



ibaPDA-Interface-TDC-TCP/UDP

Data Interface TCP/UDP to SIMATIC TDC

Manual
Issue 2.3

Measurement Systems for Industry and Energy
www.iba-ag.com

Manufacturer

iba AG
Koenigswarterstr. 44
90762 Fuerth
Germany

Contacts

Main office +49 911 97282-0
Fax +49 911 97282-33
Support +49 911 97282-14
Engineering +49 911 97282-13
E-mail iba@iba-ag.com
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 2022, 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.

Version	Date	Revision - Chapter / Page	Author	Version SW
2.3	03-2022	Troubleshooting, Nagle's Algorithm	ip	7.2.2

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

Content

1	About this Manual	5
1.1	Target group and previous knowledge	5
1.2	Notations	5
1.3	Used symbols.....	6
2	System Requirements.....	7
3	Data Interface TCP/UDP to SIMATIC TDC.....	8
3.1	General Information	8
3.2	SIMATIC TDC Configuration & Engineering.....	9
3.2.1	3.2.1 General Settings.....	9
3.2.2	Data Structures	10
3.2.2.1	Header	10
3.2.2.2	Data Range	10
3.2.3	Connection Configuring.....	12
3.2.4	Sending Data.....	13
3.3	Configuration & Engineering ibaPDA.....	15
3.3.1	General Settings.....	15
3.3.2	General Interface Settings	16
3.3.3	General Module Settings	17
3.3.4	Signal configuration	18
3.3.5	Module Type Integer	19
3.3.6	Module Type Real	19
3.3.7	Module Type Generic	20
3.3.8	Module Diagnostics	21
4	Diagnostics.....	22
4.1	License	22
4.2	Visibility of the interface.....	22
4.3	Log files.....	23
4.4	Connection diagnostics with PING.....	24
4.5	Connection table	25
4.6	Performance	26
4.6.1	TCP telegrams	26

- 4.6.2 UDP telegrams 26
- 5 Appendix 27**
- 5.1 Troubleshooting..... 27
- 5.1.1 TCP performance or data corruption problems resulting from the Delayed ACK mechanism..... 27
- 5.1.2 TCP data corruption resulting from the Nagle’s Algorithm..... 29
- 5.2 Configuration Example SIMATIC TDC..... 31
- 5.2.1 Overview..... 31
- 5.2.2 Ethernet Interface of CP51M1 Configuration 32
- 5.2.3 Telegram Configuration 32
- 5.2.3.1 TCP Telegram with Module Type Integer 32
- 5.2.3.2 TCP Telegram with Module Type 32-Real 34
- 5.2.3.3 TCP Telegram with Module Type Generic..... 35
- 5.2.3.4 UDP Telegrams..... 35
- 5.2.3.5 ibaPDA Watchdog Telegram 36
- 5.3 Configuration Example ibaPDA 37
- 5.3.1 Data Telegram Configuration 37
- 5.3.2 Watchdog Configuration..... 38
- 5.3.3 Online view..... 38
- 6 Support and contact..... 39**



1 About this Manual

This document describes the function and application of the software interface

ibaPDA-Interface-TDC-TCP/UDP

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

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

- Windows operating system
- Knowledge of configuration and operation of the relevant control system

1.2 Notations

In this manual, the following notations are used:

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

1.3 Used symbols

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

Danger!



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

- Observe the specified measures.

Warning!



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

- Observe the specified measures.

Caution!



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

- Observe the specified measures

Note



A note specifies special requirements or actions to be observed.

Tip



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

Other documentation



Reference to additional documentation or further reading.

2 System Requirements

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

- *ibaPDA* v7.0.0 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 or data corruption problems resulting from the Delayed ACK mechanism, page 27](#) (all *ibaPDA* versions).

Licenses

Order no.	Product designation	Description
31.001056	ibaPDA-Interface-TDC-TCP/UDP	Extension license for an <i>ibaPDA</i> 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 <i>ibaPDA-Interface-TDC-TCP/UDP</i> 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: any data blocks 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, we recommend the use of the interface *ibaPDA-Interface-Sisteam-TCPIP* (License number 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 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. Since 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, since 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:

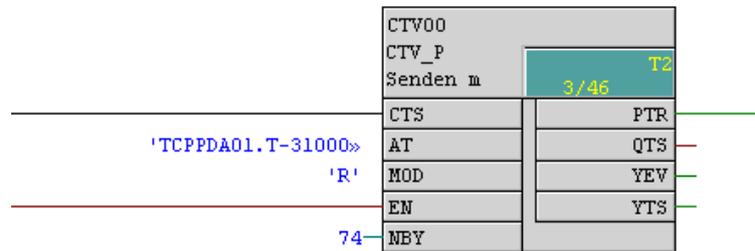
SINT, BYTE, INT, WORD, DINT, DWORD, FLOAT, DOUBLE, STRING[32]

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 Connection Configuring

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:

'aaaaaaa.b-cccc.ddddddddddd-eeee' with

aaaaaa: unique channel name, max. 8 characters

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

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

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

eeee: 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

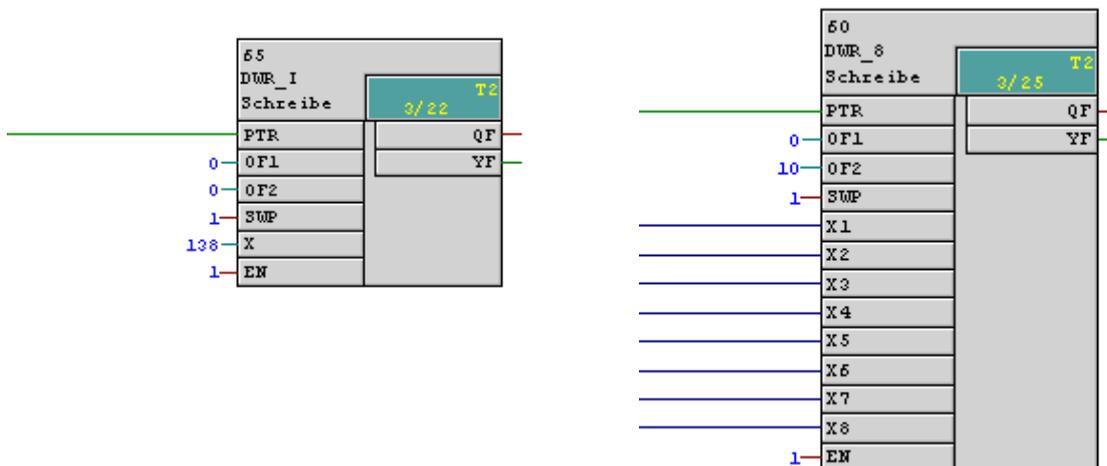
The data to be transmitted are written with the block DWR in the telegram buffer There are several variants of this block for the data types BYTE, INT and DINT and the number of signals.

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.



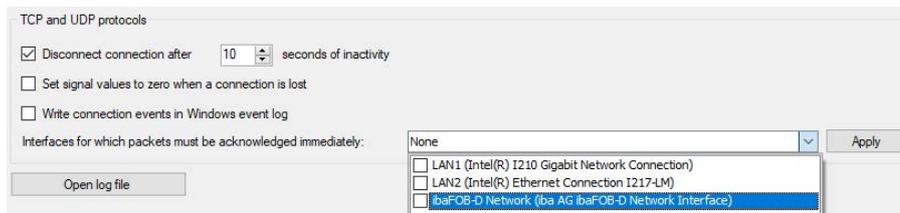
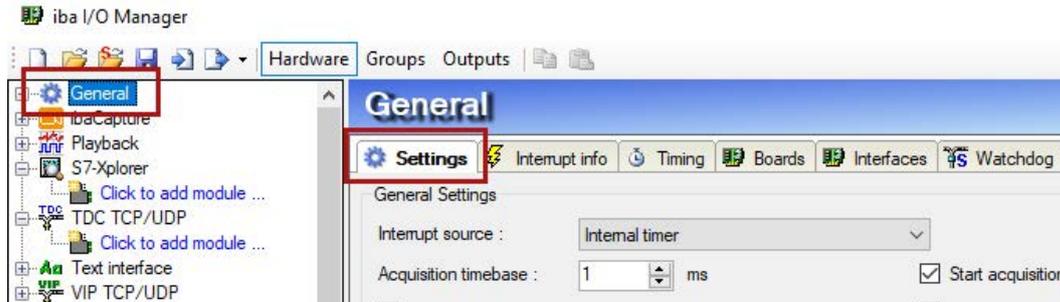
Connection	Meaning	Meaning
PTR	Buffer indicator	Connection to PTR of the transmitter block CTV_P
OF1	Offset 1	The offset in the telegram buffer is formed from OF1 + OF2
OF2	Offset 2	
SWP	SWAP	Byte rotation. For <i>ibaPDA</i> telegrams it must be set on 1
Xn	Data transmitted	
EN	Enable	
QF	Block status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

3.3 Configuration & Engineering ibaPDA

Subsequently, the engineering for *ibaPDA* is described. If all system requirements are met, the interface „TDC TCP/UDP" is displayed in the signal tree.

3.3.1 General Settings

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



Disconnect connection after x 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

Interfaces for which packets must be acknowledged immediately

Selection of required interfaces

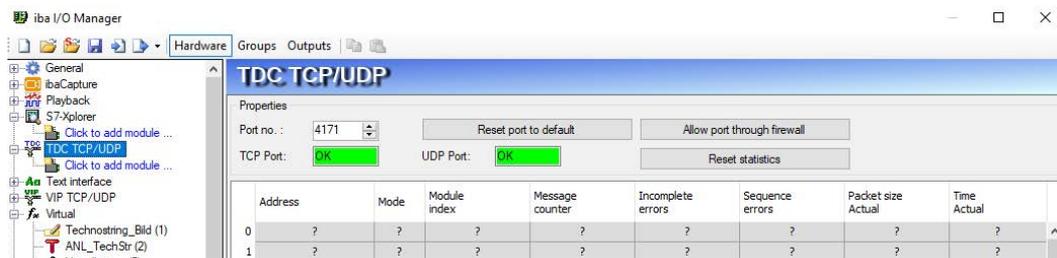
Note



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 provides the following functions and configuration options:



Port

Used port PC side. The port number has to be used identically in the SIMATIC TDC connection configuration (see [SIMATIC TDC Configuration & Engineering](#), page 9).

Reset port to default

The port number 4171 will be set.

Allow port through firewall

During the installation of *ibaPDA* the standard port numbers of the protocols used will be filled in automatically in the firewall. When the port number is changed here or when the interface was activated subsequently, it is necessary to allow this port in the firewall.

TCP Port / UDP Port

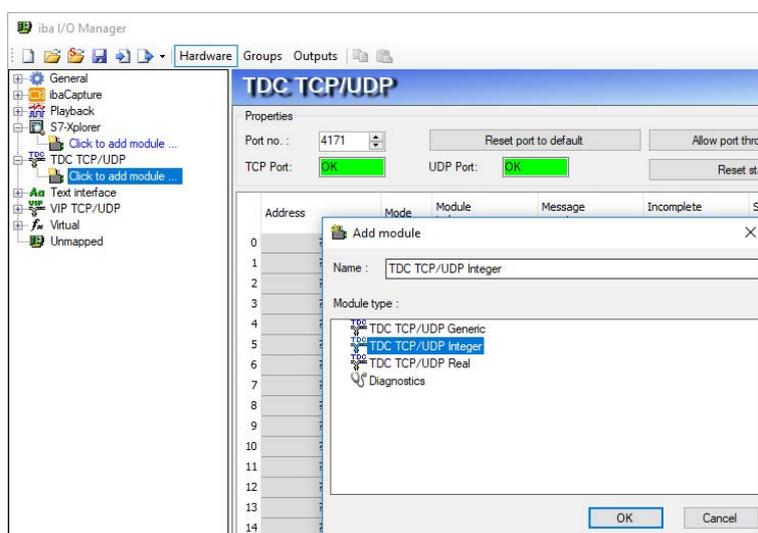
Here the notification OK is shown when the socket on the port can be opened. ERROR is displayed if conflicts occur, e. g. if the port is already occupied.

Connection table

see [Connection table](#), page 25

Adding a module

To add a module, click below the interface and select the desired module type.



Tip

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.

3.3.3 General Module Settings

If you want to configure a module, mark the module in the tree structure and configure the following settings in the dialog:

TDC TCP/UDP Generic (59)	
General	
Basic	
Module Type	TDC TCP/UDP Generic
Locked	False
Enabled	True
Name	TDC TCP/UDP Generic
Module No.	59
Timebase	1 ms
Use name as prefix	False
Text encoding	Default system locale
Advanced	
Swap analog signals	Depending on datatype
Swap digital signals	True
Module Layout	
No. analog signals	32
No. digital signals	32
TCP/IP	
Module Index	201

Basic settings

Module Type (information only)

Indicates the type of the current module.

Locked

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

Enabled

Disabled modules are excluded from signal acquisition.

Name

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

Module No.

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

Timebase

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

Use name as prefix

Puts the module name in front of the signal names.

Text encoding

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

Advanced**Swap analog signals**

Option to change the order of the byte evaluation.

Swap digital signals

Option to change the order of the byte evaluation.

Module Layout**No. analog/digital signals**

You can set up the number of analog and digital signals to be measured. The maximum number of signals is 1000 (analog/digital signals).

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 10

3.3.4 Signal configuration

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

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

Integer TCP (0)						
TDC General Analog Digital						
Name	Unit	Min	Max	Active	Actual	
0 Sinus		-32768	32767	<input checked="" type="checkbox"/>	-321	
1 Cosinus		-32768	32767	<input checked="" type="checkbox"/>	946	
2 Triangle		-32768	32767	<input checked="" type="checkbox"/>	2827	
3 Counter T1		-32768	32767	<input checked="" type="checkbox"/>	9057	
4 Counter T2		-32768	32767	<input checked="" type="checkbox"/>	25707	
5 Counter T3		-32768	32767	<input checked="" type="checkbox"/>	22810	
6 Counter T4		-32768	32767	<input checked="" type="checkbox"/>	22086	
7 Counter T5		-32768	32767	<input checked="" type="checkbox"/>	5521	
8		-32768	32767	<input type="checkbox"/>	0	

Integer TCP (0)			
TDC General Analog Digital			
Name	Active	Actual	
0 Bit 0	<input checked="" type="checkbox"/>	0	
1 Bit 1	<input checked="" type="checkbox"/>	0	
2 Bit 2	<input checked="" type="checkbox"/>	1	
3 Bit 3	<input checked="" type="checkbox"/>	1	
4 Bit 4	<input checked="" type="checkbox"/>	0	
5 Bit 5	<input checked="" type="checkbox"/>	0	
6 Bit 6	<input checked="" type="checkbox"/>	0	
7 Bit 7	<input checked="" type="checkbox"/>	0	
8 Bit 8	<input checked="" type="checkbox"/>	0	

Tip



You can use the “automatically filling in” in the column (see *ibaPDA* manual or online help).

Other documentation



For a detailed description of the settings, please refer to *ibaPDA* manual, part 2.

3.3.5 Module Type Integer

The 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.

3.3.6 Module Type Real

The 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).

3.3.7 Module Type Generic

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

The following module settings are module-specific:

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].

Generic TCP (4)								
TDC General Analog Digital								
Name	Unit	Gain	Offset	Address	DataType	Active	Actual	
0 Digitals 0-31			1	0	0 DINT	<input checked="" type="checkbox"/>	196633	
1 Sinus Integer			1	0	4 INT	<input checked="" type="checkbox"/>	745	
2 Cosinus Integer			1	0	6 INT	<input checked="" type="checkbox"/>	-667	
3 Triangle Integer			1	0	8 INT	<input checked="" type="checkbox"/>	2917	
4 Counter Integer T1			1	0	10 INT	<input checked="" type="checkbox"/>	-31422	
5 Counter Integer T2			1	0	12 INT	<input checked="" type="checkbox"/>	-28503	
6 Counter Integer T3			1	0	14 INT	<input checked="" type="checkbox"/>	25641	
7 Counter Integer T4			1	0	16 INT	<input checked="" type="checkbox"/>	22794	
8 Counter Integer T5			1	0	18 INT	<input checked="" type="checkbox"/>	5698	
9 Sinus Real			1	0	20 FLOAT	<input checked="" type="checkbox"/>	0,745044	
10 Cosinus Real			1	0	24 FLOAT	<input checked="" type="checkbox"/>	-0,667015	
11 Triangle Real			1	0	28 FLOAT	<input checked="" type="checkbox"/>	2917,7	
12 Counter Real T1			1	0	32 FLOAT	<input checked="" type="checkbox"/>	2,9177e+06	
13 Counter Real T2			1	0	36 FLOAT	<input checked="" type="checkbox"/>	364713	
14 Counter Real T3			1	0	40 FLOAT	<input checked="" type="checkbox"/>	91177	
15 Counter Real T4			1	0	44 FLOAT	<input checked="" type="checkbox"/>	22794	
16 Counter Real T5			1	0	48 FLOAT	<input checked="" type="checkbox"/>	5698	

Note



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

3.3.8 Module Diagnostics

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

Generic TCP (4)								
TDC General  Analog  Digital								
	Name	Unit	Gain	Offset	Address	Data Type	Active	Actual
0	Digitals 0-31		1	0	0	DINT	<input checked="" type="checkbox"/>	196633
1	Sinus Integer		1	0	4	INT	<input checked="" type="checkbox"/>	745
2	Cosinus Integer		1	0	6	INT	<input checked="" type="checkbox"/>	-667
3	Triangle Integer		1	0	8	INT	<input checked="" type="checkbox"/>	2917
4	Counter Integer T1		1	0	10	INT	<input checked="" type="checkbox"/>	-31422
5	Counter Integer T2		1	0	12	INT	<input checked="" type="checkbox"/>	-28503
6	Counter Integer T3		1	0	14	INT	<input checked="" type="checkbox"/>	25641
7	Counter Integer T4		1	0	16	INT	<input checked="" type="checkbox"/>	22794
8	Counter Integer T5		1	0	18	INT	<input checked="" type="checkbox"/>	5698
9	Sinus Real		1	0	20	FLOAT	<input checked="" type="checkbox"/>	0,745044
10	Cosinus Real		1	0	24	FLOAT	<input checked="" type="checkbox"/>	-0,667015
11	Triangle Real		1	0	28	FLOAT	<input checked="" type="checkbox"/>	2917,7

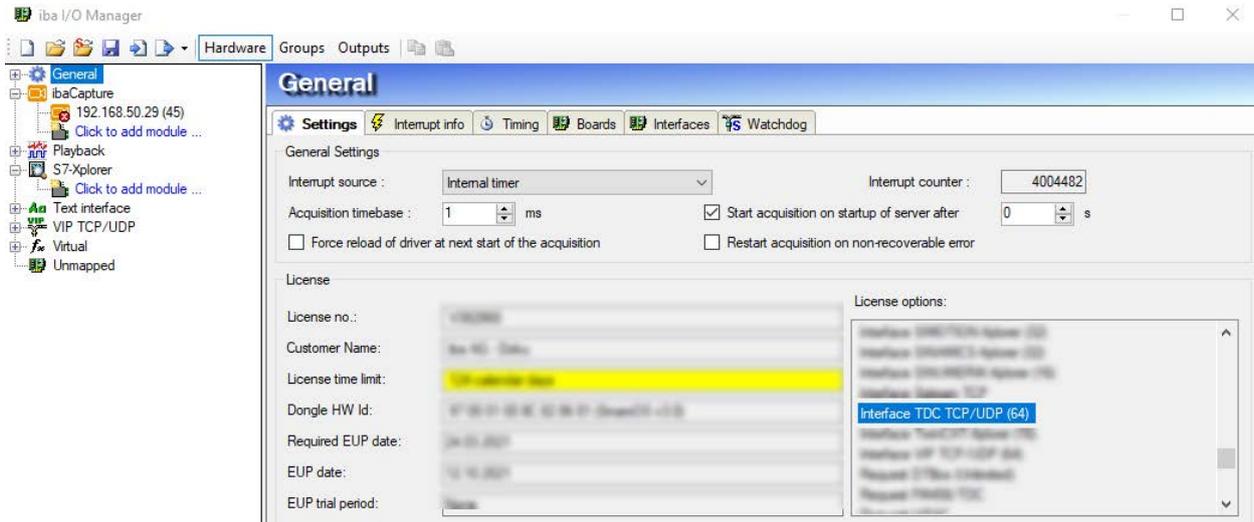
The following errors may occur:

- No data is shown:
 - 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)
 - Byte order is set incorrectly, see [General Module Settings](#), page 17 and [Sending Data](#), page 13
 - There are several modules with the same module index.
- The digital signals are sorted incorrectly:
 - Byte order is set incorrectly, see [General Module Settings](#), page 17 and [Sending Data](#), page 13
- The telegrams arrive not faster than ca. 200 ms with sequence error:
 - Problem with "Delayed Acknowledge", see [TCP performance or data corruption problems resulting from the Delayed ACK mechanism](#), page 27
 - Problem caused by "Nagle's Algorithm", see [TCP data corruption resulting from the Nagle's Algorithm](#), page 29

4 Diagnostics

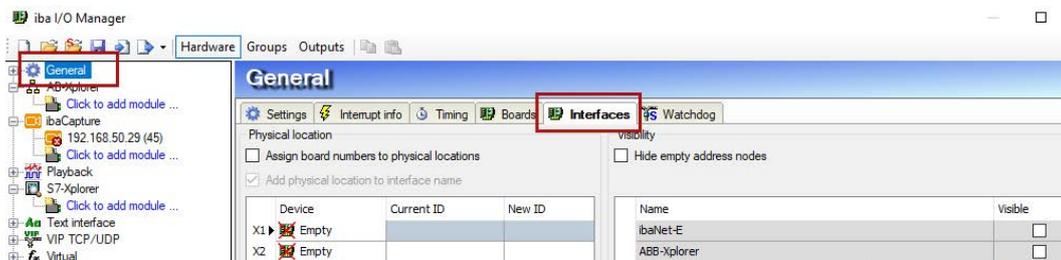
4.1 License

If the interface „TDC TCP/UDP" is not shown in the signal tree, you can check in the I/O manager under *General - Settings - License* or in the *ibaPDA* service status application whether your license has been correctly detected. The number of licensed connections is indicated in brackets.



4.2 Visibility of the interface

If the interface is not visible despite a valid license, it might be hidden. Click on the tab *Interfaces* and activate the interface “TDC TCP/UDP”.



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.

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

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

In the file system on the hard drive, you will find the log files in the program path of the *ibaPDA* server (...\\Programs\\iba\\ibaPDA\\Server\\Log\\). The file names of the log files include the name or abbreviation of the interface type.

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

Examples:

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

4.4 Connection diagnostics with PING

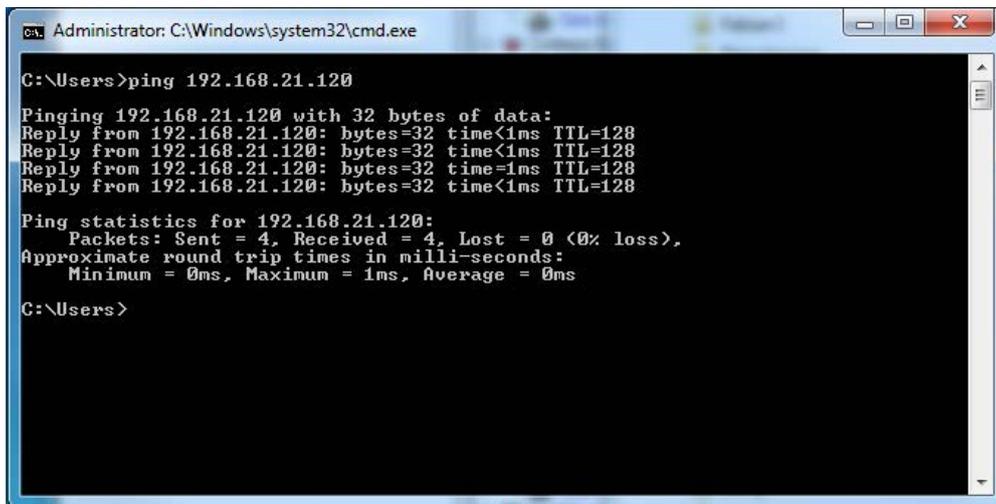
PING is a system command with which you can check if a certain communication partner can be reached in an IP network.

Open a Windows command prompt.



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

With an existing connection you receive several replies.

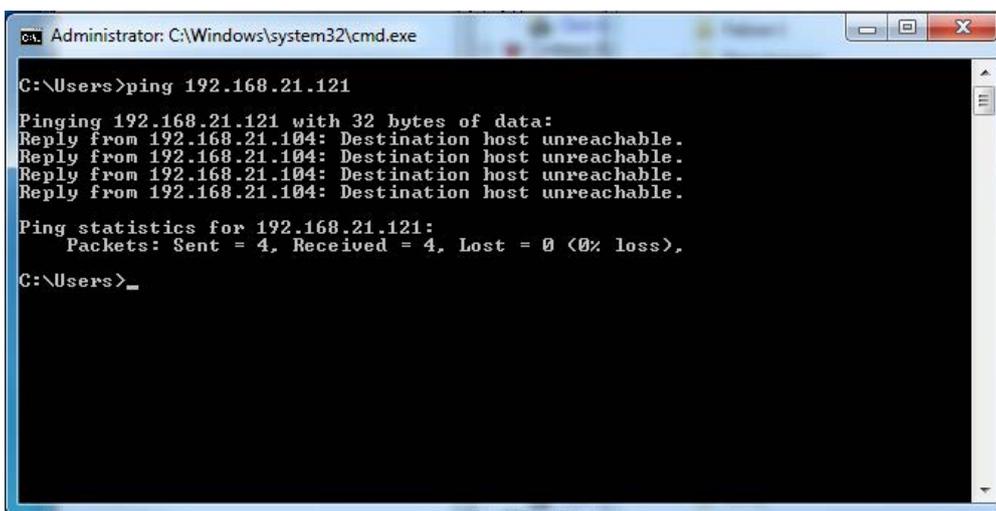
A screenshot of a Windows command prompt window titled 'Administrator: C:\Windows\system32\cmd.exe'. The user has entered the command 'ping 192.168.21.120'. The output shows four successful replies from 192.168.21.120, each with 32 bytes of data, a time of less than 1ms, and a TTL of 128. Ping statistics for 192.168.21.120 are shown as: Packets: Sent = 4, Received = 4, Lost = 0 (0% loss), and approximate round trip times in milliseconds: Minimum = 0ms, Maximum = 1ms, Average = 0ms.

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

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

C:\Users>
```

With no existing connection you receive error messages.

A screenshot of a Windows command prompt window titled 'Administrator: C:\Windows\system32\cmd.exe'. The user has entered the command 'ping 192.168.21.121'. The output shows four failed replies from 192.168.21.104, each with the message 'Destination host unreachable'. Ping statistics for 192.168.21.121 are shown as: Packets: Sent = 4, Received = 4, Lost = 0 (0% loss).

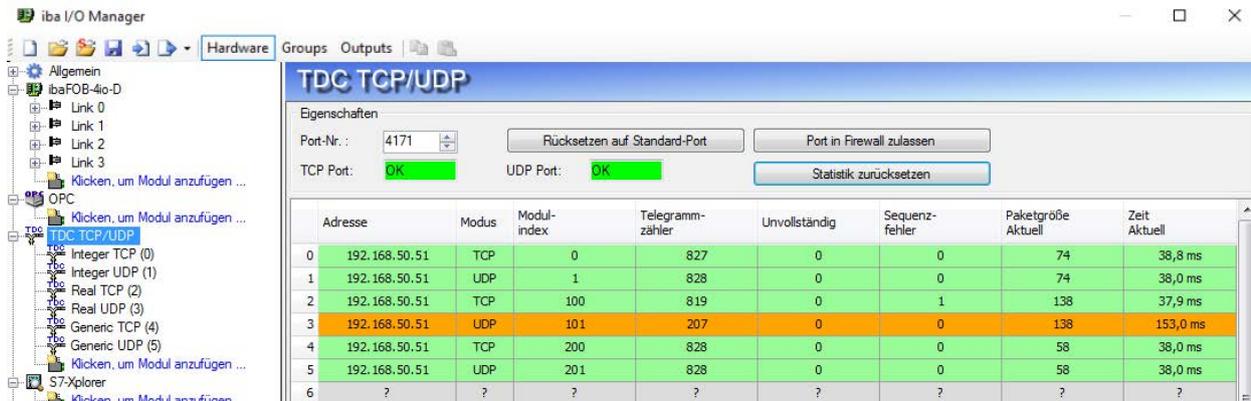
```
Administrator: C:\Windows\system32\cmd.exe
C:\Users>ping 192.168.21.121
Pinging 192.168.21.121 with 32 bytes of data:
Reply from 192.168.21.104: Destination host unreachable.

Ping statistics for 192.168.21.121:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

C:\Users>_
```

4.5 Connection table

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



The background color of the lines has the following meaning:

Color	Meaning
Green	The connection is OK. The <i>ibaPDA</i> module timebase is the same or slower than the telegram cycle. The current telegram cycle is indicated in the column "Time Actual".
Orange	The connection is OK, but the telegram cycle is significantly slower than the <i>ibaPDA</i> module timebase. It is recommended to adjust the module timebase to the telegram cycle.

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

- SIMATIC TDC is in stop mode
- No Ethernet connection between *ibaPDA PC* and SIMATIC CP51M1
- Error in configuration:
 - local port number is not unique
 - false remote IP address
 - configured port number does not comply with *ibaPDA* port.
 - port number is blocked by 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 Acknowledge” problem occurs (see ↗ *TCP performance or data corruption problems resulting from the Delayed ACK mechanism, page 27*)

4.6 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.

4.6.1 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

4.6.2 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

5.1.1 TCP performance or data corruption problems resulting from the Delayed ACK mechanism

Symptoms:

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).

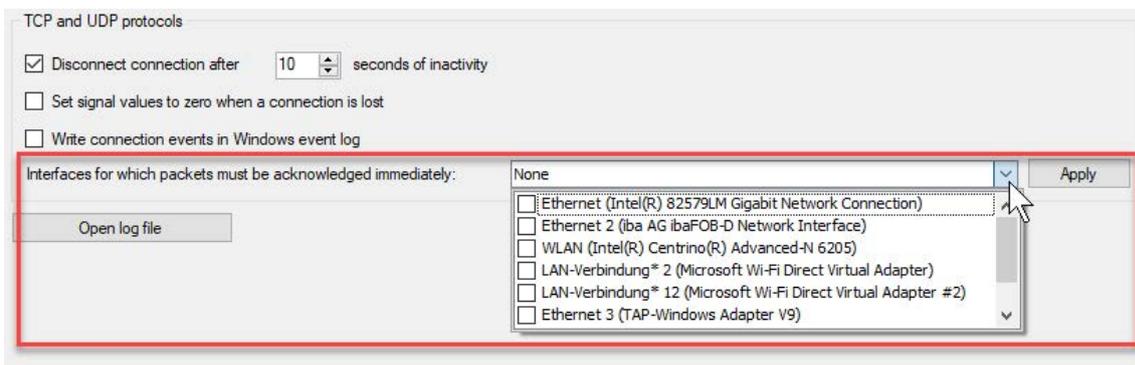
The data flow is controlled by a "sliding window" (parameter Win=nnnn). The recipient specifies how many bytes it can receive without sending an acknowledgment.

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 old one was received correctly.

Remedy:

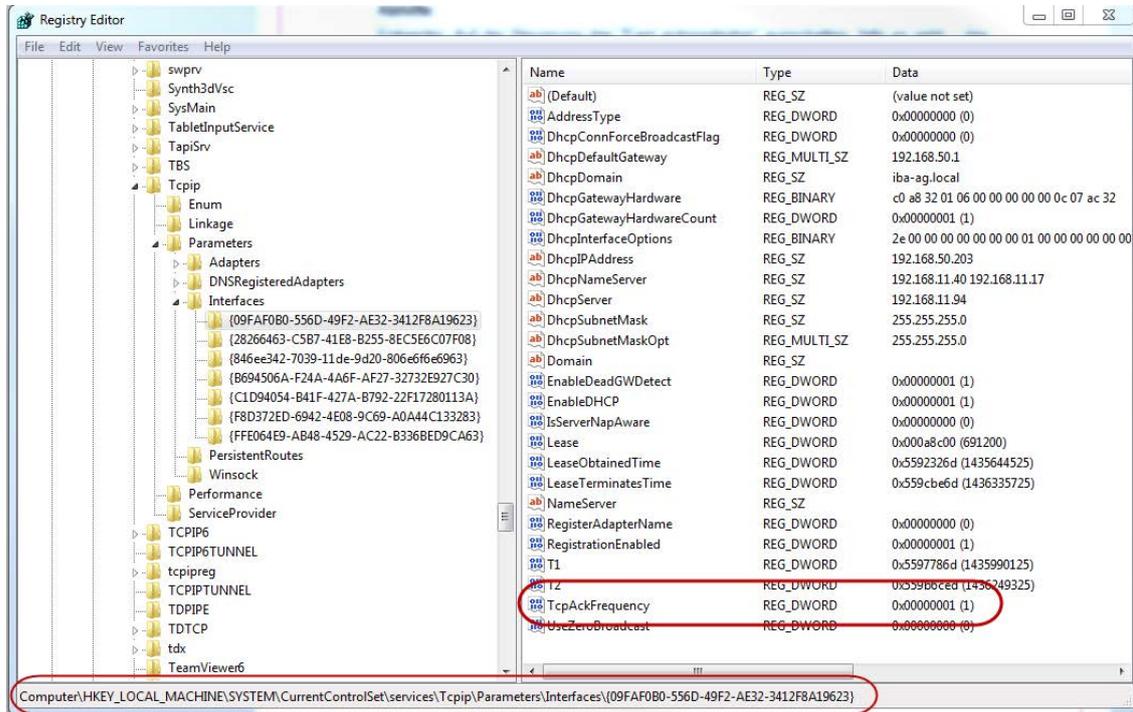
The "delayed acknowledge" can be switched off individually for each network adapter via an entry in the Windows Registry. For easy modification, *ibaPDA* offers a corresponding dialog in the I/O manager under *General* in the tab *Settings*.

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



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}



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*.

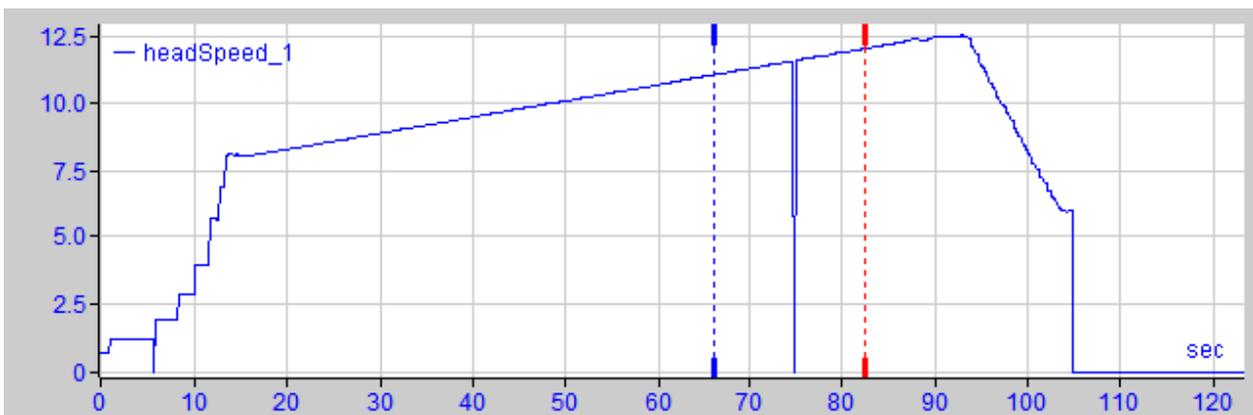
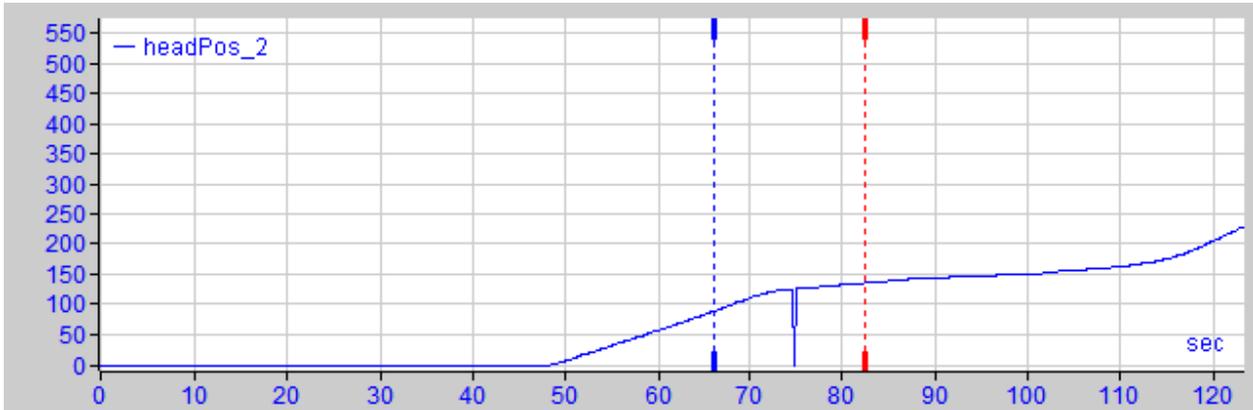
5.1.2 TCP data corruption resulting from the Nagle's Algorithm

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, named after its creator John Nagle, 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.

Since 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 27) 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.

Nagle's algorithm and Delayed ACKs together create a problem because Delayed ACKs are waiting around to send the ACK while "Nagle's" is waiting around to receive the ACK!

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

Remedy:

We recommend starting with disabling the *Delayed ACK* mechanism as explained in chapter 5.1.1, page 27. In a typical real-time application, the transmitter will then send the new data to *ibaPDA* with a certain cycle time, since 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

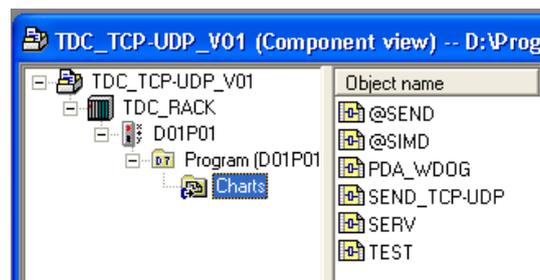
5.2.1 Overview

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.

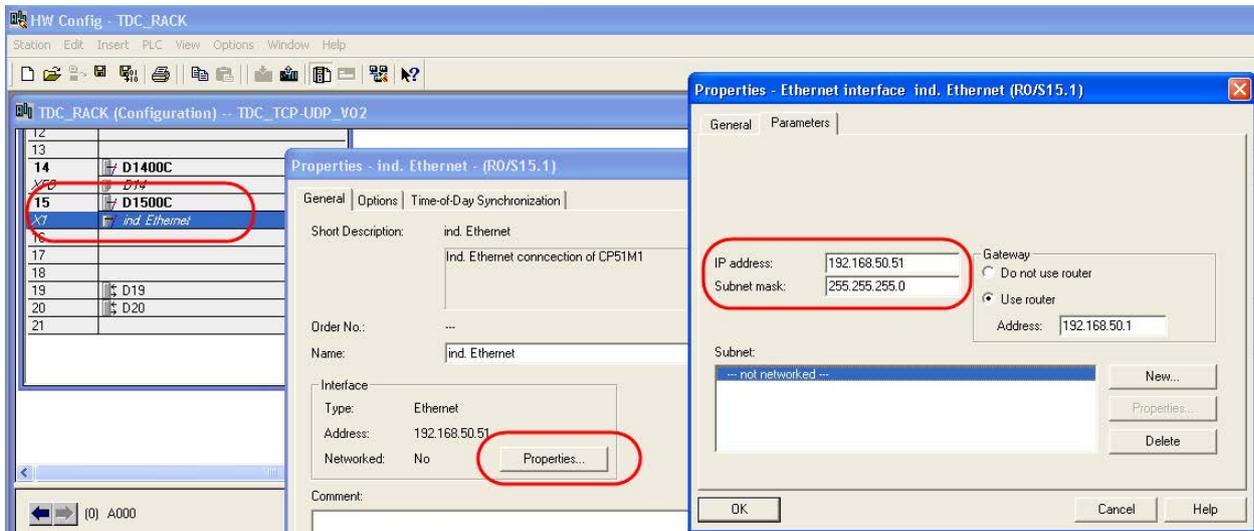
Communication parameter

- 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
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.



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.

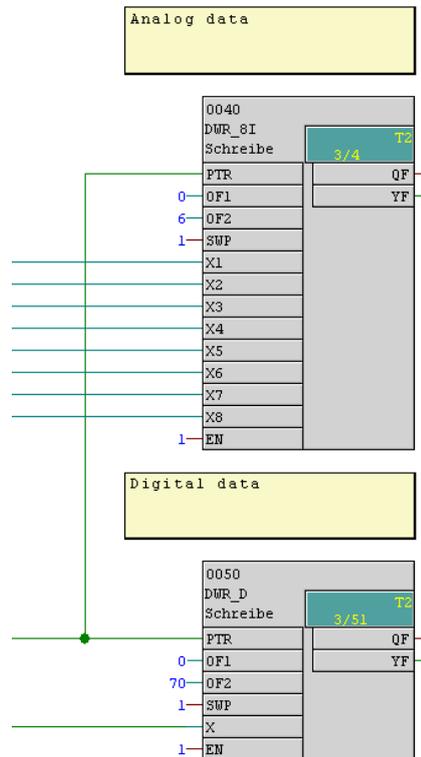
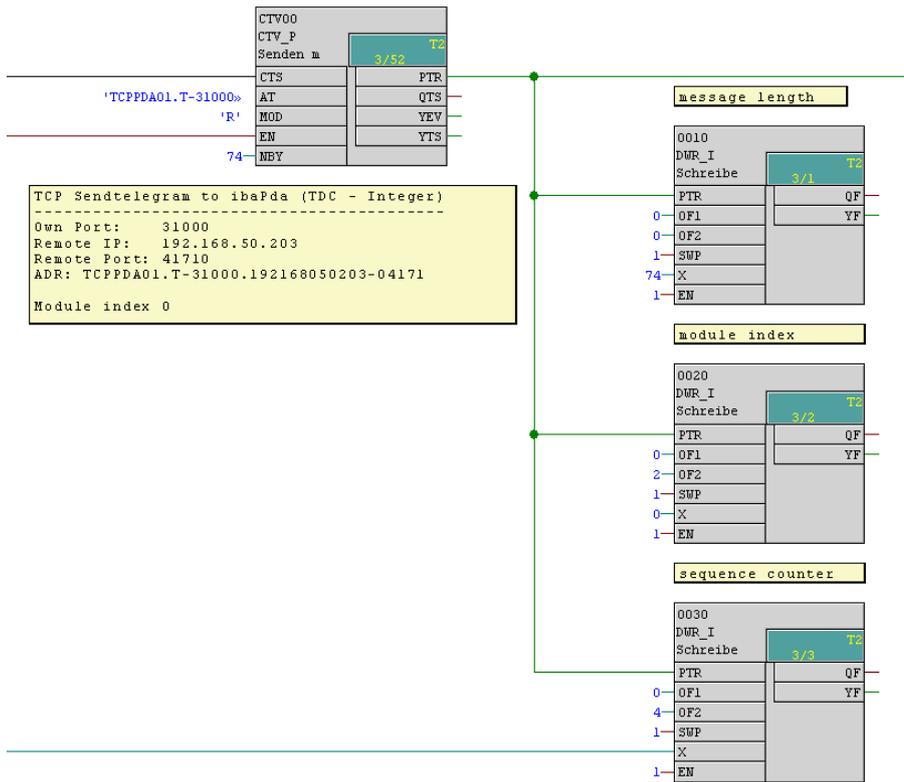
6 telegrams in total are created and sent in the plan SEND_TCP-UDP.

5.2.3.1 TCP Telegram with Module Type Integer

Parameterizing 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

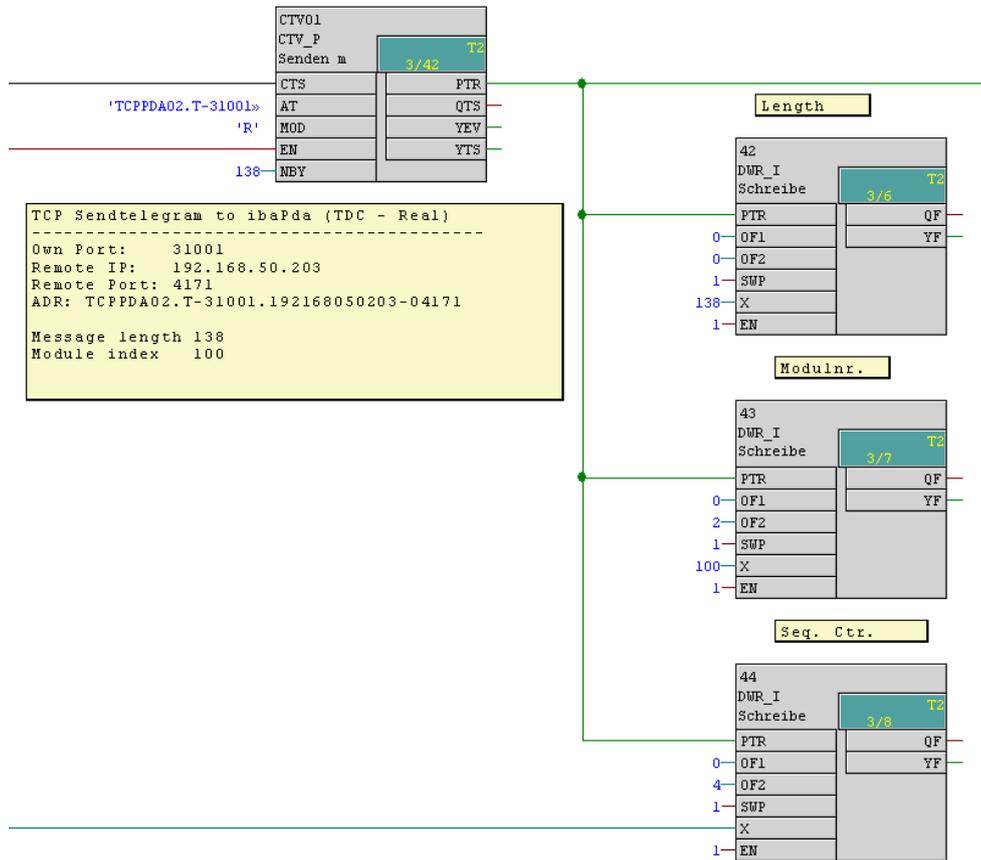


5.2.3.2 TCP Telegram with Module Type 32-Real

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

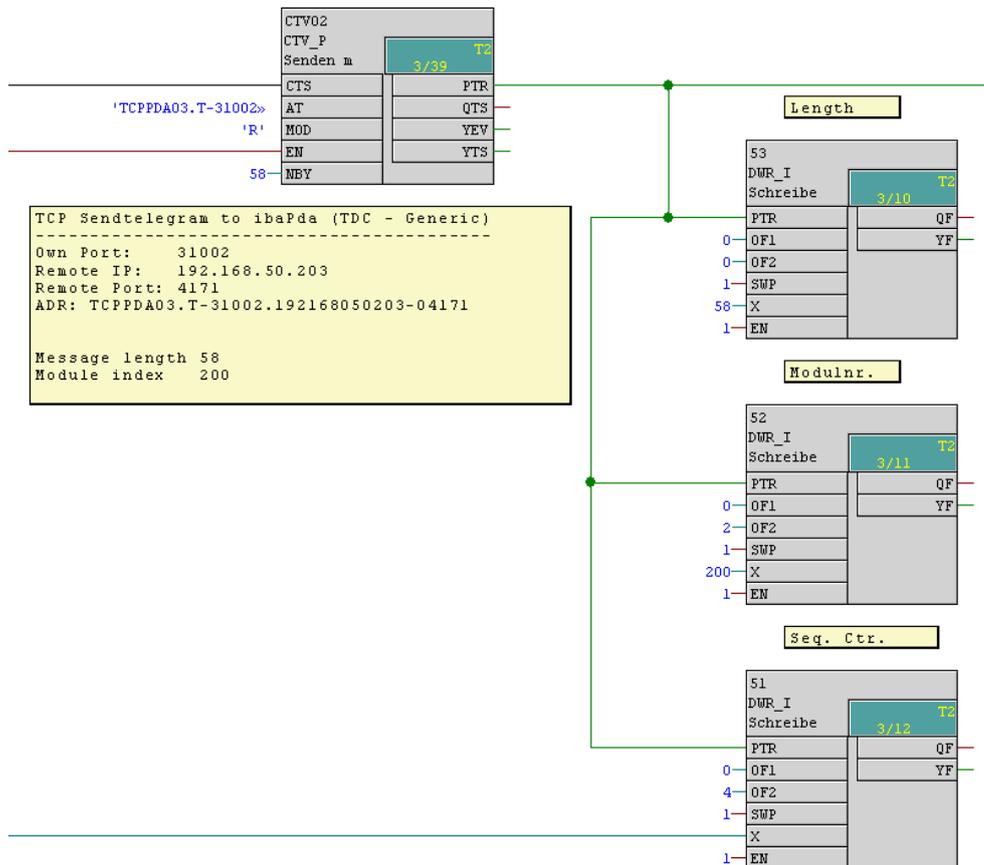


5.2.3.3 TCP Telegram with Module Type Generic

Parameterizing 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



5.2.3.4 UDP Telegrams

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

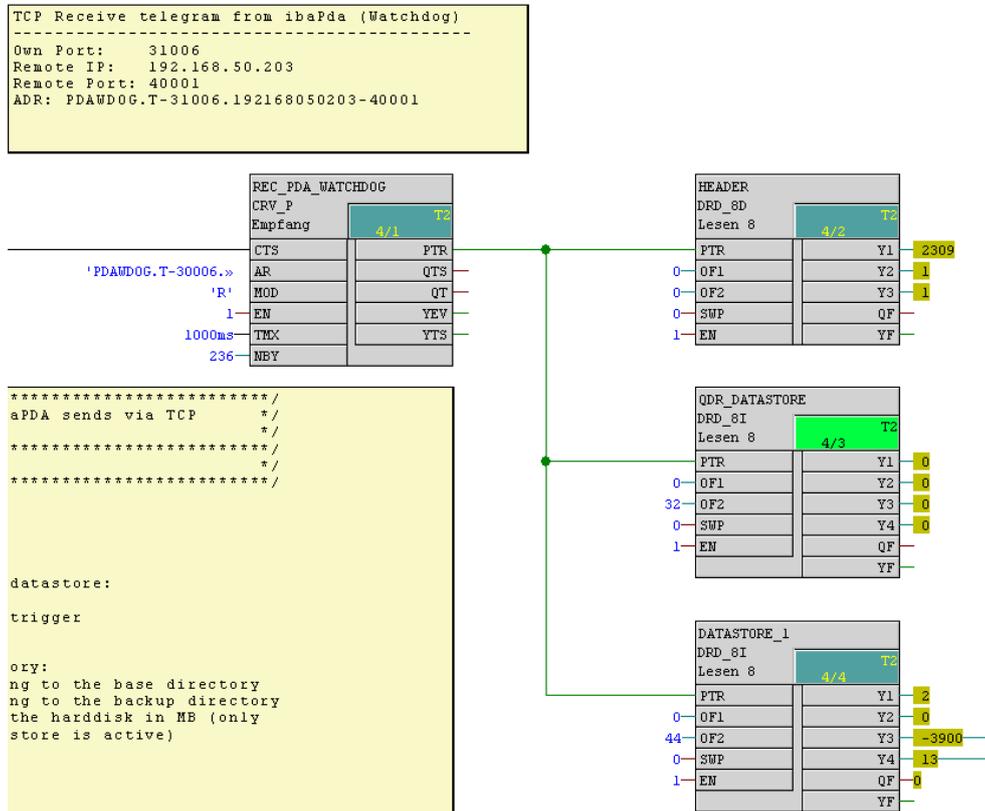
In the address connector AT, the identifier “U” is used instead of “T”.

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.



Other documentation



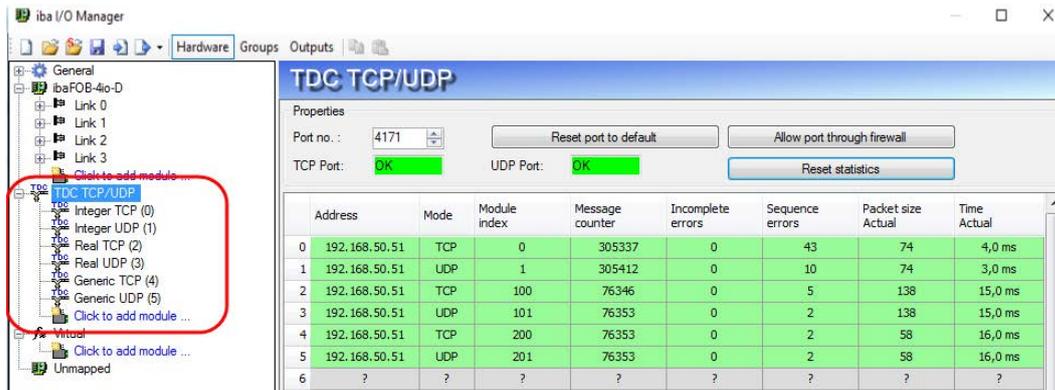
You will find the content and structure of the *ibaPDA* watchdog telegram in the *ibaPDA* manual.

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:



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

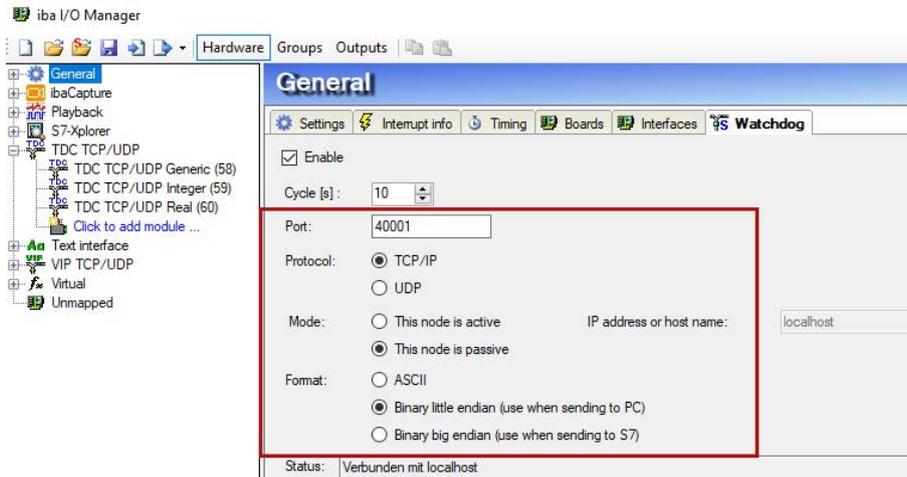
Integer TCP (0)						
TDC General Analog Digital						
Name	Unit	Min	Max	Active	Actual	
0 Sinus		-32768	32767	<input checked="" type="checkbox"/>	766	
1 Cosinus		-32768	32767	<input checked="" type="checkbox"/>	-642	
2 Triangle		-32768	32767	<input checked="" type="checkbox"/>	5022	
3 Counter T1		-32768	32767	<input checked="" type="checkbox"/>	17001	
4 Counter T2		-32768	32767	<input checked="" type="checkbox"/>	-30644	
5 Counter T3		-32768	32767	<input checked="" type="checkbox"/>	25106	
6 Counter T4		-32768	32767	<input checked="" type="checkbox"/>	-10108	
7 Counter T5		-32768	32767	<input checked="" type="checkbox"/>	-18912	

Integer TCP (0)			
TDC General Analog Digital			
Name	Active	Actual	
0 Bit 0	<input checked="" type="checkbox"/>	0	
1 Bit 1	<input checked="" type="checkbox"/>	1	
2 Bit 2	<input checked="" type="checkbox"/>	1	
3 Bit 3	<input checked="" type="checkbox"/>	0	
4 Bit 4	<input checked="" type="checkbox"/>	0	
5 Bit 5	<input checked="" type="checkbox"/>	0	
6 Bit 6	<input checked="" type="checkbox"/>	0	
7 Bit 7	<input checked="" type="checkbox"/>	0	
8 Bit 8	<input checked="" type="checkbox"/>	0	

5.3.2 Watchdog Configuration

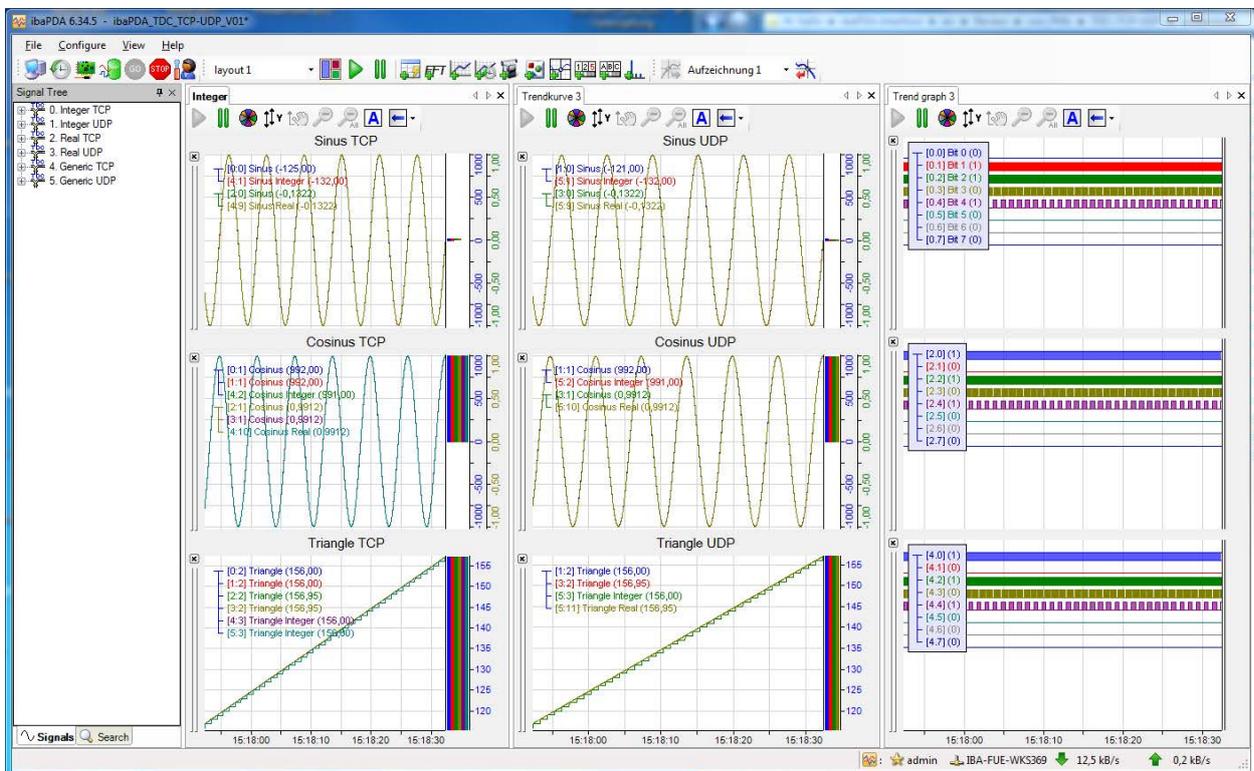
Under *Hardware - General* you find the tab *Watchdog*. The watchdog telegram with the following parameters is defined there:

Port number: 40001, Protocol: TCP/IP, Mode: passive, Format: Binary Little Endian



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

Note



If you need support for software products, please state the license number or the CodeMeter container number (WIBU dongle). 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
Fax: +49 911 97282-33
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.