ibaPDA-S7-TCPIP

Manual

Edition V 2.1

Measurement and Automation Systems

Manual

ibaPDA-S7-TCPIP

Data Interface for ibaPDA and Simatic S7 over TCP/IP

Edition V 2.1

iba AG

Printed 12 June, 2009

ibaPDA-S7-TCPIP - Manual

Issued by:

iba AG

Koenigswarterstr. 44

90762 Fuerth

Germany

Tel.: + 49 (0)911 9 72 82-0

Sales -27

Support -14

R&D -13

FAX -33

Email: iba@iba-ag.com

Web: www.iba-ag.com

ibaPDA-S7-TCPIP manual

We have checked that the contents of this manual match the hardware and software described here. However, deviations cannot be fully ruled out, so that we cannot assume any warranty should any deviations actually exist. This manual is regularly updated. Necessary revisions are included in future editions, or can be downloaded from the Internet.

The latest version is always available for downloading at

http://www.iba-ag.com

We would welcome any suggestions for improvements which you may have.

Version	Date	Revision	Author	Version SW
2.1_en	12.06.2009	New communication methods	rm	ibaPDA-V6.17.0

Table of Contents

1 General

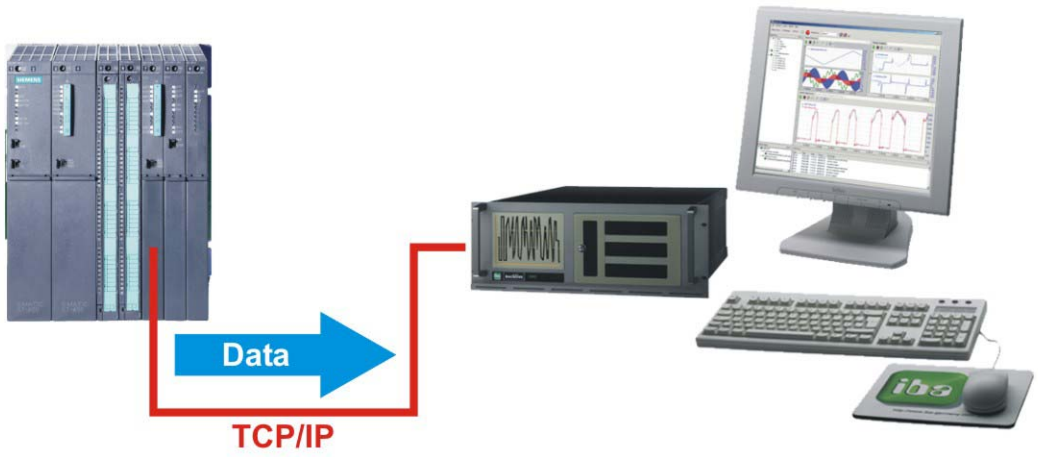
This manual describes the engineering of a TCP/IP connection between a Simatic S7 PLC and an ibaPDA system in order to acquire data (S7 variables) from the S7 CPU and record them with ibaPDA.

Furthermore, the manual offers hints for commissioning and is aimed at the experienced user and programmer.

Sample topology

Simatic S7-300 with CP 343-1 ibaPDA-PC

Simatic S7-400 with CP 443-1



1.1 System pre-requisites PC

❑ Generic PC or notebook computer

- Windows XP Professional, 2000 or 2003 Server
- P3 / 2 GHz
- 512 MB RAM
- 10 MB free harddisk space for program plus space for data files
- Network interface 10/100 Mbps
- 1 free USB interface for dongle
- iba dongle with ibaPDA basic license and license "ibaPDA-Interface-S7-TCPIP“
- ibaPDA-V6.17.0 or higher installed (client and server)

1.2 System pre-requisites Simatic S7

- CP 443-1 for S7-400 CPUs
- CP 343-1 for S7-300 CPUs
- Step7 version V4.0 or higher

2 ibaPDA solutions for Simatic S7

iba offers three different solutions for measurement of data from a Simatic S7 PLC:

2.1 S7-Analyzer

The easiest way to measure signals from a Simatic S7 PLC is to use the product ibaPDA-S7-Analyzer.

ibaPDA-S7-Analyzer was developed as a dedicated tool for electrical suppliers, control engineers, service engineers and commissioning experts. It provides the function of measured data acquisition and recording of up to 64 signals of a Simatic S7 PLC from the start of engineering phase through system test and commissioning.

ibaPDA-S7-Analyzer uses the same interfaces as used by the Simatic programming device for the measurement of data: TCP/IP, MPI and Profibus. System services of the S7 which are running separately from the cyclic application program are used for measurement. Thus, the measurement may not be as precise as the control cycles (timewise).

For the connection to TCP/IP either the generic network interface of the PC or the network adapter CP1613 can be used.

2.2 Profibus

For measuring signals from a Simatic S7 via Profibus DP two different methods can be used with a standard ibaPDA-V6 system:

1. Use of Profibus-DP interface board ibaCom-L2B-4-8 or -8-8

Up to 4 PCI cards of this type can be installed in an ibaPDA PC. Each card provides up to 4 or 8 slaves with 32 analog and 32 digital signals each (S7 Reals: only 28). the slaves must be considered in the Profibus configuration of the DP master.

The ibaCom-L2B card allows the use of symbolic request (ibaPDA-Request-S7) for measurement of symbolic variables in the S7.

2. Use of bus monitor device ibaBM-DPM-S

The ibaBM-DPM-S device can work as active slave or as mere sniffer on the Profibus DP.

One device can read up to 512 analog and 512 digital signals from the Profibus and transmit the data to an ibaPDA-V6 system.

The ibaBM-DPM-S device allows the use of symbolic request (ibaPDA-Request-S7) for measurement of symbolic variables in the S7.

2.3 S7-TCP/IP

This solution uses a so called „programmable connection“ on the S7 side. Therefore, a TCP/IP compatible S7-CP must be inserted in the Simatic S7 rack and configured with the Simatic net tool “NetPro“.

The ibaPDA PC may be connected over a usual Ethernet TCP/IP network adapter.

First, the data to be measured will be shunted by the S7 application program into DBs, containing the data in a fixed structure.

This structure derives from ibaPDA module types. Three types are available:

- Integer: 32 analog values (Integer) and 32 digital signals
- Real: 8, 16 or 32 analog values (Real) and 32 digital signals
- Generic: any data structure with maximum length of 400 bytes (*)

Then the DBs will be sent to the ibaPDA PC by S7 communication FCs as telegrams. For each telegram a dedicated connection must be configured on the S7-CPU.

A maximum of 64 connections can be configured on the ibaPDA side. On the S7 side the limitation of number of connections depends on the CPU type.

Please check the matter of compatibility of CPU and CP in the S7 documentation and the number of configurable TCP/IP connections (CPU property), if required.

The major advantage of this solution is that no special hardware components are required if a TCP/IP compatible S7 CP is already available in the PLC.

(*) The length is defined by properties of the S7 communication processors (see also chapter 8.2). Exceeding this length a DB will be split into 2 telegram blocks. This creates telegram errors on the ibaPDA side which are indicated by the max. package size 400 and periodical occurrence of sequence errors (see also chapter 5.3.1).

3 Software package S7-TCP/ IP

The software package includes the S7 sample project, created by iba AG and engineered with a CPU S7-412 and a TCP/IP compatible CP 443-1.

The project consists of a possible hardware configuration and function blocks for a TCP/IP connection between S7 and ibaPDA.

For each module type "Integer", "Real" and "Generic" there are preconfigured transmissions included which consist of a connection, a data block, a send block and instructions for data simulation and creation of a telegram counter.

The OB35 has been used for the programming in order to send the data not periodically but time controlled (by cyclic interrupt) to ibaPDA in this example. If required, you may also program the data connection in other function blocks.

Overview:

Module type	Connection ID	Data block	Length	Send block	Call in...
Integer	1	DB222	74	AG_LSEND	OB35
Real	2	DB223	138	AG_LSEND	OB35
Generic	3	DB224	206	AG_LSEND	OB35

If required you can copy the DBs with their structure. Also, you may rename the designation of the DBs according to your needs.

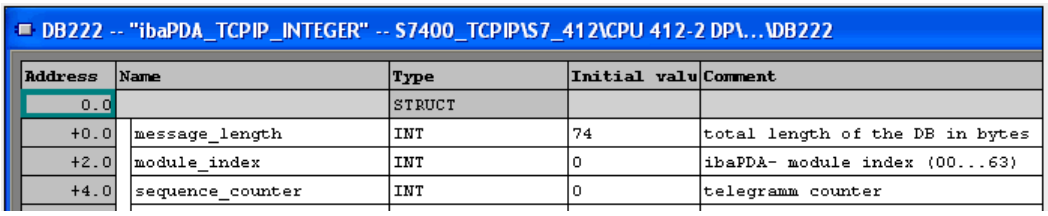
The described S7 sample project "S7400_TCPIP.zip" and the corresponding ibaPDA project are stored on the delivery CD-ROM. The software is also available for download on our website or can be ordered by phone via the iba service line.

Name of the demo project: S7400_TCPIP.zip

In the demo project there are 3 programmed connections to ibaPDA.

4 Used function blocks

Die data is transmitted by one TCP/IP telegram per module according to the ibaPDA module structure. Each telegram is based on a data block. The data blocks have a uniform header and data structure which complies with the module type.



DB222 -- "ibaPDA_TCPIP_INTEGER" -- S7400_TCPIP\S7_412\CPU 412-2 DP\...\DB222

Address	Name	Type	Initial valu	Comment
0.0		STRUCT		
+0.0	message_length	INT	74	total length of the DB in bytes
+2.0	module_index	INT	0	ibaPDA- module index (00...63)
+4.0	sequence_counter	INT	0	telegramm counter

4.1 Header

The header consists of 3 integer values.

❑ message_length

…is the total size (given in byte) of the data package (DBs). The initial value must remain unchanged during data transmission! This value is required again for the call of a send block (FC5 for S7-300, FC50 for S7-400).

The length depends on the module type:

- module type Integer: always 74
- module type Real: 38 (for 8 Reals)
  70 (for 16 Reals)
  138 (for 32 Reals)
- module type Generic: 8 to 400

❑ module_index

This label is used for assignment of a data set to the interface module in ibaPDA. The index contains also the module type in an encoded manner: the index is created by an increasing number from 00 to 63 and an offset which refers to the module type.

- module type Integer: Offset 0
- module type Real: Offset 100
- module type Generic: Offset 200

The module index corresponds to the index in the ibaPDA module settings. The initial value must remain unchanged during data transmission!

❑ sequence_counter

Each successful execution of the transmission order will increase this value by one. The counting has to be programmed in the S7. If the counter value doesn't change, ibaPDA will show a sequence error in its TCP/IP connection table.

4.2 Data ranges

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

4.2.1. Module type Integer

After the header, the 32 integer analog values follow at offset 6 and 4 bytes with digital values follow at offset 70.

DB222 -- "ibaPDA_TCPIP_INTEGER" -- S7400_TCPIP\S7_412\CPU 412-2 DP\...\DB222

Address	Name	Type	Initial valu	Comment
0.0		STRUCT		
+0.0	message_length	INT	74	total length of the DB in bytes
+2.0	module_index	INT	0	ibaPDA- module index (00...63)
+4.0	sequence_counter	INT	0	telegramm counter
+6.0	analog_signals	ARRAY[0..31]		32 analog values INT
*2.0		INT		
+70.0	digital_signals	ARRAY[0..31]		32 binary values
*0.1		BOOL		
=74.0		END_STRUCT		

Please note that S7 and ibaPDA have different byte orders: if you set the bit DB222.DBX70.0 it will be received on bit 24 in ibaPDA. But if you write 16#00000001 on DB222.DBD70 then bit 0 will be set in ibaPDA.

4.2.2. Module type Real

After the header, 4 bytes with digital values follow at offset 6 and 8, 16 or 32 real analog values follow at offset 10.

DB223 -- "ibaPDA_TCPIP_REAL" -- S7400_TCPIP\S7_412\CPU 412-2 DP\...\DB223

Address	Name	Type	Initial valu	Comment
0.0		STRUCT		
+0.0	message_length	INT	138	total length of the DB in bytes
+2.0	module_index	INT	101	ibaPDA- module index (00...63) + 100 (Offset for type REAL)
+4.0	sequence_counter	INT	0	telegramm counter
+6.0	digital_signals	ARRAY[0..31]		32 analog values INT
*0.1		BOOL		
+10.0	analog_signals	ARRAY[0..31]		32 binary values
*4.0		REAL		
=138.0		END_STRUCT		

4.2.3. Module type Generic

After the header, any data sequence of different data types can follow at offset 6. The following data formats are supported by ibaPDA:

BYTE, INT, WORD, DINT, DWORD and FLOAT.

The same data structure as defined in this DB must be reproduced in ibaPDA. The variables of type BYTE, WORD and DWORD can also be taken in ibaPDA as sets of 8, 16 or 32 bits (and vice versa).

DB224 -- "ibaPDA_TCPIP_GENERIC" -- S7400_TCPIP\S7_412\CPU 412-2 DP\...\DB224

Address	Name	Type	Initial valu	Comment
0.0		STRUCT		
+0.0	message_length	INT	206	total length of the DB in bytes
+2.0	module_index	INT	202	ibaPDA- module index (00...63) + 200 (Offset for type GENERIC)
+4.0	sequence_counter	INT	0	telegramm counter
+6.0	digitals	ARRAY[0..63]		64 digitals
*0.1		BOOL		
+14.0	analog_integers	ARRAY[0..31]		32 integers
*2.0		INT		
+78.0	analog_real	ARRAY[0..31]		32 reals
*4.0		REAL		
=206.0		END_STRUCT		

Please note:

The maximum length of a DB of 400 bytes must not be exceeded, else ibaPDA will generate a sequence error (see also chapter 8.2.).

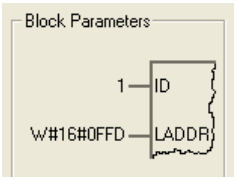
Recording of variables of types DWORD or DINT may cause problems in analysis because ibaAnalyzer does not support these data types. These types are converted automatically into Real format by ibaPDA during recording. As a consequence the 32 bit resolution will get lost. If the recording of 32 bit vector signals (arrays) is required we recommend to receive these variables in ibaPDA as 2 x WORD or as single bits.

4.3 Send blocks

In order to send the data via the configured connection please use the block AG_SEND / FC5 (with CP343 – old versions) or AG_LSEND / FC50 (with CP343-1 and CP443-1). Please refer to S7 Online-Help, FC5 or FC50, for detailed description.

Connecting the block:

```
CALL  "AG_LSEND"
 ACT   :="TCPIP_act"              //statisch "1- Merker"
 ID    :=1                        //Connection ID (from Connection in Netpro)
 LADDR :=W#16#FFD                 //Address (from Connection in Netpro)
 SEND  :=P#DB222.DBX0.0 BYTE 74   //Structure "INT"
 LEN   :=74
 DONE  :="M0_done"                //Job done
 ERROR :="M0_error"               //Error
 STATUS:="M0_status"
```



Connectors:

- ACT Initialization by flag
- ID Connection ID according to NetPro
- LADDR Address of the CP
- SEND Pointer to DB
- LEN Length value, equal to message_length
- DONE "Finished" indicator
- ERROR Error indicator
- STATUS Status display

Please note:

If the AG_LSEND is executed every cycle the data transmission cycle is only half the execution cycle because every second execution the result connectors are updated instead of triggering a new transmission.

5 Engineering guide

This chapter describes the steps of setting up the CPU- and CP parameters and explains how to establish a connection between the S7-CP and the ibaPDA-PC.

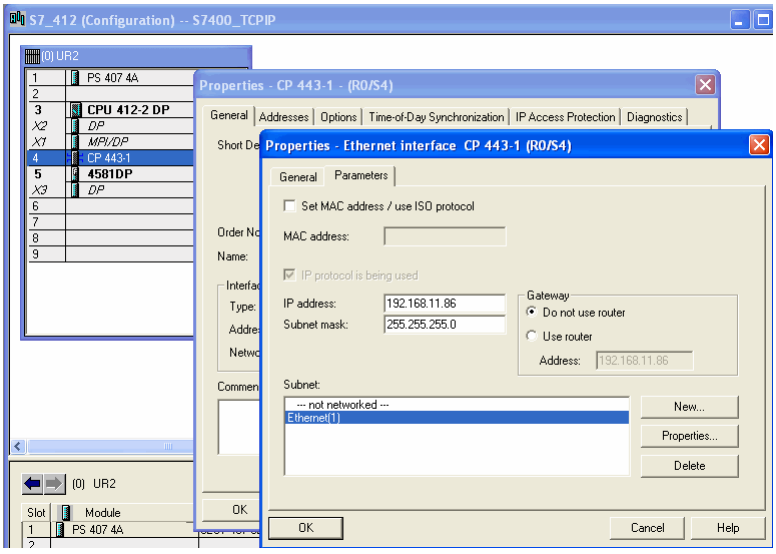
The software tools “HW-Konfig.” and “NetPro”, which are part of STEP7, will be used for configuration on S7 side.

5.1 Simatic S7

Please follow the next steps in the given order by using STEP7 software!

5.1.1. Engineering with STEP 7 tool "HW-Konfiguration“

1. If not done yet: insert the CP to be used.
2. Create a new subnet (e.g. Ethernet(1)).
3. Enter IP address and subnet mask for the CP.



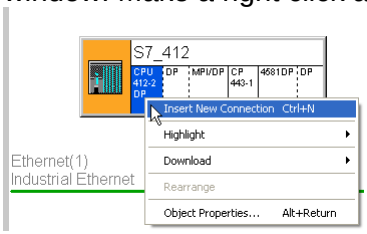
4. Save and load the hardware configuration.

5.1.2. Engineering with STEP 7 tool „NetPro“

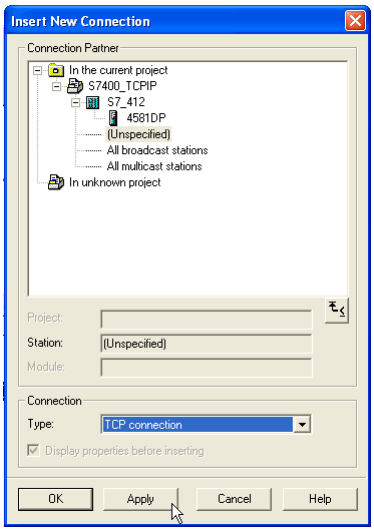
For each data package to be sent a TCP connection must be configured.

The following steps 2, 3 and 4 should be repeated for each connection. Each connection gets a different ID.

1 Click on "Configure network" in the "Extras" menu.
Mark the CPU. The (empty) connection table appears at the lower rim of the window. Make a right click and choose "Insert new connection".

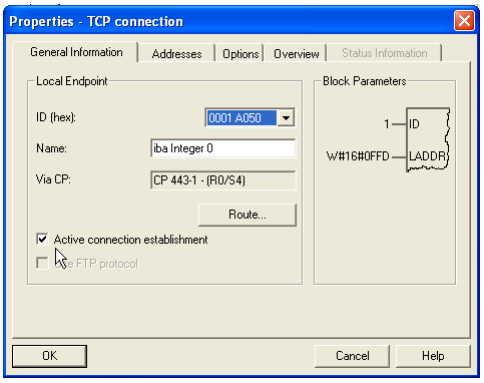


2 Insert connection(s)
- (Partner) Station: unspecified
- Connection type: TCP connection
- <Apply>

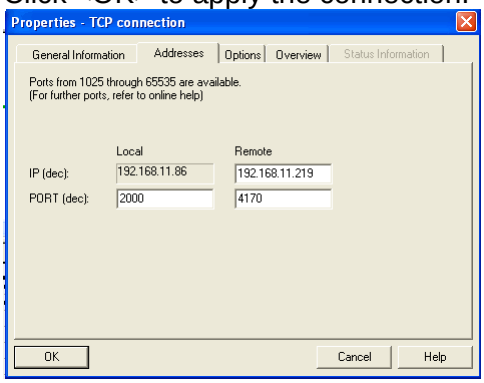


Notice:
When inserting the new connection the parameter "Station" should be set to "unspecified"! Else, the port address 5001 could not be set multiple times for multiple connections.

3 Set general parameters
- Edit connection name
- Enable "Active connection establishment"
- Keep block parameters (ID and LADDR) in mind

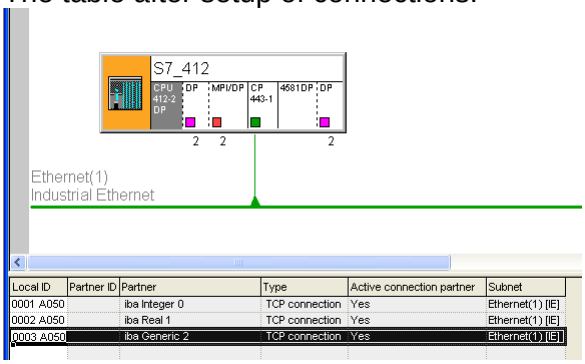


4 Set addresses for “Local“ and "Remote“
"Local": Retain default value (one port for each connection)
"Remote":
- IP address of the ibaPDA PC,
- Port number, free choice, e.g. "4170"
Click <OK> to apply the connection.



Notice:
The remote port must also be set in ibaPDA (TCPIP interface).
The remote port must be a member of the exceptions in the ibaPDA PC firewall.
The remote port must not be used anywhere else.

5 Insert further connections by using the same method
Always edit new connection names
Always set the same remote IP address
Always set the same remote port number
The connection IDs and the local port numbers are configured automatically.
The table after setup of connections:



6 Save, compile and load configuration data.

5.1.3. Programming with the STEP 7 tool "Block editor"

In the following example the measured data are simulated by means of the message counter, sawtooth and sine curves. Digital signals are simulated by the single bits of the counters.

The transmission of the data occurs in the cyclic interrupt OB35 but may also work in another OB, e.g. periodically.

The message counters are cleared in start-up OB100, 101 (depending on CPU).

Modify the example according to your application.

1 Transmission of integer data (iba module type INTEGER)
Edit / copy the DB for INT data (DB222)
Define the data structure. Prepare header and execute the menu command “Initialize data block” (only possible in data view).

DB222 -- "ibaPDA_TCPIP_INTEGER" -- S7400_TCPIP\S7_412\CPU 412-2 DP\...\DB222

Address	Name	Type	Initial valu	Comment
0.0		STRUCT		
+0.0	message_length	INT	74	total length of the DB in bytes
+2.0	module_index	INT	0	ibaPDA- module index (00...63)
+4.0	sequence_counter	INT	0	telegramm counter
+6.0	analog_signals	ARRAY[0..31]		32 analog values INT
*2.0		INT		
+70.0	digital_signals	ARRAY[0..31]		32 binary values
*0.1		BOOL		
=74.0		END_STRUCT		

Map the signals to be measured for DB222 (module type INT).

```
Network 1: Preset Integer values for DB 222 (type INTEGER)
Comment:

//Analog values

      OPN   "ibaPDA_TCPIP_INTEGER"
      L     P#6.0
INT:  T     #pointer_INT                //Loop to fill DB222 with counter
      L     "ibaPDA_TCPIP_INTEGER".sequence_counter
      T     DBW [#pointer_INT]
      L     #pointer_INT
      L     P#2.0
      +D
      L     P#68.0
      TAK
      >=D
      JC    INT


//Digital values

      L     "ibaPDA_TCPIP_INTEGER".sequence_counter
      T     DB222.DBB   70            // digital signal bits 31..24
      T     DB222.DBB   71            // digital signal bits 23..16
      T     DB222.DBB   72            // digital signal bits 15..8
      T     DB222.DBB   73            // digital signal bits 7..0
```

Send DB222 by FC50 (AG_LSEND) to ibaPDA.

```
Network 4: Send DB222 to ibaPDA S7-TCPIP modul index 0 (type INT)
Comment:

      CALL  "AG_LSEND"
       ACT   :="TCPIP_act"              //statisch "1- Merker"
       ID    :=1                        //Connection ID (from Connection in Netpro)
       LADDR :=W#16#FFD                 //Address (from Connection in Netpro)
       SEND  :=P#DB222.DBX0.0 BYTE 74   //Structure "INT"
       LEN   :=74
       DONE  :="M0_done"                //Job done
       ERROR :="M0_error"               //Error
       STATUS:="M0_status"
```

Increment message counter, after order was executed.

```
Network 5: Sequence counter in DB222
Comment:

      A     "M0_done"                   //Job done      M180.0
      FP    "M0_fp"                                   M180.2
      JCN   nw4e
      L     "ibaPDA_TCPIP_INTEGER".sequence_counter   DB222.DBW4       -- t
      L     1
      +I
      T     "ibaPDA_TCPIP_INTEGER".sequence_counter   DB222.DBW4       -- t
nw4e: NOP   0
```

2 Transmission of 32 real data (ibaPDA module type REAL)
Edit / copy the DB for REAL data.
Define the data structure. Prepare header and execute the menu command “Initialize data block” (only possible in data view).

DB223 -- "ibaPDA_TCPIP_REAL" -- S7400_TCPIP\S7_412\CPU 412-2 DP\...\DB223

Address	Name	Type	Initial valu	Comment
0.0		STRUCT		
+0.0	message_length	INT	138	total length of the DB in bytes
+2.0	module_index	INT	101	ibaPDA- module index (00...63) + 100 (Offset for type REAL)
+4.0	sequence_counter	INT	0	telegramm counter
+6.0	digital_signals	ARRAY[0..31]		32 analog values INT
*0.1		BOOL		
+10.0	analog_signals	ARRAY[0..31]		32 binary values
*4.0		REAL		
=138.0		END_STRUCT		

Map the signals to be measured for DB223 (module type REAL).
Send DB223 by FC50 (AG_LSEND) to ibaPDA.
Increase message counter, after order was executed.

3 Transmission of mixed data (ibaPDA module type GENERIC)
Edit / copy the DB for mixed data (DB224).
Define the data structure. Prepare header and execute the menu command “Initialize data block” (only possible in data view).

DB224 -- "ibaPDA_TCPIP_GENERIC" -- S7400_TCPIP\S7_412\CPU 412-2 DP\...\DB224

Address	Name	Type	Initial valu	Comment
0.0		STRUCT		
+0.0	message_length	INT	206	total length of the DB in bytes
+2.0	module_index	INT	202	ibaPDA- module index (00...63) + 200 (Offset for type GENERIC)
+4.0	sequence_counter	INT	0	telegramm counter
+6.0	digitals	ARRAY[0..63]		64 digitals
*0.1		BOOL		
+14.0	analog_integers	ARRAY[0..31]		32 integers
*2.0		INT		
+78.0	analog_real	ARRAY[0..31]		32 reals
*4.0		REAL		
=206.0		END_STRUCT		

Map the signals to be measured for DB224 (module type GENERIC).
Send DB224 by FC50 (AG_LSEND) to ibaPDA.
Increase message counter, after order was executed.

4 Reset message counter at CPU start in OB100

```
OB100 :  "Complete Restart"
Comment:
Network 1: RESET sequence counters for ibaPDA-TCPIP telegra
Comment:
      L     0
      T     "ibaPDA_TCPIP_INTEGER".sequence_counter  DB222.DBW4
      T     "ibaPDA_TCPIP_REAL".sequence_counter     DB223.DBW4
      T     "ibaPDA_TCPIP_GENERIC".sequence_counter  DB224.DBW4
```

5 Save all edited blocks and load them into the S7-CPU.

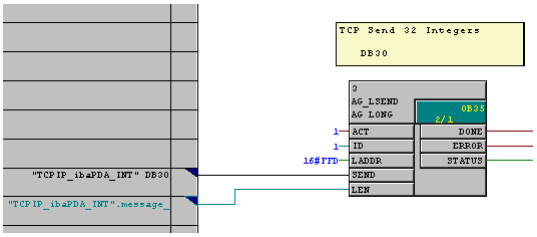
Attention:
For module type REAL: module_index = counting number + 100

For module type GENERIC: module_index = counting number + 200

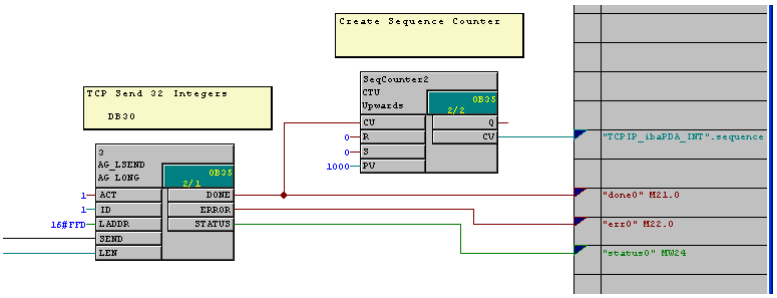
5.1.4. Programming with CFC editor

1 Create data block with AWL editor (see above).

2 Create a CFC diagram.
Call AG_LSEND.



Create the sequence counter.



3 Compile and load.

5.2 Configuration of ibaPDA-V6

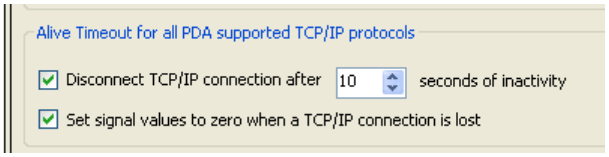
The following description only applies to version V6.17.0 or higher of ibaPDA. If you use libaPDA V5 please ask for edition 1.1 of this manual. If you use a lower version of ibaPDA-V6 than V6.17.0 we recommend that you update your ibaPDA to the current version.

Furthermore, you'll need an additional software license (S7-TCPIP) on your ibaPDA dongle to establish a connection between Simatic S7 and ibaPDA-V6 via S7-TCP/IP.

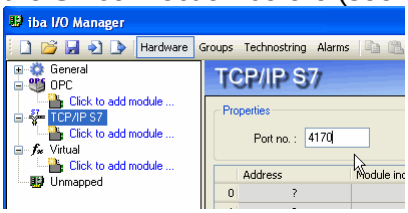
Start the ibaPDA client software, open the I/O manager and follow these steps:

If a firewall is enabled in your ibaPDA PC (standard with Windows XP, SP2) the port for the S7-TCP/IP connection must be member of the exceptions.

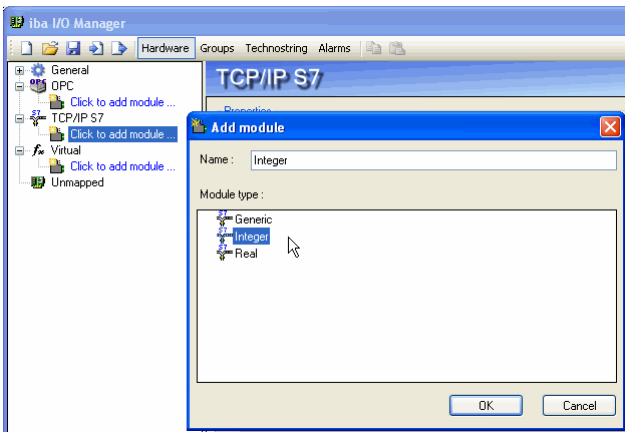
1 General settings
In the general settings (branch "General") you can setup the behaviour in case of connection breakdown:



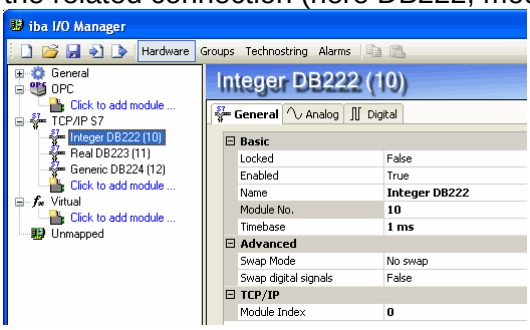
2 Define the port
If the dongle was detected correctly and the S7-TCPIP license is enabled you will see an interface TCP/IP S7 in the tree in the left pane of the I/O manager. Mark the TCP/IP S7 interface and enter the port number in the corresponding entry field. The port number must be the same as configured in the setup of the S7 connection before (see chapter 5.1.2, examle: port no. 4170).



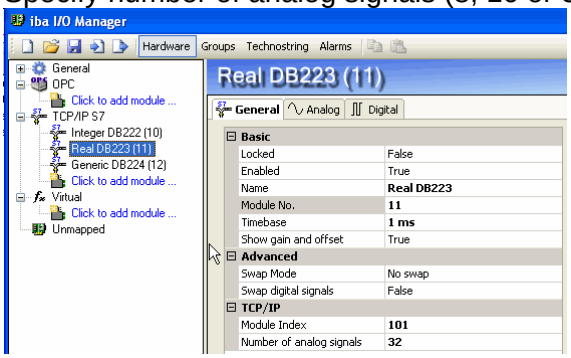
3 Define modules
If you click on the branch "Click to add module..." beneath the marked TCP/IP S7 interface you can select the module type you need. Up to 64 modules are permitted.



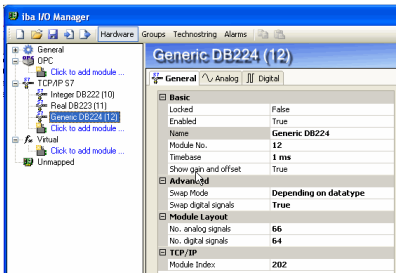
4 Module settings for module type Integer
Enter module name and number.
Retain default swap mode (no swap).
Set module index 0..63; must be equal to the module index of the S7 DB of the related connection (here DB222, module index 0).



5 Module settings for module type Real
Enter module name and number.
Retain default swap mode (no swap).
Set module index 100..163; must be equal to the module index of the S7 DB of the related connection (here DB223, module index 101).
Specify number of analog signals (8, 16 or 32).

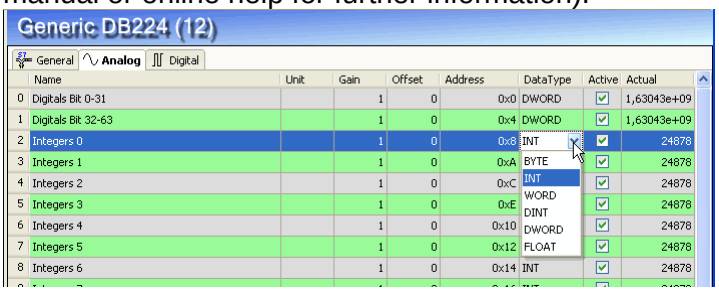


6 Module settings for module type Generic
Enter module name and number.
Retain default swap mode (no swap).
Set module index 200..263; must be equal to the module index of the S7 DB of the related connection (here DB224, module index 202).
Specify number of analog and digital signals (free choice).



7 Definition of data structure of generic modules
Select the tab "Analog" (signal table of analog signals).
For each signal select the correct data type from the drop-down list in the column "Data type".

Tip: You can use the automatic fill-in function of the column (see ibaPDA manual or online help for further information).



Enter the address for each signal which is in fact the offset in the message buffer. Please note, that counting refers to net data without header. Thus, the address equals the offset in the DB minus 6.

Tip: If the signals are packed in a compact order without gaps you can use the automatic fill-in function of the column. ibaPDA will then evaluate the correct size according to the data type. Please note that variables of all data types except bits and bytes should always have an even offset in the S7 DB.

8 Apply the settings (click on <Apply>).

9 Create a view in the ibaPDA client.

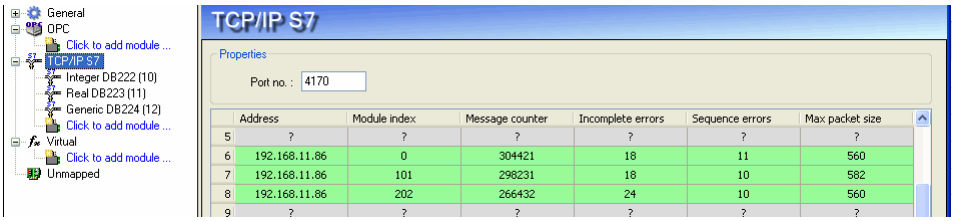
10 Setup the data storage for the measured data.

For more detailed information about setting up data storage and signal display please refer to the ibaPDA-V6 manual.

5.3 Diagnostics

5.3.1. Validating the connection

After acceptance of the configuration all connections are listed in the connection table.



Notice:
The lines which show the connections are assigned internally with increasing number by each establishment of connection.

Connections may be displayed only in parts or not at all due to the following reasons:

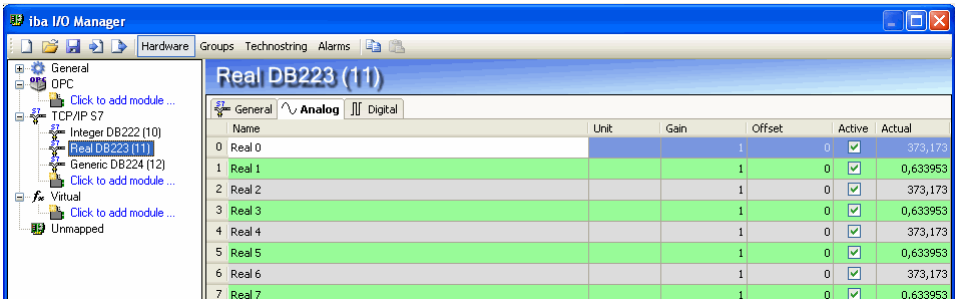
- ❑ S7 is in "Stop" status
- ❑ no Ethernet link between ibaPDA PC and S7 CP
- ❑ mistake in S7 / NetPro configuration:
  - wrong remote IP address
  - port number does not comply with S7 connection
  - port blocked by firewall

Other errors:

- ❑ If the sequence counters do not count continuously the send blocks AG_LSEND are not executed in every cycle on the S7 side.
- ❑ If values increase in columns "Incomplete errors" and / or "Sequence errors" one of the following errors might be the reason:
  - the "message_length" in the DB has another value as expected
  - the "sequence_counter" in the DB is not incremented
  - the DB length is larger than 400 bytes (for type Generic, see chapter 8.2)
  - the "Delayed Acknowledge" problem occurs (see chapter 8.1)

5.3.2. Validating the message contents

The contents of the TCPIP messages is displayed in the column "Actual" of the signal tables in tabs "Analog" and "Digital" of the TCP/IP S7 modules. If this is not the case the reason may be an incorrect filling of the DB on S7 side or a bad wiring of the connectors on the send block.



If messages are received approximately every 200 ms although they are sent in a faster cycle this is due to different variants of the TCP/IP protocol (problem "Delayed Ackowledge", see chapter 8.1).

6 Test results for cycle time extension

CPU- Type: S7- 416-2DP (416-2XK00-0AB0)

Please note that the measured time periods are CPU-specific.

The time measurement includes the execution times of the FC50 (AG_LSEND) and of the S7-operations of the message counter per connection.

6.1 Test „3 ibaPDA connections“

Number of programmed S7-connections: 3 (96 A + 96 D signals)

-> Cycle time extension 1-2 ms

6.2 Test „6 ibaPDA connections“

Number of programmed S7-connections: 6 (192 A+ 192 D signals)

-> Cycle time extension 2-3 ms

7 Support and contact

For technical support or sales information, please call the following numbers:

Telephone: +49 911 97282-14

Fax: +49 911 97282-33

email: support@iba-ag.com

For downloads of the latest software versions as well as hardware and software manuals, please use our website at: http://www.iba-ag.com

Any feedback, comments or tips on errors in this documentation or suggestions for improvement will be appreciated. Simply send an e-mail or fax to us, thank you for your support.

Headquarters

iba AG
Koenigswarterstrasse 44
90762 Fuerth
Germany
Tel.: +49 (911) 97282-13
Fax: +49 (911) 97282-33
Contact: Harald Opel
iba@iba-ag.com

Belgium, Netherlands, Luxemburg, France, Spain, Great Britain

IBA-Benelux BVBA
Rivierstraat 64
B-9080 Lochristi
Belgium
Tel.: +32 9 226 2304
Fax: +32 9 226 2902
Contact: Roeland Struye
roeland.struye@iba-benelux.com

North America, US Territories, Caribbean, Bermuda

iiba America, LLC
6845 Shiloh Road East, Suite D-7
Alpharetta, GA 30005
USA
Tel.: +1 (770) 886-2318
Fax: +1 (770) 886-9258
Contact: Scott Bouchillon
sb@iba-america.com

South America, Venezuela

iba LAT, S.A.
C.C San Miguel 1, Piso 1, Oficina 1.
Calle Neveri, Redoma de Harbor
YV 8050 Puerto Ordaz
Venezuela
Tel.: + 58 (286) 951 9666
Fax.: + 58 (286) 951 2915
Cell: + 58 (414) 386 0427

Contact: Eric Di Luzio
eric.di.luzio@iba-ag.com

ibaChina, ibaKorea, ibaIndia, ibaIndonesia ibaMalaysia, ibaThailand

ibaASIA GmbH & Co. KG
Saturnstrasse 32
90522 Oberasbach
Germany
Tel.: +49 (911) 969 4346
Fax: +49 (911) 969 4351
Contact: Mario Gansen
iba@iba-asia.com

8 Appendix

8.1 Problem variants of TCP/IP protocol

Error:

Sometimes, measurements by ibaPDA of automation devices via TCPIP (Simatic S7 - CP443 and CP343 , Simatic TDC - CP5100 and CP51M1, or others) don't work for cycle times of < 200ms.

Error manifestation in ibaPDA: sequence error or incomplete error.

Reason:

The TCPIP protocol offers different variants of acknowledge:

The WinSocket standard uses the "delayed acknowledge" method according to RFC1122. This means that an acknowledge is delayed until further messages are received in order to send one acknowledge for all of these together. If no further messages are received the acknowledge (ACK message) will be sent after a maximum delay time of 200 ms (depending on socket).

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

Some controllers don't accept this method but expecting an acknowledge after each data message instead. If this acknowledge is not received within a specified time (200 ms) the data message is sent again, occasionally filled up with new data to be sent, leading to an error on the receiver's side because the original was already received correctly.

How to fix it:

1st method: Switch off the "Fast acknowledge" on the CP; but this might cause also problems because other connections to other remote stations may be relayed over the same CP.

2nd method: Switch off the "delayed acknowledge" in Windows. This can be done by setting a parameter in the Windows Registry:

- Windows XP: parameter "TcpAckFrequency" REG_DWORD = 1;
- Windows 2000: parameter "TcpDelAckTicks" REG_DWORD = 0;

These parameters are not available by default and should be entered in the following path:

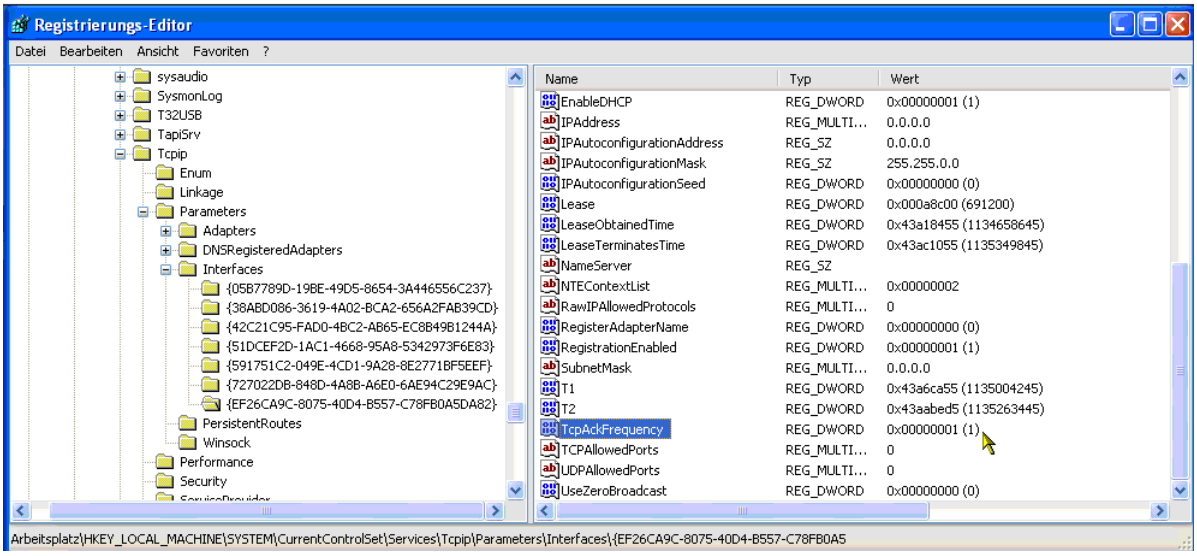
"HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\{InterfaceGUID}

One should select the correct interface. The correct interface can be found by checking the actual IP address, for instance.

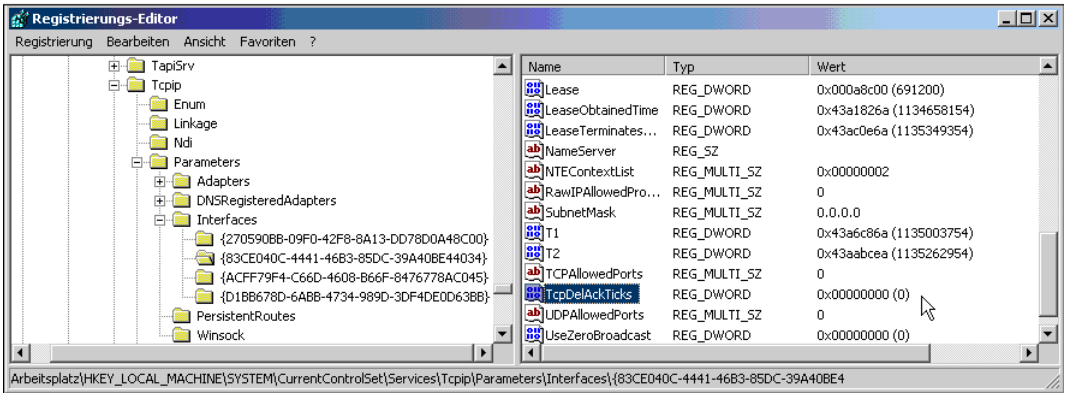
See figures below or refer to Microsoft web site

http://support.microsoft.com/kb/328890

Windows XP and Window Server 2003:



Windows 2000:



8.2 Problem data block length

Excerpt from the Siemens product information for communication processors:

Configuration Notes:

This entry is designed to explain a comment in the CP443-1EX40 device manual. Section 5.3 "Characteristic data for SEND/RECEIVE interface" provides information about data block lengths for the various protocols of Send/Receive communication.

LAN interface - data block length generated by th CP per protocol unit	
For sending	ISO transport, ISO-on-TCP, TCP: 400 bytes / TPDU
For receiving	ISO transport: 512 bytes / ISO-on-TCP: 1024 bytes / TCP: 1460 bytes / TPDU

The CP443-1EX40 device manual is available in Entry ID: 19308871

The values specified there are the number of user data bytes that are sent or received simultaneously by the module. It might well happen that the user sends/receives jobs to/from the communication blocks with longer user data lengths. These are divided into segments of appropriate length by the transport layer of the CP. These values are valid for the LAN interface of the module (local area network).

Values for sending user data blocks

The block size is limited to 400 bytes. But this doesn't depend on the protocol used. The buffer sizes are organized in the firmware so that always data blocks of a maximum length of 400 bytes are sent.

This means that a user data package of 401 bytes is sent in 2 separate messages. The first message has a length of 400 bytes and the second a length of one byte. The behavior described here applies for all 3 protocols of the Send/Receive interface used:

- ISO transport protocol
- ISO-on-TCP protocol
- TCP protocol