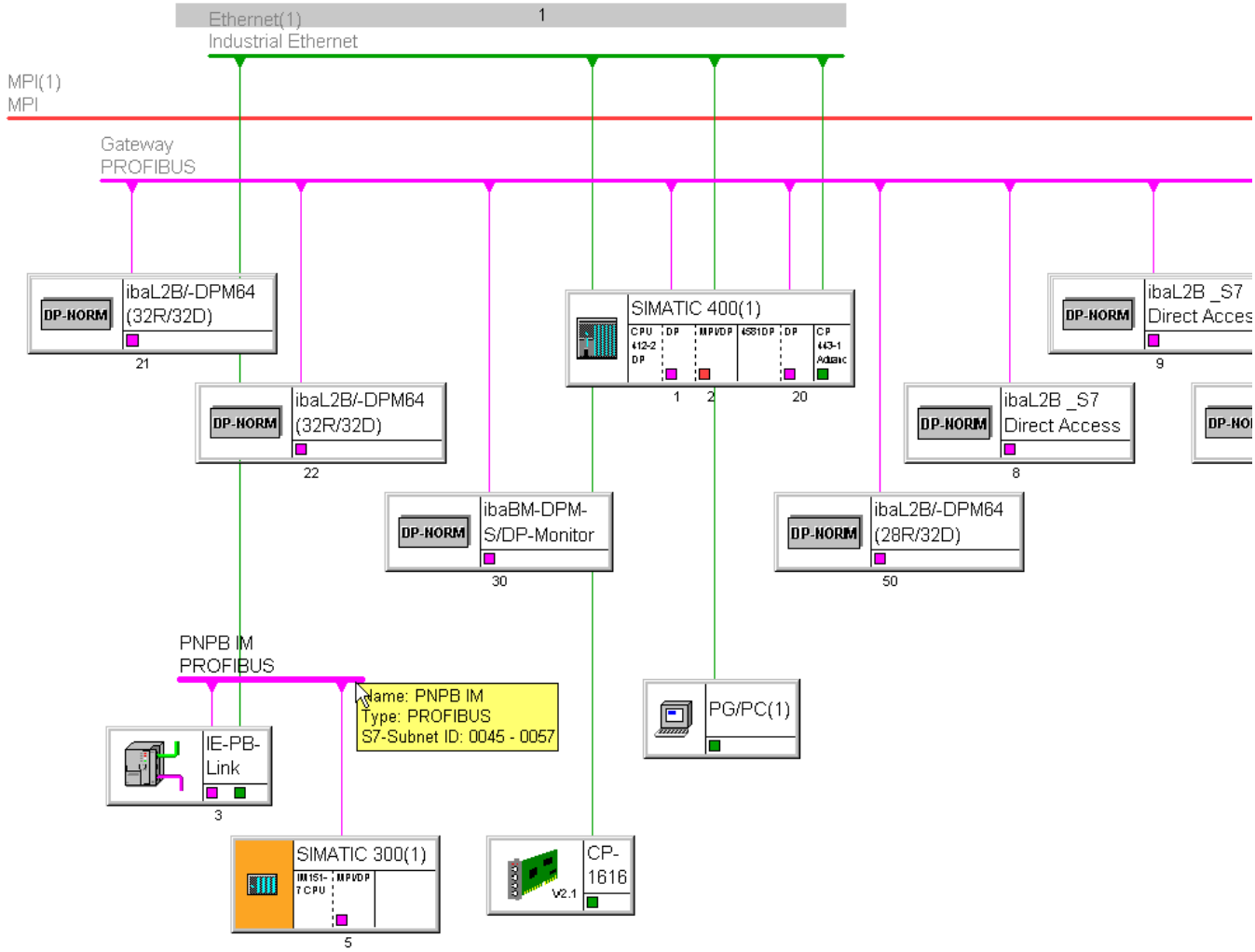
New features in ibaPDA v6.28.0

1 S7 changes

1.1 S7 routing

According to Siemens S7 routing is the ability to transfer data beyond you own network boundaries. It means that a PG can exchange data with a PLC that is not directly connected to the PG but that is connected via one or more gateways. The gateway can be another PLC or a protocol converter like the IE/PB link.

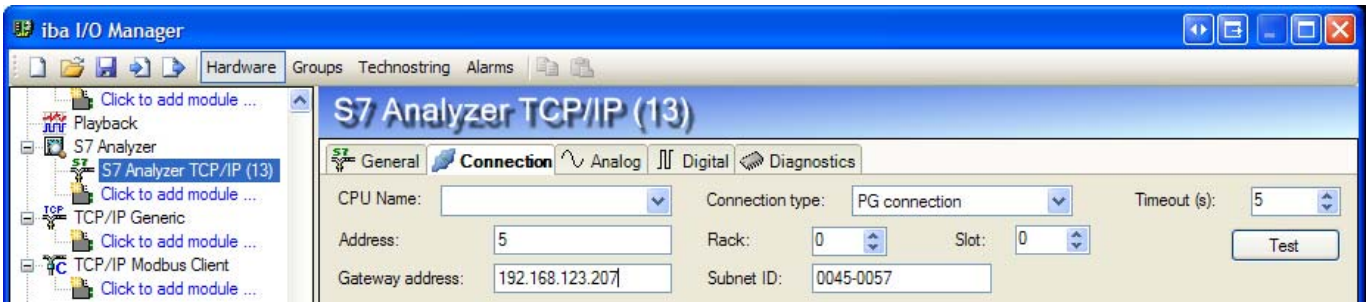
In the office we have tested the S7 routing with the pda PC connected to an IE/PB-Link device via ethernet. The IE/PB link is connected via profibus to an IM151-7 (which is in principle an S7-300 CPU). The following screenshot shows the NetPro configuration for this setup.



The PG is connected to the industrial Ethernet network. The IE/PB link has IP address 192.168.123.207. The target CPU IM151-7 is connected to the profibus PNPB IM and has profibus address 5. The profibus network itself has a subnet ID 0045-0057.

In S7 analyzer we have to specify the address, rack and slot fields for the actual target CPU so for the IM151-7 CPU. We enter profibus address 5 and rack 0 and slot 0. If we would be connected to the profibus directly via e.g. a CP5613 board then that is all we needed to fill in. But now we are not connected directly but via the IE/PB link via Ethernet. The IE/PB link acts as our gateway. We are connected via Ethernet so we have to fill in the IP address of the IE/PB

link. Next we have to enter the subnet ID of the network that the target CPU is connected to. This is 0045-0057 in our case.



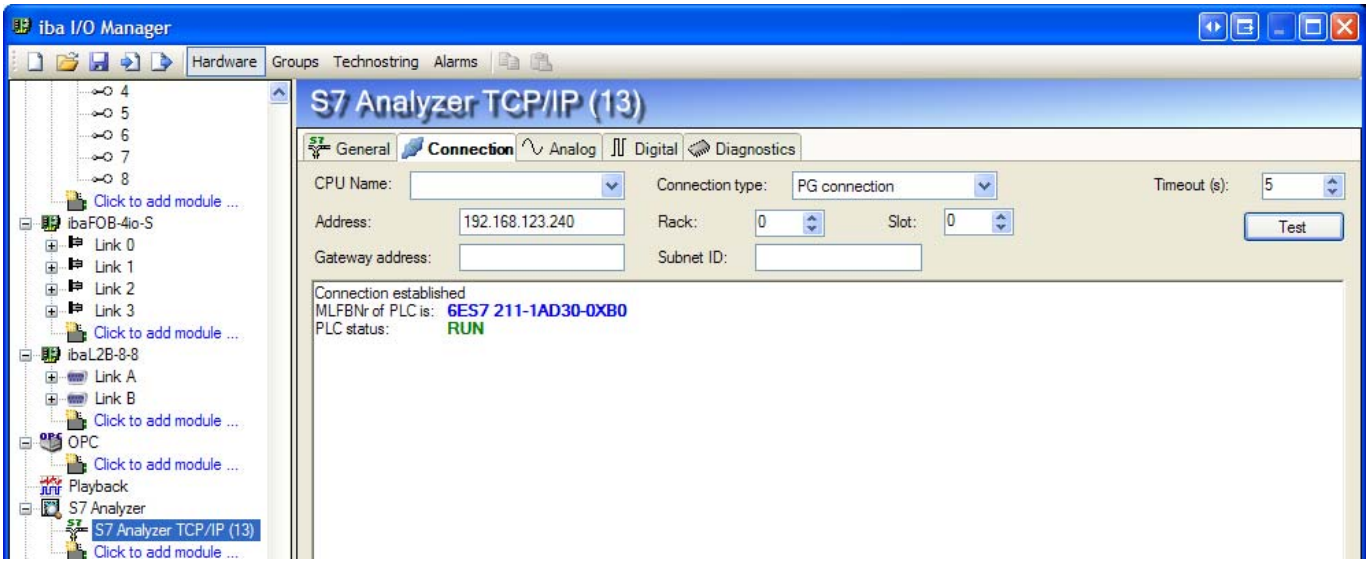
S7 routing works also in case you need to make more than 1 network transition to reach the target. In S7 analyzer you always have to specify the gateway address of the first gateway and the subnet ID of the network the target is connected to. The route in between is automatically determined by the PLCs and CPs in the path.

You can use S7 routing in the following module types:

- S7 analyzer TCP/IP
- S7 analyzer PC/CP
- S7 request via DPM-S
- FM458 request via DPM-S
- FM458 request via L2B
- TDC request via DPM-S
- TDC request via L2B

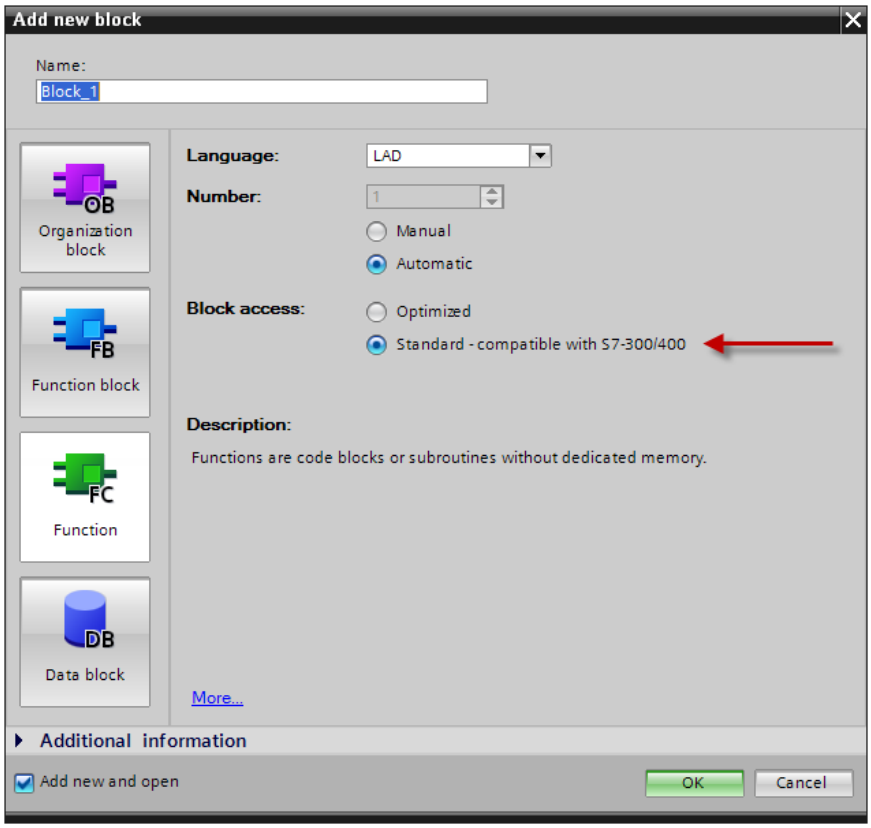
1.2 S7-1200

The S7-1200 family of PLCs can be accessed via S7 analyzer. The standard TCP/IP and PC/CP modules now work for S7-300, S7-400 and S7-1200. Pda will automatically detect which CPU type is connected. There are some limitations for the S7-1200. The S7-1200 can't return cycle times, the number of DBs, Markers and other statistics. So when you test the connection to an S7-1200 CPU you will only see his MLFB number and his working status.

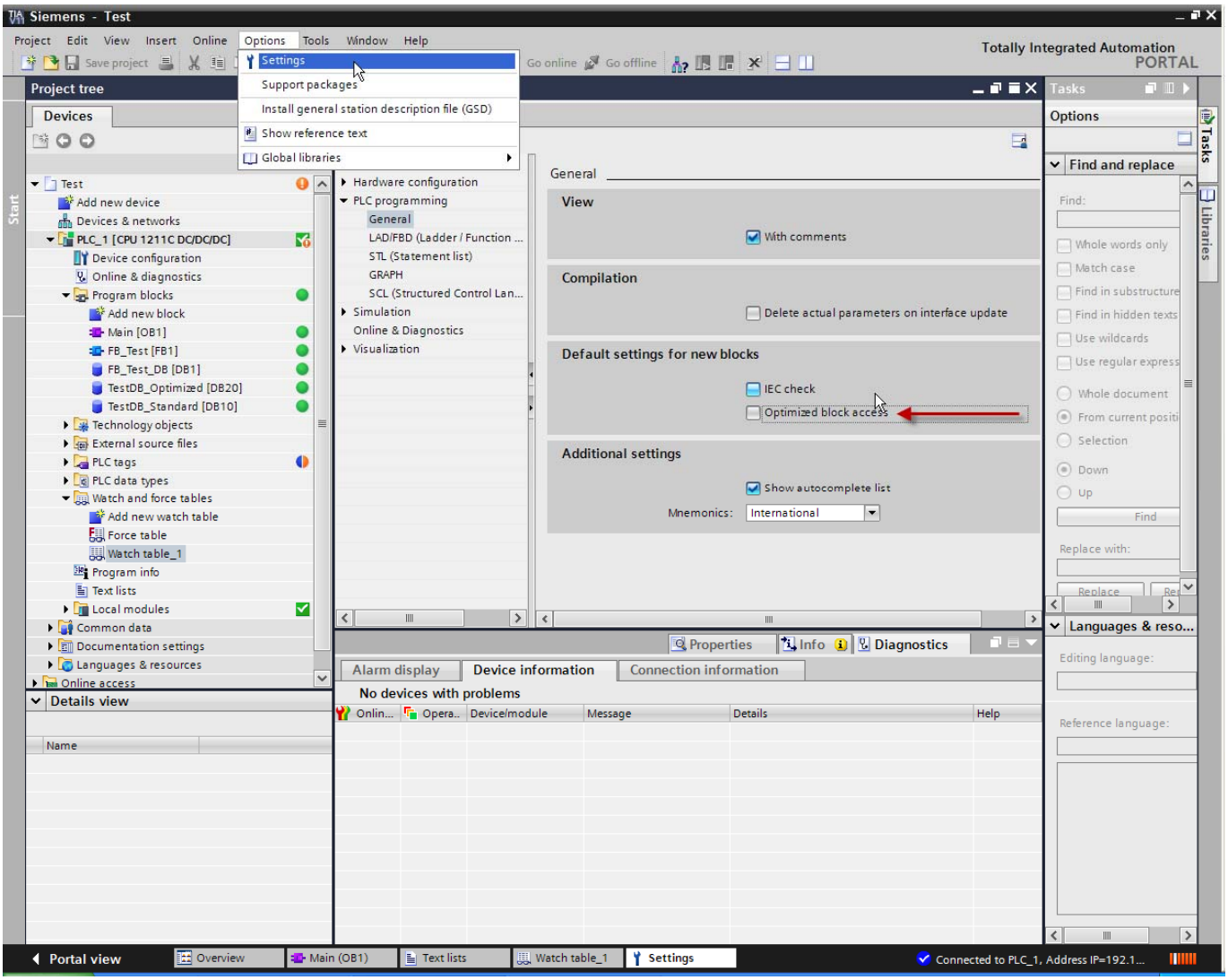


The S7-1200 is not programmed via Step7 but via the new TIA portal of Siemens. This new project type can't be read by the S7 address book generator so there is no symbolic request possible. You can only use operands to read data from an S7-1200. The S7-1200 has 2 types of DB:

- The DB with standard access which is the same as the DBs in S7-300 and S7-400
- The DB with optimized access



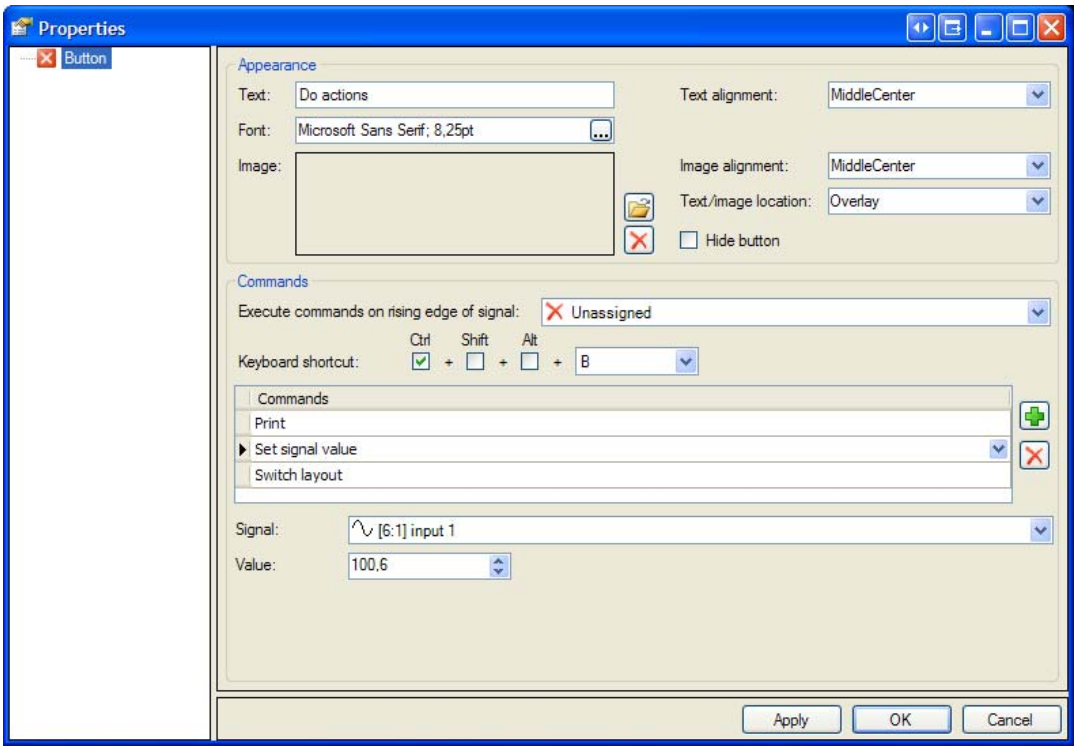
The data for the DB with optimized access can only be retrieved by name and not by operand. So since we don't have symbolic information S7 analyzer can't read DBs with optimized access. If you want to use S7 analyzer with the S7-1200 then you will need to use DBs with standard access. You can set the default access mode via Options -> Settings -> PLC programming -> General.


Siemens - Test
Project Edit View Insert Online Options Tools Window Help
Settings
Support packages
Install general station description file (GSD)
Show reference text
Global libraries
Totally Integrated Automation
PORTAL
Project tree
Devices
Test
Add new device
Devices & networks
PLC_1 [CPU 1211C DC/DC/DC]
Device configuration
Online & diagnostics
Program blocks
Add new block
Main [OB1]
FB_Test [FB1]
FB_Test_DB [DB1]
TestDB_Optimized [DB20]
TestDB_Standard [DB10]
Technology objects
External source files
PLC tags
PLC data types
Watch and force tables
Add new watch table
Force table
Watch table_1
Program info
Text lists
Local modules
Common data
Documentation settings
Languages & resources
Online access
Details view
Name
Hardware configuration
PLC programming
General
LAD/FBD (Ladder / Function ...
STL (Statement list)
GRAPH
SCL (Structured Control Lan...
Simulation
Online & Diagnostics
Visualization
General
View
With comments
Compilation
Delete actual parameters on interface update
Default settings for new blocks
IEC check
Optimized block access
Additional settings
Show autocomplete list
Mnemonics:
International
Properties
Info
Diagnostics
Alarm display
Device information
Connection information
No devices with problems
Onlin...
Opera..
Device/module
Message
Details
Help
Tasks
Options
Find and replace
Find:
Whole words only
Match case
Find in substructure
Find in hidden texts
Use wildcards
Use regular express
Whole document
From current positi
Selection
Down
Up
Find
Replace with:
Replace
Languages & reso...
Editing language:
Reference language:
Libraries
Start
Portal view
Overview
Main (OB1)
Text lists
Watch table_1
Settings
Connected to PLC_1, Address IP=192.1...

2 ibaHD event store

See separate manual ibaHD Event Store.docx

3 QPanel button changes

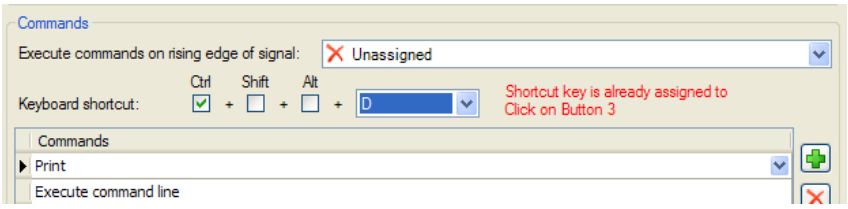


A button can now execute multiple commands when it is clicked. The commands are entered into a table. You can add a command by clicking the + button and you can remove one by clicking the x button. The commands are executed in the order they appear in the table. When you select a row you will see the properties for that command under the table.

There is a new command called “Set signal value”. This command allows you to set the value of a QPanel analog or digital input signal to a specific value.

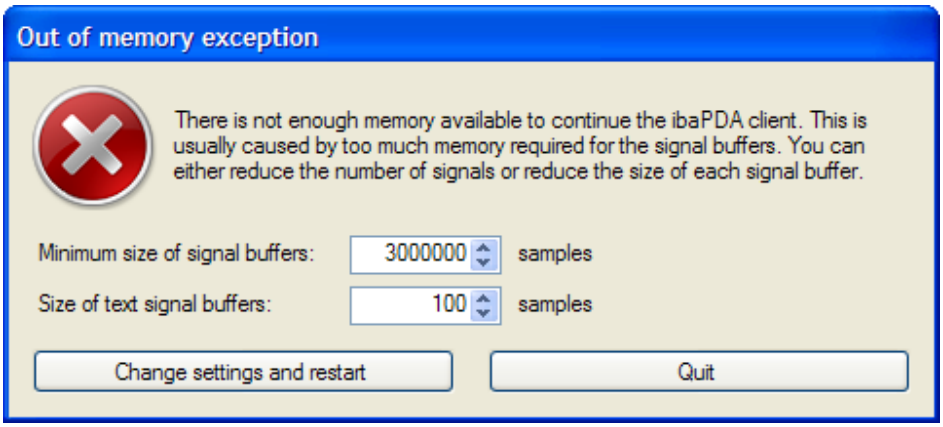
The “Execute command line” command can now expand environment variables in the command line. You can do something like: %TEMP%\ProcessClick.bat

The last change is the ability to assign a keyboard shortcut to the button. You can define a combination of Ctrl, Shift, Alt and a number, letter or function key. This shortcut will work even if the button’s QPanel is not focused. If the shortcut has already been assigned to another button or to a pda built-in command then a message will appear.



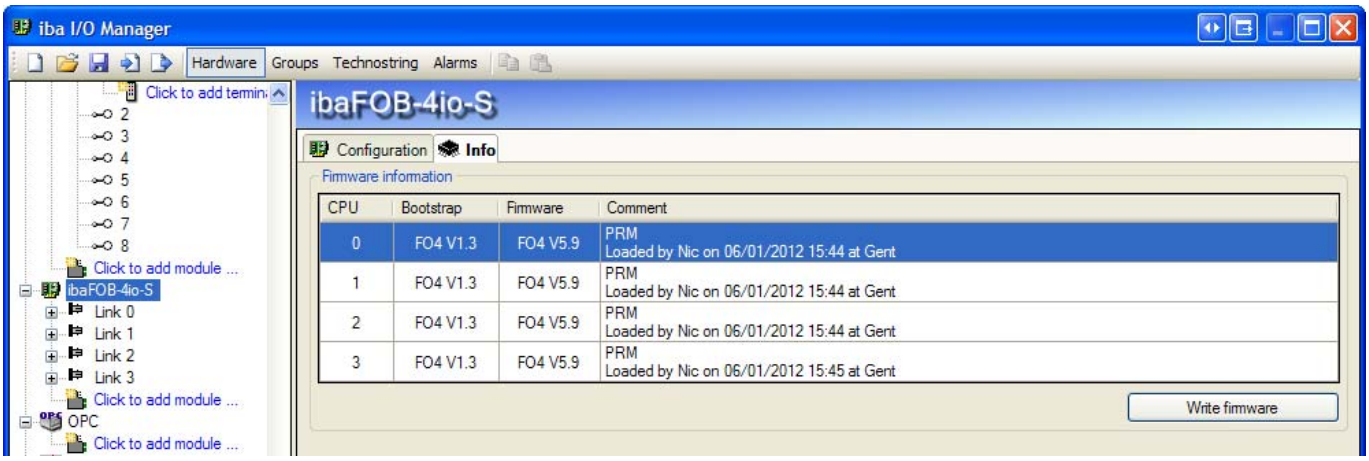
4 Handling of client OutOfMemory exception

If the pda client runs out of memory then this is usually caused by the size of the signal buffers. In the client you can specify how big each signal buffer is by specifying the minimum number of samples per signal buffer in the preferences. If you use large buffers and view lots of signals in multiple layouts then you can run into out of memory errors. When this happens then pda will show the following dialog instead of the standard .NET crash dialog.

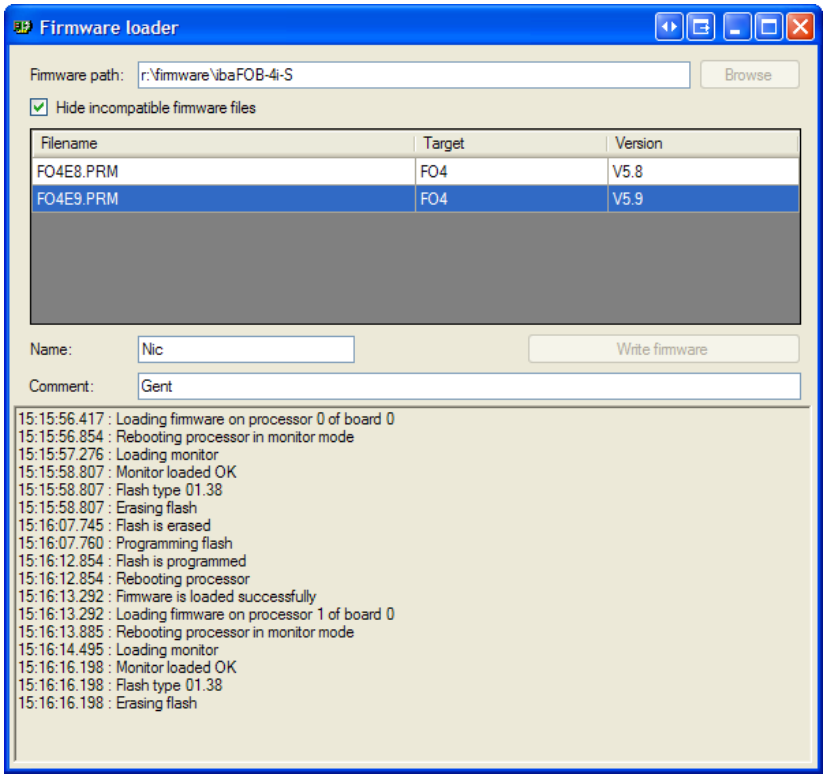


In this dialog you can change the size of the signal buffers and let the pda client restart.

5 Firmware loading on older iba boards

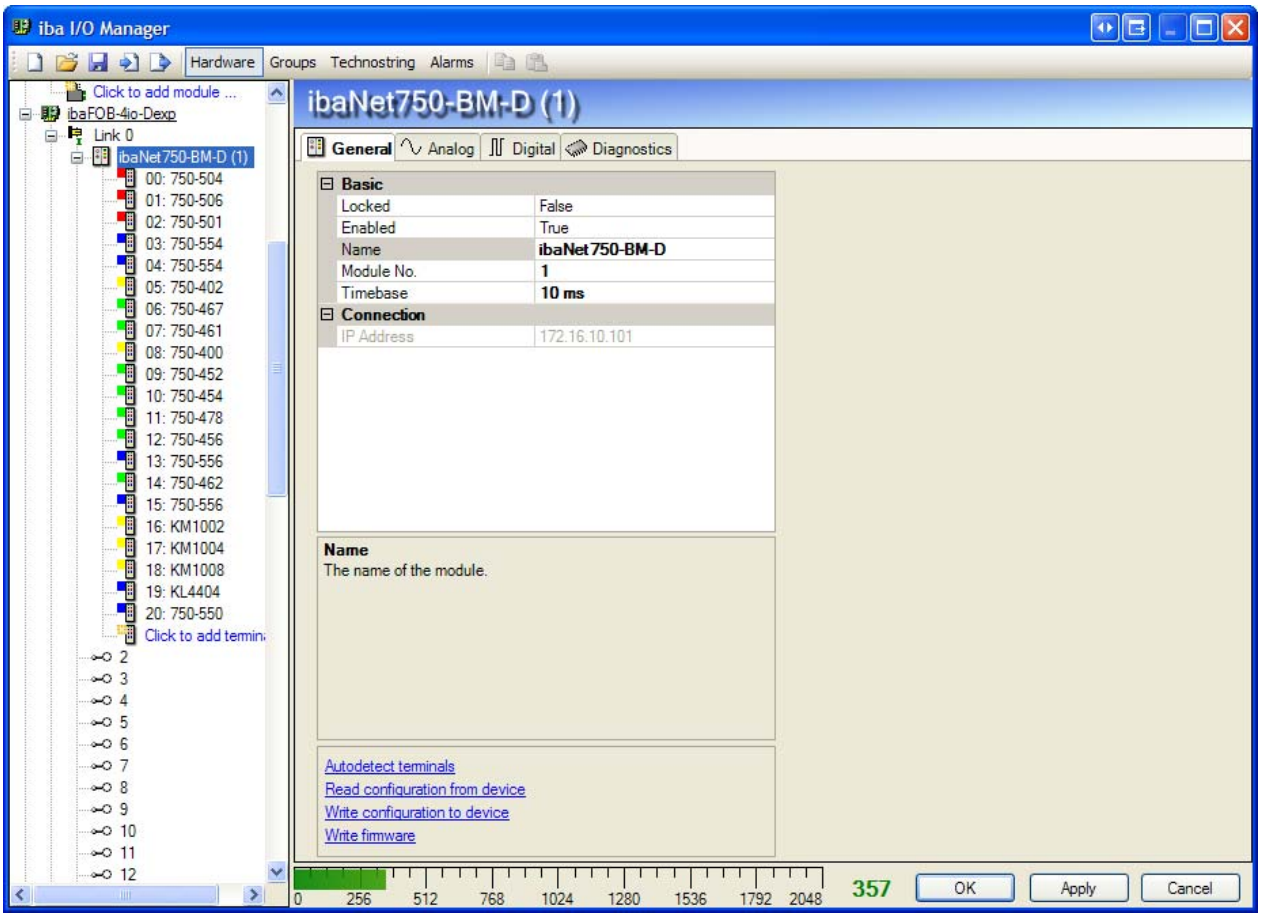


Pda can now update the firmware on FOB, FOB-S, FOB-X, FOB-TDC, FOB-SD and L2B boards. This functionality is copied from ibaDiag. The interface nodes for these boards have a new tab ‘Info’ that shows the loaded firmware version on the different CPUs. There is also a button to write the firmware. This opens the same firmware loader dialog as for FOB-D.



When you press write firmware the selected firmware file (.PRM file) is written to all CPUs on the board.

6 ibaNet750-BM-D



The ibaNet750-BM-D device is the successor to the ibaNet750 device. It supports most Wago and Beckhoff terminals. The new device has 2 modes determined by the mode switch S1:

- S1 = 0: 3.3Mbit/s protocol and behaves like old ibaNet750 device
- S1 = F: 32Mbit/s flex protocol with more data and extra diagnostics

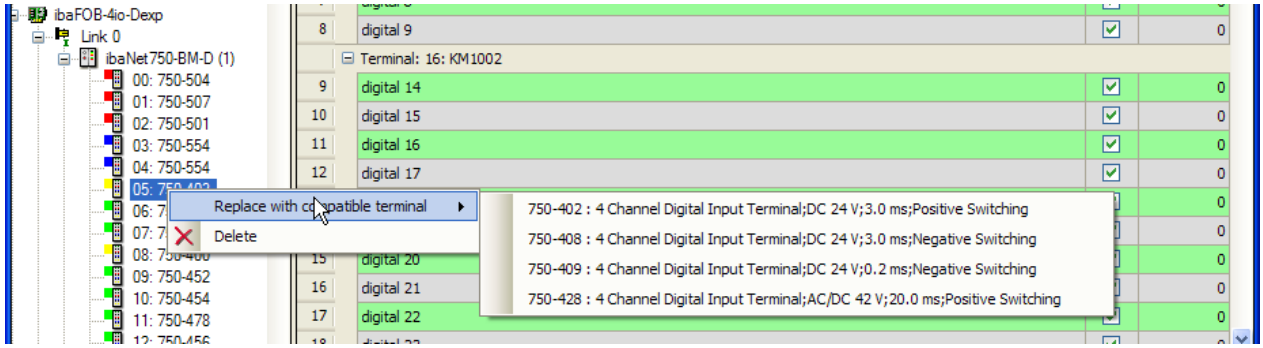
In mode 0 the 3.3Mbit/s protocol is used on the fiber optic link. The device behaves like the old ibaNet750 device and in pda an old ibaNet750 module needs to be used. Pda doesn't see any difference between an ibaNet750 device and an ibaNet750-BM-D device in mode 0.

In mode F the 32Mbit/s flex protocol is used on the fiber optic link. This allows bidirectional communication between pda and the device. Pda uses this to read out the connected terminals, to provide extra diagnostics and to decide which values get put on the fiber optic link. The rest of this chapter will deal with mode F.

Pda can autodetect the terminals connected to the ibaNet750-BM-D. You can trigger an autodetect by clicking the "Autodetect terminals" link on the general tab of the module or via the context menu of the module node in the tree. The autodetect terminals procedure is also performed when you do autodetect on the FOB-D link. The current list of terminals is shown in the tree. When you insert terminals and you perform autodetect terminals then pda will try to insert the new terminals in between the already configured terminals. When you remove terminals then pda will remove them from the configured terminals.

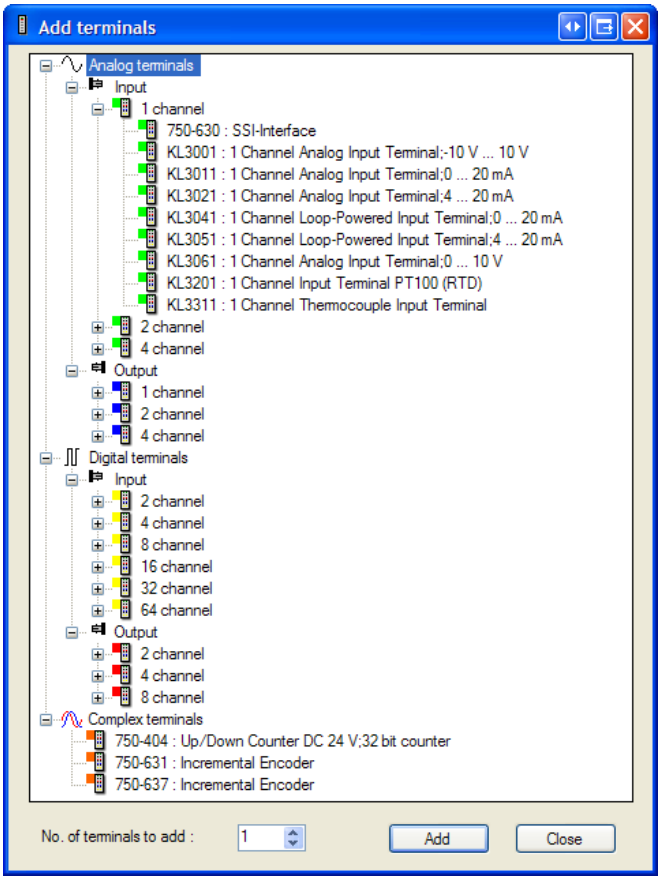
There are some limitations to the autodetect terminals procedure. The autodetect can't detect the type number of the digital terminals. It can only detect how many digital inputs and digital outputs the terminal has. Pda will then insert the first terminal it knows that has the same number of inputs and outputs. For example pda can detect a digital

terminal with 2 inputs and no outputs. Pda will then insert a 750-400 terminal but it could very well be that in reality a 750-401 terminal is present. The 750-401 also has 2 digital inputs and no outputs but it is faster than the 750-400. For pda this actually doesn't matter. The only thing that matters is the number of inputs and outputs. If you wish then you can right-click the terminal and replace it with a compatible terminal.

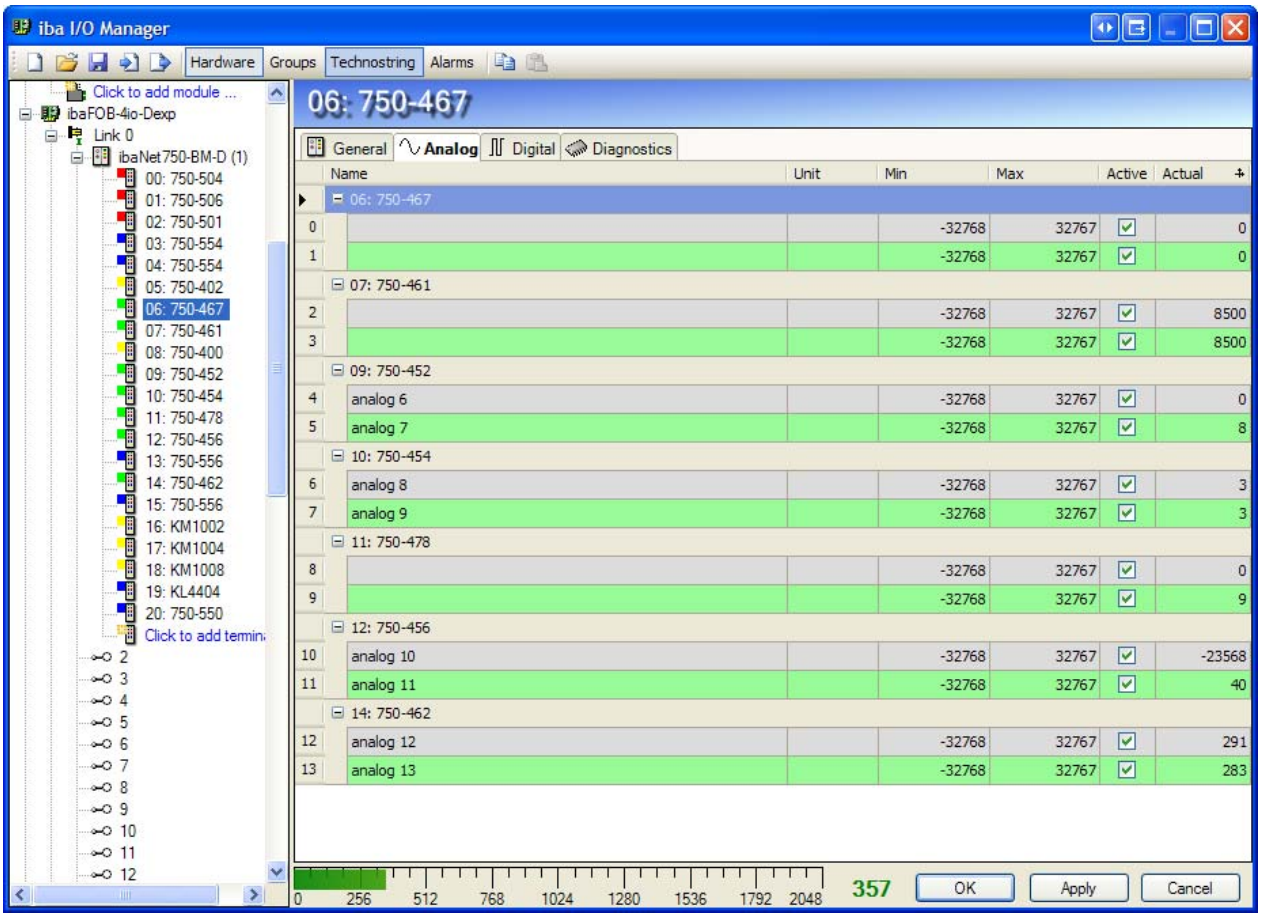


The autodetect procedure can detect the type number for the analog and complex terminals. So for the analog and complex terminals there is no need for a replace function.

You can also add terminals manually like for the old ibaNet750 device. Click the "click to add terminal" node and the terminal wizard will appear.

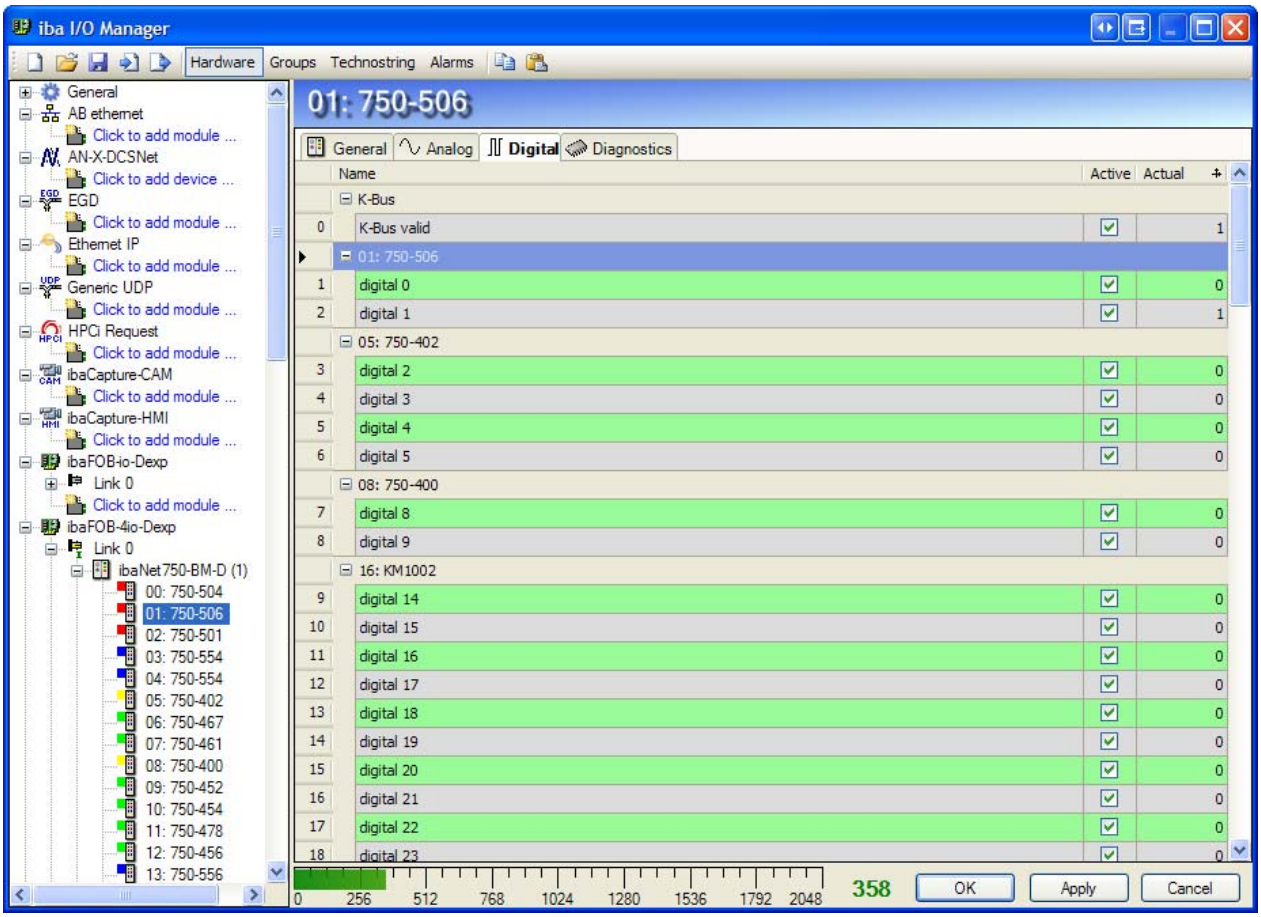


The terminals are sorted by type (analog, digital, complex), by direction (input, output) and by number of channels. The color of the terminal icon corresponds with the color of the actual terminal.

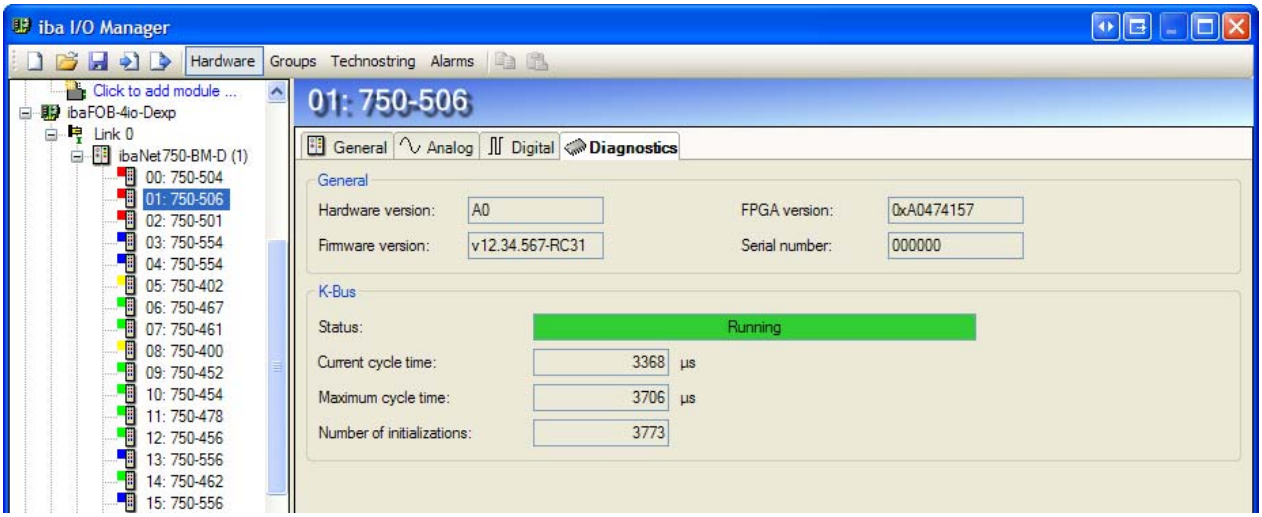


The analog tab shows the analog input signals grouped by terminal. When you click on a signal row then the corresponding terminal node will be selected in the tree. When you select a terminal in the tree then the corresponding signal row will be selected.

The actual column shows the current value of the signals. The actual value is only shown if it is being measured. So if you add a new ibaNet750-BM-D device then the actual value column will be empty. After you apply the configuration the actual value column will show values for the active signals.



The digital grid contains 1 virtual terminal called the K-Bus. The K-Bus is the name for the internal bus between the ibaNet750-BM-D device and the terminals. This virtual terminal contains 1 digital input signal called K-Bus valid. It will be 1 when the K-Bus is ok. It will be 0 when the K-Bus is not ok (e.g. not properly terminated). When the K-Bus is not ok then all the signal values will be 0 as well.



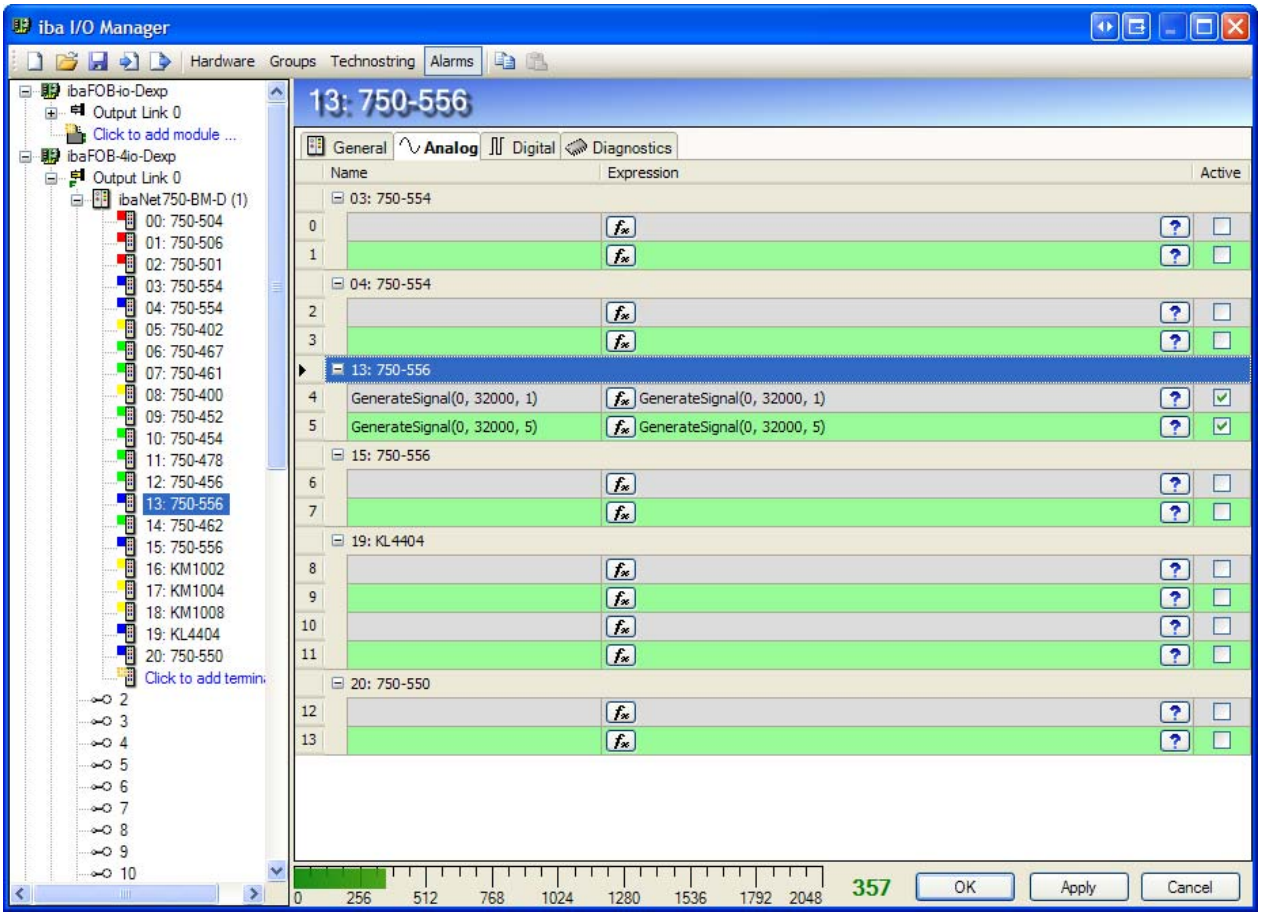
The last tab is the diagnostics tab. It shows some general version information about the device. It also shows the status of the K-Bus. The status can have the following values:

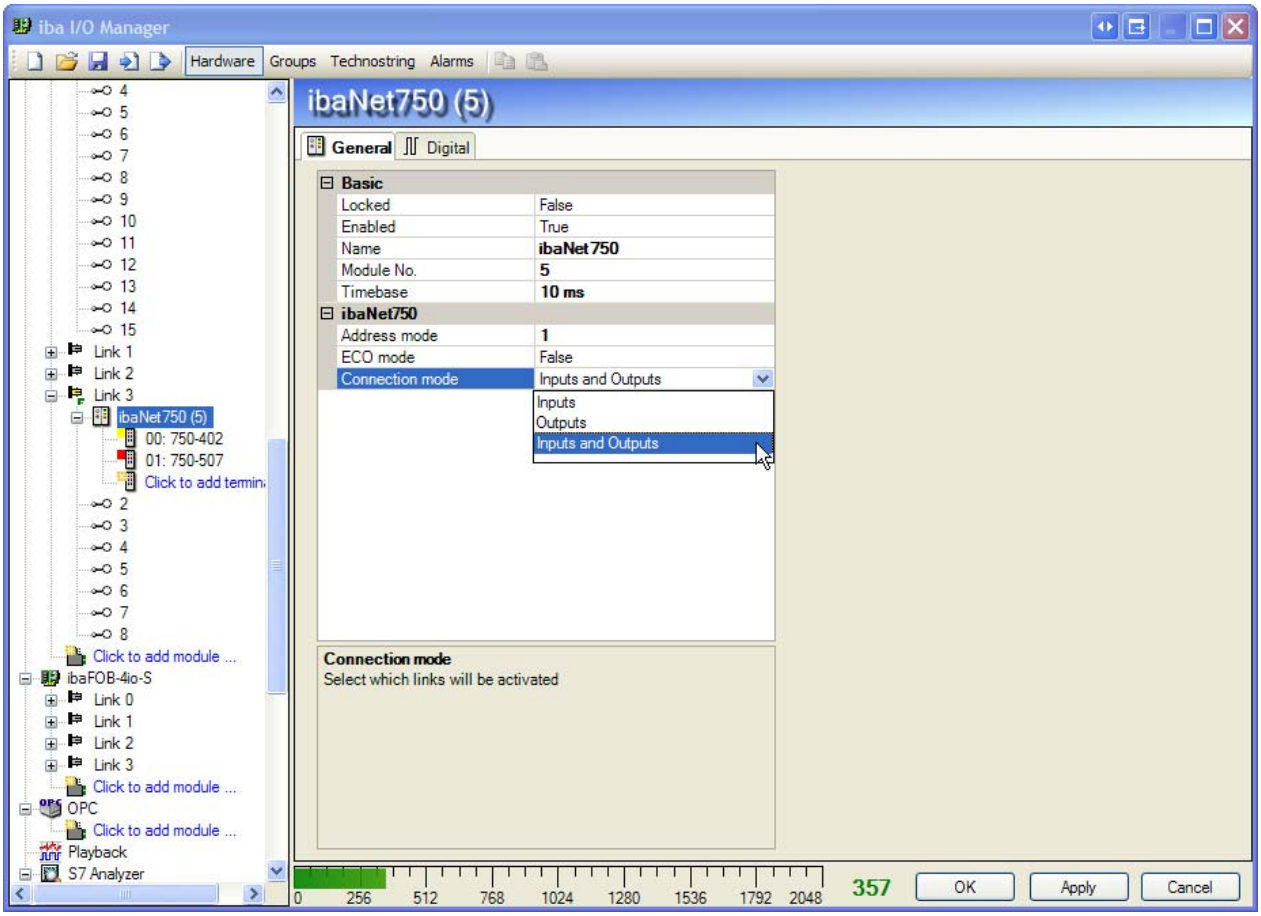
- Running: Everything is ok and the device is measuring
- I/O error: The K-Bus is interrupted. The device can't measure on the K-Bus and will send all zeros to pda.

- Configuration error: The K-Bus is ok but the currently placed terminals are incompatible with the configured terminals. The device will send all zeros to pda.

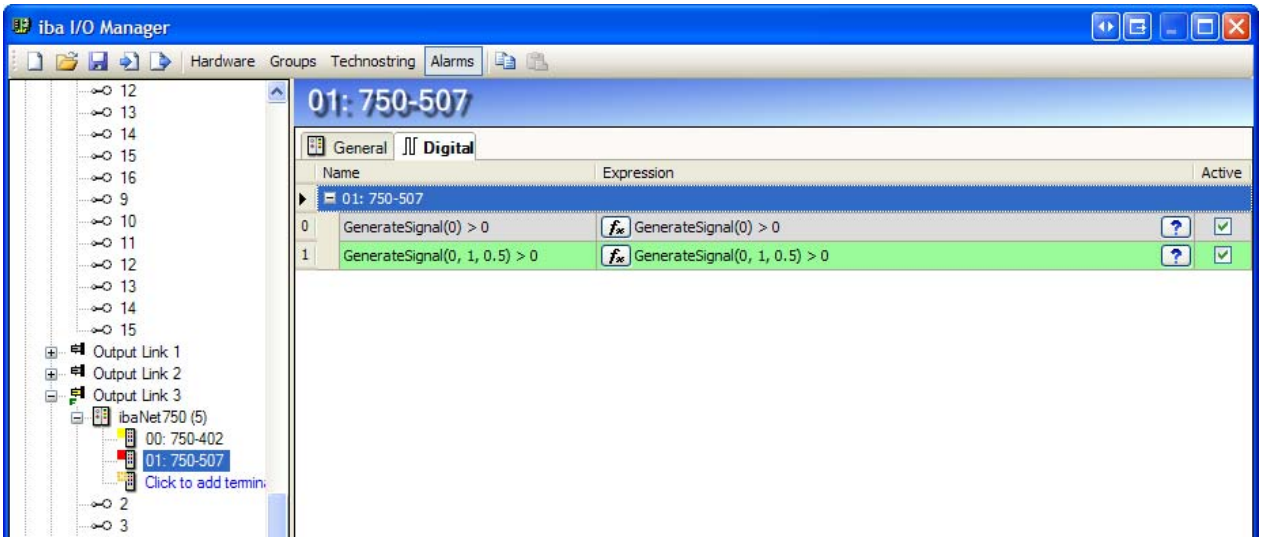
When the K-Bus is running then the cycle times are measured. The current and the maximum cycle times are reported. The cycle time will depend on the number and the type of the connected terminals. The cycle time should be relatively stable.

You can also connect output terminals to the device. Pda fully supports output terminals on the ibaNet750-BM-D. In the alarms mode on the I/O manager you will see the device and all its connected terminals (both input and output). The analog and digital grids now contain the signals corresponding to the output terminals. For each signal you can specify an expression. The result of the expression is sent to the device and put on the correct output. The expression is evaluated with the timebase of the module but the result is only transferred to the device every 50 ms (or slower depending on the timebase of all modules).





The old ibaNet750 module now also supports outputs. There is a new property called “Connection mode”. This property determines if you see inputs only, outputs only or inputs and outputs. When you load an existing configuration then the mode will be set to inputs only. This way the module will behave just as before. If you set it to input and outputs then the output link of the module is mapped to the output link of the FOB board.In the alarms mode you now see the signals corresponding to the output terminals on the module.

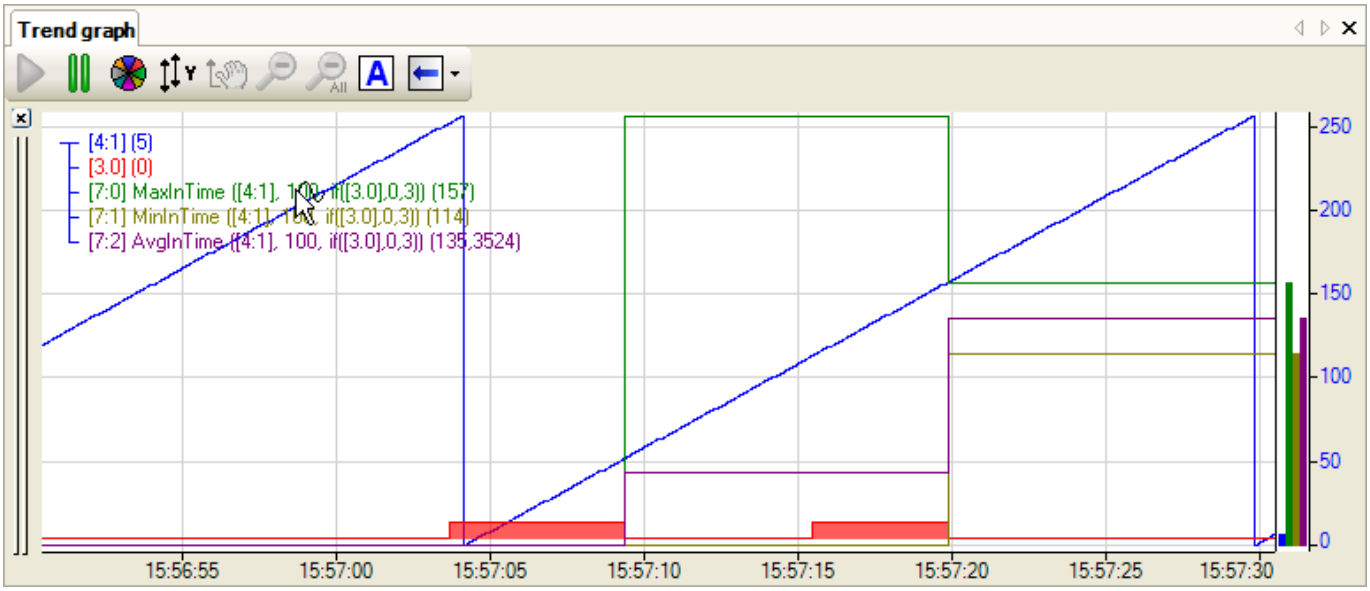


7 Virtual signal changes

There is a new function called PerformanceCounter. It allows you to measure any performance counter available on the system. It is meant to be used for diagnostics. The expression PerformanceCounter("Process", "Private Bytes", "ibaPDAService") for example can be used to log the memory usage of the ibaPDA service.

All the statistical “InTime” functions (e.g. AvgInTime, MaxInTime, …) have an extra reset parameter. All these functions calculate some statistical value during a certain interval. The reset parameter can be used to restart the calculation interval. When reset is zero then the normal calculation is done. When reset is 1 then the calculation is stopped and the result is set to 0. When reset is 2 then the calculation is stopped and the result is unchanged. When reset is 3 then the calculation is performed and the result is updated. While reset is not 0 the result isn’t changed. When reset is set to 0 again then the calculation is restarted again and also the interval input is sampled again.

The following example shows the reset parameter used to calculate the average, min and max of an analog signal (blue) over a time interval where a digital signal is true (red). The average, min and max values are updated at the end of the interval marked by the digital signal.

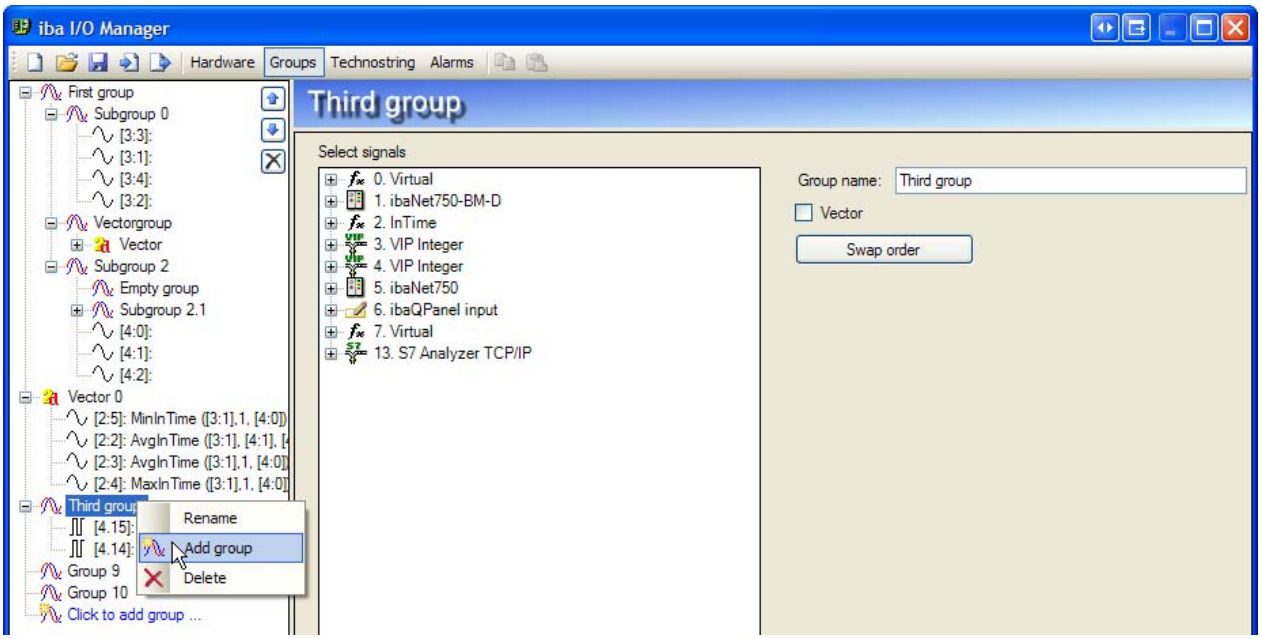


In previous versions of pda each time you switched to a virtual module in the I/O manager all the expression of the complete I/O configuration were compiled. This could cause a significant delay when you had a lot of expressions. Now pda only compiles the expressions of the selected module and the expressions that they might reference. The compilation itself has also been improved especially when signals are referenced by name. You should see significantly better performance when you have such a configuration.

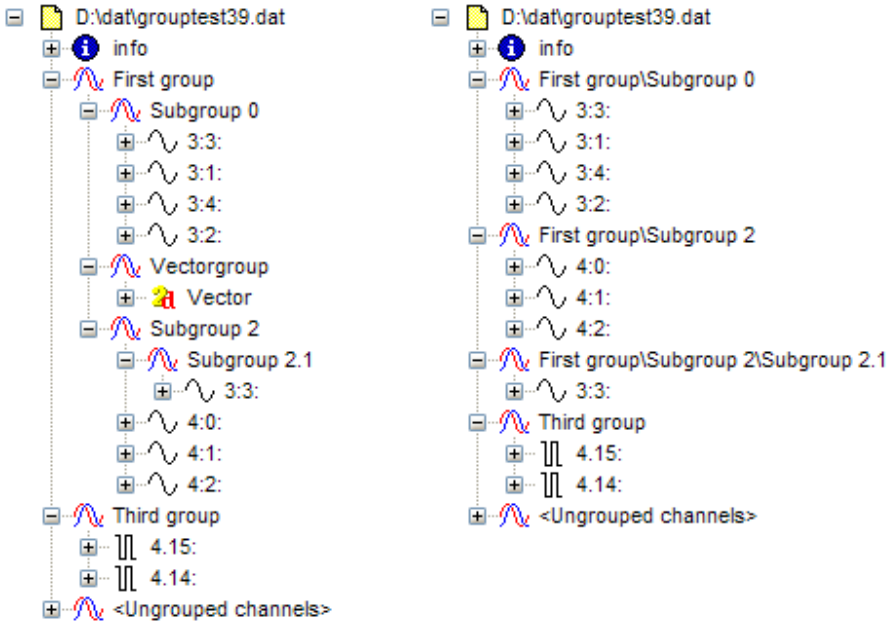
In previous versions the order of operators was not always honored correctly. It only worked if there were only 2 operators after each other. In the other cases you had to use brackets to get the correct behavior. This has been fixed. The order of the operators is this (highest precedence -> lowest precedence):

- *, /
- +, -
- >, >=, <, <=
- =, <>
- AND
- XOR
- OR

8 Sub groups



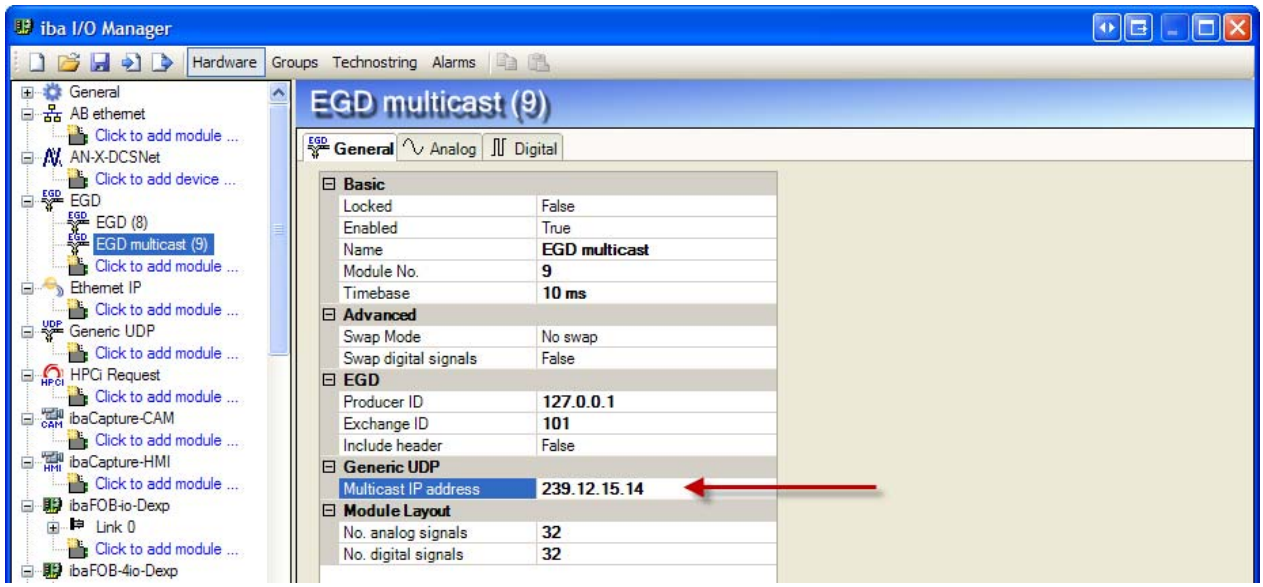
Signal groups can now have sub groups. You can add a subgroup by right-clicking on an existing group and selecting add group. The position of a sub group within its parent can be changed by the arrow buttons. Sub groups are always positioned before any signals that belong to the parent group. You can drag a group and drop it onto signals or another group to make it a sub group. You can make a vector out of a group when the group only contains signals and no sub groups.

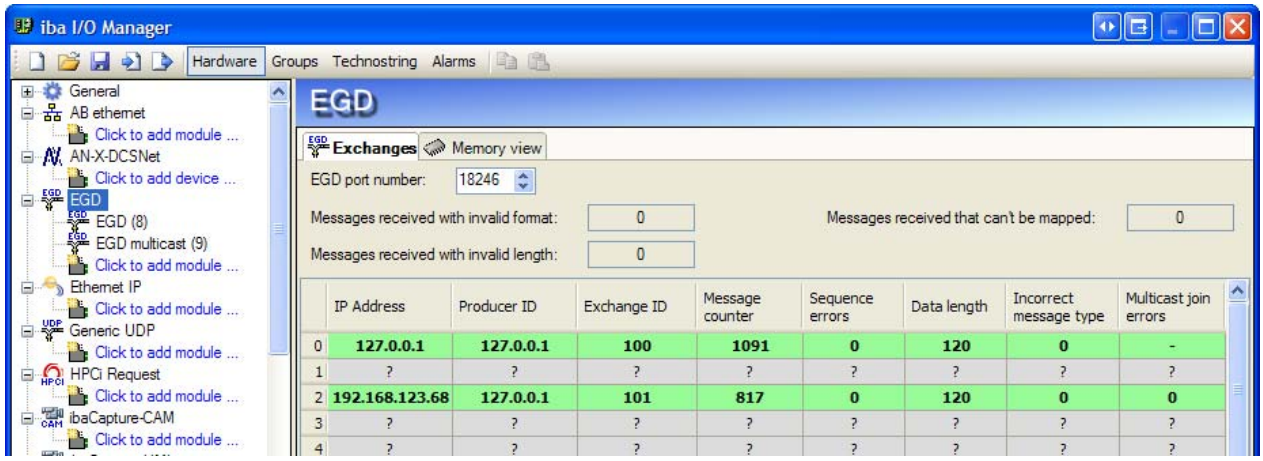


The group hierarchy is also saved in the dat file. If you have ibaAnalyzer v5.22.2 or higher the groups are shown as sub groups. If you have an older version of ibaAnalyzer then you will see all the groups at the root level with '\' used as separator between the group levels.

9 EGD multicast

The EGD interface has been extended to support EGD multicast. Now all forms of EGD communication are supported: unicast, broadcast and multicast. For multicast a new module type is added called EGD multicast. The only difference with the normal EGD module is a new property called “Multicast IP address”. Here you have to enter the IP address of the multicast group that the EGD source is sending to. Pda will join the multicast group and start receiving multicast EGD data when the acquisition is started.





The connection grid on the EGD interface has some 2 extra columns: Incorrect message type and Multicast join errors. The incorrect message type counter will be counting if unicast messages are received on an exchange that is configured for multicast. It will also count when multicast messages are received on an exchange that is configured for unicast. The multicast join errors counter will count each time pda fails to join the multicast group. It will have a ’-‘ symbol when this exchange is a unicast exchange.