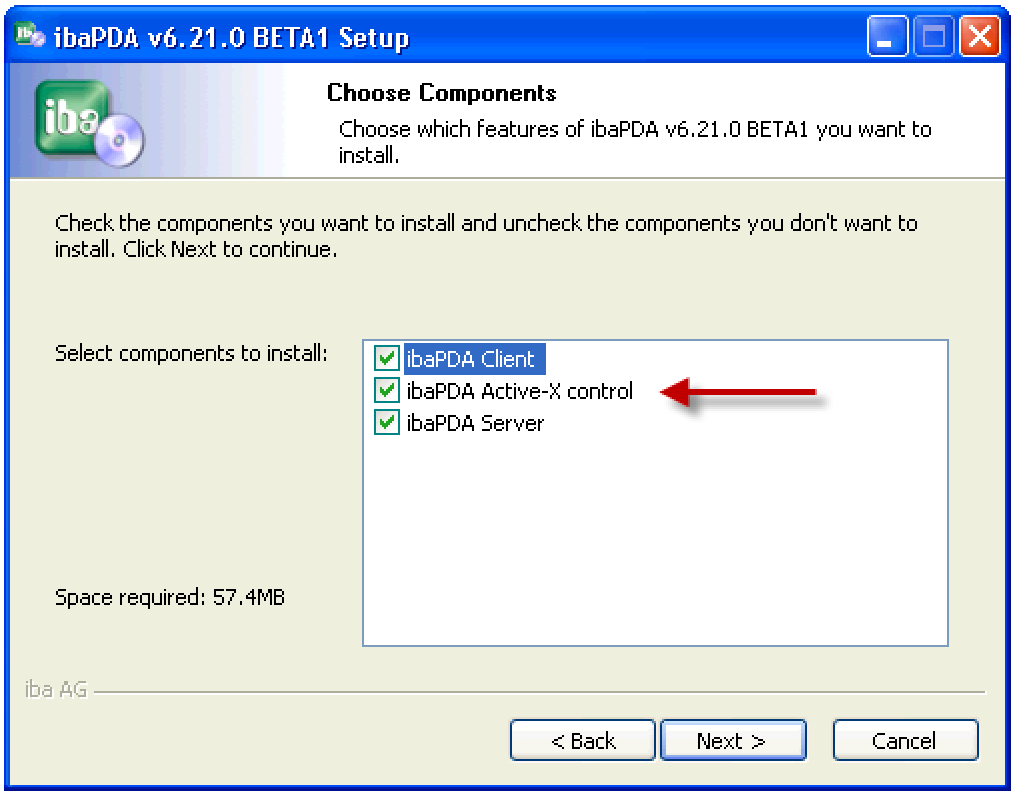
New features in ibaPDA v6.21.0

1 Active-X control

The ibaPDA client can be used as an active-x control. This means it can be embedded into any active-x container. Almost all HMIs are active-x containers. In this manual we will use WinCC as example active-x container.

During the installation of ibaPDA you have to select the ibaPDA active-x control.


ibaPDA v6.21.0 BETA1 Setup

Choose Components

Choose which features of ibaPDA v6.21.0 BETA1 you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

Select components to install:

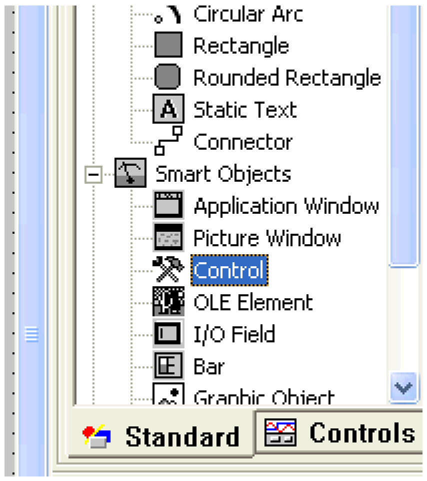
- ✓ ibaPDA Client
- ✓ ibaPDA Active-X control
- ✓ ibaPDA Server

Space required: 57.4MB

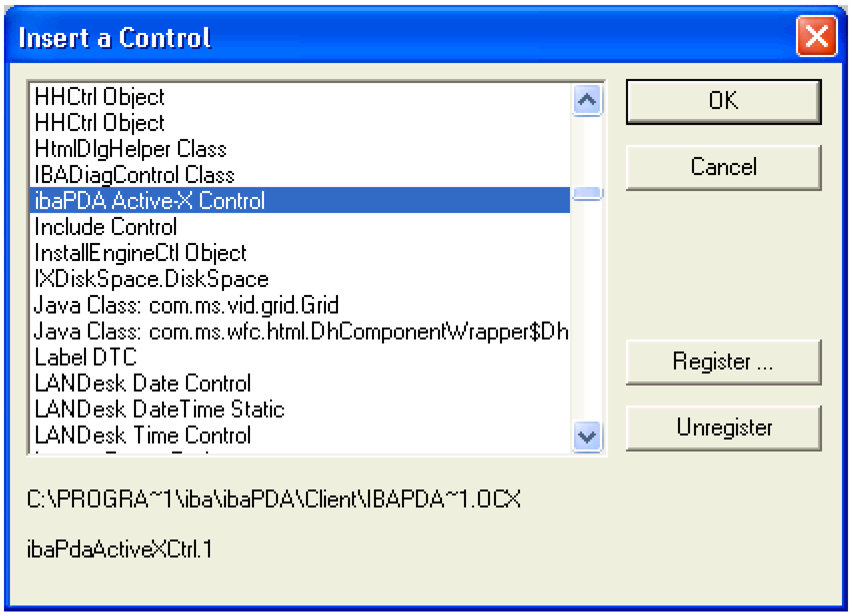
iba AG

< Back | Next > | Cancel

Create a new graphic in WinCC. In the empty graphic insert a control.



A dialog will open where you have to choose which active-x control to insert. Select *ibaPDA Active-X Control*.

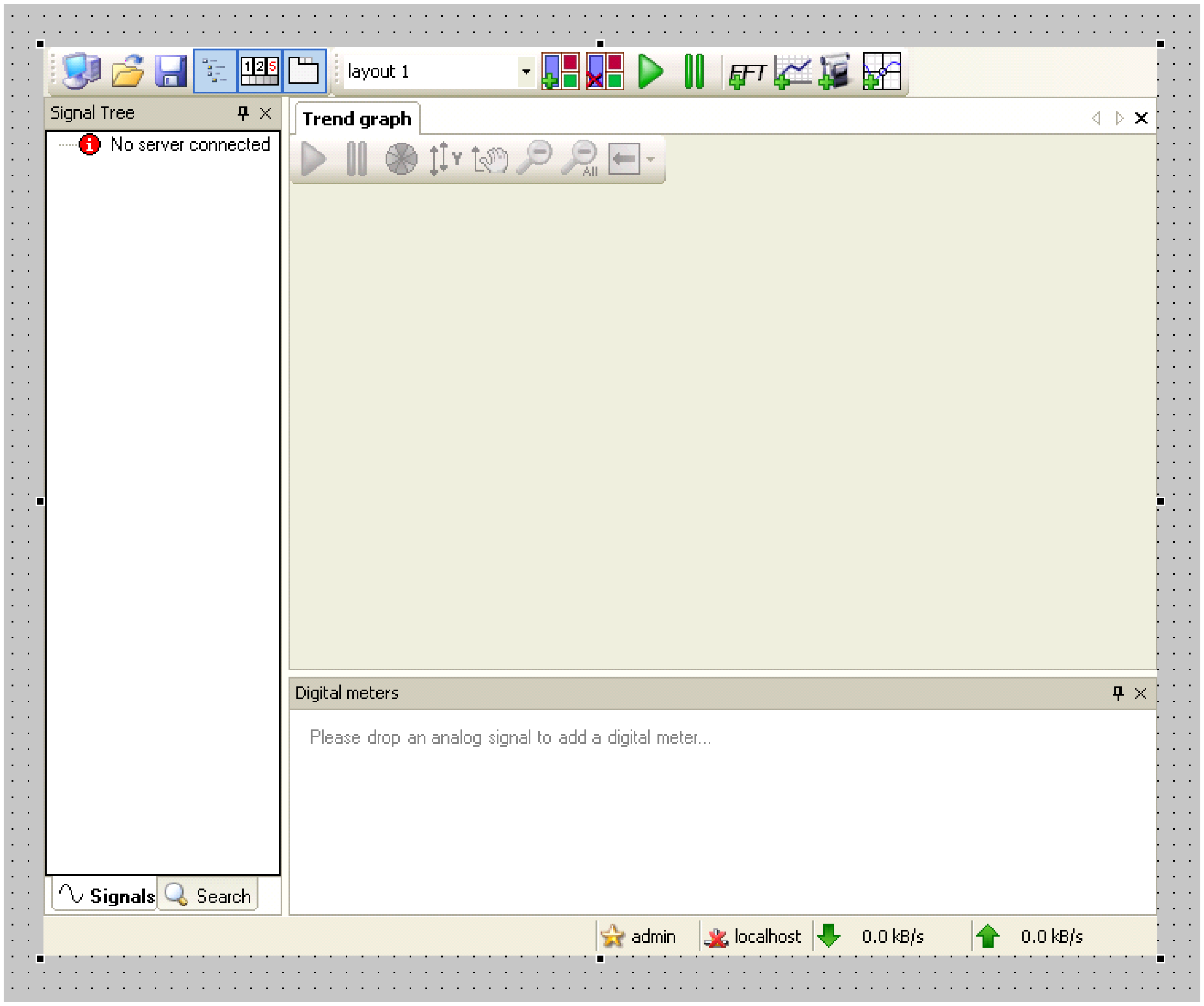


HHCtrl Object
HHCtrl Object
HtmlDlgHelper Class
IBADiagControl Class
ibaPDA Active-X Control
Include Control
InstallEngineCtl Object
IXDiskSpace.DiskSpace
Java Class: com.ms.vid.grid.Grid
Java Class: com.ms.wfc.html.DhComponentWrapper$Dh
Label DTC
LANDesk Date Control
LANDesk DateTime Static
LANDesk Time Control

C:\PROGRA~1\iba\ibaPDA\Client\IBAPDA~1.OCX

ibaPdaActiveXCtrl.1

A new instance of the ibaPDA active-x control will be inserted.

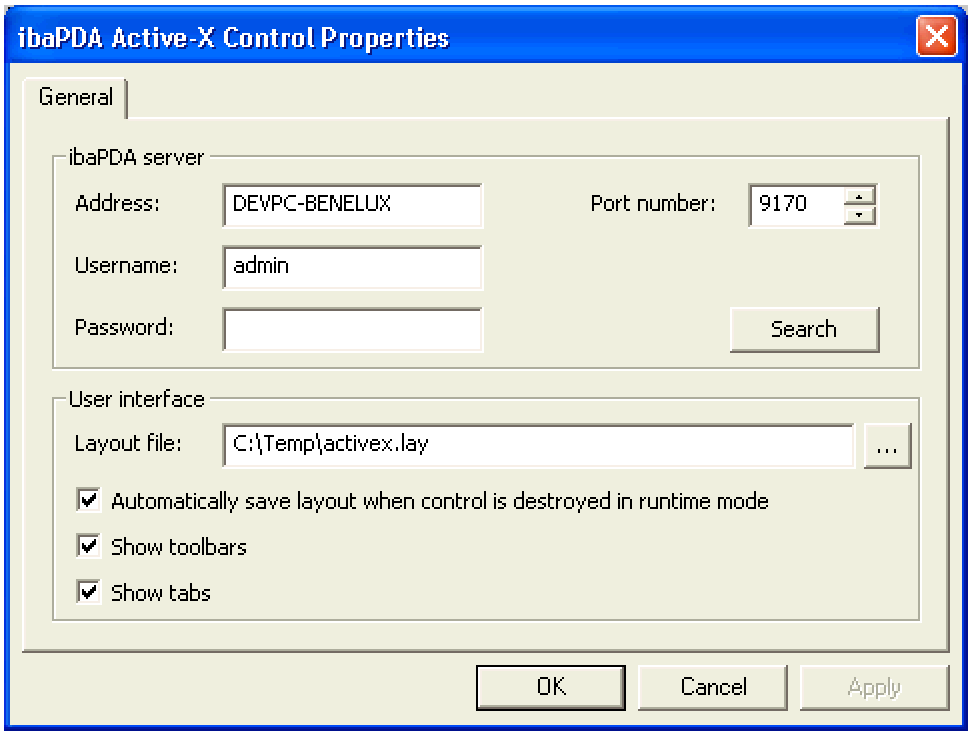


The active-x control looks like the normal standalone client. The active-x control can only be used for visualization. So you can't start/stop the acquisition or configure the I/O manager and data storage.

There are two toolbars: a special active-x toolbar and the standard layouts toolbar. The active-x toolbar contains six buttons:

- Server selection
- Open layout
- Save layout
- Show signaltree
- Show digital meters panel
- Show tabs

An active-x container can be in 2 modes: design mode and runtime mode. Some containers can switch between these modes and others can only be in 1 mode. In the case of WinCC the graphics designer is an active-x container that is always in design mode. The runtime of WinCC is another active-x container and this one is always in runtime mode. In design mode the mouse clicks and keyboard events are captured by the active-x container and they are not forwarded to the active-x control itself. So in design mode you can't interact with the active-x control. You can only set its properties. In WinCC double-click the active-x control to show its properties.


ibaPDA Active-X Control Properties

General

ibaPDA server

Address: DEVPC-BENELUX

Port number: 9170

Username: admin

Password:

User interface

Layout file: C:\Temp\activex.lay

[✓] Automatically save layout when control is destroyed in runtime mode

[✓] Show toolbars

[✓] Show tabs

In the top half you can specify the server you want to connect to. The search button will open the standard server selection form of the ibaPDA client. In runtime mode you can also change the server by clicking the server selection button on the active-x toolbar.

The bottom half contains properties concerning the user interface. You have to specify a layout file. This can be a new file or an already existing file. If you check the *automatically save layout* option then the layout will be saved in the specified file whenever the control is destroyed. This only applies to runtime mode because in design mode you aren't able to interact with the layout anyway. In runtime mode you can also load a different layout by clicking the *load layout button* on the active-x toolbar. The *save layout button* on the active-x toolbar will save the current visible layouts into the layout file specified in the properties. The *show toolbars* option controls the visibility of the active-x toolbar, layout toolbar and the status bar. In runtime mode you can show/hide the toolbars via the F10 function key. The *show tabs* option controls the visibility of the tab headers. In runtime mode you can use the *show tabs* button on the active-x toolbar to change this property.

In design mode the active-x control won't try to connect to the ibaPDA server. In runtime mode the active-x control will automatically try to connect to the ibaPDA server. If there are multiple instances of the active-x control that all connect to the same server with the same username and password then only 1 connection is made to the server. This also means that only 1 client license is required per active-x container and not a license per instance of the active-x control. The same applies for Q-Panel licenses. An active-x container can contain multiple instance of the active-x control. Each instance can connect to a different server.

When you switch to a new page in WinCC then the old page and every control that is on it are destroyed. When an active-x control is destroyed in runtime mode then it doesn't disconnect from the server. The connection is kept alive and also all the signals that were requested by the active-x control will still be requested. When you switch back to the first page then the active-x control is created again and it will use the connection to the server that still exists. It will also be able to show the historical data of the signals for when the active-x control wasn't visible because the data for all its signals was still being requested when the active-x control was removed.

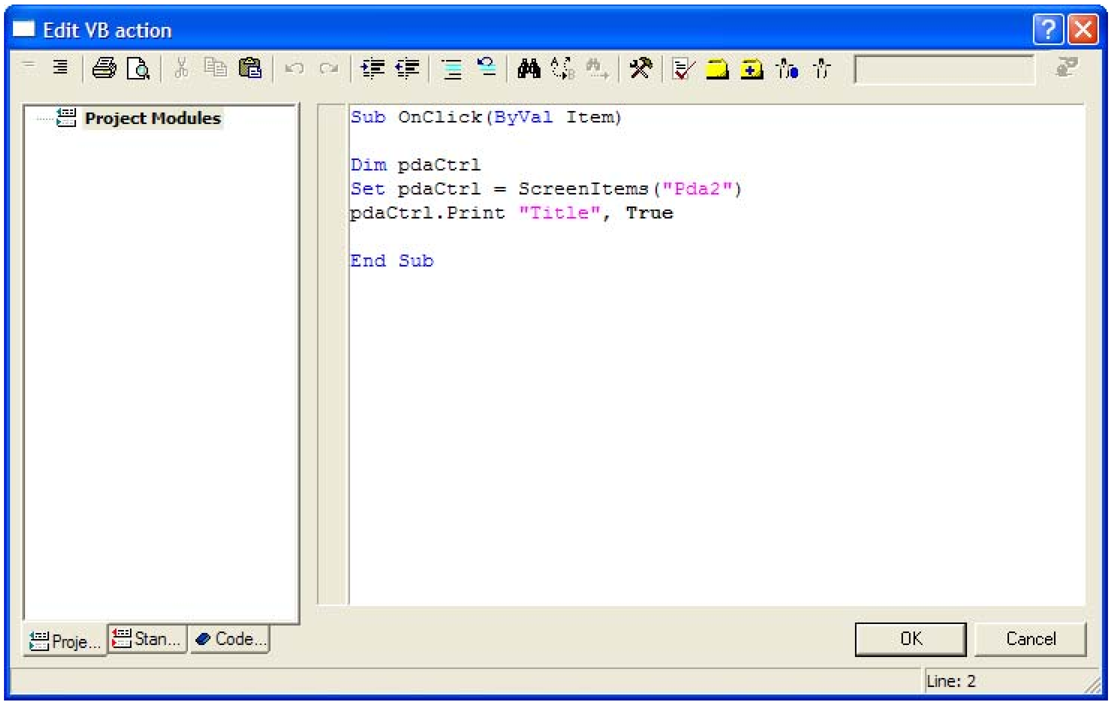
The active-x control can connect to any version of the ibaPDA server. Unlike the normal ibaPDA client the versions don't have to be exactly the same. It is anyway recommended to connect to ibaPDA servers with version 6.21.0 or later.

The active-x control creates log files in the all users application folder, default: c:\documents and settings\all users\Application data\iba\ibaPdaActiveX. It will create a subdirectoy with the name of the active-x container. All active-x control instances within one active-x container will use the same log file.

The active-x control has a *Print* function that can be called from the active-x container. The print function has 2 arguments:

- STRING title: The title for the document
- BOOL showPrintSetup: Set this to true to show the print setup form before printing. The print settings are saved in the same folder as the log files.

In WinCC you can access the active-x control *Print* function via VB-script.



```
Sub OnClick(ByVal Item)

Dim pdaCtrl
Set pdaCtrl = ScreenItems("Pda2")
pdaCtrl.Print "Title", True

End Sub
```

2 Windows Vista and Windows 2008 Server support

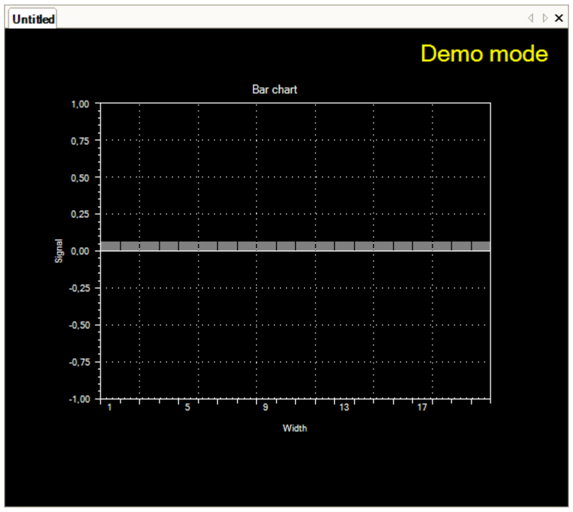
The only difference between the Vista version and the XP version is that the default directory for the client configuration files is the application data folder of the current user instead of the default user.

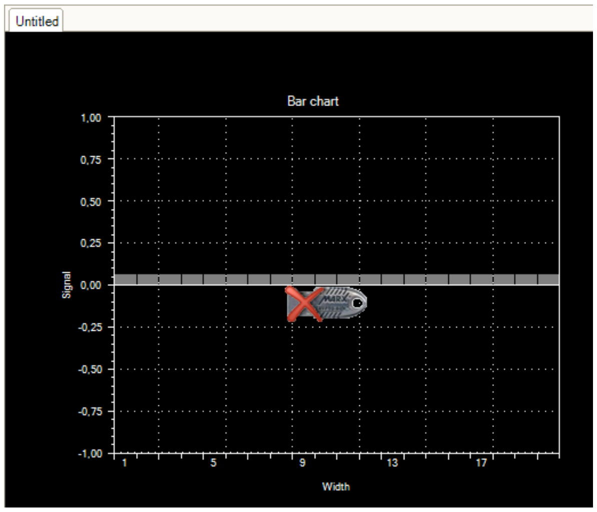
Pda now also supports the large fonts option in windows. The dialogs and controls are scaled automatically so that the texts don’t overlap.

3 ibaQPanel related changes

3.1 ibaQPanel is part of ibaPDA install

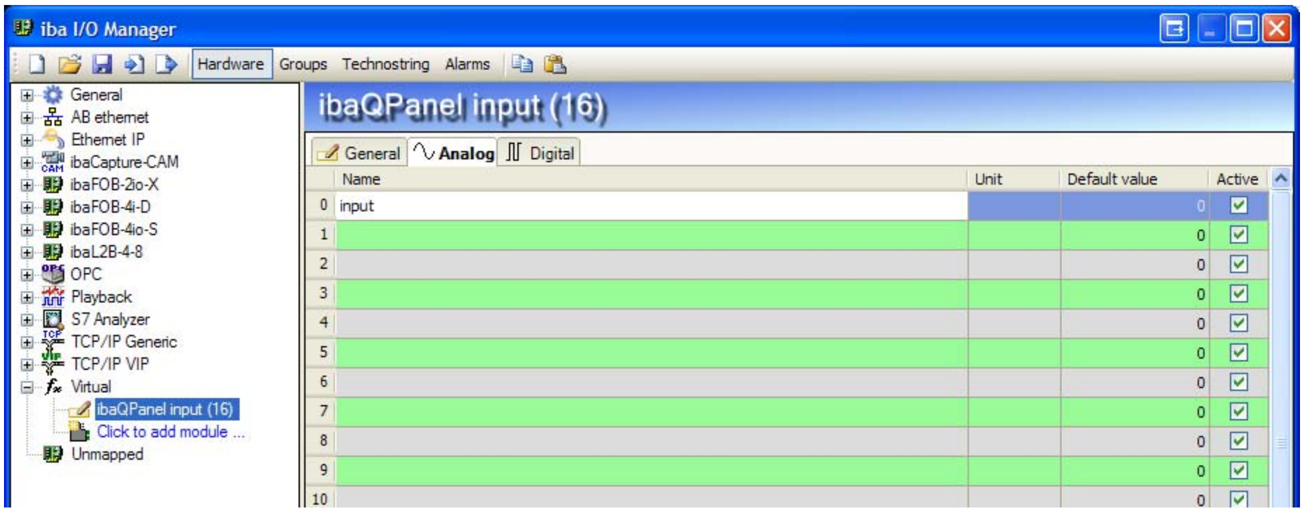
ibaQPanel is now automatically installed together with ibaPDA. If you don't have an ibaQPanel license then the ibaQPanel functionality is limited. You can only create 1 QPanel and you can only use 2 signals in that QPanel. The text "Demo Mode" is painted in the top right corner to indicate that you don't have an ibaQPanel license. If you create more than 1 QPanel or you use more than 2 signals then a dongle image will appear on top of the QPanel(s) and they won't work until you get an ibaQPanel license.





3.2 Input controls

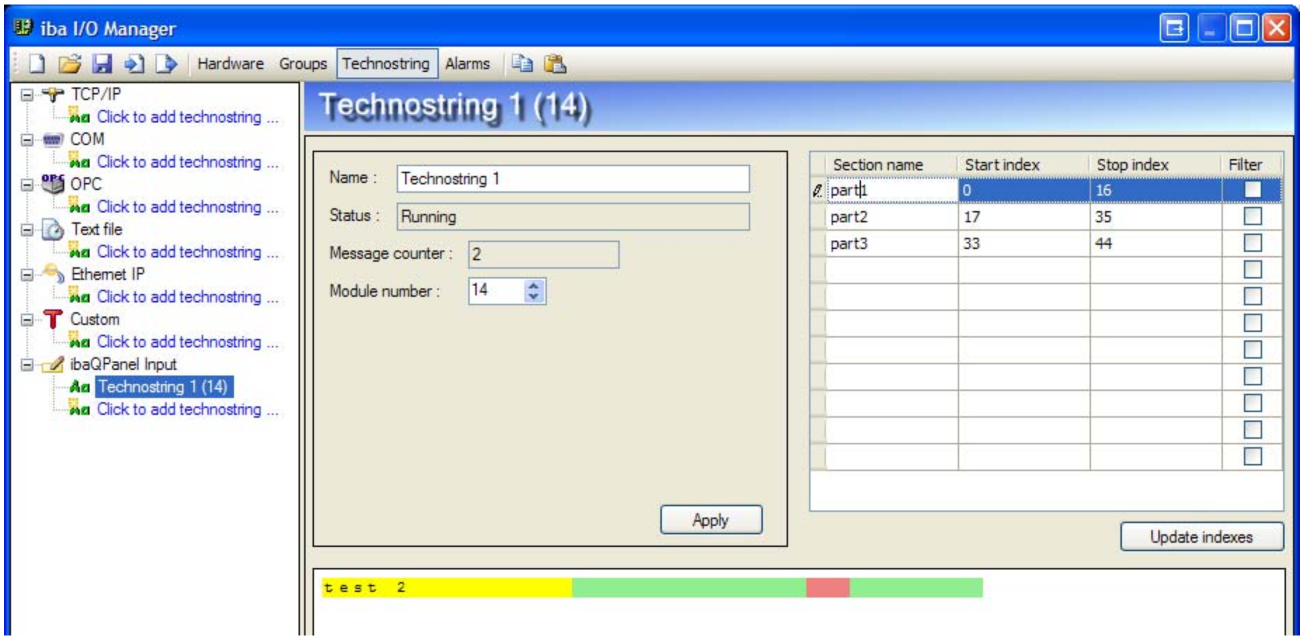
There is a new module type "ibaQPanel input module" that can be added to the virtual interface. The signals of such a module are writeable by the text input control and the command button control within QPanel.



	Name	Unit	Default value	Active
0	input		0	✓
1			0	✓
2			0	✓
3			0	✓
4			0	✓
5			0	✓
6			0	✓
7			0	✓
8			0	✓
9			0	✓
10			0	✓

The default value of the signals can be specified. The signal will have the default value each time the acquisition is started.

There is also a new technostring type called “ibaQPanel technostring”. The sections of such a technostring are writeable by the text input control within QPanel.


iba I/O Manager
Hardware
Groups
Technostring
Alarms
TCP/IP
Click to add technostring ...
COM
Click to add technostring ...
OPC
Click to add technostring ...
Text file
Click to add technostring ...
Ethernet IP
Click to add technostring ...
Custom
Click to add technostring ...
ibaQPanel Input
Technostring 1 (14)
Click to add technostring ...
Technostring 1 (14)
Apply
Update indexes
t e s t   2

Name :	Technostring 1
Status :	Running
Message counter :	2
Module number :	14

Section name	Start index	Stop index	Filter
part1	0	16	☐
part2	17	35	☐
part3	33	44	☐

The technostring preview control has been changed slightly in ibaPDA v6.21.0. It now shows a green background for all the sections. The overlapping parts of 2 sections have a red background. The currently selected section has a yellow background. This makes it easier to see which parts of the technostring have been marked as sections.

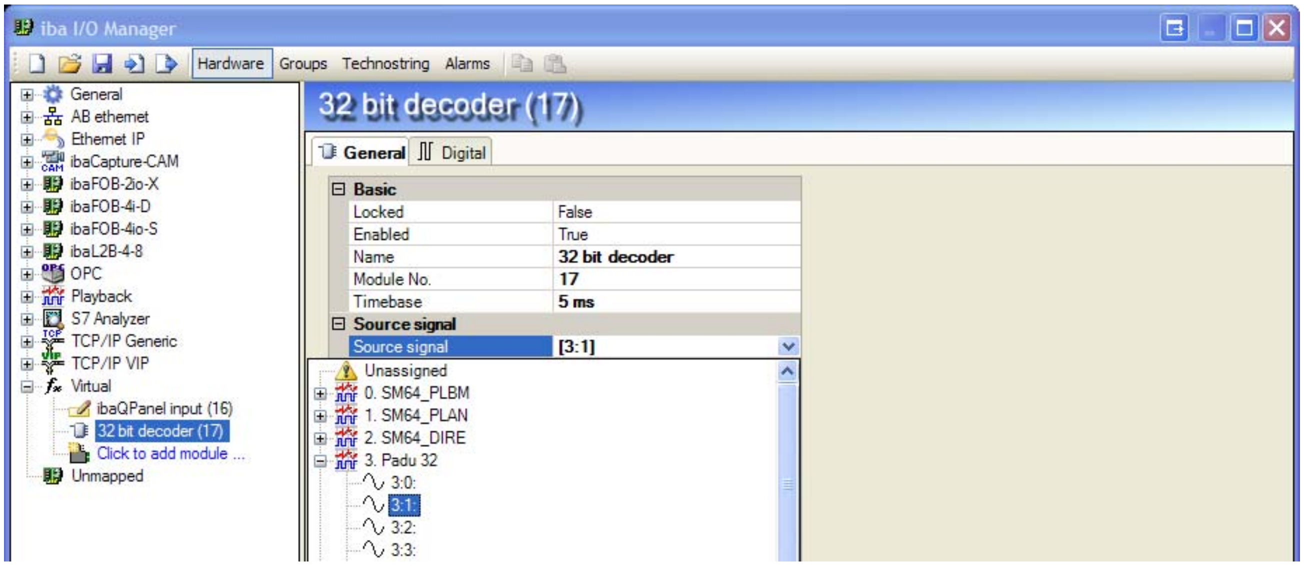
Check out the updated ibaQPanel document for more information about the text input control and command button control.

3.3 Bar chart updates

Check out the updated ibaQPanel document for more information about the new bar chart features.

4 New module types

4.1 32 bit decoder



Basic	
Locked	False
Enabled	True
Name	32 bit decoder
Module No.	17
Timebase	5 ms
Source signal	
Source signal	[3:1]

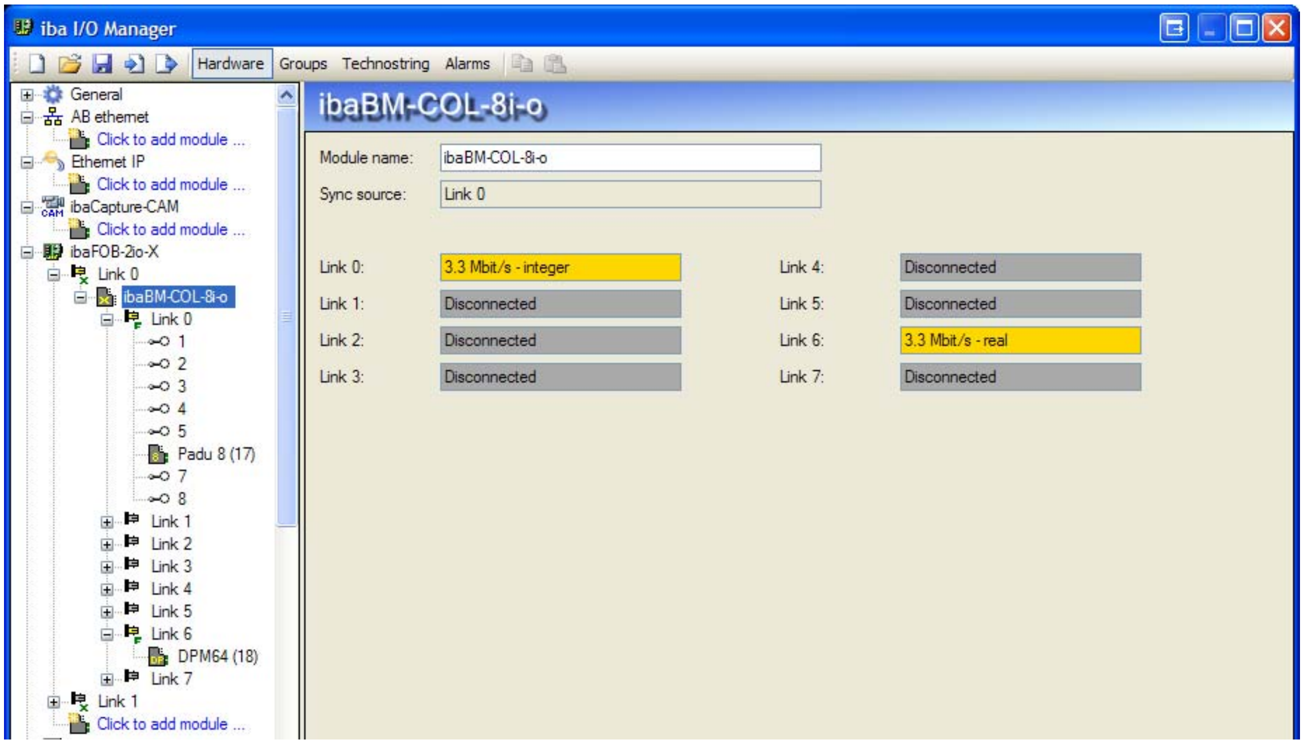
The 32 bit decoder module is similar to the 16 bit decoder module. The behavior of this module depends on the datatype of the source signal. If the datatype is floating-point then the value is treated as a bitmask and then the bits are extracted. This is the same behavior as the GetFloatBit function in ibaPDA and the GetBitMask function in ibaAnalyzer. If the datatype is something else than floating-point then the value is first rounded to the nearest integer and then the bits are extracted. This is the same behavior as the GetIntBit function in ibaPDA and the GetBit function in ibaAnalyzer.

Certain interfaces automatically convert 32 bit integer values into floats during the data transfer between source system and ibaPDA. If this is the case for the source signal then the original bit pattern will be broken. A warning is generated during the I/O configuration validation in this situation. Currently the affected interfaces are:

- OPC interface
- Simolink module
- HPCi request interface
- Simadyn interface
- TDC interface
- Paco4 module
- X-Pact interface
- Playback interface

4.2 ibaBM-COL-8i-o

The ibaBM-COL-8i-o is a device with 8 3.3Mbit/s or 2Mbit/s input links and 1 32Mbit/s output link. The data of the 8 input links is put on the single output link. This device can be connected to a FOB-X or a FOB-D board. The autodetect functionality in ibaPDA will detect the ibaBM-COL-8i-o module and also all modules connected to its input links. Any 3.3Mbit/s module that can be connected to a FOB link can be connected to an ibaBM-COL-8i-o input link.



Module name: ibaBM-COL-8i-o

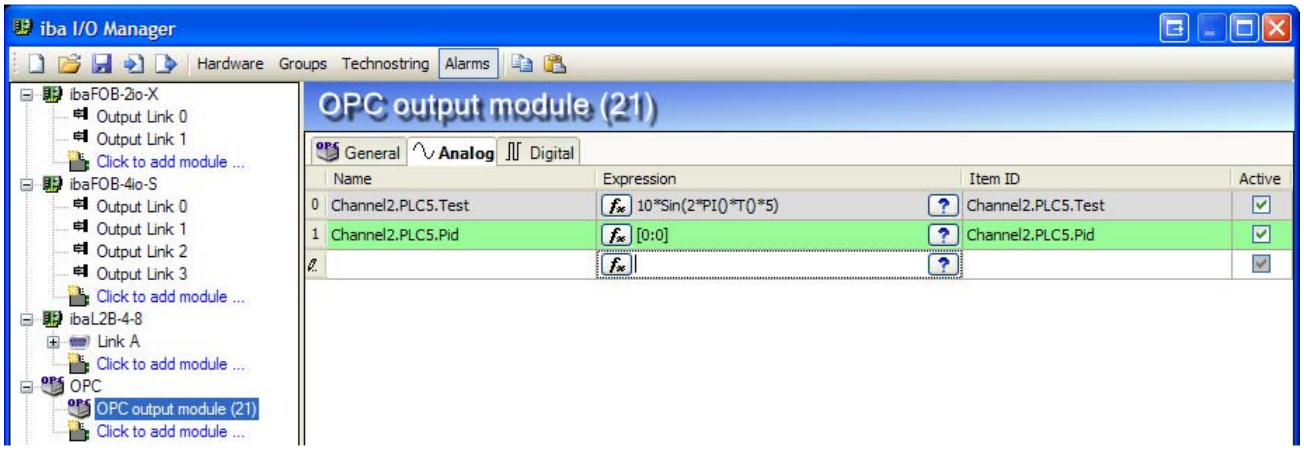
Sync source: Link 0

Link 0:	3.3 Mbit/s - integer	Link 4:	Disconnected
Link 1:	Disconnected	Link 5:	Disconnected
Link 2:	Disconnected	Link 6:	3.3 Mbit/s - real
Link 3:	Disconnected	Link 7:	Disconnected

On the device itself you can only specify a name. The diagnostics show the current link modes of all the 8 input links. The sync source shows which link is used as clock for the output link. Normally this is the first input link that has something connected to it.

4.3 OPC output module

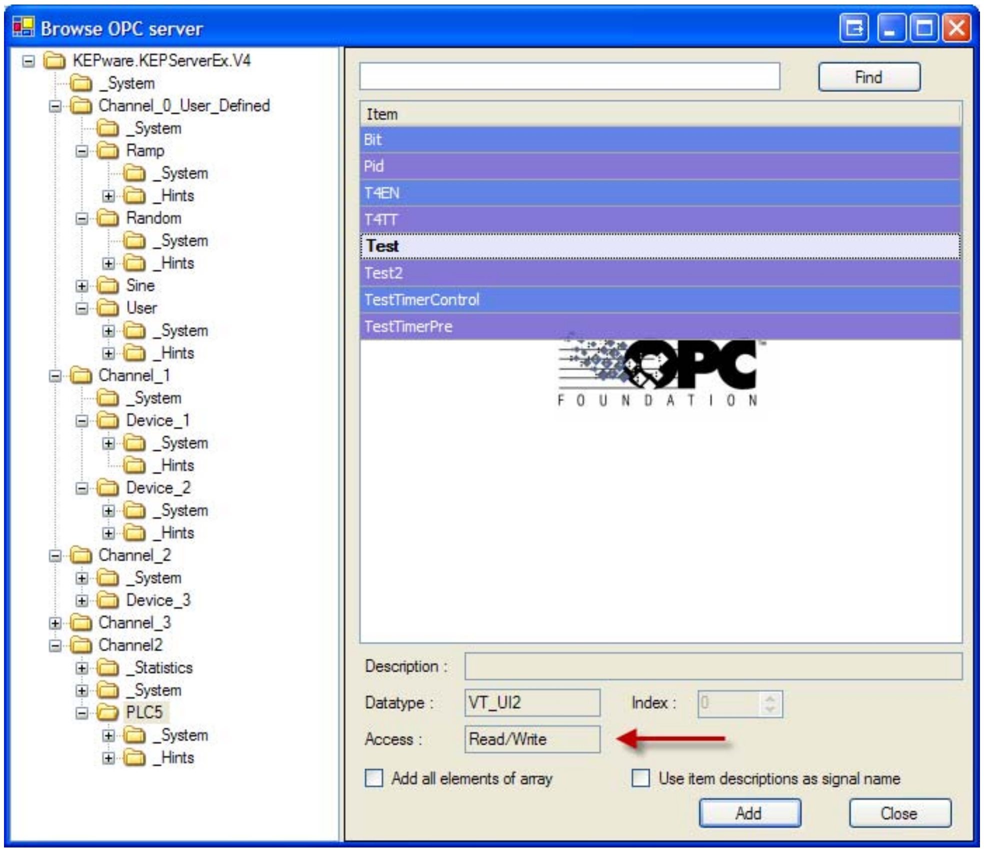
The OPC output module connects to an OPC server and writes values to writeable OPC items.



	Name	Expression	Item ID	Active
0	Channel2.PLC5.Test	10*Sin(2*PI()*T()*5)	Channel2.PLC5.Test	✓
1	Channel2.PLC5.Pid	[0:0]	Channel2.PLC5.Pid	✓
				✓

For each signal you have to select the OPC item ID you want to write to. You also have to specify an expression that calculates the value you want to write. Like with all alarm modules the maximum write rate is 50 ms.

The OPC browser has been extended with the “Access” field. It shows if an OPC item is readable and/or writeable.

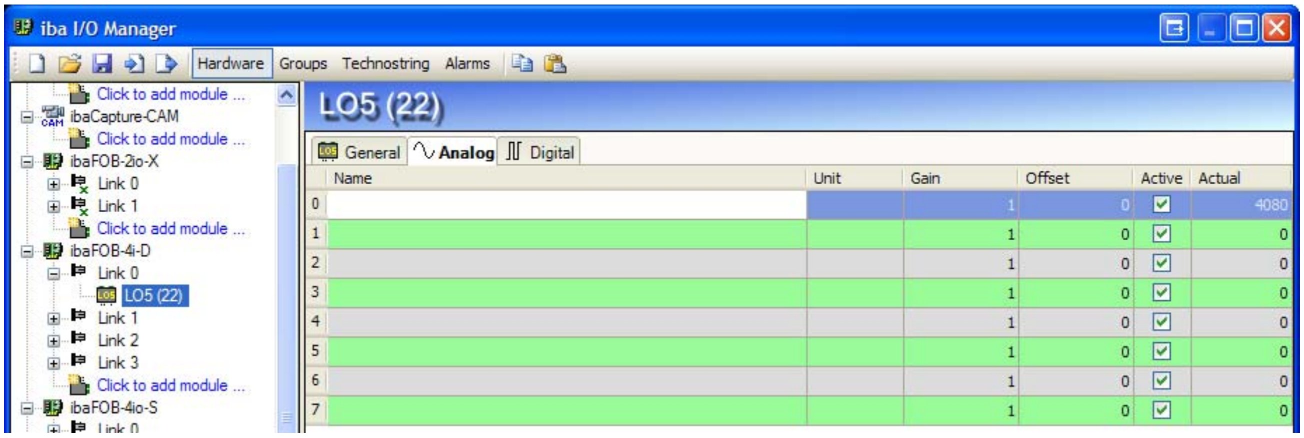


Description :

Datatype : VT_UI2 Index : 0

Access : Read/Write

4.4 FOB LO5 module

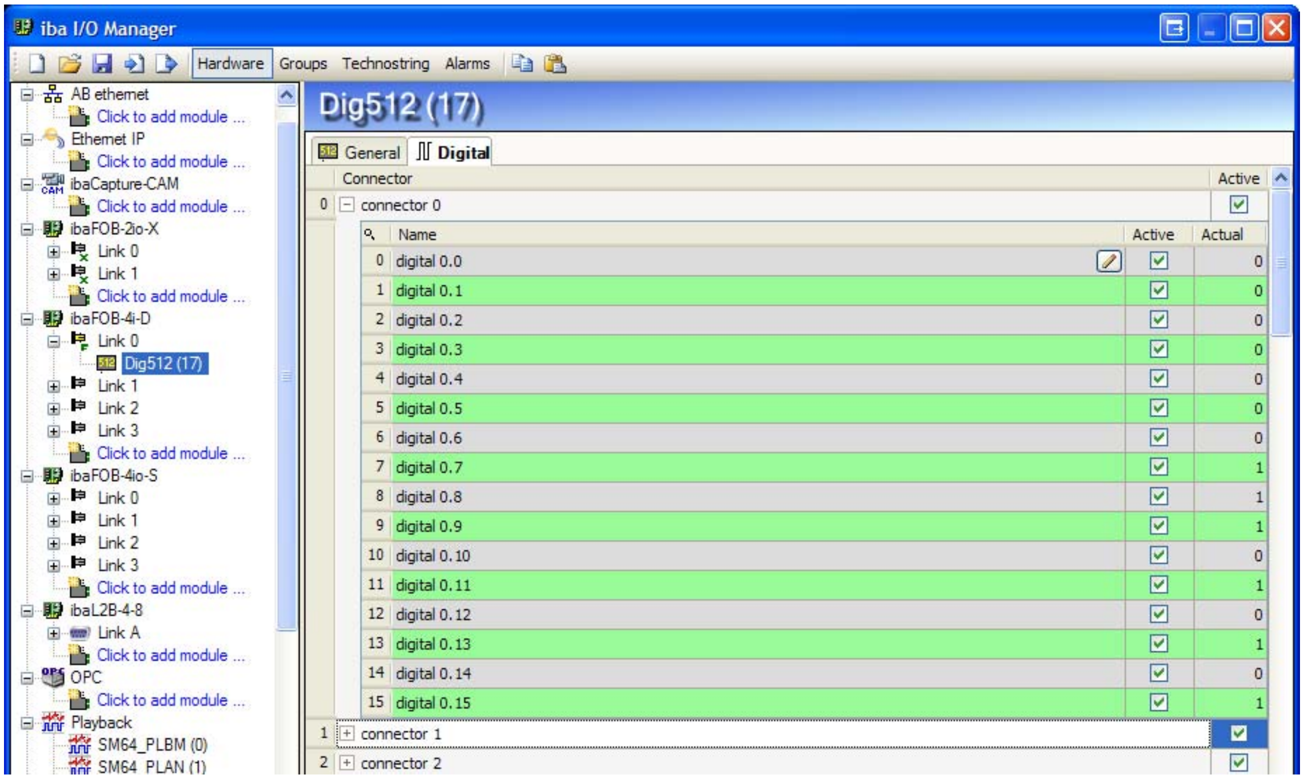


	Name	Unit	Gain	Offset	Active	Actual
0			1	0	✓	4080
1			1	0	✓	0
2			1	0	✓	0
3			1	0	✓	0
4			1	0	✓	0
5			1	0	✓	0
6			1	0	✓	0
7			1	0	✓	0

Siemens has an LO5 board that fits into a TDC rack. This board sends a telegram with 8 integers and 8 bits every 156 microseconds on a 5MBit/s FO link. This board can be connected to a FOB-D board in the PC. In ibaPDA this board is handled by the LO5 module.

4.5 FOB Dig512 module

A long time ago iba has developed a device that has 512 digital inputs. It sends this data on a 2MBit/s FO link. ibaPDA now supports this device via the FOB Dig512 module. It can be connected to a FOB-S, FOB-X and FOB-D board. ibaPDA v5 also supported this module. The ibaPDA v5 configuration for this module can be imported into ibaPDA v6.

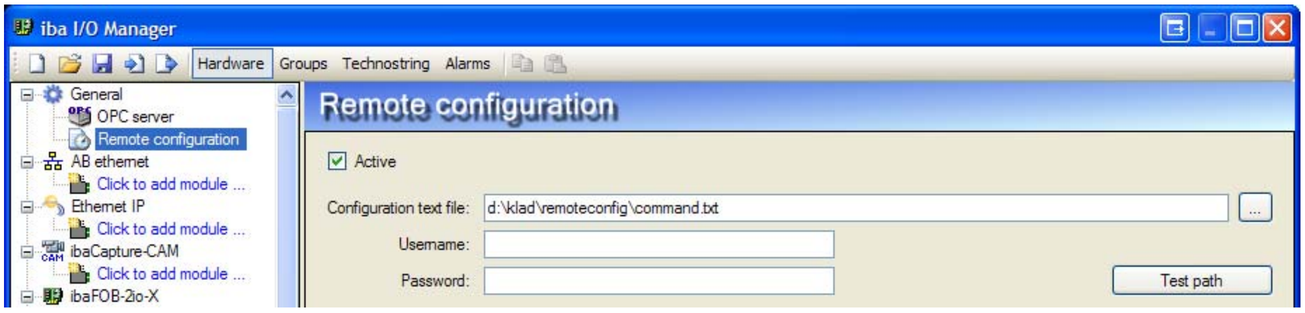


	Connector	Active
0	connector 0	✓

	Name	Active	Actual
0	digital 0.0	✓	0
1	digital 0.1	✓	0
2	digital 0.2	✓	0
3	digital 0.3	✓	0
4	digital 0.4	✓	0
5	digital 0.5	✓	0
6	digital 0.6	✓	0
7	digital 0.7	✓	1
8	digital 0.8	✓	1
9	digital 0.9	✓	1
10	digital 0.10	✓	0
11	digital 0.11	✓	1
12	digital 0.12	✓	0
13	digital 0.13	✓	1
14	digital 0.14	✓	0
15	digital 0.15	✓	1

	Connector	Active
1	connector 1	✓
2	connector 2	✓

5 Remote configuration changes



☑ Active

Configuration text file: d:\klad\remoteconfig\command.txt

Username:

Password:

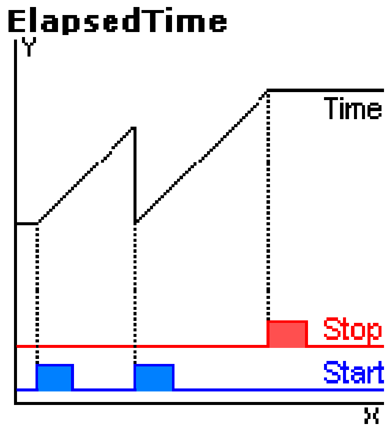
The remote configuration feature periodically checks if the configured file is available. If the file is available then it is processed and afterwards deleted. The processing depends on the extension of the file.

- **.zip file**: ibaPDA tries to open this file as a project file. If it is successful then the new project is loaded and the acquisition is started.
- **.io file**: ibaPDA tries to load this file as a new I/O configuration. If it is successful then the new I/O configuration is applied and the acquisition is started.
- **.ds file**: ibaPDA tries to load this file as a new datastore configuration. If it is successful then the new datastore configuration is applied.
- **other files**: ibaPDA interprets this file as a text file. It reads the first line. This line can contain one of the following commands:
    - **STOP**: ibaPDA stops the acquisition.
    - **RESUME**: ibaPDA starts the acquisition.
    - **EXPORT**: ibaPDA exports the current I/O configuration to the file specified on the rest of the first line

  If none of these commands are on the first line then ibaPDA tries to import this file as a new I/O configuration. If it is successful then the new I/O configuration is applied and the acquisition is started.

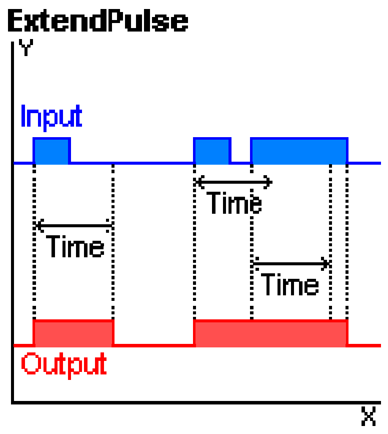
6 New functions

6.1 ElapsedTime



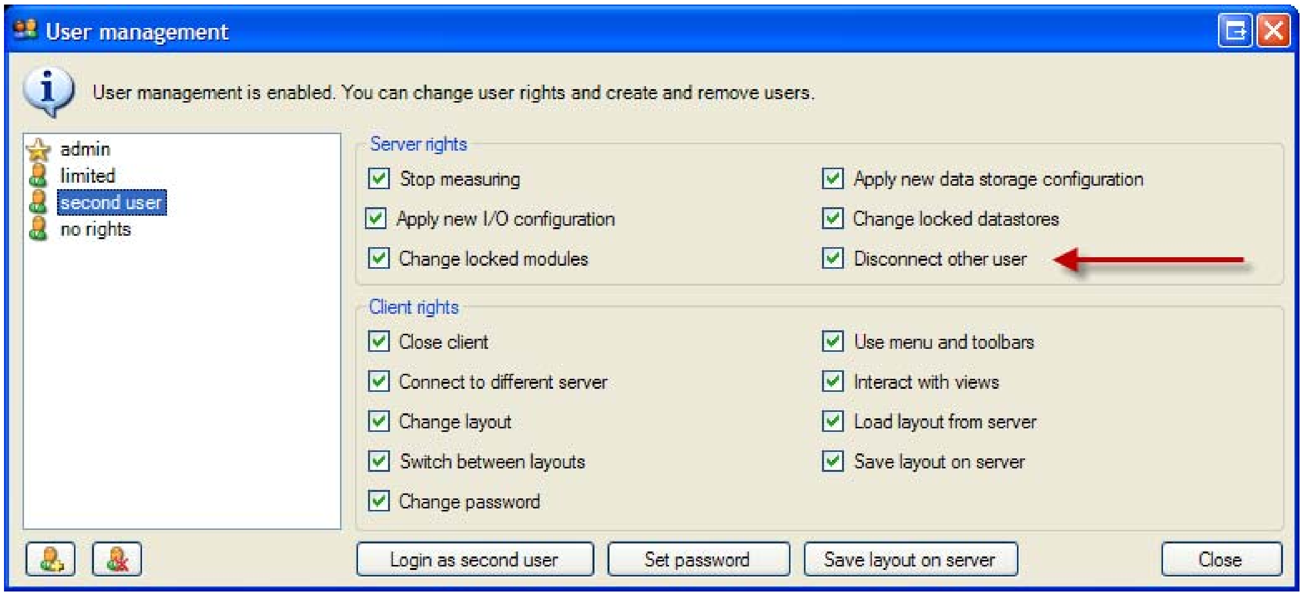
ElapsedTime('start', 'stop'). This function returns the time since the last rising edge on 'start'. The time is stopped when there is a rising edge on 'stop'.

6.2 ExtendPulse



ExtendPulse('input', 'time'). This function extends a pulse on 'input' so that it is at least 'time' seconds long. A rising edge on 'input' will restart the time.

7 “Disconnect other client” right


User management

User management is enabled. You can change user rights and create and remove users.

- admin
- limited
- second user
- no rights

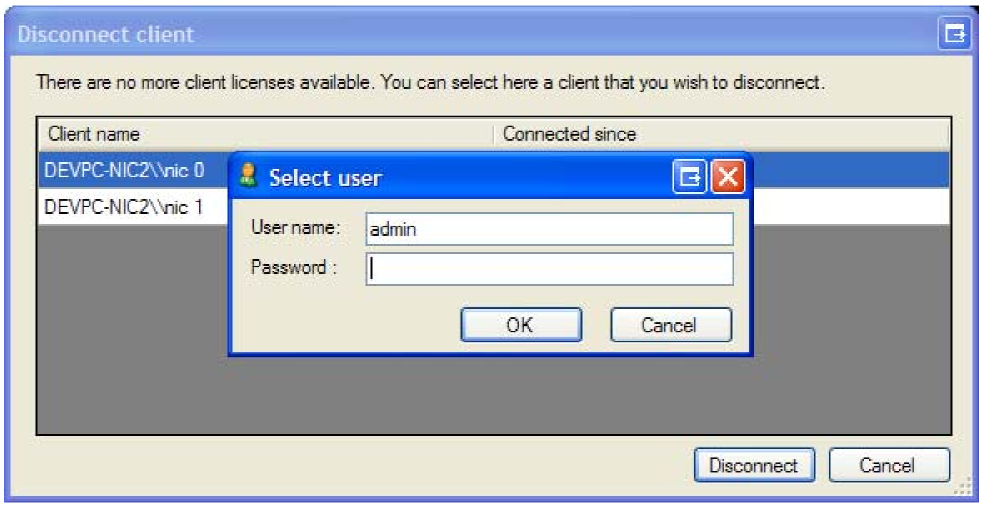
Server rights

- ✓ Stop measuring
- ✓ Apply new data storage configuration
- ✓ Apply new I/O configuration
- ✓ Change locked datastores
- ✓ Change locked modules
- ✓ Disconnect other user

Client rights

- ✓ Close client
- ✓ Use menu and toolbars
- ✓ Connect to different server
- ✓ Interact with views
- ✓ Change layout
- ✓ Load layout from server
- ✓ Switch between layouts
- ✓ Save layout on server
- ✓ Change password

There is a new server right: disconnect other user. If a user tries to connect to an ibaPDA server that has no more free client licenses left then ibaPDA ask if you wish to disconnect another client.


Disconnect client

There are no more client licenses available. You can select here a client that you wish to disconnect.

Client name	Connected since
DEVPC-NIC2\\nic 0	
DEVPC-NIC2\\nic 1	

Select user

User name: admin

Password :

In previous version this required the administrator password. Now you can enter the credentials of a user with the “disconnect other user” right.

8 Updated binary watchdog

The binary watchdog now also contains the status of the ibaCapture-HMI and ibaCapture-CAM connections. Here is the C-structure of the watchdog message.

```
#define N_MAX_DATASTORES      16

typedef struct _PDA_DATASTORE_INFO
{
  unsigned short status;       // (0) Status of the datastore:
                               //       =0: Inactive
                               //       =1: Waiting for trigger
                               //       =2: Recording
                               //       =3: Post-trigger
  unsigned short directory;    // (2) Current directory:
                               //       =0: Pda is writing to the base directory
                               //       =1: Pda is writing to the backup directory
  unsigned int   freeSpace;    // (4) Free space on the harddisk in MB (only available if
datastore is active)

  unsigned char  reserved[4];  // (8)

} PDA_DATASTORE_INFO, *PPDA_DATASTORE_INFO;

typedef struct _QDR_DATASTORE_INFO
{
  unsigned short status;       // (0) Status of the QDR datastore:
                               //       =0: Inactive
                               //       =1: Not synchronized
                               //       =2: Synchronized
  unsigned short directory;    // (2) Current directory:
                               //       =0: Pda is writing to the base directory
                               //       =1: Pda is writing to the backup directory
  unsigned int   freeSpace;    // (4) Free space on the harddisk in MB (only available if
datastore is active)

  unsigned char  reserved[4];  // (8)

} QDR_DATASTORE_INFO, *PQDR_DATASTORE_INFO;

typedef struct _PDA_WATCHDOG
{
  unsigned int  msgCounter;              // (0)  Message counter is incremented after each message
  unsigned int  msgVersion;              // (4)  Version number (currently = 1)

  unsigned int  isMeasuring;             // (8)  =1: Pda is measuring

  unsigned char reserved[4];             // (12)

  unsigned int  captureCamStatus[2];     // (16) Bit per connection to ibaCapture-CAM server
                                         //      bit = 1: connection ok
  unsigned int  captureHmiStatus[2];     // (24) Bit per connection to ibaCapture-HMI server
                                         //      bit = 1: connection ok

  QDR_DATASTORE_INFO qdrDatastore;       // (32)   Information about the QDR datastore
  PDA_DATASTORE_INFO datastores[N_MAX_DATASTORES];    // (44) Information about all datastores

} PDA_WATCHDOG, *PPDA_WATCHDOG;
```

The captureCamStatus and captureHmiStatus fields are new compared to previous versions. In previous versions these fields were reserved so the change is backwards compatible.