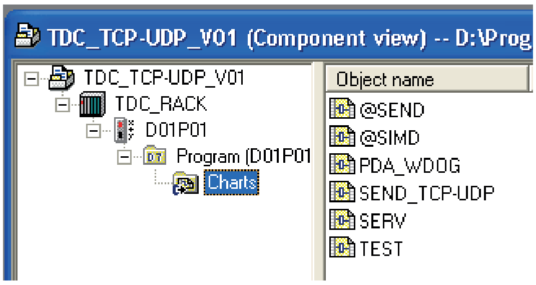
5.2.1 Overview

Example	ibaPDA Project	SIMATIC TDC Project
Project	ibaPDA_TDC_TCP-UDP_Vxx.zip	TDC_TCP-UDP_Vxx.zip

Hardware

Slot	Name	Type	
1	D01P01	CPU551	CPU
15	D1500C	CP51M1	Communication processor

Software



- The demo signals are created in the plan TEST.
- The data is sent to *ibaPDA* in the plan SEND_TCP-UDP.
- The *ibaPDA* watchdog telegram is received in the plan PDA_WDOG.

Communication parameter

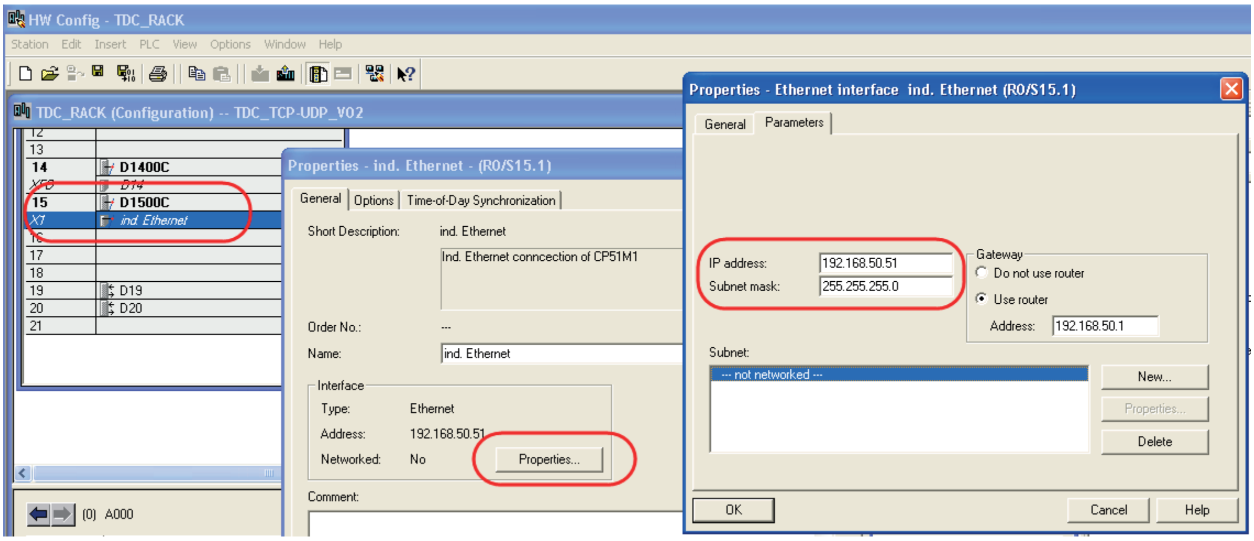
- Remote IP address (*ibaPDA* PC): 192.168.50.203
- Remote Port (ibaPDA): 4171
- Telegram overview:

Local port in TDC	Protocol	ibaPDA Module type	Modul index
30000	TCP/IP	Integer	0
30001	TCP/IP	Real	100
30002	TCP/IP	Generic	200
30003	UDP	Integer	1

Local port in TDC	Protocol	ibaPDA Module type	Modul index
30004	UDP	Real	101
30005	UDP	Generic	201
30006	TCP/IP	Watchdog	-

5.2.2 Ethernet interface of CP51M1 configuration

The connection to *ibaPDA* is executed via the communication processor CP51M1. The IP address and the subnet mask are set in HW Config.



Properties - ind. Ethernet - (R0/S15.1)

Short Description: ind. Ethernet

Ind. Ethernet conncection of CP51M1

Order No.: ---

Name: ind. Ethernet

Interface
Type: Ethernet
Address: 192.168.50.51
Networked: No

Comment:

Properties - Ethernet interface ind. Ethernet (R0/S15.1)

IP address: 192.168.50.51
Subnet mask: 255.255.255.0

Gateway
○ Do not use router
● Use router
Address: 192.168.50.1

Subnet:
--- not networked ---

5.2.3 Telegram configuration

The following send data will be created in the plan "TEST": sine, cosine, triangular signal, one counter per time slice.

The initialization block for the HW modules is placed in the plan @SEND.

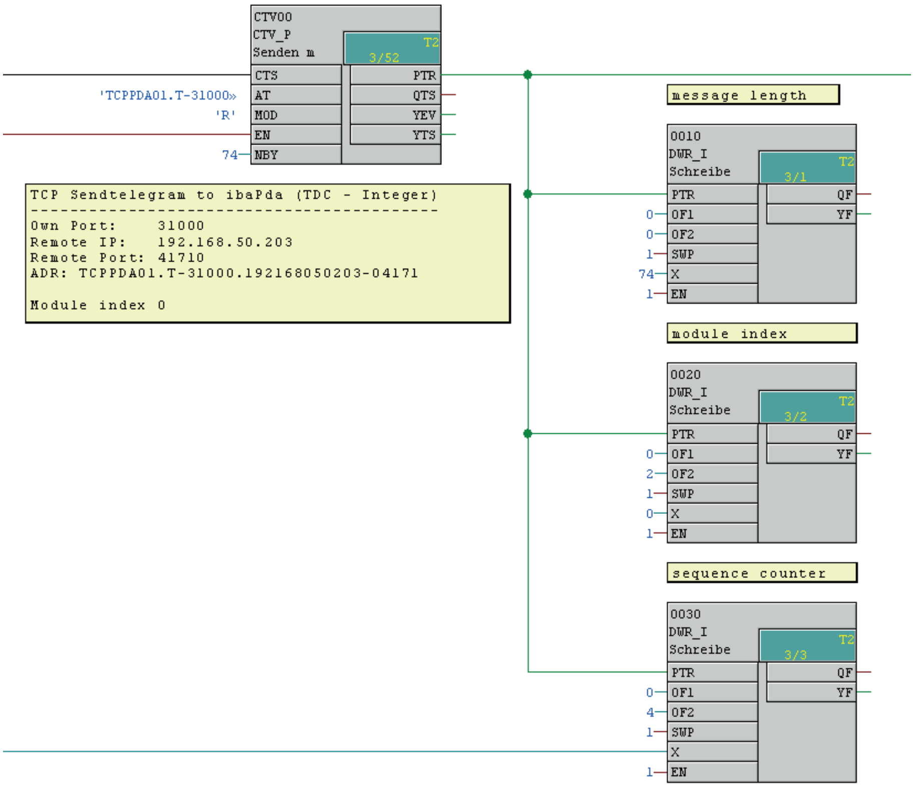
6 telegrams in total are created and sent in the plan SEND_TCP-UDP.

5.2.3.1 TCP telegram with module type Integer

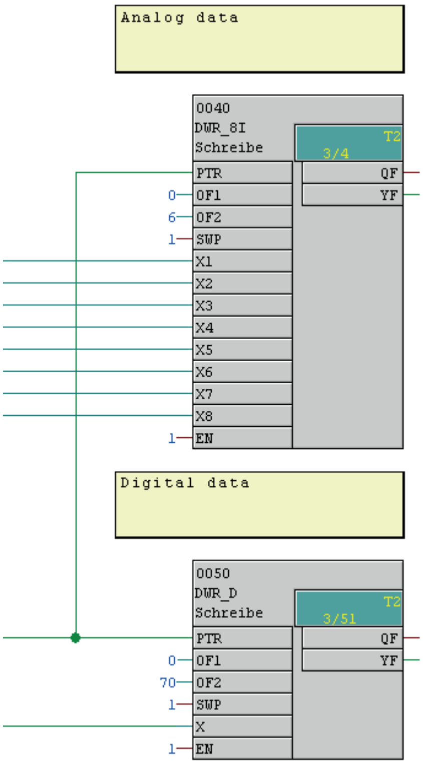
Parametering of the transmitter block and enter the telegram header.

Enter the telegram data:

- 8 analog values type INT from Offset 6
- 32 digital values (1 DWORD) from Offset 70



```
TCP Sendtelegram to ibaPda (TDC - Integer)
------------------------------------------
Own Port:      31000
Remote IP:     192.168.50.203
Remote Port:   41710
ADR: TCPPDA01.T-31000.192168050203-04171

Module index 0
```

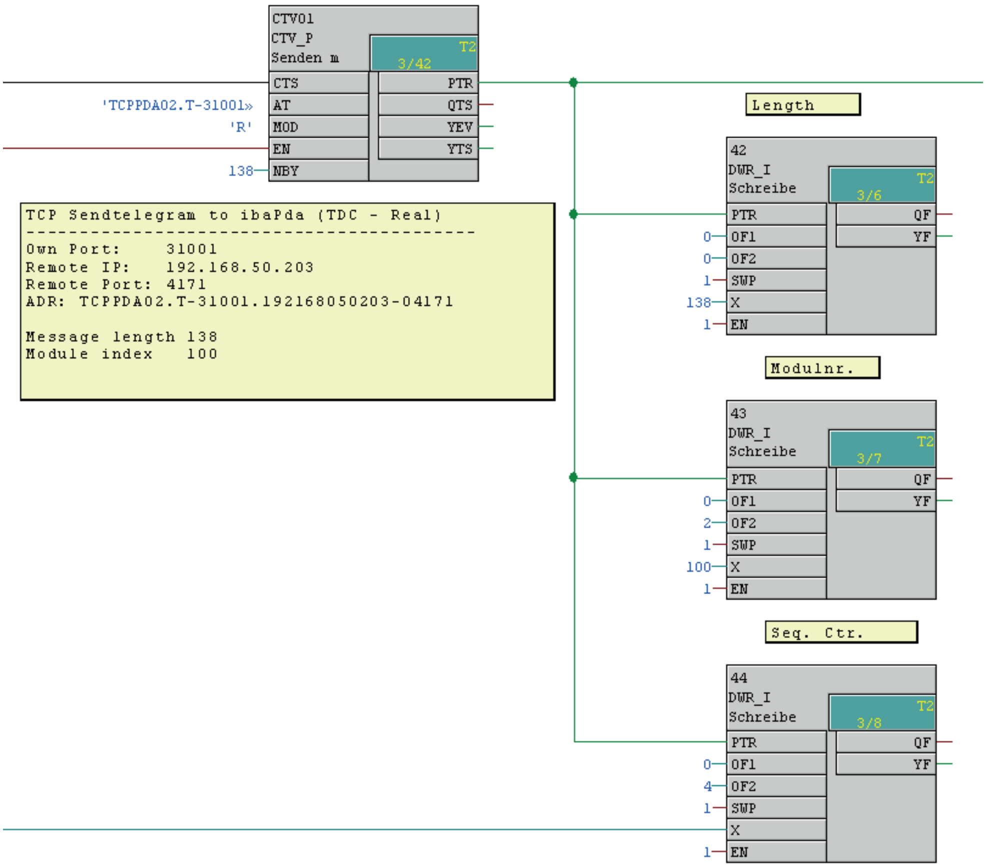


5.2.3.2 TCP telegram with module type 32-Real

Parametering of the transmitter block and enter the telegram header.

Enter the telegram data

- 32 digital signals (1DWORD) from Offset 6
- 8 analog values type FLOAT from Offset 10



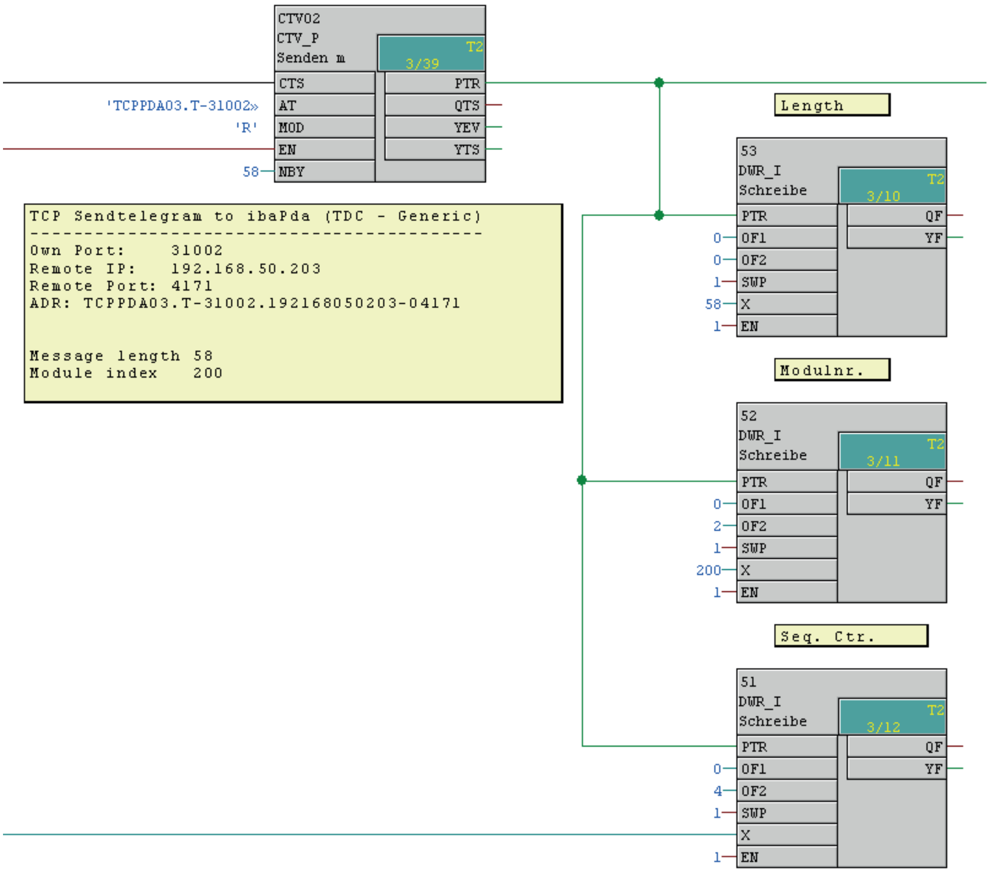
```
TCP Sendtelegram to ibaPda (TDC - Real)
-----------------------------------------
Own Port:     31001
Remote IP:    192.168.50.203
Remote Port: 4171
ADR: TCPPDA02.T-31001.192168050203-04171

Message length 138
Module index   100
```

5.2.3.3 TCP telegram with module type Generic

Parametering of the transmitter block and enter the telegram header.

Enter the telegram data:

- 32 digital values (1DWORD) from Offset 6
- 8 analog values type INT from Offset 10
- 8 analog values type FLOAT from Offset 26



```
TCP Sendtelegram to ibaPda (TDC - Generic)
-----------------------------------------
Own Port:     31002
Remote IP:    192.168.50.203
Remote Port:  4171
ADR: TCPPDA03.T-31002.192168050203-04171


Message length 58
Module index    200
```

5.2.3.4 UDP telegrams

The parametering and sending of the UDP telegrams is identical to the TCP telegrams with the following difference:

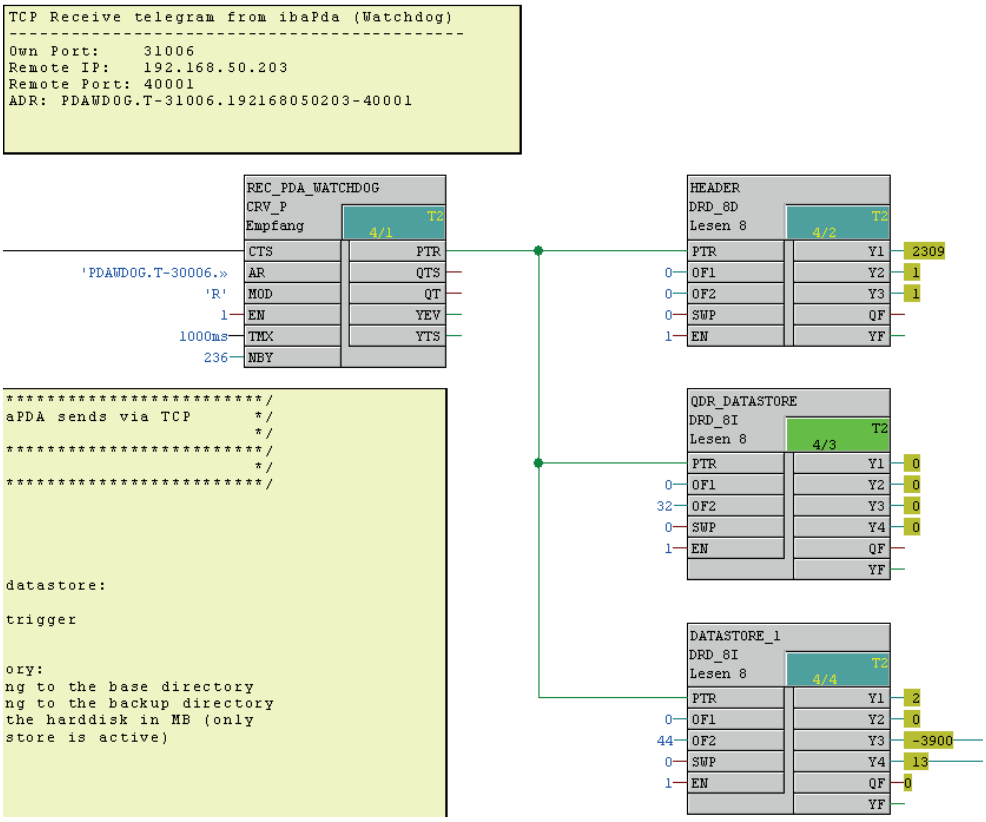
In the address connector AT, the identifier “U” is used instead of “T”.

5.2.3.5 ibaPDA watchdog telegram

SIMATIC TDC is the active partner as well.

From the telegram buffer, the header (offset 0), the QDR_DATASTORE information (Offset 32) and the DATASTORE_1 information (Offset 44) will be read.

If needed further DATASTORE reading blocks can be implemented.



```
TCP Receive telegram from ibaPda (Watchdog)
-------------------------------------------
Own Port:     31006
Remote IP:    192.168.50.203
Remote Port:  40001
ADR:  PDAWDOG.T-31006.192168050203-40001
```

```
*************************/
aPDA sends via TCP      */
                        */
*************************/
                        */
*************************/





datastore:

trigger


ory:
ng to the base directory
ng to the backup directory
the harddisk in MB (only
store is active)
```

Other documentation

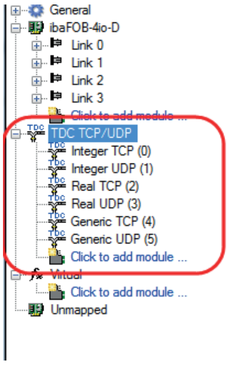
You will find the content and structure of the *ibaPDA* watchdog telegram in the *ibaPDA* manual.

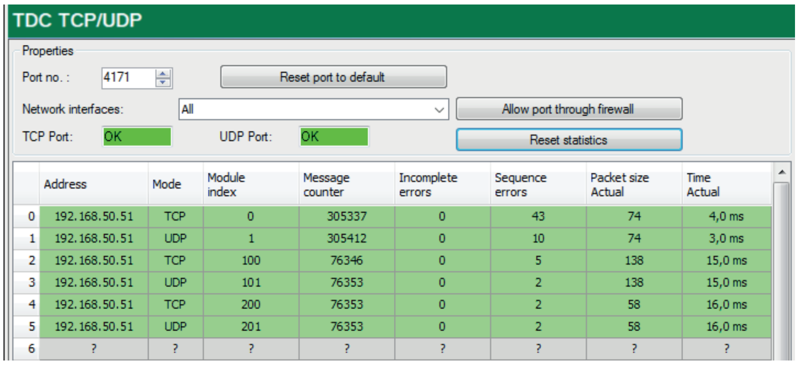
5.3 Configuration example ibaPDA

This *ibaPDA* configuration applies to the above mentioned SIMATIC TDC configuration.

5.3.1 Data telegram configuration

6 modules are available in the I/O manager. The standard port number 4171 is used for all connections:




TDC TCP/UDP

Properties

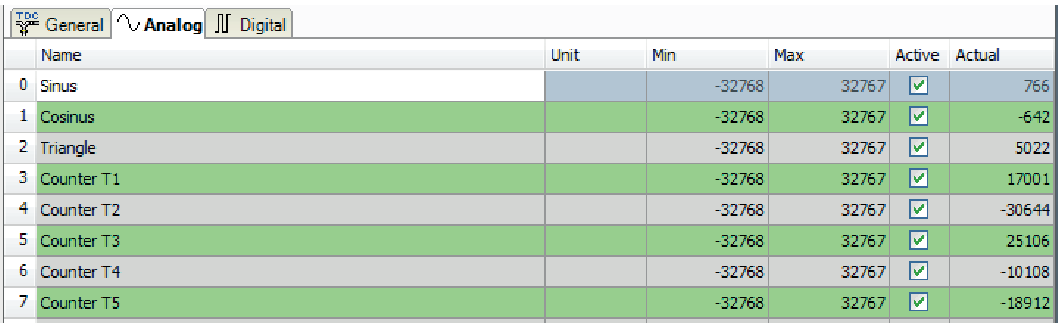
Port no. : 4171

Network interfaces: All

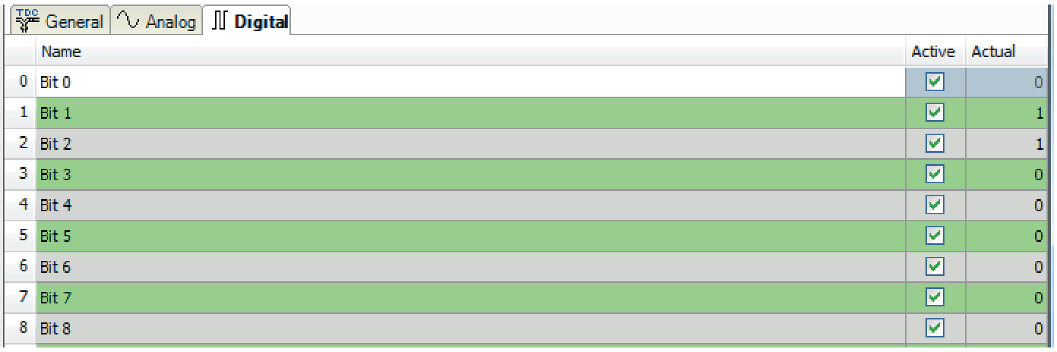
TCP Port: OK UDP Port: OK

	Address	Mode	Module index	Message counter	Incomplete errors	Sequence errors	Packet size Actual	Time Actual
0	192.168.50.51	TCP	0	305337	0	43	74	4,0 ms
1	192.168.50.51	UDP	1	305412	0	10	74	3,0 ms
2	192.168.50.51	TCP	100	76346	0	5	138	15,0 ms
3	192.168.50.51	UDP	101	76353	0	2	138	15,0 ms
4	192.168.50.51	TCP	200	76353	0	2	58	16,0 ms
5	192.168.50.51	UDP	201	76353	0	2	58	16,0 ms
6	?	?	?	?	?	?	?	?

The acquired signals are entered and activated in the tabs *Analog* and *Digital* of the individual modules.



	Name	Unit	Min	Max	Active	Actual
0	Sinus		-32768	32767	☑	766
1	Cosinus		-32768	32767	☑	-642
2	Triangle		-32768	32767	☑	5022
3	Counter T1		-32768	32767	☑	17001
4	Counter T2		-32768	32767	☑	-30644
5	Counter T3		-32768	32767	☑	25106
6	Counter T4		-32768	32767	☑	-10108
7	Counter T5		-32768	32767	☑	-18912

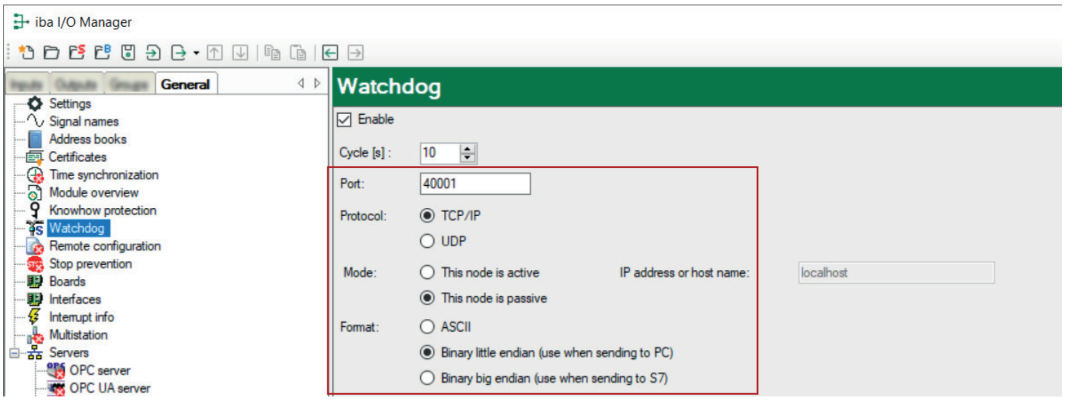


	Name	Active	Actual
0	Bit 0	☑	0
1	Bit 1	☑	1
2	Bit 2	☑	1
3	Bit 3	☑	0
4	Bit 4	☑	0
5	Bit 5	☑	0
6	Bit 6	☑	0
7	Bit 7	☑	0
8	Bit 8	☑	0

5.3.2 Watchdog configuration

In the I/O Manager under *General* - *Watchdog* the following parameters for the watchdog telegram are defined:

Port number: 40001, Protocol: TCP/IP, Mode: passive, Format: Binary little endian



Watchdog

☑ Enable

Cycle [s]: 10

Port: 40001

Protocol: ● TCP/IP ○ UDP

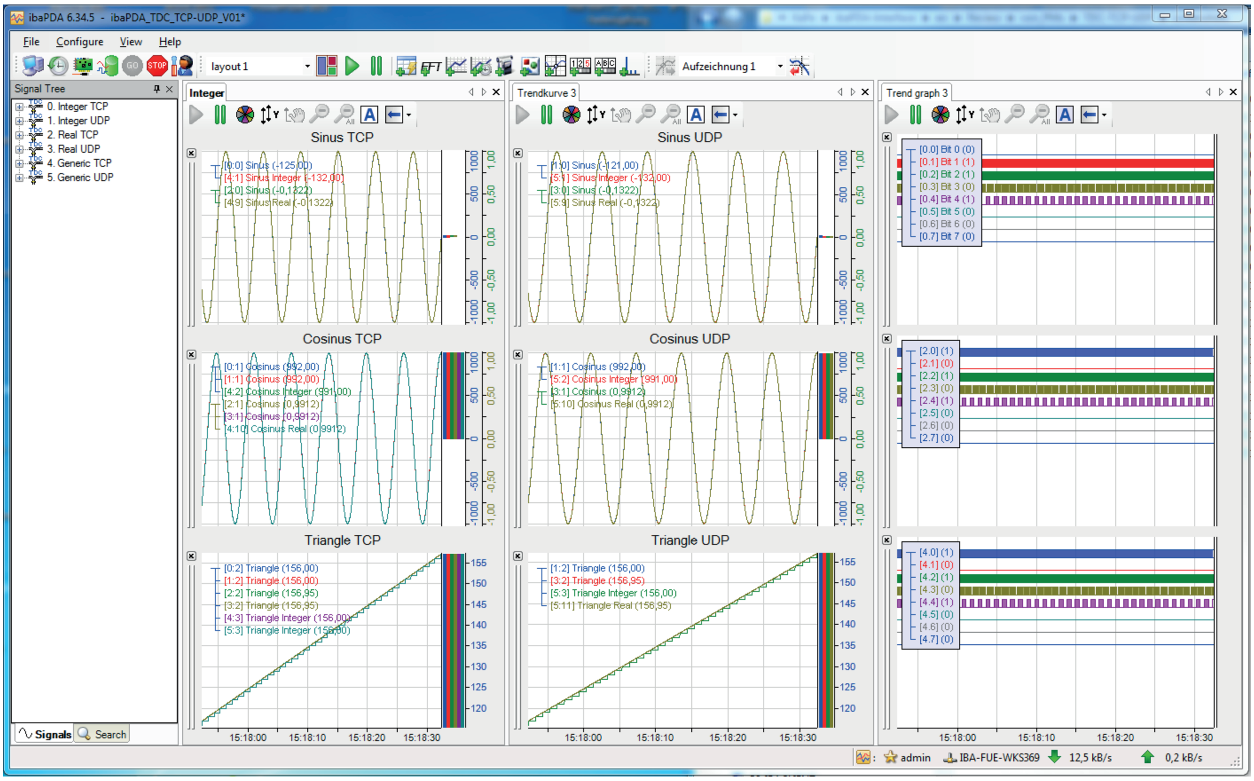
Mode: ○ This node is active ● This node is passive

IP address or host name: localhost

Format: ○ ASCII ● Binary little endian (use when sending to PC) ○ Binary big endian (use when sending to S7)

5.3.3 Online view

Trend curves show the currently transmitted analog and digital values.



6 Support and contact

Support

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

Note

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

Contact

Headquarters

iba AG
Koenigswarterstrasse 44
90762 Fuerth
Germany

Phone:	+49 911 97282-0
Email:	iba@iba-ag.com

Mailing address

iba AG
Postbox 1828
D-90708 Fuerth, Germany

Delivery address

iba AG
Gebhardtstrasse 10
90762 Fuerth, Germany

Regional and Worldwide

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

www.iba-ag.com