a b r i d g e b e t w e e n w o r l d s

NETxKNX OPC Server 3.5
System documentation

t h e n e w d i m e n s i o n

Member of the OPC Foundation.
Member of Konnex Association.

December 2010, Version: 03.5.0004

Index

Copyright

NETxKNX OPC Server 3.5

The system allows the control and - in connection with an applicable OPC client - the visualization of small KNX plants. It builds a connection between the world of KNX and other systems.

The experience and the know-how out of the large projects were used during developing the large amount of small systems. So the system has been realized in a very reliable, open and user friendly way.

Important Information

A workspace from NETxEIB OPC Server 3.0 cannot be used without manual changes within NETxKNX OPC Server 3.5 workspaces directory. Filenames and Parameter names have changed. So best way to upgrade is to create a new workspace, open each configuration file with an Editor and fill out manually each definition using copy and paste.

Also OPC ItemIDs will change, because the name part for the Server “\NETxEIB\” has to be changed to “\NETxKNX\”. By using of Alias feature in telegram definition it is possible to keep old OPC Item IDs for preventing to change item IDs in OPC Clients.

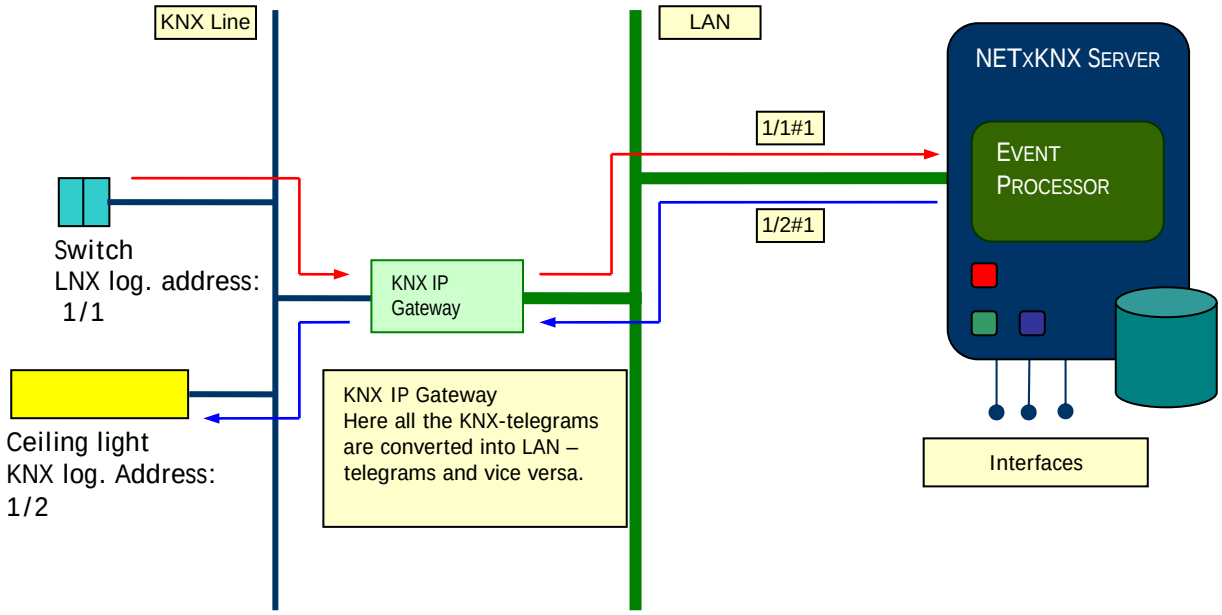
Main criteria:

- Up to 1000 gateways can be administrated
- Unified driver – different gateway types can be handled at once. ABB IG/S 1.1, KNX NetIP gateways or eibNode can be used together in one Workspace, also for Windows Server Operating Systems
- several OPC Clients can be connected parallel
- All officially used EIS data types are supported
- Up to 100.000 and more telegram definitions
- High data transfer rate
- VNET Interface for OPC Tunneling between Voyager 4.1 OPC and NETxKNX OPC Server 3.5
- Redundancy by Main- and Backup Server Configuration
- Clustering of NETxKNX Servers by N-Mesh Configuration
- KNX Device Availability Checking (in Unified Driver Version only)
- Multi-Project kernel
- Workspace Management – multiple workspaces can be administrated
- LUA Script Language Engine for programming own logic with script editor
- Event Processor – cyclic, time and event based actions can be defined in the OPC Server in Live mode
- Link Manager - the administration of linked group addresses can be realized now directly in the OPC server in Live mode
- Task Definition table for linking OPC Items and executing LUA scripts in Live mode
- KNX Telegram Overflow Manager – an extended Protection against overloading of KNX
- Real Database Refresh - automatically updates of the OPC server database
- Direct value polling of KNX devices for initializing the virtual model of the plant
- NETxKNX OPC Studio 3.5
- Additional address space - the real address of the data point consists of the logical KNX address and the IP address
- Virtual OPC items for extend address range 16/0/0 – 32/0/0

- Custom Items created within LUA script for project structuring e.g. Hotel/Building/Floor/Room
- Data recording in the queue buffer - up to one million sent or received telegrams
- Current state of the whole system can be saved and will be loaded automatically when starting
- Online check of the consistency of all connected gateways
- Real-time view of the telegram traffic with clear text description and further information
- Geo Data Interface for sun position, sunrise, sunset, age of the moon
- Several extension modules available like SQL Interface for Microsoft SQL Databases, Micros Fidelio Hotel Information Interface, …

How the system works

Data flow in the NETxKNX OPC System, Version 3.5:



KNX telegrams are converted by the gateways into LAN telegrams, which will be forwarded via the LAN to the NETxKNX Server.

The server receives, analyses, checks for errors, protocols and sends the data via the OPC interface to all connected OPC Clients.
All events are logged in the LOG data and can be used for system analyses and troubleshooting.
Every malfunction of a gateway will be detected, displayed and additionally logged.

A virtual model of the whole KNX-plant is administrated by the server. Each telegram definition is shown as a cell. In this cell not only the current value of the data point is stored, but also a list of further information (e.g. time of the last change).

The current value of a cell can be checked on the cell monitor in the studio.

The Event Processor is an extension of the NETxKNX Server kernel and makes complex, event based commands possible.

System structure

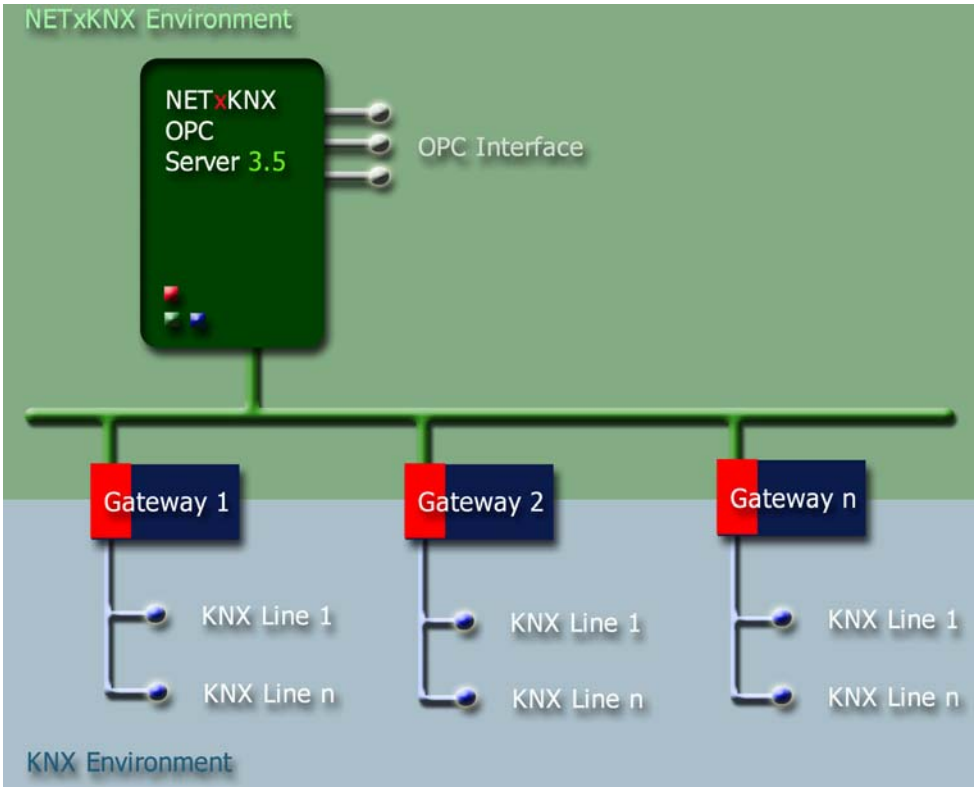
The system consists of two main parts: the server and the NETxKNX OPC Studio

The server is an autonomous program, which implements and controls all the different steps.

The Studio is the interface between the server and the administrator (only during the time of the analysis).

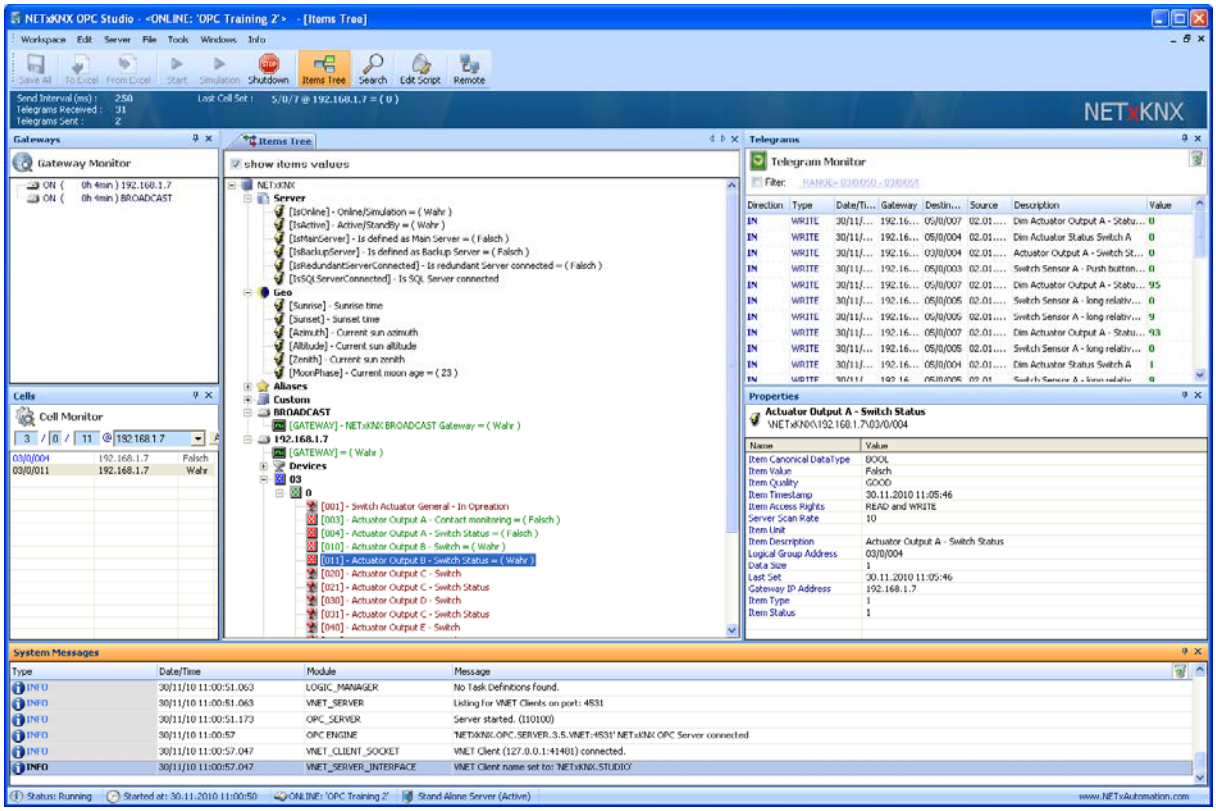
This workbench includes some useful tools, which facilitate the administrator the management, the analyses and the error search in the whole KNX-project.

Physical construction of the systems:



OPC Studio

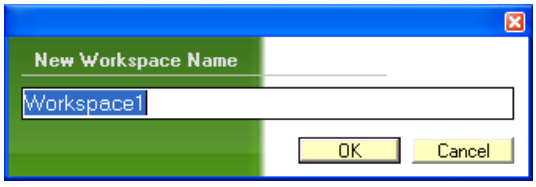
The NETxKNX OPC Studio is divided into many areas.
By the help of the window manager the user can choose how to plan the environment in accordance to his needs.



Menu

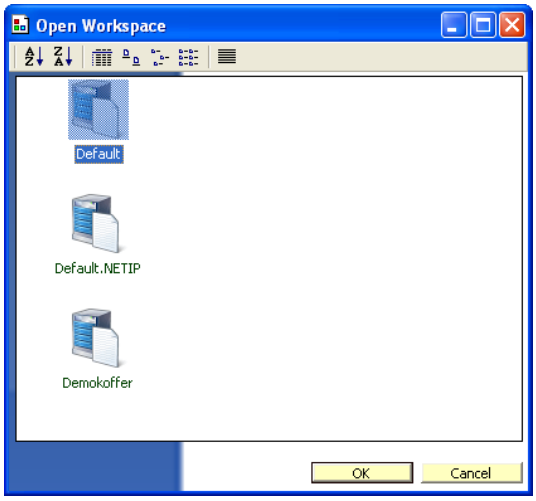
Workspace

New Workspace
This creates a new workspace - contents of the "Workspaces\Default" directory are copied into the new created directory.
The new directory gets the name of the new workspace.

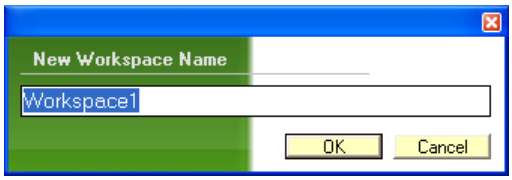


Open Workspace
This opens an existing workspace. In the "open Workspace" dialog are listed all valid workspaces "Workspaces" available.

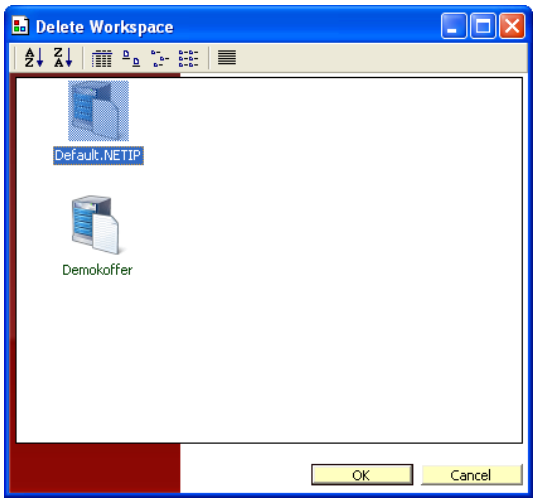
If a new "Workspace" is opened, the server will be restarted with the new configuration again.



Save Workspace As
It stores the current workspace with a new name - all relevant files of the source directory are copied into the new directory.



Delete Workspace
This deletes irrevocably the workspace with all contents and its directory.



Exit
It closes the Studio and the server will not be shut down.

Edit

Cut
It is enabled during editing in configurations tables and cuts marked text.

Copy
It is enabled during editing in configurations tables and copies marked text in clipboard.

Paste
It is enabled during editing in configurations tables and pastes text from clipboard to edit line.

Search
It searches for a text in the definition table (it will be activated, if the table is selected as the current window).

Search and Replace
It searches for and replaces a text in the definition table (it will be activated, if the table is selected as the current window)

Select all
It is enabled during editing in configurations tables and select all text.

Server

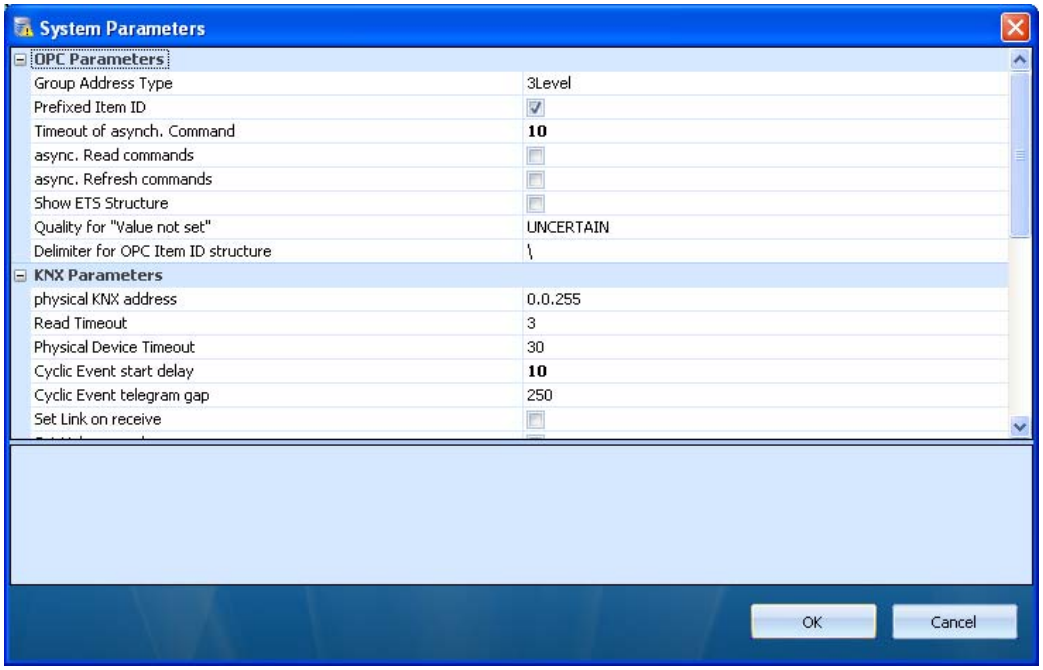
Import ETS(c) Project

This starts the importing tool, which is able to convert the exported ETS© OPC Files (*.esf) in telegram- und link definition files.
More detailed information can be found at “The converting tool: NETxKNXConvertETS”.

After each change in the telegram definition file the OPC Server have to be restarted to apply changes. Following dialog will be displayed accordingly:

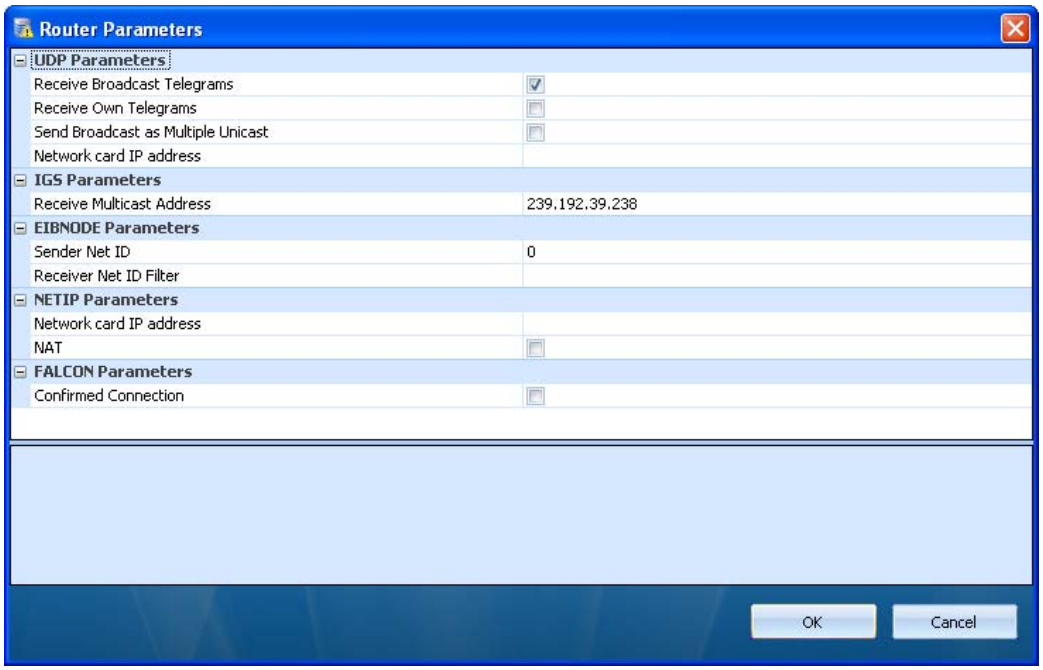
SystemConfiguration

Here special Server Parameters can be changed. The options are described and selectable. These parameters are also listed in the System Configuration File.



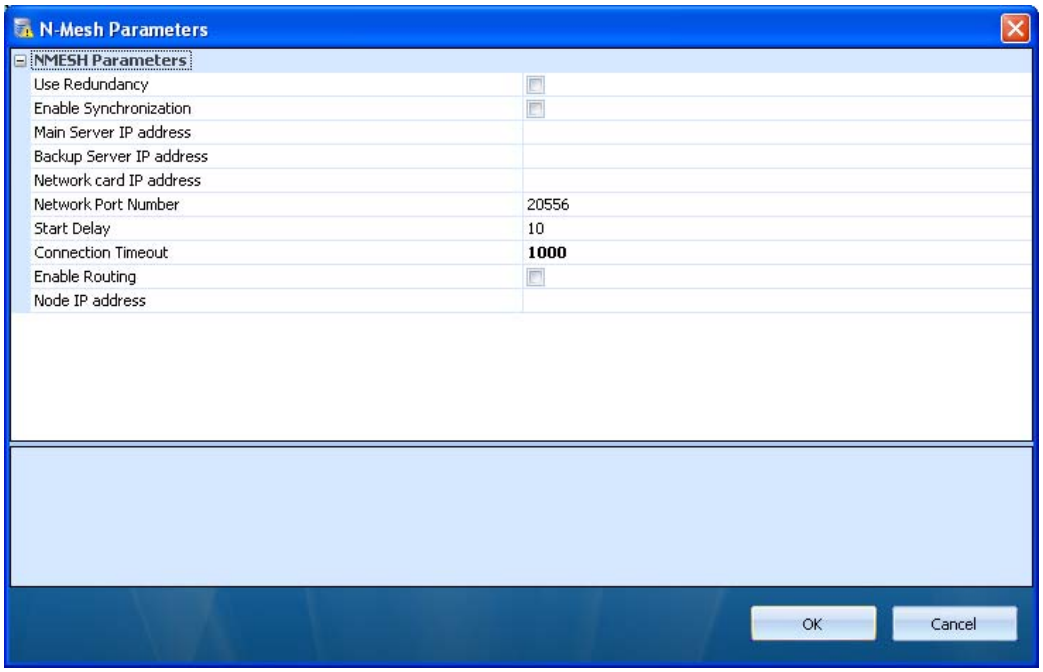
Router Configuration

Here special Router Parameters can be changed. The options are described and selectable. These parameters are also listed in the Router Configuration File.



N-Mesh Configuration

Here special N-Mesh Parameters can be changed. The options are described and selectable. These parameters are also listed in the N-Mesh Configuration File.



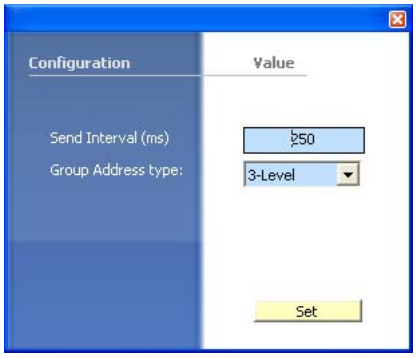
For using N-Mesh routing tables it is not necessary to configure N-mesh Node IP address as well. If N-Mesh Node IP address is configured, all changes are sent to N-Mesh nodes additionally to N-mesh routing table definitions. So you can turn this off.

Main / Backup Server Configuration

For Main / Backup Server Operation just configure “Use Redundancy”, “Enable Synchronization”, the “Main Server IP address” of Main Server, the “Backup Server IP address” of Backup Server and at last the IP address of the network card the server use to communicate with each other. By the last information the server knows whether it is Main or Backup Server.
Network Port number can be changed and firewall has to open this port.
At first configure Main Server and then copy workspace to Backup Server and change “Network card IP address”. That’s all. You can also see in the status line the operation mode of server. Either it will be shown “Main” or “Backup Server” in “active” or “standby” mode.

If Main Server stops, Backup Server will overtake all KNX IP gateways and has the same status as Main Server, because it has listened to all changes in the past. If Main Server starts again, Backup will release KNX IP gateways and Main Server overtakes them. Also if connection between Main and Backup Server is lost, Backup will try after “Connection Timeout” to overtake the KNX IP gateways.
In conjunction with Voyager 4.1 OPC the Main Server disconnects the Voyager before stopping, so that the Voyager connects the Backup Server automatically and vice versa.
For Main / Backup Operation Voyager 4.1 OPC visualization is optimized to be the perfect frontend for highly redundant system requirements.

System Settings
It shows the configuration window. Here all the system parameters which do have influence on the System can be changed. All other parameters can only be changed in the configuration data („nxaOPCRouter.35.cfg“ and „nxaOPCSystem.35.cfg“).



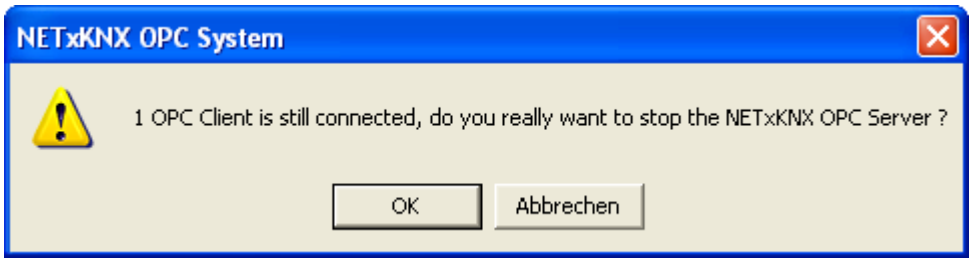
Send Interval
It defines the interval of telegrams, which are sent to a defined gateway. Because the KNX can send about 17 telegrams a second, the data management module is responsible that the telegrams are sent to the gateway in the defined time. If for example 2000 telegrams are sent to the gateway 192.168.1.1, it will take 2000x200ms=6.6 minutes, until all of them will be sent out. We suggest calculating with 10 telegrams per second per KNX line to have enough capacity for higher short period load and not to loose any telegrams.

Group Address Type
Defines how the logical KNX address is shown in the studio (in 2 or 3 steps). This parameter does not have any influence on the OPT ITEM ID – (which mainly consists of the log. KNX group address and is used for the OPC ITEM ID and is placed in the data file „NxEopOPCSystem.35.cfg“.

Start Server
Starts the OPC server (it will be activated, if the server is started or if the studio lost the connection to the server)

Shutdown Server
The server shuts down.

If at present OPC Clients are connected to the server, this dialogue window will be shown:



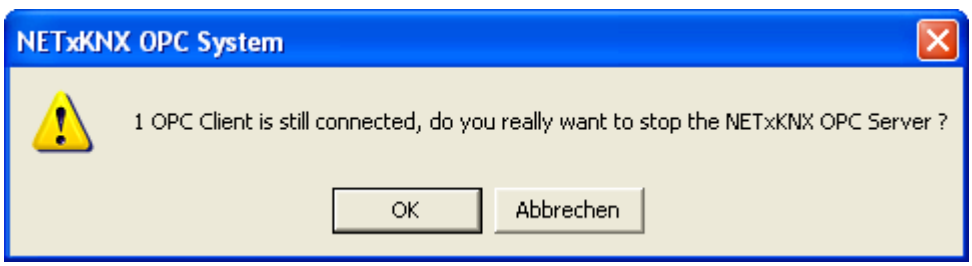
This informs that still another OPC Client is attached, and it asks whether the server is really to be stopped.

Attention: although the system has an OPC shutdown interface and the information regarding the shutdown is sent to all the clients by the help of the server, some clients do not consider this information and report a connection error!

Also some clients try automatically to connect the OPC Server again and would restart the server. So it is a good practice to stop this behavior at the clients beforehand.

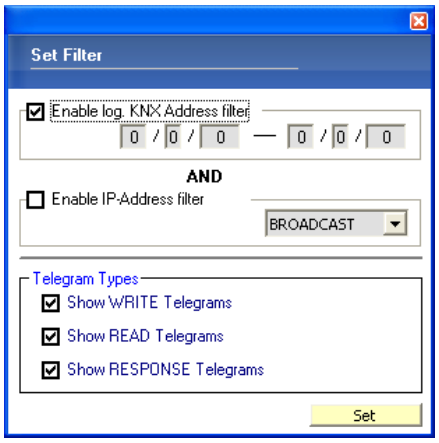
Restart Server
So the system is initialized new and the current definition tables are loaded. In this case also the server is shut down, and the connection to the OPC clients is interrupted.

If at present OPC Clients are connected to the server, this dialogue window will be shown:

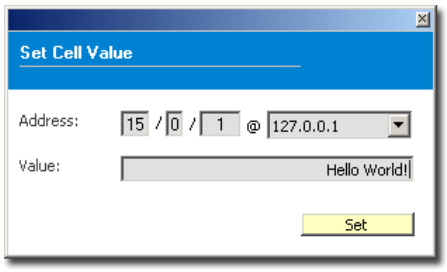


This informs that still another OPC Client is attached, and it asks whether the server is really to be stopped.

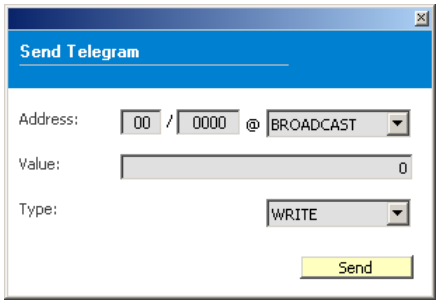
Reload N-Mesh Routing
With this command the N-Mesh Routing file is reloaded manually.



Set Filter
This function simplifies the analysis of the data traffic. This filter tells us which telegrams are shown in the telegram monitor. The filter consists of three parts. The first part defines the range of the log. KNX addresses, which should be shown. In the second part the IP address of a gateway can be defined. So only the telegrams are listed, which are sent or received by this gateway.
The last part defines the type of the telegrams which should be shown.
The filter parts can be combined.



Set Cell Value
This tool enables the direct placing of a cell value. The value is changed in the virtual model of the plant - in detail in the server storage - and sent to the OPC client. No telegram is sent to the KNX. Particular data points can be initialised with values. The requirement is that the entered parameters are valid. Otherwise you will get an error message and everything will be stopped.



Send Telegram
By the help of this tool the sending of a telegram is possible. Not only the cell value is changed, also the telegram is sent to the KNX. If no telegram is defined, an error message will be created and everything will be stopped. If all the inputs are correct, but the connection to the LAN is interrupted, no action will be taken place.

The type of the telegram defines the function:
"WRITE" - write telegram
"RESPONSE" - answer telegram
"READ" - read telegram

Advanced Configuration

Extended Logging ON/OFF
The extended logging of the OPC interface is switched in and out.

File

System Log File

Open
The System-LOG-File of the current workspace is opened and all content is displayed in a table. This process can take longer on larger files.

Close
It closes the view.

Gateway Definitions

Here all KNX NETIP Gateways have to be defined.

Open
This opens the gateway definition table

Save
It stores the gateway definition table (is going to be activated when the table is chosen as a current window). After the storage you will be asked regarding a new initialization of the system.

Close
This closes the gateway definition table, without storing the current situation (is going to be activated when the table is chosen as current window).

Telegram Definitions

This table has been created automatically with the import of ESF file. It defines which KNX telegrams should be listed to and how to hold its value in Items together with the KNX IP Gateway, where the log. KNX group address is situated.
It is also possible to edit this in the table for small changes or for larger changes to export the telegram definitions to Excel and import them back after the changes.

Open
This opens the telegram definition table

Save
It stores the telegram definition table (is going to be activated when the table is chosen as a current window). After the storage you will be asked regarding a new initialization of the system.

Close

It closes the telegram definition table, without storing the current situation (is going to be activated when the table is chosen as current window).

Device Definitions

The built in Device Manager (only Unified Driver!) checks the defined KNX Devices cyclically and will show them on or off, if they can be connected.

Open
This opens the device definition table

Save
It stores the device definition table (is going to be activated when the table is chosen as a current window). After the storage you will be asked regarding a new initialization of the system.

Close
It closes the device definition table, without storing the current situation (is going to be activated when the table is chosen as current window).

[Live] Task Definitions

Task Definitions are used to make a link between Items and with the possibility to execute LUA scripts, if Source Item has been set, received or sent. The value of Source item is forwarded to Destination Item. This behavior can also be delayed.

Open
This opens the Task definition table

Save
It stores the Task definitions table (is going to be activated when the table is chosen as a current window). Changes are applied live at the running system. No restart is necessary.

Close
It closes the Task definition table, without storing the changes (is going to be activated when the table is chosen as current window).

[Live] N-Mesh Routing Definitions

If in N-Mesh Configuration “Enable Routing” is enabled, these definitions are executed. With them it is possible to route telegrams from one OPC Server to another OPC Server.

Open
This opens the N-Mesh Routing definition table

Save
It stores the N-Mesh Routing table (is going to be activated when the table is chosen as a current window).
Changes are applied live at the running system. No restart is necessary.

Close
It closes the N-Mesh Routing table definition table.

[Live] KNX Link Definitions

This links logical group addresses together like in KNX field, if e.g. more than one group address is used to control the same input (master control).

Open
This opens the link definition table

Save
It stores the link table (is going to be activated when the table is chosen as a current window). Changes are applied live at the running system. No restart is necessary.

Close
It closes the link definition table, without storing the changes (is going to be activated when the table is chosen as current window).

[Live] KNX Response Event Definitions

Here you can define events based on receiving KNX telegrams and to create new KNX telegrams.

Open
This opens the response event definition table

Save
It stores the response event table.
Changes are applied live at the running system. No restart is necessary.

Close
Closes the response event definition table, without storing the changes (is going to be activated when the table is chosen as current window).

[Live] KNX Timer Event Definitions

Here you can define Timer events to create new KNX telegrams.

Open
Opens the timer event definition table

Save
It stores the timer event table (is going to be activated when the table is chosen as a current window).
Changes are applied live at the running system. No restart is necessary.

Close

Closes the timer event definition table, without storing the changes (is going to be activated when the table is chosen as current window).

[Live] KNX Cyclic Event Definitions

Here you can define Cyclic events to create new KNX telegrams.

Open
This opens the cyclic event definition table

Save
It stores the cyclic event table (is going to be activated when the table is chosen as a current window).
Changes are applied live at the running system. No restart is necessary.

Close
Closes the cyclic event definition table, without storing the changes (is going to be activated when the table is chosen as current window).

Advanced Configuration

System Configuration File

Open
It opens the system configuration file

Save
This stores the system configuration file.
After storing the changes are not taken over - that's why the server must be stopped manually and started again.

Close
It closes the system configuration file, without storing the changes.

Router Configuration File

Open
It opens the router configuration file

Save
This stores the router configuration file.
After storing the changes are not taken over - that's why the server must be stopped manually and started again.

Close
It closes the router configuration file, without storing the changes.

N-Mesh Configuration File

Open
It opens the N-Mesh configuration file

Save
This stores the N-Mesh configuration file.
After storing the changes are not taken over - that's why the server must be stopped manually and started again.

Close
It closes the N-Mesh configuration file, without storing the changes.

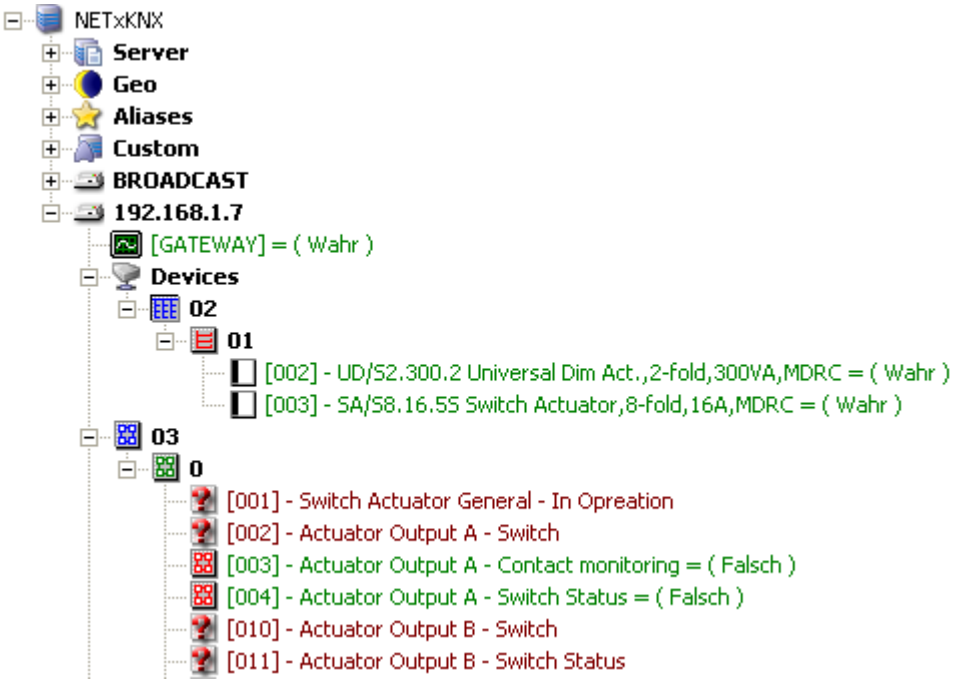
Print
Prints the definition tables (activated if the table is selected as current window)

Tools

Telegram History Explorer
It starts the Telegram History Explorer, with which the logged telegrams can be analyzed.

Windows

Items Tree
It turns the „OPC Items" – View on and off.



System Messages
This switches on / off the System Messages.

Telegram Monitor
It switches on / off the telegram monitor.

Cell Monitor
This switches on / off the cell monitor.

Gateway Monitor
It turns on / off the gateway monitors.

Search
It is a search tool to find OPC items by ItemID or description search pattern e.g.”1/*”.

Item Properties
It turns on / off the Item Properties window.

Restore Positions
It restores the position of all windows to its default position.

Cascade
It positions the definition windows in the center area as cascade.

Vertical
It positions the definition windows in the center area vertically.

Horizontal
It positions the definition windows in the center area vertically.

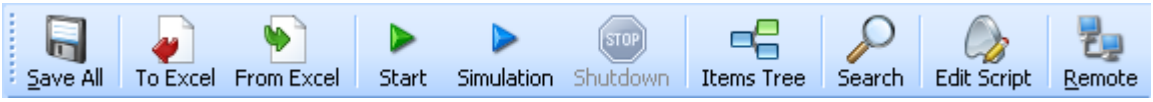
Info

License Manager
Use this tool to license the software with hardware dependent local system ID. This is our Softlock code system to protect the software.
Do not use this tool to check your license. Please look at System messages or System Log File.

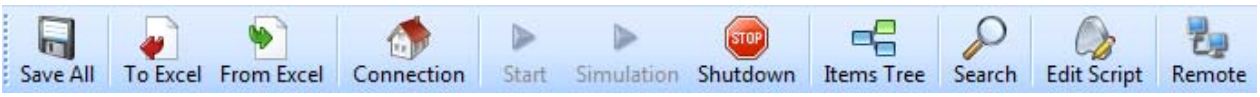
About
General information dialog can be seen.

Toolbar

At UnifedDriver Version:



At Direct(KNX) Version:



Save All

It saves all open definition tables. If the table is saved, a query dialog will be displayed whether to initialize the server or not. If this query is confirmed by YES, the system will initialize with the new table.
Attention: The OPC Server will shut down and the connection to the OPC clients will be interrupted.

To Excel

It exports a definition table in Microsoft© Excel (it will be activated, if the table is selected as current window).

From Excel

This imports a definition table from a Microsoft© Excel file (it will be activated, if the table is selected as current window).

Connection

It opens the Falcon Connection manager. (Just in Direct(KNX) version).

Start

It starts the OPC server (it will have been activated, if the server has not started or if the studio has lost the connection to the server).

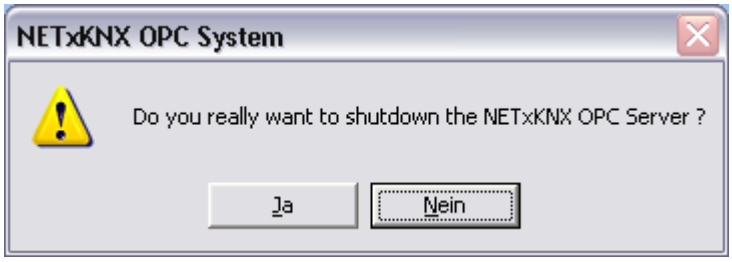
Simulation

It starts the OPC server in Simulation mode. This means the connection of KNX NetIP gateways is simulated. Of course the KNX Devices are not simulated, but it is possible to set Items in Item tree and so manually test your OPC Clients etc. There is also the possibility to let the system generate random values for the Items and then you can see in e.g. your visualization some activity.

Shutdown

The server is shutting down.
(It will have been activated, if the studio has got the connection to the server).
Attention: although the system has an OPC shutdown interface and the information about the shutdown is sent to the clients by the server, some clients do not consider this information and report a connection error.

At present if OPC Clients are connected to the server, this dialogue window is indicated:



Here informed that still another OPC Client is attached, and it is queried whether the server is to be really stopped.

Items Tree

It starts the Items tree.

Search

It starts the Search tool for OPC Items.

Edit Script

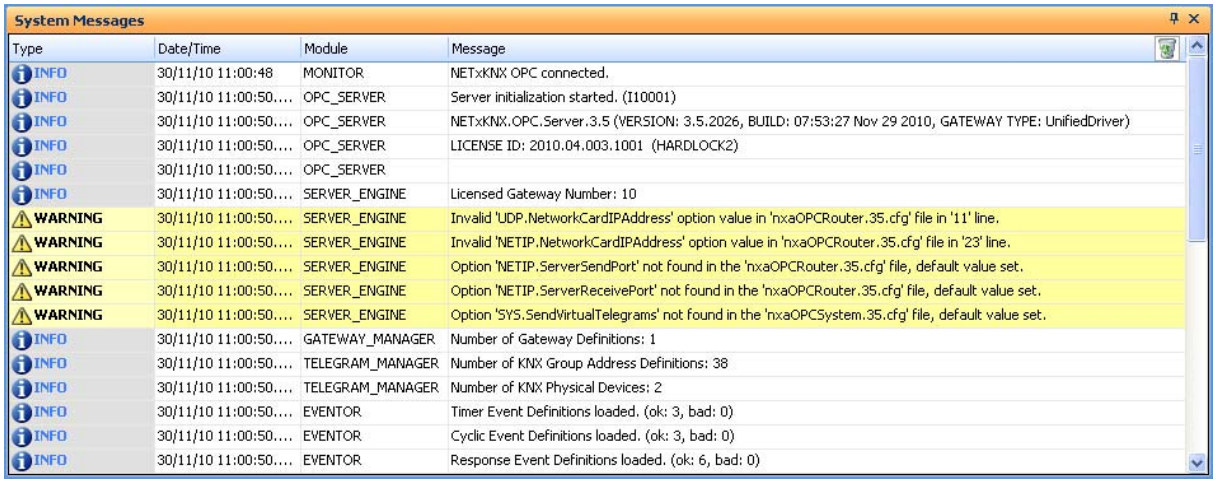
It starts the built in script editor for LUA scripts.

Remote

It starts the built in Quick Support Tool, which can be used with Teamviewer application to create a remote desktop connection for maintaining or support.

Window

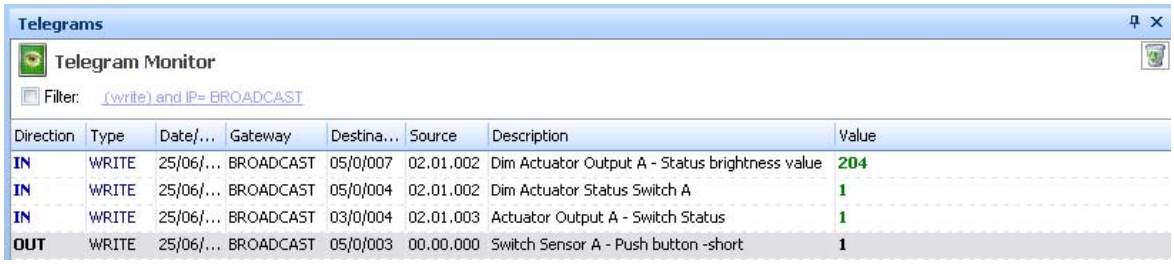
System Messages



Here all the relevant events of the system are shown. Every information type is marked. All shown pieces of information are stored in the „nxaOPCSystem.35.log“ file or later dated system log files.
This makes a system analysis afterwards possible.

If a column header is dragged and dropped to the group field above the table, the header will be pulled out and the table will be grouped by this header. If you press additionally “Ctrl” key while dragging, the header will remain in the table.
With the trash icon all system messages are deleted.
Error messages are red colored.
Warning messages are yellow colored.
Info messages are not colored.

Telegram Monitor



All received or sent telegrams are shown in the telegram monitor.
Only those telegrams are listed, which were defined in the system. All others are going to be ignored. The first column shows whether a telegram has been sent (OUT) or received (IN).The second column displays the type of the telegram. In the third column the date/time of the event is shown.

When receiving a telegram the IP address of the sending gateway is shown, when sending a telegram the destination IP address is shown.
„BROADCAST“ means that a telegram has been sent to all gateways at the same time (central telegram).

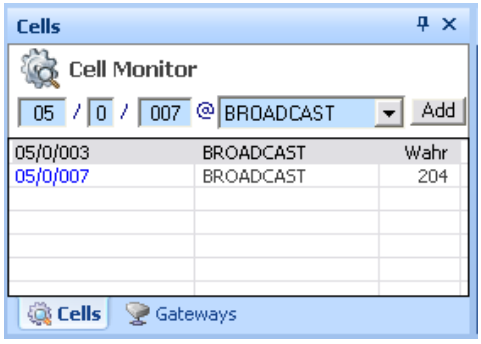
The next column shows the logical KNX-address, which has sent the telegram.
The physical KNX address of the source of the received telegram can be seen in the fifth column. The value of the telegram is followed by the clear text description.

In plants, where many telegrams are sent or received, the table should only be used for system checks. Afterwards it should be closed again, just to save the resources for the server.

Configuration of the table
The width of a column of the table can be changed. If you would like to have a different order of columns, just drag and drop the header.

Filter
The filter enables the monitoring on particular log. KNX addresses - areas or gateways.

Set Filter
This function simplifies the analysis of the data traffic. This filter tells us which telegrams are shown in the telegram monitor. The filter consists of three parts. The first part defines the range of the log. KNX addresses, which should be shown. In the second part the IP address of a gateway can be defined. So only the telegrams are listed, which are sent or received by this gateway.
The last part defines the type of the telegrams which should be shown.
The filter parts can be combined.

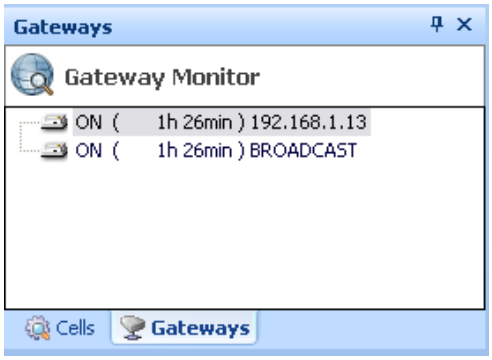


Cell Monitor

The virtual model of the plant is mapped in the server memory. The single data points are implemented as cells. The value changes of a cell can be seen by the help of the cell monitor in real time. To insert a cell into the Cell Monitor can be done by entering on the whole address (log.KNX address + IP Address) and pressing the "ADD“ Buttons. A

cell can be deleted by clicking on it and pressing the Delete button on the keyboard. If no value is referenced to the cell, the sign “???” will arise in the value column. Only the cells which are defined in the telegram definition table can be checked.

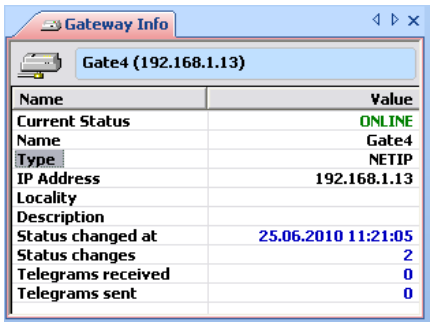
Gateway Monitor



The "Gateway Monitor" lists all gateways which are defined in the file of the gateway definition table.
Here also the current situation (OFF – means Offline, ON – means Online), the duration since the last change and the IP address can be seen. If a gateway cannot be reached by the system it will be marked with a red cross and "OFF".

The "BROADCAST" gateway is imaginary equipment. It is used to send all the central telegrams, which are sent to all gateways at the same time.
If the "BROADCAST" equipment is offline, the server does not have any connection to the network. In this case the network configuration of the PC has to be checked.
By making a double click on the listed gateway, the gateway info window is opened.

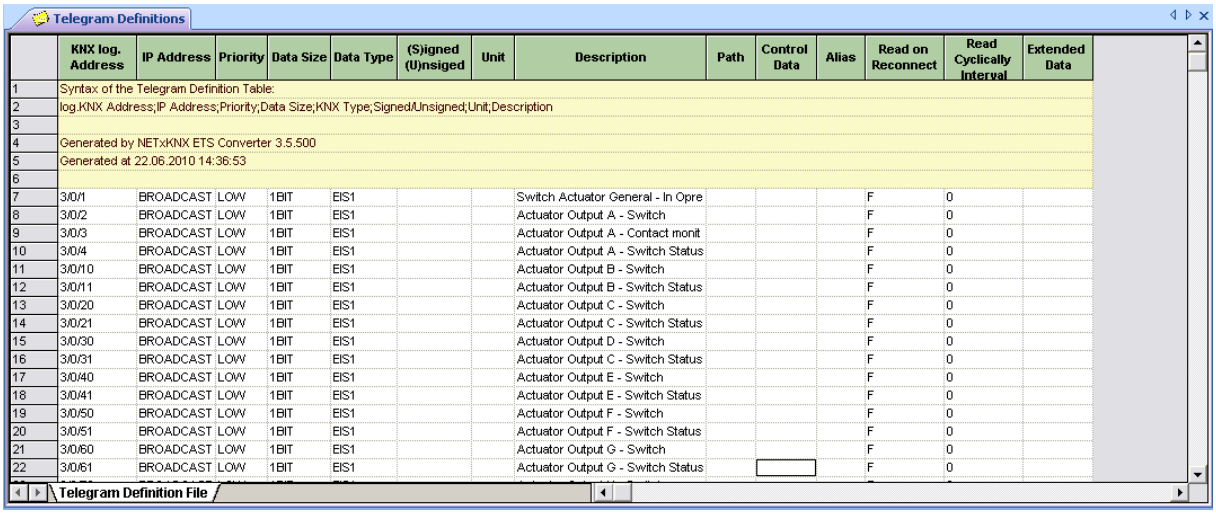
Gateway Info



The Gateway Info displays the available information about the selected gateway. The window can be activated by making a double click on the gateway icon in the gateway monitor.

Current Status – the current status of the equipment (ONLINE/OFFLINE)
Name – the name of the gateway
IP Address – the IP address
Locality – location
Description – shows a short description
Status changed at – time of the last status changed
Status changes – the amount of status changes since the last initialization of the system
Telegrams received – number of telegrams the server has received via this equipment
Telegrams Sent – number of telegrams which were sent via this gateway

Telegram Definition Table



Telegram Definitions

	KNX log. Address	IP Address	Priority	Data Size	Data Type	(S)igned (U)nsiged	Unit	Description	Path	Control Data	Alias	Read on Reconnect	Read Cyclically Interval	Extended Data
1	Syntax of the Telegram Definition Table:													
2	log.KNX Address;IP Address;Priority;Data Size;KNX Type;Signed/Unsigned;Unit;Description													
3														
4	Generated by NETxKNX ETS Converter 3.5.500													
5	Generated at 22.06.2010 14:36:53													
6														
7	3/0/1	BROADCAST	LOW	1BIT	EIS1			Switch Actuator General - In Opre				F	0	
8	3/0/2	BROADCAST	LOW	1BIT	EIS1			Actuator Output A - Switch				F	0	
9	3/0/3	BROADCAST	LOW	1BIT	EIS1			Actuator Output A - Contact monit				F	0	
10	3/0/4	BROADCAST	LOW	1BIT	EIS1			Actuator Output A - Switch Status				F	0	
11	3/0/10	BROADCAST	LOW	1BIT	EIS1			Actuator Output B - Switch				F	0	
12	3/0/11	BROADCAST	LOW	1BIT	EIS1			Actuator Output B - Switch Status				F	0	
13	3/0/20	BROADCAST	LOW	1BIT	EIS1			Actuator Output C - Switch				F	0	
14	3/0/21	BROADCAST	LOW	1BIT	EIS1			Actuator Output C - Switch Status				F	0	
15	3/0/30	BROADCAST	LOW	1BIT	EIS1			Actuator Output D - Switch				F	0	
16	3/0/31	BROADCAST	LOW	1BIT	EIS1			Actuator Output C - Switch Status				F	0	
17	3/0/40	BROADCAST	LOW	1BIT	EIS1			Actuator Output E - Switch				F	0	
18	3/0/41	BROADCAST	LOW	1BIT	EIS1			Actuator Output E - Switch Status				F	0	
19	3/0/50	BROADCAST	LOW	1BIT	EIS1			Actuator Output F - Switch				F	0	
20	3/0/51	BROADCAST	LOW	1BIT	EIS1			Actuator Output F - Switch Status				F	0	
21	3/0/60	BROADCAST	LOW	1BIT	EIS1			Actuator Output G - Switch				F	0	
22	3/0/61	BROADCAST	LOW	1BIT	EIS1			Actuator Output G - Switch Status				F	0	

Telegram Definition File

The whole data structure of the system is fixed in the telegram definition table.
This table is read out of the „nxaTelegramDefinitions.35.dat“file and can be edited by the user.
For working in this file it is recommended to export to Microsoft© Excel (XLS File), edit there and import it from XLS file back again.
In this view it is possible to the make some little changes and some extension of the table. Eventual occurred errors can be corrected. The pop up for working in the table can be opened via the right mouse button.

Insert new Definition – inserts a new line in the table, this line can be filled with telegram definition data.
Insert new Comment – inserts a comment line, which can be filled with a text
Convert to Definition – converts to telegram definition data.
Convert to Comment – converts to comment line.
Delete – deletes a chosen row
Send Telegram… - sends telegram of chosen group address.
Set Cell Value… - sets only cell value without sending a telegram.
Add to monitor - adds group address to Cell Monitor.

A telegram definition line consists of many columns. For detailed information see Telegram definitions: nxaTelegramDefinitions.35.dat

Gateway Definition Table

Gateway Definitions

	IP Address	Type	Port	Name	Locality	Description	Options	Path	Extended Data 1	Extended Data 2	Extended Data 3
1	Syntax of the Gateway Definition Table ver. 3.5:										
2	IP Address;Port;Type;Name;Locality;Description										
3											
4	192.168.1.1;IGS;52000;GATE 1;Room 51;pre-defined Gateway 1										
5	192.168.1.7;NETIP;3671;Gate2										
6	192.168.1.8;NETIP;3671;Gate3										
7	192.168.1.13	NETIP	3671	Gate4							
8	192.168.1.2;EIBNODE;1634;Gate4										

All gateways, which should be administrated by the system, have to be defined in this table.
This table is read out by the „nxaGatewayDefinitions.35.dat“ file and is available as shown above. For working in this file it is recommended to export to Microsoft© Excel (XLS File), edit there and import it from XLS file back again.

This view allows the making of some little changes and extensions in the table. Eventually occurred errors can be corrected in a rather easy way. The pop up menu for working in the table can be opened by clicking on the right mouse button.

Insert new Definition – inserts a new line in the table, which can be filled with gateway definition data
Insert new Comment – inserts a comment line, where a text can be inserted.
Delete Row – deletes a chosen line or row
Convert to Definition – converts to gateway definition data.
Convert to Comment – converts to comment line.
Delete – deletes a chosen line

A gateway definition line consists of many columns. For detailed information see: Gateway definitions: nxaGatewayDefinitions.35.dat

System protocol data

System Log File

Drag a column header here to group by that column.

Type	Date/Time	Module	Message
INFO	03/09/09 ...	SERVER_ENGINE	Licensed Gateway Number: 3
INFO	03/09/09 ...	SERVER_ENGINE	Option 'OPC.GroupAddressType' = '2Level'
INFO	03/09/09 ...	SERVER_ENGINE	Option 'OPC.AsyncReadFromDevice' = 'FALSE'
INFO	03/09/09 ...	SERVER_ENGINE	Option 'OPC.AsyncWriteConfirmation' = 'FALSE'
INFO	03/09/09 ...	SERVER_ENGINE	Option 'OPC.AsyncRefreshFromDevice' = 'FALSE'
INFO	03/09/09 ...	SERVER_ENGINE	Option 'OPC.AsyncTimeout' = '10'
INFO	03/09/09 ...	SERVER_ENGINE	Option 'KNX.Timeout' = '3'
INFO	03/09/09 ...	SERVER_ENGINE	Option 'SYS.EnableDeviceManager' = 'TRUE'
INFO	03/09/09 ...	SERVER_ENGINE	Option 'SYS.GatewayConnectionTimeout' = '100'
INFO	03/09/09 ...	SERVER_ENGINE	Option 'SYS.GatewayConnectionTimeoutAttemptCount' = '3'
INFO	03/09/09 ...	SERVER_ENGINE	Option 'OPC.QualityForValueNotSet' = 'UNCERTAIN'
INFO	03/09/09 ...	SERVER_ENGINE	Option 'OPC.Delimiter' = '\'
INFO	03/09/09 ...	SERVER_ENGINE	Option 'KNX.ServerPhysicalAddress' = '0.0.255'
WARNI...	03/09/09 ...	SERVER_ENGINE	Network Server disabled.
INFO	03/09/09 ...	SERVER_ENGINE	Option 'UDP.ReceiveOwnTelegrams' = 'FALSE'
INFO	03/09/09 ...	SERVER_ENGINE	Option 'UDP.ReceiveBroadcastTelegrams' = 'TRUE'
WARNI...	03/09/09 ...	SERVER_ENGINE	Invalid 'UDP.NetworkCardIPAddress' option value in 'nxaOPCRouter.35.cfg' file in '11' line.

All relevant information is stored in a file. This file can be loaded in the Studio for reviews. So you can analyze older events.

If a column header is dragged and dropped to the group field above the table, the header will be pulled out and the table will be grouped by this header. If you press additionally “Ctrl” key while dragging, the header will remain in the table.

Error messages are red colored.
Warning messages are yellow colored.
Info messages are not colored.

Link Manager

The NETxKNX Server can manage the linked KNX group addresses now.

Managing interlinked group addresses can be realized now directly in the OPC server and therefore the OPC clients can be relieved considerably. The values of the linked group addresses are set and the attached OPC clients informed about it by the OPC server automatically.

The concept "interlinked addresses" is explained at the following example:

An Example:

Two KNX devices have been defined in the ETS© with the physical addresses 1.2.3 and 1.2.4.

The **1/1** and the **10/0** group addresses are assigned to the device 1.2.3 and the **1/2** and the **10/0** are assigned to the device 1.2.4.

- Is a telegram 1/1 sends out with the value 1 to KNX, the value of the device 1.2.3 is put on 1.
- Is a telegram 1/2 sends out with the value 2 to KNX, the value of the device 1.2.4 is put on 2.
- Is a telegram 10/0 sends out with the value 3 to KNX, the value of the device 1.2.3 and the device 1.2.4 is put on 3.

However, the cells of the database of the NETxKNX server have the following values:

1/1 = 1, 1/2 = 2 und 10/0 = 3

If the values on to the 1/1 and the 1/2 addresses are queried by KNX read telegrams, the value 3 will be responded.

You can define in the current version of the system, that these addresses 1/1 and 1/2 shall be updated automatically with the value of the address 10/0 in the database of the server.

This functionality can be achieved about the link definition file.
For the upper example a link definition can be specified as follows:

10/0;BROADCAST;1/1;1/2

Therefore you define that if the value of the address 10/0 ("master link") changes, the values of the addresses 1/1 and 1/2 ("Sub link") will be set to the value of the 10/0 address in the servers' database automatically. **Important:** no telegrams will be sent, however, the attached OPC clients are informed about this change of the OPC items.

The link managers' mechanism can only be triggered by the arriving or outgoing telegrams. If the value of the master link address is set by other events, e.g. by "set Cell Value" tool in the Studio, the values of the Sub Link addresses won't be changed since the real value (of the KNX device) hasn't changed.

! The link manager must be activated in the system configuration file of the server. The system parameter should be set to "ON": **KNX.SetLinkOnReceive:ON, KNX.SetLinkOnSend:ON**.

The consequence can be restricted to the appropriate area with the detail of specific IP-addresses.

An Example:

10/0;192.168.1.100;1/1;1/2

only the addresses which are connected with Gateway "192.168.1.100" are set here.

It can be defined in a line up to 1000 Sub links; however, the definition can be inserted in several lines. There isn't any restriction for the number of assignments:

10/0;192.168.1.100;1/1;1/2;1/3;1/4
10/0;192.168.1.100;2/1;2/2;2/3;2/4

With this definition the value of addresses 1/1, 1/2, 1/3, 1/4, 2/1, 2/2, 2/3 and 2/4 become the value of address 10/0 automatically.

A bidirectional link can be established by defining an address as Master Link and as Sub Link as well:

10/0;192.168.1.100;1/1
1/1;192.168.1.100;10/0

If the value of address 10/0 changes the value of address 1/1 will be set to the same value. If the value of address 10/0 changes the value of address 1/1 will be set to the same value. Because these changes do not cause real KNX telegrams, it can not start an endless Loop.

Link Definition Table

The table is stored in the „nxaLinkDefinitions.35.dat" file, and can be edited.
For working on the data file please use the program Microsoft© Excel (CSV File Import).
This view allows the making of some little changes and extensions in the table. Eventually occurred errors can be corrected in a rather easy way. The pop up menu for working in the table can be opened by clicking on the right mouse button.

Insert new Definition – inserts a new line in the table, which can be filled with link definition data
Insert new Comment – inserts a comment line, where a text can be inserted.
Delete – deletes a chosen line
Convert to Definition – converts to definition data.
Convert to Comment – converts to comment line.

Column 1 „Master Link“
The KNX-group address („Master Link“). Its value will set to the Sub Links.
It can be used several times.

Column 2 IP Address
The IP-Address of the Gateways.
If “BROADCAST” gateway is entered, Master Link and all Sub Links addresses will have to be defined as “BROADCAST” in the telegram definitions file. In this case only the value is transferred to all Sub Links as well.

Column 3 to 1000 Sub Links
The „Sub Link“–KNX group addresses.
The values of these addresses are set by the value of the “Master Link” address

The "Link-Definition Table" can be prepared automatically. See: Import definitions from a (*.esf) ETS© file with the help of the **NETxKNXConvertETS** tool.

Event Processor

The NETxKNX server has the sub system: the "Event Processor". This system manages the “Events”.

There are three kinds of events, which are handled:

- **Response Event**
  The Response Events make it possible to build up routing functionalities. So certain KNX telegrams can be passed automatically to other gateways or their values can be passed to other addresses.
  These events are released by the arriving KNX telegrams.

- **Timer Event**
  The timer events make possible to send predefined telegrams at fixed times.

- **Cyclic Event**
  The cyclic Events can be used to send telegrams in defined time intervals - (e.g. to query the current values electric meters)
  It will regularly dispatched at KNX, and can be defined as "WRITE" (set value) or as "READ" (value to query) of telegrams.

All these tasks are accomplished independently of the OPC Clients - which relieves these very much. Since the server was conceived mainly for the data management, these tasks do not represent a large load for it.

The OPC Clients are certainly informed about each change and accomplished action, so that they have the current data available in any case.

Response Event Definitions Table

The "Response Event definition" table specifies the answer events. Each line of this file (excluded the comment lines) defines an event.

Response Event Definitions

	Active	Event Name	KNX Group Address to Send	IP Address to Send	Telegram Type to Send	Send Received Value	Value to Send	KNX Group Address to Monitor	IP Address to Monitor	Equal Value to Monitor	Less Value to Monitor	Greater Value to Monitor	Use Working Time	Working Time Begin	Working Time End
27															
28	F	Test1	1/2	BROADCAS	WRITE	F	1	6/1	BROADCAS			13			
29	F	Test2	1/3	BROADCAS	WRITE	F	1	6/1	BROADCAS		12				
30	F	Test3	1/4	BROADCAS	WRITE	F	1	6/1	BROADCAS			51			
31	F	Test4	1/5	BROADCAS	WRITE	F	1	6/1	BROADCAS		50				
32	F	14/1 = 14/2	14/1	192.168.1.1	WRITE	T	1	14/2	192.168.1.8						
33	F	Test6	9/79	192.168.1.2	WRITE	F	1	9/0	192.168.1.1	10					
34															

This table is loaded from the "nxaResponeEvents.35.dat" file ("EventFiles" directory of the current work area), and can be edited by the user. For working on the data file the Microsoft© Excel program (CSV file import) is recommended.

This view allows the making of some little changes and extensions in the table. Eventually occurred errors can be corrected in a rather easy way. The pop up menu for working in the table can be opened by clicking on the right mouse button.

Insert new Definition – inserts a new line in the table, this line can be filled with definition data.
Insert new Comment – inserts a comment line, which can be filled with a text
Convert to Definition – converts to definition data.
Convert to Comment – converts to comment line.
Delete – deletes a chosen line

Column 1
Active
It defines whether this event is enabled. If the value is „F“ („FALSE“) , this event will not loaded into the global Event table of the server.

Column 2
Event Name
It should be used a unique name for the event. It is used for displaying only.

Column 3
KNX Group Address to Send
If this event is caused by an in- or outgoing telegram, the Link Manager will send an KNX telegram to the here specified address.

Column 4
IP Address to Send
If this event is caused by an in- or outgoing telegram, the Link Manager will send an KNX telegram to the here specified address.

Column 5
Telegram Type to Send
It defines the type of telegram to send back. Choices are: „WRITE“(set value) und „READ“(read value). If the type is set to „ READ “, the values of the columns “Send Received Value “and „Value to Send “will be ignored.

Column 6
Send Received Value
This column defines whether a predefined or the received value should be send to the destination address.
If it is set to „T“(„TRUE“), the Event- Processor sends a telegram to the destination address with the value of the received telegram, which caused the event. In this case the next column will be ignored. If it is set to „F“ („FALSE), the value from the column „Value to Send“ will be sent.

Column 7
Value to Send (Optional)
It defines the value of the telegram.
If the value in the column „Send Received Value“ is „F“, this value will be sent as KNX-telegram to the destination address.

Column 8
KNX Group Address to Monitor
This column defines that only telegrams with the entered KNX Group Address can raise this event. If e.g. the address 1/1 is entered, und an KNX telegram with this address is received or sent, this event will be raised.
Because a NETxKNX-Address consists of the KNX Group address and the IP-address, the IP-address (Column 9) is always considered.

Column 9
IP Address to Monitor
It defines the IP-address part of the NETxKNX-address, which is to be monitored. (see column 8).

Column 10, 11, 12
Equal Value to Monitor (Optional)
Less Value to Monitor (Optional)
Greater Value to Monitor (Optional)

The next three columns define the values, which are to be supervised.

Three comparison operations are available
- Equal Value: If a telegram with exactly this value is received, this event will be raised.

- Less Value: If a telegram with value lower than this is received, this event will be raised.

- Greater Value: If a telegram with value greater than this is received, this event will be raised.

Column 13
Use Working Time (Optional
This column defines if the working time is to be considered.
The value „T“ determines, that the in the following columns defined working time is used. The event will be executed, if the actual time is between Working time begin“(column 14) and working time end (column 15).
If this column is set to „F“ or it is empty, the event monitoring will be active the whole time.

Column 14
Working Time Begin (Optional)
It defines when working time begins.

Column 15
Working Time End (Optional)
It defines when working time ends.

Example:

Use Working Time	Working Time Begin	Working Time End
T	6:30	18:42

This event is active from 6:30 (inclusive) to 18:42 (inclusive).

Column 16
Use Calendar (Optional)
This column defines if the Yearly Calendar and the Weekly Calendar are to be used. The value „T“ determines that the in the following columns defined Yearly Calendar and Weekly Calendar are used.
With „F“ or empty column the calendar is deactivated.

Column 17
Active Weekdays (Optional)
This column determines the Weekly Calendar Filter. This filter defines that this event is to be activated only on specific week days.
The entry consists of seven places, which in each case with "F" (for "FALSE" - event not actively) and T - (for "TRUE" - event actively) is to be entered. Each place stands for a weekday. The place 1 defines the filter for Sunday, place 2 for Monday, to place 7, which specifies the filter for Saturday.

Example: „FTTTTTF“ – this filter defines that this event is not implemented for Sundays and Saturdays.

Column 18
Month Day Begin (Optional)
This column determines from which day of the month (column 19) this event is activated. It applies only together with the value of the Column 19.

Column 19
Month Begin (Optional)
This column determines from which month this event is activated.
It applies only together with the value of the Column 18.

Column 20
Month Day End (Optional)
This Column defines, up to which day of the month (Column 19) this event is active. It applies only together with the value of the Column 21.

Column 21
Month End (Optional)
This column defines, up to which month this event is active. It applies only together with the value of the Column 20.

Example:

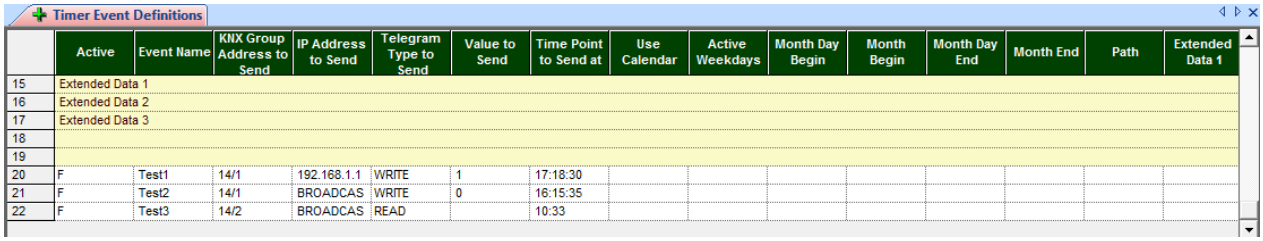
Use Calendar	Active Weekdays	Month Day Begin	Month Begin	Month Day End	Month End
T	FTTTTTF	10	1	15	1

This event will in each case be active Monday until Friday of 10 January (inclusive) until 15 January (inclusive).

Column 22 - 25
Path, Extended Data 1 - 3
These columns are for internal use only.

Timer Event Definitions Table

The "Timer Event Definition " table specifies the timer events. Each line of this file (excluded the comment lines) defines an event.



Timer Event Definitions

	Active	Event Name	KNX Group Address to Send	IP Address to Send	Telegram Type to Send	Value to Send	Time Point to Send at	Use Calendar	Active Weekdays	Month Day Begin	Month Begin	Month Day End	Month End	Path	Extended Data 1
15	Extended Data 1														
16	Extended Data 2														
17	Extended Data 3														
18															
19															
20	F	Test1	14/1	192.168.1.1	WRITE	1	17:18:30								
21	F	Test2	14/1	BROADCAS	WRITE	0	16:15:35								
22	F	Test3	14/2	BROADCAS	READ		10:33								

This table is loaded from the "nxaTimerEvents.35.dat" file ("EventFiles" directory of the current work area), and can be edited by the user. For working on the data file the Microsoft© Excel program (CSV file import) is recommended.

This view allows the making of some little changes and extensions in the table. Eventually occurred errors can be corrected in a rather easy way. The pop up menu for working in the table can be opened by clicking on the right mouse button.

Insert new Definition – inserts a new line in the table, this line can be filled with definition data.
Insert new Comment – inserts a comment line, which can be filled with a text
Convert to Definition – converts to definition data.
Convert to Comment – converts to comment line.
Delete – deletes a chosen line

Column 1
Active
It defines whether this event is enabled. If the value is „F“ („FALSE“) , this event will not loaded into the global Event table of the server.

Column 2
Event Name
It should be used a unique name for the event. It is used for displaying only.

Column 3
KNX Group Address to Send
If this event is caused by an in- or outgoing telegram, the Link Manager will send an KNX telegram to the here specified address.

Column 4
IP Address to Send
If this event is caused by an in- or outgoing telegram, the Link Manager will send an KNX telegram to the here specified address.

Column 5
Telegram Type to Send
It defines the type of telegram to send back. Choices are: „WRITE“ (write value) und „READ“(read value). If the type is set to „READ “, the values of the columns “Send Received Value“ and „Value to Send “will be ignored.

Column 6
Value to Send (Optional)
It defines the value of the telegram.

Column 7
Time Point to Send at
At this time the event is executed, what results with the dispatching of a telegram.

Column 8
Use Calendar (Optional)
This column defines if the Yearly Calendar and the Weekly Calendar are to be used. The value „T“ determines that the in the following columns defined Yearly Calendar and Weekly Calendar are used.
With „F“ or empty column the calendar is deactivated.

Column 9
Active Weekdays (Optional)
This column determines the Weekly Calendar Filter. This filter defines that this event is to be activated only on specific week days.
The entry consists of seven places, which in each case with "F" (for "FALSE" - event not actively) and T - (for "TRUE" - event actively) is to be entered. Each place stands for a weekday. The place 1 defines the filter for Sunday, place 2 for Monday, to place 7, which specifies the filter for Saturday.

Example: „FTTTTTF“ – this filter defines that this event is not implemented for Sundays and Saturdays.

Column 10
Month Day Begin (Optional)
This column determines from which day of the month (column 19) this event is activated. It applies only together with the value of the Column 19.

Column 11
Month Begin (Optional)
This column determines from which month this event is activated.
It applies only together with the value of the Column 18.

Column 12
Month Day End (Optional)
This Column defines, up to which day of the month (Column 19) this event is active. It applies only together with the value of the Column 21.

Column 13
Month End (Optional)
This column defines, up to which month this event is active. It applies only together with the value of the Column 20.

Column 14 - 17
Path, Extended Data 1 - 3
These columns are for internal use only.

Example:

Use Calendar	Active Weekdays	Month Day Begin	Month Begin	Month Day End	Month End
T	FTTTTTF	10	1	15	1

This event will in each case be active Monday until Friday of 10 January (inclusive) until 15 January (inclusive).

Cyclic Event Definitions Table

The "Cyclic Event Definition " table specifies the cyclic events. Each line of this file (excluded the comment lines) defines an event.

Cyclic Event Definitions

	Active	Event Name	KNX Group Address to Send	IP Address to Send	Telegram Type to Send	Value to Send	Time Interval (sec)	Use Working Time	Working Time Begin	Working Time End	Use Calendar	Active Weekdays	Month Day Begin	Month Begin	Month Day End
17	Path ("Root/Building1/Room1")														
18	Extended Data 1														
19	Extended Data 2														
20	Extended Data 3														
21															
22	F	Cyclic Event	14/1	192.168.1.1	READ		10	F							
23	F	Cyclic Event	14/2	BROADCAS	WRITE	1	60	T	7:15	18:00	F				
24	F	Cyclic Event	14/3	192.168.1.8	READ		180	F			T	FTTTTTF	1	2	1

This table is loaded from the "nxaTimerEvents.35.dat" file ("EventFiles" directory of the current work area), and can be edited by the user. For working on the data file the Microsoft© Excel program (CSV file import) is recommended.

This view allows the making of some little changes and extensions in the table. Eventually occurred errors can be corrected in a rather easy way. The pop up menu for working in the table can be opened by clicking on the right mouse button.

Insert new Definition – inserts a new line in the table, this line can be filled with definition data.
Insert new Comment – inserts a comment line, which can be filled with a text
Convert to Definition – converts to definition data.
Convert to Comment – converts to comment line.
Delete – deletes a chosen line

Column 1
Active
It defines whether this event is enabled. If the value is „F“ („FALSE“) , this event will not loaded into the global Event table of the server.

Column 2
Event Name
It should be used a unique name for the event. It is used for displaying only.

Column 3
KNX Group Address to Send
If this event is caused by an in- or outgoing telegram, the Link Manager will send an KNX telegram to the here specified address.

Column 4
IP Address to Send
If this event is caused by an in- or outgoing telegram, the Link Manager will send an KNX telegram to the here specified address.

Column 5
Telegram Type to Send
It defines the type of telegram to send back. Choices are: „WRITE“ (set value) und „READ“ (read value). If the type is set to „READ “, the values of the columns “Send Received Value“ and „Value to Send “will be ignored.

Column 6
Value to Send (Optional)
It defines the value of the telegram.

Column 7
Time Interval
In this time interval (in seconds) the events are cyclically executed.

Column 8
Use Working Time (Optional)
This column defines if the working time is to be considered.
The value „T“ determines, that the in the following columns defined working time is used. The event will be executed, if the actual time is between Working time begin“(column 14) and working time end (column 15).
If this column is set to „F“ or it is empty, the event monitoring will be active the whole time.

Column 9
Working Time Begin (Optional)
It defines when working time begins.

Column 10
Working Time End (Optional)
It defines when working time ends.

Example:

Use Working Time	Working Time Begin	Working Time End
T	6:30	18:42

This event is active from 6:30 (inclusive) to 18:42 (inclusive).

Column 11
Use Calendar (Optional)
This column defines if the Yearly Calendar and the Weekly Calendar are to be used.

The value „T“ determines that the in the following columns defined Yearly Calendar and Weekly Calendar are used.
With „F“ or empty column the calendar is deactivated.

Column 12
Active Weekdays (Optional)
This column determines the Weekly Calendar Filter. This filter defines that this event is to be activated only on specific week days.
The entry consists of seven places, which in each case with "F" (for "FALSE" - event not actively) and T - (for "TRUE" - event actively) is to be entered. Each place stands for a weekday. The place 1 defines the filter for Sunday, place 2 for Monday, to place 7, which specifies the filter for Saturday.

Example: „FTTTTTF“ – this filter defines that this event is not implemented for Sundays and Saturdays.

Column 13
Month Day Begin (Optional)
This column determines from which day of the month (column 19) this event is activated. It applies only together with the value of the Column 19.

Column 14
Month Begin (Optional)
This column determines from which month this event is activated.
It applies only together with the value of the Column 18.

Column 15
Month Day End (Optional)
This Column defines, up to which day of the month (Column 19) this event is active. It applies only together with the value of the Column 21.

Column 16
Month End (Optional)
This column defines, up to which month this event is active. It applies only together with the value of the Column 15.

Example:

Use Calendar	Active Weekdays	Month Day Begin	Month Begin	Month Day End	Month End
T	FTTTTTF	10	1	15	1

This event will in each case be active Monday until Friday of 10 January (inclusive) until 15 January (inclusive).

Column 17 - 20
Path, Extended Data 1 - 3
These columns are for internal use only.

Info desk

In both info areas of the Studio the user gets a series of information.
The first area can be found in the upper part of the studio user interface.
It will turn to red, if there is reported an error from NETxKNX OPC Server.

Send Interval

Shows the current value (milliseconds) of the gap, between which telegrams are sent to a gateway.

Last Cell Set

The address and the value of the last sent cell:
<Main group>/<middle group>/<sub group> @ <IP address> #<value>
or
< Main group >/< sub group > @ <IP address > # < value >

Telegrams Received

It shows the amount of the received telegrams since the last initializing of the server.

Telegrams Sent

It shows the amount of sent telegrams since the last initializing of the server.

Status

The current status of the server:
Running
Stopped

Start Date/Time

It shows the last time when the server was started.

Mode

It shows the Running Mode

- ONLINE
- OFFLINE
- SIMULATION

and name of workspace.

N-Mesh Mode
It shows the N-Mesh Mode

- Stand Alone Server
- Main Server
- Backup Server

and whether the Server is active or passive.

Date/Time
It displays the current time and date.

Server

On the one hand the server has the task to administrate all received and sent telegrams. On the other hand the server has to check the whole plant regarding occurring problems and if a problem occurs it will have to take action. This application can either be installed as "service" or as "normal" COM server.

The virtual model of the plant is continually saved on the hard disk. If the server is started again, the latest saved level of the model is loaded.
If a gateway disconnects itself from the main server and then reconnects, the current values of all the defined KNX equipment will be read out from the system.
This feature assures that the current level of the plant is mapped in total in the virtual model.

All telegrams are stored in queue and on the hard disk and can be used for analyses later on.

! No telegrams are routed to the OFFLINE gateways

The way how the server / service works

The difference between the two forms of installation is that the service is automatically started with the computer and no user has to be logged into the system.
The "normal" COM-Server form is not automatically started. An OPC client or the studio itself has to build up a connection to the server.
Both forms can be changed after installing.

Server registered as COM-Server:

NETxOPC.exe –unregserver
And then:
NETxOPC.exe –regserver

Operate in the NETxKNX System register:

Server registered as service:

NETxOPC.exe –unregserver
And then:
NETxOPC.exe –service

If the server is installed as service, it can be found under the name
„ [NXA] NETxKNX.OPC.Server.3.5“.

Configuration

The configuration of the NETxKNX OPC Server 3.5 is read out of the configuration file during starting. All those data files are stored in the subdirectory of the system.

Directories

Following directories are created during installation:

The system directories:
<Program Files>\NETxAutomation\NETxKNX.OPC.3.5.UD
Here all the application data files are stored

The sub directories:
<NXA_System>\Workspaces\<CurrentWorkspace>\ConfigFiles
Configuration data files

<NXA_System>\Workspaces\<CurrentWorkspace>\DataFiles
The telegram definition data files

<NXA_System>\Workspaces\<CurrentWorkspace>\EventFiles
The Event definition files.

<NXA_System>\Workspaces\<CurrentWorkspace>\LogFiles
all protocol files

<NXA_System>\Workspaces\<CurrentWorkspace>\ScriptFiles
all LUA ScriptFiles

<NXA_System>\Drivers
Only at Hardlock version

Router configuration: nxaOPCRouter.35.cfg

This file contains the configuration parameters of the gateway driver.

Parameter: **UDP.ReceiveOwnTelegrams**
Scope: <ON/OFF>
Default value: OFF
Unit: None
Description:
ON – it specifies that the telegrams with local IP address are accepted
OFF – the local telegrams are ignored

Parameter: **UDP.ReceiveBroadcastTelegrams**
Scope: <ON/OFF>
Default value: ON
Unit: None
Description:
ON – it specifies that the "BROADCAST" telegrams are accepted
OFF – the "BROADCAST" telegrams are ignored

Important:
The receiving of "BROADCAST" telegrams:

> If a KNX group address is assigned only to the "BROADCAST" gateway, all telegrams with this KNX address, which come from defined gateways (!) are interpreted as "BROADCAST" telegrams.
>
> However even if this KNX group address is still assigned to another gateway (e.g.: "192.168.1.2") - and a telegram comes with this KNX address from this gateway, the telegram will be interpreted as "192.168.1.2"-Gateway telegram.

Parameter: **UDP.SendBroadcastAsMultipleUnicast**
Scope: <ON/OFF>
Default-Wert: OFF
Unit: None
Description:
It defines whether the BROADCAST defined telegrams are really sent as IP broadcast packets or as unicast packets to each single gateway.

Parameter: **UDP.NetworkCardIPAddress**
Scope: IP address
Default-Wert: None
Unit: None
Description:
It specifies the network card which is to be used for the KNX telegrams for sending and receiving.

The network card is defined by its IP address.
If no IP address is defined, the valid network cards will be determined automatically by the system.

For **eibNode**:

Parameter: **eibNode.SenderNetID**
Scope: from 0 to 255
Default value: 0
Unit: None
Description:
It specifies the NetID for the outgoing telegrams.

Parameter: **eibNode.ReceiverNetIDFilter**
Scope: a list from 0 to 255 values separated by comma (e.g.: 1,2,33,4)
Default value: <empty>
Unit: None
Description:
It specifies the NetIDs of telegrams, which are to be accepted.
Is this parameter not specified all telegrams are accepted.

Example:
eibNode_ReceiverNetIDFilter: 0,2,3,7
Only the telegrams are accepted, which have the NetID 0,2,3, or 7.

For **IG/S 1.1**:

Parameter: **IGS.ReceiveMulticastAddress**
Scope: IP address
Default value: 239.192.39.238
Unit: None
Description:
It defines the multicast group (multicast IP address) of the project.

For **KNX NETIP**:

Parameter: **NETIP.NetworkCardIPAddress**
Scope: IP address
Default-Wert: None
Unit: None
Description:
It specifies the network card which is to be used for the KNX NETIP telegrams for sending and receiving.

Parameter: **NETIP.NAT**
Scope: <ON/OFF>
Default value: OFF
Unit: None
Description:
ON – should be used if there is a NAT Router or Firewall between IP gateway and OPC Server.

OFF – the default value

For **Direct(KNX)** Version:

Parameter: **Falcon.ConfirmedConnection**
Scope: <On/OFF>
Default-Wert: OFF
Unit: None
Description:
If connection to KNX Interface does not work, try Confirmed Connection set “ON”.

System configuration: nxaOPCSystem.35.cfg

Here the parameters relevant for the NETxKNX server are specified.

The parameters, which can be defined in this file, are divided in three groups:

- OPC
- KNX
- SYSTEM

Every parameter is identified with a name. The name is not "case sensitive" (small capitalization is not considered). The scope and the default value are fixed for each parameter in the system, and they are considered in the initialization phase.

Die OPC-Parameters:

Parameter: **OPC.GroupAddressType**
Scope: 2level, 3level
Default value: 3level
Unit: None
Description:
It defines how the OPC ItemIDs are build up.

2Level – defines, that the ITEM ID displays the log. KNX address part two-step
3Level – or three-step

ATTANTION: In the opposite of the in the studio defined parameter this has a crucial influence on the structure of all OPC ITEM IDs.

Example:

The same dada point:
OPC.GroupAddressType: 2Level -> ITEM ID = „\NETxKNX\192.168.1.1**00/0513**“
OPC.GroupAddressType: 3Level -> ITEM ID = „\NETxKNX\192.168.1.1**00/2/001**“

Parameter: **OPC.PrefixedItemID**
Scope: ON, OFF
Default value: ON
Unit: None
Description:
It defines whether the leading „0“-characters shall be added to OPC ITEM ID.

ON – defines that the ITEM ID displays the log. KNX-address part with leading „0“-characters
OFF – defines that the ITEM ID displays the log. KNX-address part without leading „0“-characters

ATTANTION: This parameter has crucial influence on the structure of all OPC ITEM IDs.

Example:

The same dada point:
OPC.PrefixedItemID: ON -> ITEM ID = „\NETxKNX\192.168.1.1**00/2/001**“
OPC.PrefixedItemID: OFF -> ITEM ID = „\NETxKNX\192.168.1.1**0/2/1**“

Parameter: **OPC.AsyncTimeout**
Scope: from 1 to 60
Default value: 5
Unit: seconds
Description:
It defines the timeout value for asynchronous OPC queries.
If no answer to the OPC query comes within this time, the quality of all queried data points will be set to UNCERTAIN.

Parameter: **OPC.AsyncReadFromDevice**
Scope: ON, OFF
Default value: OFF
Unit: None
Description:
It specifies whether the asynchronous OPC value read queries ("READ") shall be answered with data from the data base, or over a query of the value of the associated KNX device.

ON – the KNX devices are queried
OFF – the values are read from the NETxKNX server data base

Parameter: **OPC.AsyncRefreshFromDevice**
Scope: ON, OFF
Default value: OFF
Unit: None
Description:
It specifies whether the asynchronous OPC value read queries ("REFRESH ") shall be answered with data from the data base, or over a query of the value of the associated KNX device.

ON – the KNX devices are queried
OFF – the values are read from the NETxKNX server data base

Parameter: **OPC.ShowETSStructure**
Scope: ON,OFF
Default value: OFF
Unit: None
Description:
It specifies whether to show ETS like Structure Tree

ON – show ETS like Structure Tree
OFF – shows NxA like Structure Tree

Parameter: **OPC.QualityForValueNotSet**
Scope: UNCERTAIN,BAD,NOT_CONNECTED
Default value: UNCERTAIN
Unit: None
Description:
It specifies which OPC Quality value should be shown for not set item's value.

Parameter: **OPC.Delimiter**
Scope: a string, but it should not contain "
Default value: \
Unit: None
Description:
It specifies which delimiter for OPC Item ID tree structure should be used.

Following OPC parameters are available only for compatibility reasons. In future versions they may vanish:

Parameter: **OPC.AsyncWriteConfirmation**
Scope: ON,OFF
Default value: OFF
Unit: None
Description:
It specifies whether the asynchronous OPC value read queries ("WRITE ") shall be answered with data from the data base, or over a query of the value of the associated KNX device.

ON – the KNX devices are queried
OFF – the values are read from the NETxKNX server data base

Attention: If the parameter is set on "ON", to each "write" command will result in sending up to three KNX telegrams. The telegram quantity is if necessary trebled.

Parameter: **OPC.RefreshOnEqualValue**
Scope: ON,OFF
Default value: ON
Unit: None
Description:
It specifies whether the "OnDataChange" events are generated by KNX of telegrams with unchanged values.

ON – the "OnDataChange" of events will be generated
OFF – the "OnDataChange" of events will be generated only with changes of value

Example:

If the OPC server receives two KNX telegrams with same value and same address (IP + log KNX address) and is set the option "OPC.RefreshOnEqualValue" on "OFF", the "OnDataChange" event is generated only for the first telegram, since the second telegram does not cause changes of value of the data point.
If this option is set on "ON", then two "OnDataChange" events are generated.

The KNX-Parameters:

Parameter: **KNX.ServerPhysicalAddress**
Scope: from 0.0.0 to 15.15.255
Default value: 15.15.255
Unit: none
Description:
It specifies the physical KNX address of the OPC of server.
This address is used by the outgoing KNX telegrams.

Parameter: **KNX.Timeout**
Scope: from 2 to 60
Default value: 10
Unit: seconds
Description:
It specifies the timeout value for the system internal "value read" queries of KNX devices. This value is used by "NETxKNX EventEngine" subsystem for the "READ" events and the value queries in the initialization phase of the system.

Parameter: **KNX.PhysicalDeviceTimeout**
Scope: from 1 to 600
Default value: 30
Unit: seconds
Description:
It specifies the timeout value for the KNX devices check telegrams. This value is used by "NETxKNX Device Manager" subsystem.

Parameter: **KNX.CyclicEventStartDelay**
Scope: from 10 to 600
Default value: 30
Unit: seconds
Description:
It specifies the start delay for Cyclic Events (in seconds). This value is used by "NETxKNX Cyclic Event Manager" subsystem.

Parameter: **KNX.CyclicEventTelegramGap**
Scope: from 10 to 600
Default value: 30
Unit: seconds
Description:
It specifies the time gap between Cyclic Events (in milliseconds). This value is used by "NETxKNX Cyclic Event Manager" subsystem.

Parameter: **KNX.SetLinkOnReceive**
Scope: ON,OFF
Default value: OFF
Unit: None
Description:
The "link manager" the subsystem of the NETxKNX of server makes possible the managing of linked KNX group addresses. This parameter specifies whether the mechanism is to be released by arriving KNX telegrams.

ON – sets the linked KNX group addresses when receiving KNX telegrams
OFF – not activated

Parameter: **KNX.SetLinkOnSend**
Scope: ON,OFF
Default value: OFF
Unit: None
Description:
The "link manager" the subsystem of the NETxKNX of server makes possible the managing of linked KNX group addresses. This parameter specifies whether the mechanism is to be released by sending KNX telegrams.

ON – sets the linked KNX group addresses when sending KNX telegrams
OFF – not activated

Parameter: **KNX.ShowUndefinedTelegrams**
Scope: ON,OFF
Default value: OFF
Unit: None
Description:
All undefined arriving KNX telegrams will be shown in the OPC Studio System Messages window.

ON –shows undefined KNX telegrams
OFF – not activated

Following KNX parameters are available only for compatibility reasons. In future versions they may vanish:

Parameter: **KNX.RefreshAllValuesOnConnect**
Scope: ON,OFF
Default value: OFF
Unit: None
Description:
If a gateway is reconnecting to the server, the server can poll all data points (KNX values), which for this gateway was defined, for the current values.

ON – the KNX devices are polled
OFF – not activated

Parameter: **KNX.CyclicRefreshAllValues**
Scope: ON,OFF
Default value: OFF
Unit: None
Description:
In the version 3.0 it is possible that the system automatically accomplish updating the internal data base in defined time intervals. The updates are realized over cyclic "READ" KNX telegrams, which of course will have to be considered in the maximum stress of the KNX system. The configuration of this subsystem is realized with the help of some, here listed parameters

ON – automatic updating is activated
OFF – not activated

Parameter: **KNX.CyclicRefreshAllValuesInterval**
Scope: from 5 to 86400
Default value: 10
Unit: second
Description:
It specifies, in which time intervals the value query process is to be started.
Here it must be considered that all defined data points are queried. Thus the more data points during an update process to be queried, the more largely must be this value. If the value is too small specified, it will lead to the fact that two update processes overlap temporally.

Example:

10 data points for gateway 192.168.1.1 are defined.
The global system end interval is set on 1 second ("send interval" can only be set in the OPC studio).

i.e.:
An update process takes 10 seconds in this case. If the "KNX_CyclicRefreshAllValuesInterval" parameter is set on 5 seconds, the next update process already starts after 5 seconds and tries to send query telegrams. Since they cannot be sent away immediately (because the telegrams by the process 1 are being sent away) these remain in the queue of the server.
Here also the "KNX_CyclicRefreshAllValuesTelegramGap" parameter must be considered.

Parameter: **KNX.CyclicRefreshAllValuesStartDelay**
Scope: from 10 to 600
Default value: 60
Unit: seconds
Description:
It specifies, in which time interval after the initialization of the server the first query process is to be started.

Parameter: **KNX.CyclicRefreshAllValuesTelegramGap**
Scope: from 100 to 3000
Default value: 250
Unit: **milliseconds**

Description:
It specifies, in which time interval the particular query telegrams of an update process are to be sent away.
This value must be **larger** than the value of the global system send interval.

The SYSTEM-Parameters:

Parameter: **SYS.UseTelegramDataFile**
Scope: ON,OFF
Default value: ON
Unit: None
Description:
It specifies whether at run-time the virtual image of the plant (the values of all defined data points) is to be provided and written on the disk.

Parameter: **SYS.MaxSizeOfTelegramLogFile**
Scope: from 1 to 1000
Default value: 10
Unit: Megabyte
Description:
It specifies the size of the telegram log file.

Parameter: **SYS.EnableDeviceManager**
Scope: ON,OFF
Default value: ON
Unit: None
Description:
It enables the Physical KNX Device Management.

Parameter: **SYS.GenerateRandomValues**
Scope: ON,OFF
Default value: ON
Unit: None
Description:
It generates random values for OPC Items in simulation mode.

Parameter: **SYS.EnableOperationTime**
Scope: ON,OFF
Default value: OFF
Unit: None
Description:
The OPC Server will calculate the Operation Time in Seconds of datapoints (for EIS1 data type only).

Parameter: **SYS.GeoLatitude**
Scope: from 0 to 360
Default value: 0
Unit: None
Description:

The geographic latitude in degrees, + north, - south

Parameter: **SYS.GeoLongitude**
Scope: from 0 to 360
Default value: 0
Unit: None
Description:
The geographic longitude in degrees, + east, - west

Parameter: **SYS.GeoElevation**
Scope: from 0 to 10000
Default value: 0
Unit: None
Description:
The geographic elevation in meters

Following KNX parameters are available only for compatibility reasons. In future versions they may vanish:

Parameter: **SYS.GatewayConnectionTimeout**
Scope: from 10 to 10000
Default value: 500
Unit: **milliseconds**
Description:
It specifies the timeout value for the connecting checking of the attached IP gateways.

Parameter: **SYS.GatewayConnectionTimeoutAttemptCount**
Scope: from 1 to 10
Default value: 3
Unit: None
Description:
It specifies the number of call attempts for the checking of the connection of the attached IP gateways in the case of an interruption.

The Database-Parameters:

These parameters will be only used, if the extension module Microsoft SQL Database interface is licensed.

Parameter: **DB.Enabled:**
Scope: ON,OFF
Default value: OFF
Unit: None
Description:
It enables the extension module Microsoft SQL Database interface.

Parameter: **DB.User:**
Scope: String
Default value:
Unit: None
Description:

It defines the Database User Name.

Parameter: **DB.UserPassword:**
Scope: String
Default value:
Unit: None
Description:
It defines the Database User Password.

Parameter: **DB.ODBC:**
Scope: String
Default value:
Unit: None
Description:
It defines the name of the ODBC that should be used to connect to Database.

Parameter: **DB.Database:**
Scope: String
Default value:
Unit: None
Description:
It defines the name of the database.

Parameter: **DB.Table:**
Scope: String
Default value:
Unit: None
Description:
It defines the name of the database's table.

Parameter: **DB.ExpireAfterDays:**
Scope: from 0 to 600
Default value: 0
Unit: None
Description:
The older date will be automatically deleted from the database (in days).

Telegram definitions: nxaTelegramDefinitions.35.dat

The main data file of the system is the telegram definition data.
Here all for the system relevant telegrams are stored.

The structure of the data file:

```
'Syntax of the Telegram Definition Table:
'log.KNX Address;IP Address;Priority;Data Size;EIS Type;Signed/Unsigned;Unit;Description;Path;Control Data;Alias;Read on Reconnect;Read Cyclically Interval;Extended Data
'
1/0;BROADCAST;LOW;1BIT;EIS1;;;Telegram Demo EIS1;;;;F;0;
2/0;BROADCAST;LOW;4BIT;EIS2;;;Telegram Demo EIS2;;;;F;0;
3/0;BROADCAST;LOW;3BYTE;EIS3;;;Telegram Demo EIS4;;;;F;0;
4/0;BROADCAST;LOW;3BYTE;EIS4;;;Telegram Demo EIS4;;;;F;0;
5/1;BROADCAST;LOW;2BYTE;EIS5;;°C;Telegram Demo EIS5 1;;;;F;0;
5/21;BROADCAST;LOW;2BYTE;EIS5;;mA;Telegram Demo EIS5 2;;;;F;0;
```

Each line - except the comment lines (`) - defines a telegram.

Column 1 - log.KNX Address
It defines the logical KNX address of a telegram. It can be shown in both viewing forms, e.g.: in three steps 0/2/1 or in double steps 0/513.

Column 2 - IP Address
It defines the IP address of the gateway, which can receive or send the telegram.

Those two columns define the real address of the telegram (it is internally used by the NxA system)
The system allows many telegram definitions (up to 100) with the same log. KNX address. But the requirement is that all of them do have different IP addresses.

So it is possible that behind every gateway an additional ETS address room can exist. Theoretically up to 1000 ETS projects with the same address areas are possible within the system.
The name "BROADCAST" defines a telegram which is sent to all defined gateways, but which cannot be received from the main server.

Column 3 - Priority
It defines the priority of the telegram. Possible statues: SYSTEM, HIGH, ALARM and LOW. The standard status „LOW“ should be used in general.

! The columns 4 to 7 define how to interpret the received data. The data type of a telegram is defined by a minimum of two attributes. All possible data type definitions (combination of attributes) are listed in the data type table.

Column 4– Data Size
It defines the size of the data type, where the value of the telegram is passed on.
Possible values: 1BIT, 2BIT, 4BIT, 1BYTE, 2BYTE, 3BYTE, 4BYTE, 10BYTE, 14BYTE
For simple switch telegrams the type 1BIT is recommended.

Column 5– EIS Type
Here the data type regarding the **EIB Interworking Standard (EIS)** is fixed.

Column 6– Signed/Unsigned
This attribute is only used by a few data types. Here it is defined, whether a data type is interpreted with or without the sign.
For all other data types this field has to be left out.

Column 7- Unit
It shows us the unit of the data type. This cannot be entered in a free form. It has to be selected from the list.

Column 8- Description
The last column defines the telegram text. This is also displayed in the telegram monitor of the studio.

Column 9- Path
This is used for the automatic PDA visualization to create a tree structure like “Home/Level1/living room”.

Column 10- Control Data
This is used for additional data for the automatic PDA visualization.

Column 11- Alias
This defines an alias name for a new OPC Item. This is also displayed in the OPC Tree of the studio.

Column 12- Read on Reconnect
This defines that a READ Telegram will be sent, if IP Gateway is reconnected and actual value will be received from KNX. T means true and enables the function, with F for false the function could be disabled. Read flag in KNX device has to set of course.

Column 13- Read Cyclically Interval
This defines that a READ Telegram will be sent at periodic interval and actual value will be received from KNX cyclically. Be aware that this costs bandwidth in KNX Line. If the value is set to 0, the function is disabled, if a value between 1 and 3600 is entered, the function is enabled and the value defines the interval in seconds.

Assuming the OPC Server can send maximum 4 telegrams per second per gateway, you could use 1 telegram per second and gateway for “Read Cyclically” telegrams. So if you have e.g. 1000 group addresses per gateway to read cyclically, you can calculate 1000 telegrams / 1 telegram per second = 1000 seconds as time interval for “Read Cyclically” for these 1000 group addresses for this gateway.
These send requests are cached and wait for free slot to be sent, if no other higher priority send request comes from OPC Clients or N-Mesh.

Column 14- Extended Data
This is used to add any comments.

The „nxaTelegramDefinitions.35.dat“ file can also be made by the help of the additional program „NETxKNXConvertETS“ (more or less automatically out of the exported ETS projects)
Please see: The converting tool: NETxKNXConvertETS“

A telegram is identified with the logical KNX address and the IP address. It is not allowed to have two telegram definitions with the same address combination.

Gateway definitions: nxaGatewayDefinitions.35.dat

The gateway manager administrates the LAN-coupler, which is connected to the system. The gateway definition table defines all the gateways which should be considered.

! No telegrams are sent to the OFFLINE gateways.

```
'Syntax of the Gateway Definition Table:
'IP Address;Type;Port;Name;Locality;Description ;Options
'
192.168.1.1;IGS;51000; GATE 1;Room 51;pre-defined Gateway 1
192.168.1.2;NETIP;3671;Gate2;;;;;;;
192.168.1.3;EIBNODE;1634;Gate3;;;;;;;
```

The structure of the data file

Each line - except the comment line (`) defines a gateway.

! The virtual BROADCAST gateway is made by the system and cannot be defined in a table.

Column 1 - IP - Address
The IP address of the gateway is used as a keyword and can only be entered once only.

Column 2 - Type
The type of the gateway (IGS, NETIP,EIBNODE or VIRTUAL)

Column 3 - Port
The socket-port number, which is inserted in the gateway, is shown.

Column 4 - Name
The name of the gateway (can be defined)

Column 5 - Locality
The location of the equipment (can be defined)

Column 6 - Description
The description of the equipment (can be defined)

Column 7 - Options
If used with gateway type “EIBNODE”, this is the SenderNetID for this gateway e.g. 1.

Column 8 - 11
Path, Extended Data 1 - 3
These columns are for internal use only.

Device definitions: nxaDeviceDefinitions.35.dat

The device manager can check a KNX Device availability. It tries to connect the defined device, whether it can be reached. If it fails, the OPC Item's value of this device will become false. It can be used to check for some important devices. Of course this cost bandwidth in KNX Line, so it should be used only if necessary. These telegrams are not group telegrams and are not shown in the Telegram Monitor. The device definition table defines all the devices which should be checked.

This feature is not available in Direct(KNX) version of the server!

! The system parameter "SYS.EnableDeviceManager" has to be set ON.

```
'Syntax of the Device Definition Table:
'physical EIB Address;IP Address;Polling interval (sec);Description;Alias
'
1.1.1;192.168.1.7;20;Device 1;Switch 1
1.1.2;192.168.1.8;40;Device 2;
```

The structure of the data file

Each line - except the comment line (`) defines a device.

Column 1 – physical KNX Address
The physical KNX Address of the device, which should be connected.

Column 2 - IP - Address
This defines the IP address of the gateway via which device is reachable. "BROADCAST" is not supported in this case.

Column 3 – Polling interval
This defines the polling interval in seconds.

Column 4 – Description
This column defines the device text. This is also displayed in the OPC Tree of the studio.

Column 5 – Alias
This defines an alias name for a new OPC Item. This is also displayed in the OPC Tree of the studio.

N-Mesh Subsystem Config File: nxaNMesh.35.cfg

The N-mesh subsystem is used to setup Main and Backup Server and Clustering. Main and Backup Server must have the same basic configuration (e.g. Telegram definition, gateway definition,…).
Servers for Clusters need the same telegram definition and gateway definition. Only changes in gateway ports or eibNet IDs are allowed to prevent conflicts in connecting the gateways.
N-Mesh only works within the same release version.

Parameter: **NMESH.UseRedundancy**
Scope: ON,OFF
Default value: OFF
Unit: None
Description:
It enables redundancy feature and has to be set “ON” at Main and Backup Server.

This feature is enabled at 3 IP Gateways UnifiedDriver and is not available in the Direct(KNX) Version.

Parameter: **NMESH.EnableSynchronization**
Scope: ON,OFF
Default value: OFF
Unit: None
Description:
It enables synchronization between Main and Backup Server. So the standby server will get update information of all changes in the database. This helps the server to be able to get active at very short notice.

Parameter: **NMESH.MainServerIPAddress**
Scope: IP address
Default value: 192.128.1.1
Unit: None
Description:
It defines the IP address of Main Server.

Parameter: **NMESH.BackupServerIPAddress**
Scope: IP address
Default value: 192.128.1.2
Unit: None
Description:
It defines the IP address of Backup Server.

Parameter: **NMESH.NetworkCardIPAddress**
Scope: IP address
Default value: 192.128.1.1
Unit: None
Description:

It defines the IP address of the Network card, which should be used to communicate with all N-Mesh Server.

Parameter: **NMESH.NetworkPortNumber**
Scope: 0..65355
Default value: 20556
Unit: None
Description:
It defines the Port number, which should be used to communicate with N-Mesh Server.

Parameter: **NMESH.StartDelay**
Scope: from 0 to 60
Default value: 10
Unit: seconds
Description:
It defines the delay during startup.

Parameter: **NMESH.ConnectionTimeout**
Scope: from 10 to 10 000
Default value: 500
Unit: milliseconds
Description:
It defines the time out of the connection, after that the Backup Server comes to the conclusion that Main Server is gone and has to connect gateways to assure redundancy of the system.

Parameter: **NMESH.EnableRouting**
Scope: ON / OFF
Default value: OFF
Unit: None
Description:
It enables the N-Mesh Routing of OPC Items defined in N-Mesh Routing definition file.

Parameter: **NMESH.Node**
Scope: IP address
Default value: missing
Unit: None
Description:
This is used for Clustering of NETxKNX OPC Servers.
If it is found, all update information is sent to this N-Mesh node of NETxKNX OPC Server. The other N-Mesh Node must have also this entry with the opposite IP address. No N-Mesh routing definition is necessary. N-Mesh nodes need the same telegram definition to store the receiving information, otherwise it is neglected.

N-Mesh Routing definitions: nxaNMeshRoutingDefinitions.35.dat

The N-Mesh Routing manager uses these definitions to route update information of OPC items to another NETxKNX OPC Server.

! The N-Mesh Routing parameter “NMESH.EnableRouting “ has to be set ON.

```
'Syntax of the N-Mesh Routing Definition Table:
'OnReceive;OnSent;OnSetValue;KNX Address to Monitor;IP Address to Monitor;KNX Address to
Send;IP Address to Send;Destination IP Address;Send Telegram Remotely
T;T;T;14/1;BROADCAST;14/1;BROADCAST;192.168.1.61;F
```

The structure of the data file

Each line - except the comment line (`) defines a N-Mesh Routing.

Column 1 – OnReceive
This defines if it should route on receiving of a telegram.

Column 2 – OnSent
This defines if it should route on sending of a telegram.

Column 3 – OnSetValue
This defines if it should route on setting the value of this group address.

To prevent too much traffic from above 3 configuration possibilities obey the following:

If an OPC Client changes an OPC item value, OPC Server will send it (OnSent), unless it is not in the virtual address range, and then also set the value of the cell (OnSetValue).

If KNX Gateway sends telegram to OPC server, the OPC server receives the telegram (OnReceive) and then set also the value of the cell (OnSetValue).
If N-mesh receives a request, it will set the value of the cell only (OnSetValue).

On basis of these event situations an N-mesh routing will be initiated.

Column 4 – KNX Address to Monitor
This defines the KNX Address to monitor.

Column 5 – IP Address to Monitor
This defines the IP Address of the gateway to monitor.

Column 6 – KNX Address to Send
This defines the KNX Address to send.

Column 7 – IP Address to Send
This defines the IP Address of the gateway to send.

Column 8 – Destination IP Address
This defines the IP Address of the NETxKNX OPC server to receive the update information.

Column 9 – Send Telegram Remotely
This defines whether the NETxKNX OPC server has to send also a telegram to the KNX IP gateway. This allows routing between KNX lines via different NETxKNX OPC Servers.

[Live] Task definitions: nxaTaskDefinitions.35.dat

These definitions are used to link Items inside NETxKNX OPC Server. Based on source and destination Item definition and event options the value of source Item will be linked to value of destination Item. Also it is possible to delay this forwarding of the value, to generate KNX Telegrams or to execute LUA scripts as well.

Tasks

	Source ItemID	Destination ItemID	OnReceive [T]	OnSent [F]	OnSetValue [T]	Delay in ms [0]	Command [WRITE]	Parameters [<none>]
1	\NETxKNX\BROADCAST\03/0/010	\NETxKNX\BROADCAST\03/0/020	F	T	F	0	SCRIPT	nxa.LogInfo("Hello!")
2	\NETxKNX\BROADCAST\03/0/010	\NETxKNX\BROADCAST\03/0/020	T	T	T	0	SCRIPT	nxaTest()
3	\NETxKNX\BROADCAST\03/0/010	\NETxKNX\BROADCAST\03/0/020	T	T	T	0	WRITE	

Column 1 – Source Item
This defines the Source Item.

Column 2 – Destination Item
This defines the Destination Item.

Column 3 – OnReceive
This defines if it should route on receiving of a telegram.

Column 4 – OnSent
This defines if it should route on sending of a telegram.

Column 5 – OnSetValue
This defines if it should route on setting the value of this group address.

Column 6 – Delay in ms
This delays routing in milliseconds.

Column 7 – Command
This will execute a command:

- WRITE: it will generate also a KNX write telegram for destination item
- READ: it will generate also a KNX read telegram for destination item.
- SET: it will set the value in destination item
- SCRIPT: it will execute a LUA script.

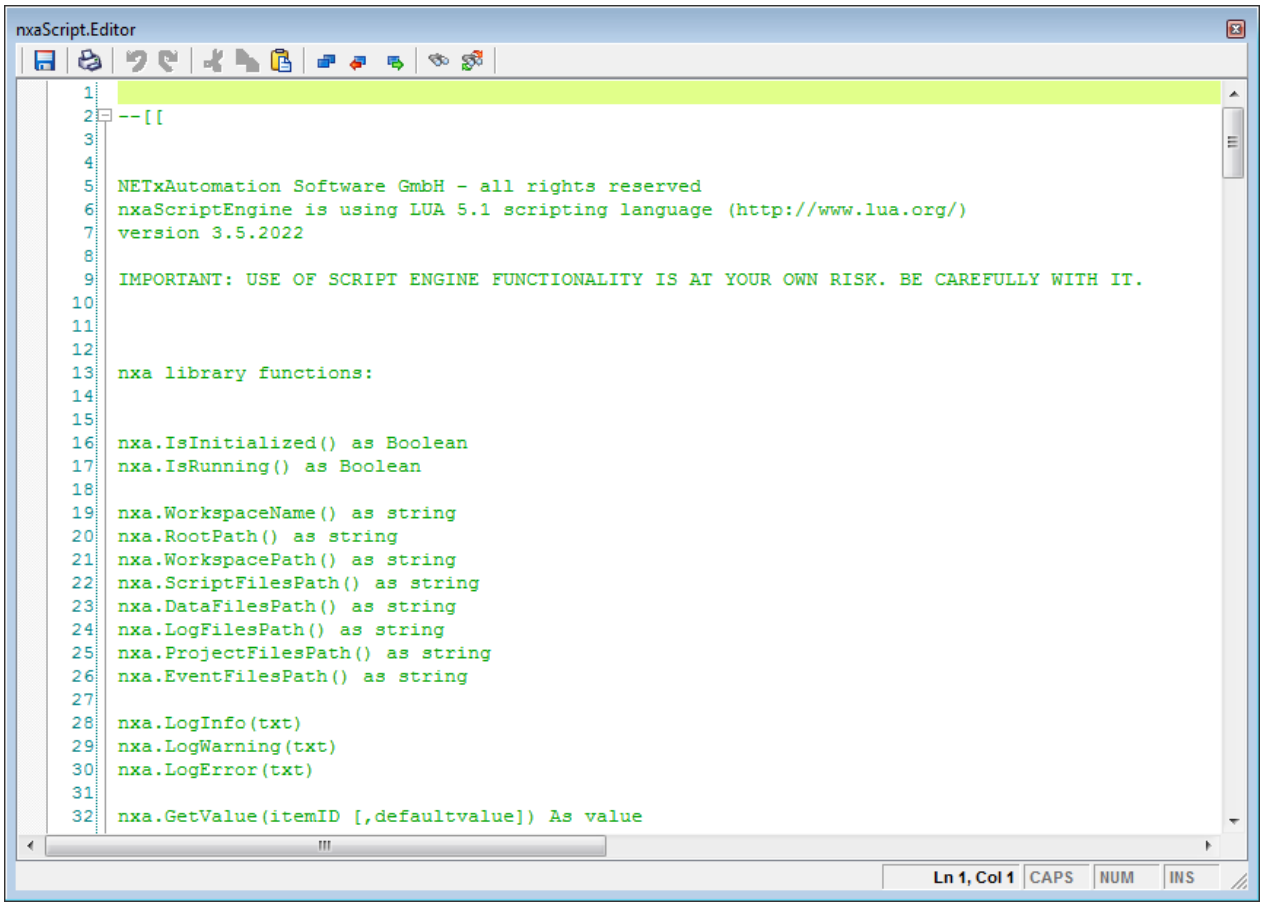
Column 8 – Parameters
It is used to define the LUA script, which should be executed.

LUA script file: nxaDefinitions.lua

If you click at the Edit Script Button, it will open the nxaDefinitions.lua file, which you find in the ScriptFiles directory of your workspace directory.

In this file in the comment section you see, which built-in functions from the scripting API are available.

For syntax of LUA script programming language please look at www.lua.org, where you find a lot of information about how to program with LUA. LUA is an open source scripting language, which is easy to program and is extremely fast.



The nxa Library functions

nxa.IsInitialized() as Boolean
It will deliver TRUE, if server is initialized, otherwise FALSE.

nxa.IsRunning() as Boolean
It will deliver TRUE, if server is running, otherwise FALSE.

nxa.WorkspaceName() as string
It will deliver the name of the current workspace as string.

nxa.RootPath() as string
It will deliver the name of the root path of the installed server application as string.

nxa.WorkspacePath() as string
It will deliver the name of the current workspace path as string.

nxa.ScriptFilesPath() as string
It will deliver the name of the current ScriptFiles path of the current opened workspace as string.

nxa.DataFilesPath() as string
It will deliver the name of the current DataFiles path of the current opened workspace as string.

nxa.LogFilesPath() as string
It will deliver the name of the current LogFiles path of the current opened workspace as string.

nxa.ProjectFilesPath() as string
It will deliver the name of the current ProjecttFiles path of the current opened workspace as string. This is the path of the Smart Voyager project files (*.vxf). It is used only at Voyager Server.

nxa.EventFilesPath() as string
It will deliver the name of the current EventFiles path of the current opened workspace as string.

nxa.LogInfo(txt)
This will generate an Info message with given parameter “txt” as String in System messages and System Logfile.

nxa.LogWarning(txt)
This will generate a Warning message with given parameter “txt” as String in System messages and System Logfile.

nxa.LogError(txt)
This will generate an Error message with given parameter “txt” as String in System messages and System Logfile.

nxa.GetValue(itemID [,defaultvalue]) As value
This will query an item referred by itemID as string and return it’s actual value, if quality of item is GOOD or if UNCERTAIN, it will return the given default value.

nxa.SetValue(itemID, value [,delay_in_ms])
This will set the value of an item referred by itemID as string. Optional it is possible to delay the setting of the value.

nxa.WriteValue(itemID, value [,delay_in_ms])
This will generate a WRITE telegram with the value of an item referred by itemID as string. Optional it is possible to delay the writing of the value.

nxa.ReadValue(itemID [,delay in ms])
This will generate a READ telegram with the value of an item referred by itemID as string. Optional it is possible to delay the reading of the value.

nxa.PropertyValue(itemID, propertyID)
This will query a property referred by propertyID as string of an item referred by itemID as string and return its actual property value as value.

NXA Library functions for Task Definitions

These functions are used in combination of Task definitions, where script files are executed and help to implement reactions.

nxa.SourceItemID() as itemID
This will query the itemID of the source item of the task definition, which has called the function.

nxa.DestinationItemID() as itemID
This will query the itemID of the destination item of the task definition, which has called the function.

nxa.SetDestinationValue(value)
This will set the value of the destination item of the task definition, which has called the function.

nxa.WriteDestinationValue(value)
This will generate a WRITE telegram with the value to the destination item of the task definition, which has called the function.

nxa.ReadDestinationValue()
This will generate a READ telegram with the value to the destination item of the task definition, which has called the function.

nxa.InputValue() as value
This will query the input value of the source item of the task definition, which has called the function and return it as value.

nxa.GetItemID(knx_address [, gateway_ip_address]) as itemID
This will query the itemID refered by KNX logical group address as string and the Gateway IP Address as string, where this logical group address is defined and return it as itemID.

nxa.ExecuteDelayedScript(script [,delay_in_ms])
This will execute a script given by parameter as string and optional it will be executed by given delay in milliseconds.

nxa.AddWriteTask(sourceItemID, destinationItemID, onReceive, onSend, onSet, delay_in_ms [,constValue])
This will add a new task definition with WRITE command during runtime.
Parameters are:

- source itemID as string
- destination item ID as string
- onReceive as bool
- onSend as bool
- onSet as bool
- delay in milliseconds as value
- optional: constValue as value, which will be used instead of input value from source item

nxa.AddReadTask(sourceItemID, destinationItemID, onReceive, onSend, onSet, delay_in_ms)
This will add a new task definition with READ command during runtime and it will be executed, if an onReceive, onSend or onSet event happens.
Parameters are:

- source itemID as string
- destination item ID as string
- onReceive as bool
- onSend as bool
- onSet as bool
- delay in milliseconds as value
- optional: constValue as value, which will be used instead of input value from source item

nxa.AddSetTask(sourceItemID, destinationItemID, onReceive, onSend, onSet, delay_in_ms [,constValue])
This will add a new task definition with SET command during runtime and it will be executed, if an onReceive, onSend or onSet event happens.
Parameters are:

- source itemID as string
- destination item ID as string
- onReceive as bool
- onSend as bool
- onSet as bool
- delay in milliseconds as value
- optional: constValue as value, which will be used instead of input value from source item

nxa.AddScriptTask(sourceItemID, destinationItemID, onReceive, onSend, onSet, delay_in_ms, script)
This will add a new task definition with SCRIPT command during runtime and it will be executed, if an onReceive, onSend or OnSet event happens.
Parameters are:

- source itemID as string
- destination item ID as string
- onReceive as bool
- onSend as bool
- onSet as bool
- delay in milliseconds as value
- script as string, which will be executed, if an onReceive, onSend or onSet event happens.

NXA Library functions for creating Custom Items

These functions are to create custom items, which will appear after start of server.

nxa item data types:

These define the data types for items:

nxa.type.Integer
nxa.type.Real
nxa.type.Date
nxa.type.String
nxa.type.Boolean

nxa item access rights types:

An item can be read or written.

nxa.access.Readable
nxa.access.Writeable
nxa.access.All

nxa.AddCustomItem(itemName, description, nxa_access_rights, nxa_data_type [,delimiter, sub_path_1 ,sub_path_2, ..]) as itemID
This will add a new custom item during first initial phase of server and returns the itemID as string.

This function should be used in the "OnInitEvent" function only!

Parameters are:

- itemName as string
- description text as string
- nxa_access_rights as nxa.access
- nxa_data_type as nxa.type
- optional delimiter as string, which is used between the sub paths to form item ID, which follow to be defined.
- optional sub path and more optional sub paths

e.g.:
nxa.AddCustomItem("Test.Item.dbl", "Custom Item 1", nxa.access.All, nxa.type.Real, "/", "MySubDir")

This will create a custom item with the name “Test.Item.dbl” in Item Tree beneath Custom/MySubDira, Description “Custom Item 1” and data type real.

The following functions are often used in combination of custom items, but can also be used with other items, where itemID is known.

nxa.AddItemLink(sourceItemID, destinationItemID [,delay_in_ms])
This function links source and destination item, so that a value from source will be forwarded to destination item.

nxa.AddItemEvent(itemID, onReceive, onSend, onSet, delay_in_ms, script)
This will add a new Event based ion an item also during runtime.
Parameters are:

- itemID as string
- onReceive as bool
- onSend as bool
- onSet as bool
- delay in milliseconds as value
- script as string, which will be executed, if an onReceive, onSend or onSet event happens.

Global Event functions

These event functions must not be changed, but you can add your own functions inside.

OnInitEvent()
Code inside this function will be excuted during 1. Phase of initialization of the server. Here custom Items can be created.

OnStartEvent()
This is the 2. Phase of initialization of the server.
At this moment all items are available to be used for initialization of custom LUA scripts.

OnStopEvent()
This function will be called, if server stops.

OnSecondTimerEvent()
This function will be called every second.

OnMinuteTimerEvent()
This function will be called every minute.

OnHourTimerEvent()
This function will be called every hour.

OnKNXGatewayConnectedEvent(ipaddress)
This function will be called every time, a KNX gateway has been connected.

OnKNXGatewayDisconnectedEvent(ipaddress)
This function will be called every time, a KNX gateway has been disconnected.

After the Global Event functions own written LUA functions can be written or if the development is finished, they can be saved into a different file with *.lua extension in ScriptFiles directory.

Such files are included to main script file with the command:

require “ name_of_script_file”

System - protocol file: nxaOPCSystem.35.log

All relevant events are logged in this data file. Those are the same pieces of information as shown in the "Event Monitor" of the studio. There is only one difference: those are not stored on the hard disk for a later on check or analyses.
If a problem shows up and interrupts the starting of the studio, the current system information can be checked here.
When maintaining the system please always check this data, if it is getting larger than 1MB, please store it in another file and delete it in here. The data file is automatically created new during starting the system.

The data can be loaded in the studio3 and there it can be checked as well (Menu „File -> System Log File -> Open“).

Example:

```
INFO;05/03/09 08:11:35.234;SERVER_ENGINE;Licensed Gateway Number: 32;0
INFO;05/03/09 08:11:35.249;SERVER_ENGINE;Option 'OPC.GroupAddressType' = '2Level';0
INFO;05/03/09 08:11:35.249;SERVER_ENGINE;Option 'OPC.AsyncReadFromDevice' = 'FALSE';0
INFO;05/03/09 08:11:35.249;SERVER_ENGINE;Option 'OPC.AsyncWriteConfirmation' = 'FALSE';0
INFO;05/03/09 08:11:35.249;SERVER_ENGINE;Option 'OPC.AsyncRefreshFromDevice' =
'FALSE';0
INFO;05/03/09 08:11:35.249;SERVER_ENGINE;Option 'OPC.AsyncTimeout' = '10';0
INFO;05/03/09 08:11:35.249;SERVER_ENGINE;Option 'KNX.Timeout' = '3';0
```

Data structure drawing: nxaOPCData.35.log

The whole KNX plant is shown in a virtual model in the memory. This actual state of the process is saved on the hard disk as well. If the server is started, it will search for this file and then will load it automatically. The data are stored in binary form and cannot be used without the system.

The system parameter SYS.UseTelegramDataFile:ON turns it on.

Telegram-protocol data: nxaOPCTelegram.35.log

The telegram drawing is realized by the help of the data. All received and sent telegrams are added with additional information and are saved. This allows an analysis later on. If the data achieves the maximum size, it will be overwritten - the overwriting starts at the beginning and deletes the first parts.

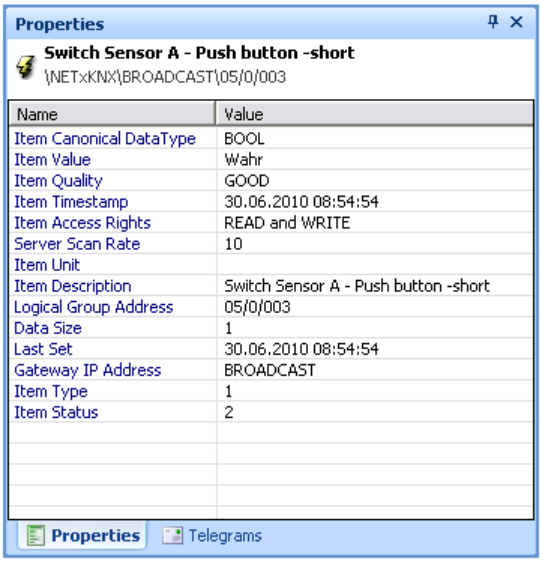
The data is limited to 10MB, if the server is started, the old data will be stored and a new one will be created.

The system parameter SYS.MaxSizeOfTelegramLogFile defines the maximum size.

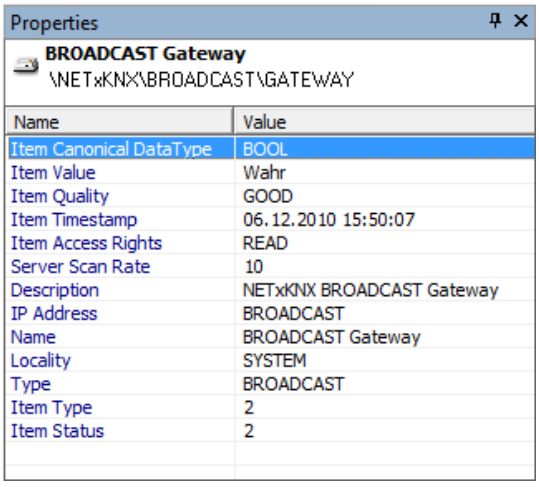
The OPC ITEM Properties

An OPC ITEM consists of several properties.
The NETxKNX OPC System 3.5 differs between two different types of OPC ITEMs.

- The OPC ITEM, which shows a data point (KNX group address), has the following properties:



- The OPC ITEM, which defines a gateway, is described as follows:



The structure of the OPC system

The OPC ITEM structure looks like the following:

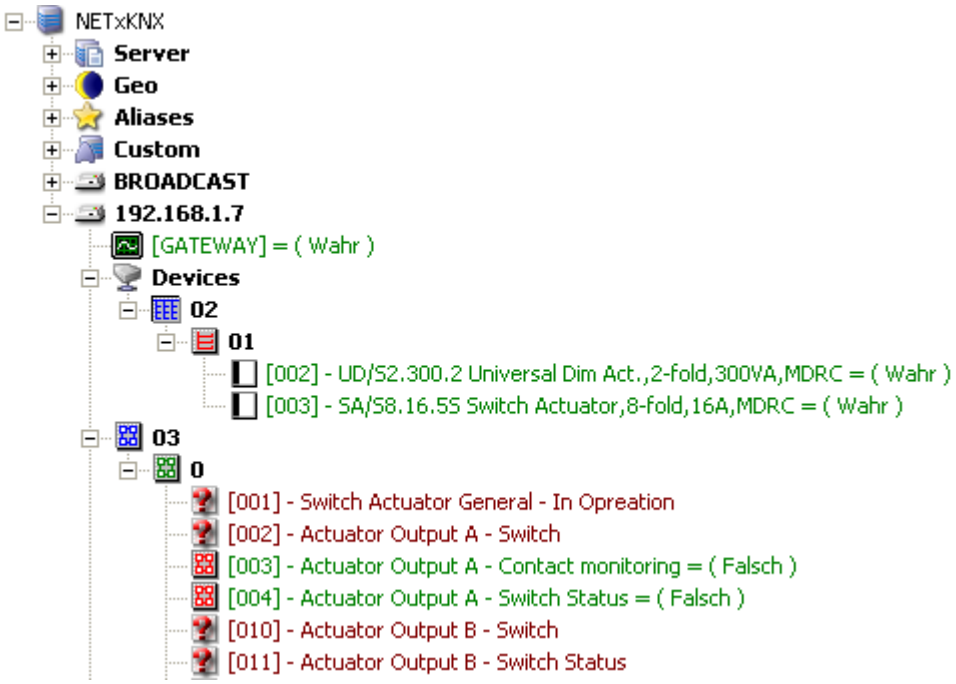
„Root Branch“ of the data structure of the server:
„\NETxKNX\“

for every gateway a branch is made in the OPC structure

„\NETxKNX\BROADCAST\“
„\NETxKNX\192.168.1.1\“
„\NETxKNX\127.0.0.1\“

Every data point is specified with a complete path.

„\NETxKNX\BROADCAST\00/2/001“
„\NETxKNX\BROADCAST\00/2/002“
„\NETxKNX\192.168.1.1\14/0/001“



In Server branch there are items about the status the server.
In Geo branch sunrise, sunset time and moon age items are listed.
If in telegram definitions aliases have been defined, they will appear beneath Aliases branch.

In LUA Scripts custom items can be defined in OnInitEvent function and will be shown beneath “Custom”.Beneath every "IP address" branch the OPC ITEM with the definition "GATEWAY" is defined.
The value of the ITEMs is shown after the equal sign, if the item quality is good. The availability of the gateway can be seen (if the gateway is offline, the value of the gateway ITEM will be set to "false" and quality is set to “UNCERTAIN”)

There are items of the Server, which shows the actual state of the Server.
Beneath Geo there is the astronomical watch, which presents in items sunrise, sunset time, current azimuth and moon age.
Also devices can be seen, if defined and the Device Manager is activated.

Connection of OPC Clients

The NETxKNX OPC Server " is logged in to the system as follows:

„NETxKNX.OPC.Server.3.5“

Each OPC Client makes the selection available of the server on own way.
Most of them monitor the existing OPC server on the PC and the user can select it.

The NETxKNX OPC Server 3.5 can have several OPC Clients at the same time and feed them with data.

Every connection or disconnection of the OPC Client is logged in the NETxKNX System.

Possible Data types

Every row of the below-mentioned table (except the column "description") defines a data type.

Only the following data types are supported

- EIS Data types

Data size	EIS Type	(S)igned / (U)nsigned	Unit	Meaning
1BIT	EIS1			Switch
4BIT	EIS2			Dim object Bit 0=Up/Down Bit 1..3=Speed, if Bit1..4=0 Stop
3BYTE	EIS3			Time
3BYTE	EIS4			Date(Attention: century 2-digits)
2BYTE	EIS5		°C	Temperature
2BYTE	EIS5		K	Temperature difference
2BYTE	EIS5		K/h	Temperature gradient
2BYTE	EIS5		Lux	Light intensity
2BYTE	EIS5		m/s	Wind velocity
2BYTE	EIS5		Pa	Air pressure
2BYTE	EIS5		s	Time difference
2BYTE	EIS5		ms	Time difference
2BYTE	EIS5		mV	Voltage
2BYTE	EIS5		mA	Current
1BYTE	EIS6		%	Relative Luminance (0...100)
1BYTE	EIS6		%	Relative Humidity (0...100)
1BYTE	EIS6		°	Wind direction (0...360)
1BYTE	EIS6	U		8-Bit unsigned (unofficial Type)
1BYTE	EIS6	S		8-Bit signed (unofficial Type)
1BIT	EIS7			Motor Motion
2BIT	EIS8			Priority Control
4BYTE	EIS9		m/s²	Acceleration
4BYTE	EIS9		rad/s²	
4BYTE	EIS9		J/mol	
4BYTE	EIS9		1/s	
4BYTE	EIS9		mol	
4BYTE	EIS9			
4BYTE	EIS9		rad	
4BYTE	EIS9		°	
4BYTE	EIS9		Js	
4BYTE	EIS9		rad/s	
4BYTE	EIS9		m²	
4BYTE	EIS9		F	

4BYTE	EIS9		C/m²	
4BYTE	EIS9		C/m³	
4BYTE	EIS9		m²/N	
4BYTE	EIS9		S	
4BYTE	EIS9		S/m	
4BYTE	EIS9		kg/m	
4BYTE	EIS9		C	
4BYTE	EIS9		A	
4BYTE	EIS9		A/m²	
4BYTE	EIS9		Cm	
4BYTE	EIS9		C/m²	
4BYTE	EIS9		V/m	
4BYTE	EIS9		C	
4BYTE	EIS9		C/m²	
4BYTE	EIS9		C/m²	
4BYTE	EIS9		V	
4BYTE	EIS9		V	
4BYTE	EIS9		Am	
4BYTE	EIS9		V	
4BYTE	EIS9		J	
4BYTE	EIS9		N	
4BYTE	EIS9		1/s	
4BYTE	EIS9		rad/s	
4BYTE	EIS9		J/K	
4BYTE	EIS9		W	
4BYTE	EIS9		J	
4BYTE	EIS9		Ohm	
4BYTE	EIS9		m	
4BYTE	EIS9		J	
4BYTE	EIS9		cd/m²	
4BYTE	EIS9		lm	
4BYTE	EIS9		cd	
4BYTE	EIS9		A/m	
4BYTE	EIS9		Wb	
4BYTE	EIS9		T	
4BYTE	EIS9		Am	
4BYTE	EIS9		T	
4BYTE	EIS9		A/m	
4BYTE	EIS9		A	
4BYTE	EIS9		kg	
4BYTE	EIS9		kg/s	
4BYTE	EIS9		N/s	
4BYTE	EIS9		rad	
4BYTE	EIS9		°	
4BYTE	EIS9		W	
4BYTE	EIS9			

4BYTE	EIS9		Pa	
4BYTE	EIS9		Ohm	
4BYTE	EIS9		Ohm	
4BYTE	EIS9		Ohm m	
4BYTE	EIS9		H	
4BYTE	EIS9		sr	
4BYTE	EIS9		W/m²	
4BYTE	EIS9		m/s	
4BYTE	EIS9		Pa	
4BYTE	EIS9		N/m	
4BYTE	EIS9		°C	
4BYTE	EIS9		K	
4BYTE	EIS9		K	
4BYTE	EIS9		J/K	
4BYTE	EIS9		W/mK	
4BYTE	EIS9		V/K	
4BYTE	EIS9		s	
4BYTE	EIS9		Nm	
4BYTE	EIS9		m³	
4BYTE	EIS9		m³/s	
4BYTE	EIS9		N	
4BYTE	EIS9		J	
2BYTE	EIS10	U		16-Bit unsigned
2BYTE	EIS10	S		16-Bit signed
4BYTE	EIS11	U		32-Bit unsigned
4BYTE	EIS11	S		32-Bit signed
4BYTE	EIS12			Access control
1BYTE	EIS13			ASCII Character
1BYTE	EIS14			ASCII Character
14BYTE	EIS15			14 * EIS13
10BYTE	EIS15			10 * EIS13 (unofficial Type)
14BYTE	EIS15A			14 * EIS13 as BYTE ARRAY (VT_UI1|VT_ARRAY)
8BYTE	EIS29		Wh	data type for power meter
8BYTE	EIS29		VARh	data type for power meter

- Canonical Data types

Data size	Type	(S)igned / (U)nsigned	Unit	Meaning
4BYTE	UI4			Unsigned Integer
3BYTE	UI4			Unsigned Integer only 3 of 4 bytes used
8BYTE	UI8			Unsigned Integer

! If you want to use one of these data types in a telegram, all indicated data type attributes must be entered in the telegram definition table:

Examples:

Correct:	Incorrect:
;2BYTE;EIS10;U;;	**;2BYTE;EIS10;;;** here the "Signed/Unsigned" attribute is missing

Correct:	Incorrect
;1BIT;EIS1;;;	**;1BIT;EIS1;;°C;** EIS 1 does not hold any units

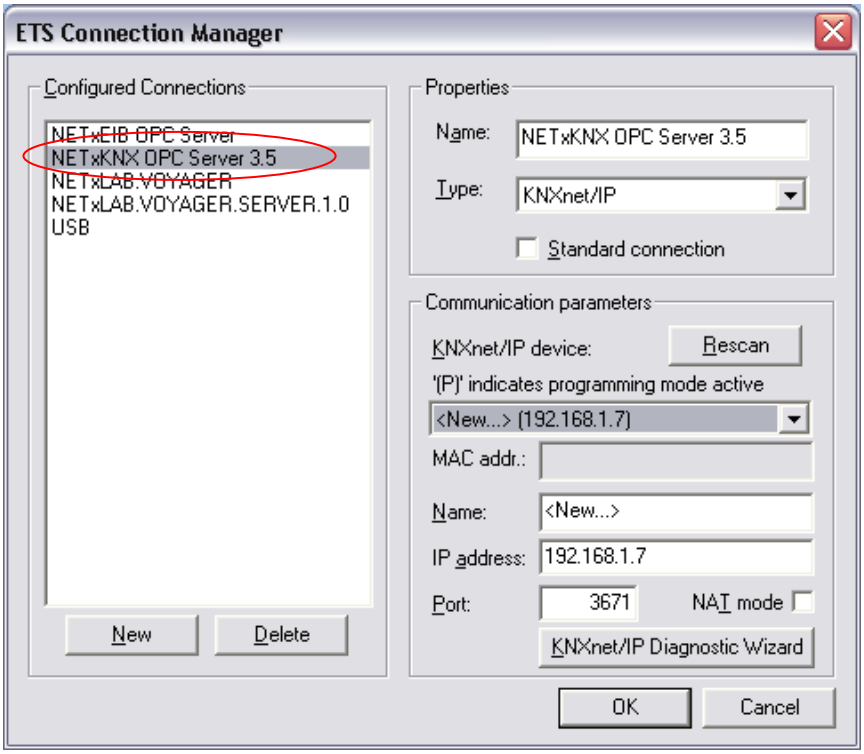
---

NETxKNX OPC Server Direct(KNX) Version

The version of the OPC server supports all KNX protocols like in ETS, but only for one KNX gateway.

The configuration of the Direct(KNX) interface can be accomplished by the help of the "ETS Connection manager" of program.

The OPC server uses the "NETxKNX OPC Server 3.5" connection.



ETS Connection Manager

At present the following interface types are available:

- RS.232 Standard
- RS.232 FT1.2
- USB
- KNXnet/IP Tunneling
- KNXnet/IP Routing
- IP(EIBlib/IP)

After each change of the connecting parameters the OPC server must be started again.

The converting tool: NETxKNXConvertETS

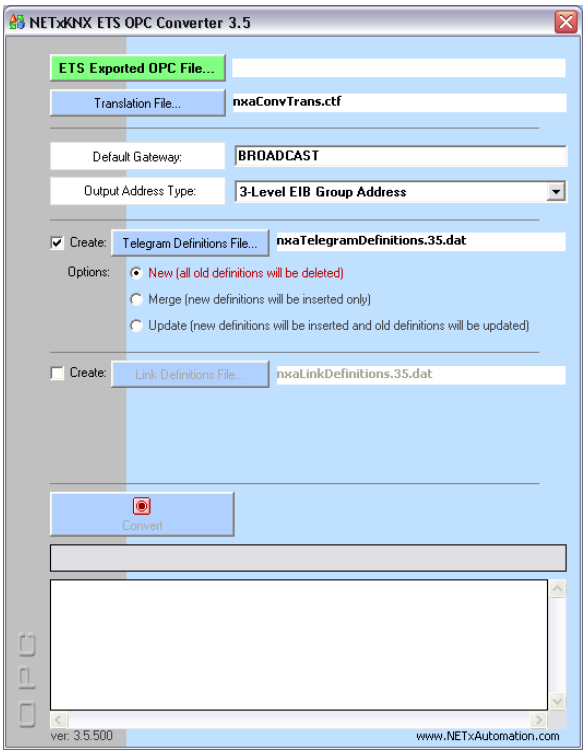
The data migration from the ETS can be made easier by converting the exported OPC data (from version ETS2 1.3) in a telegram definition table and the link definition table by the help of a tool.

! It's an intermediate step, in which the produced file has to be adapted to the project, because the gateway link is not available in the OPC file.

All made telegram definitions and the Link Definitions are linked to the "Default Gateway" gateway, the exact gateway link has to be done by the system integrator.

It is possible to overwrite the old telegram definition, merge them with the new definition or to update them.

The converted definition files can be edited directly with Microsoft© Excel (as CSV file)



The conversion bases on a data type converting file (nxaConvTrans.ctf).

Example of the nxaConvTrans.ctf data:

```
'NETxKNX ETS OPC Converter Translation File
'
'Syntax:
'ETS OPC DataType Entry;NETxKNX DataSize;NETxKNX EIS Type;(S)igned/(U)nsigned;Unit;
'
EIS 1 'Switching' (1 Bit);1BIT;EIS1;;
Uncertain (1 Byte);1BYTE;EIS6;U;;
Uncertain (2 Byte);2BYTE;EIS5;;°C;
EIS 2 'Dimming - control' (4 Bit);4BIT;EIS2;;;
```

Here the data type translation is defined and can be saved regarding to the relevant project.

A definition can be inserted in the table only once and is described in four columns.

Column 1 - ETS OPC DataType Entry
It defines the data type definition part which is listed in the OPC file („**EIS 1 'Switching' (1 Bit)**").
This value is used as a key word, and can only occur once.

Column 2 to 4
The columns 2 to 4 define the data type, which is interpreted by the NETxKNX OPC3 system.

! Only those data types are supported, which are listed in the Data Type table.

A NETxKNX OPC3 data type consists of a minimum of two parts, e.g.:

„**EIS 1 'Switching' (1 Bit)" = 1BIT;EIS1;;;**

The input is interpreted as a data type with the size of 1 bit and as EIS1.

Also an unspecific data type can be converted.

"**Uncertain (2 Byte)" = 2BYTE;EIS5;;°C;**

All those inputs are converted to the data type EIS5 with the size 2 Byte and the unit **„°C"**.

Different forms of licenses:

The form of the license of the system depends on the number of connected gateways. It is invariant from the number of defined group telegrams, but we suggest planning the KNX line, where the IP gateway is connected, not to exceed a maximum of 10 telegrams per second to prevent telegram loses.

Following forms are possible:

1 Gateway
3 Gateways
5 Gateways
10 Gateways
20 Gateways
32 Gateways
50 Gateways
100 Gateways
200 Gateways
500 Gateways
Up to 1000 gateways

Software security

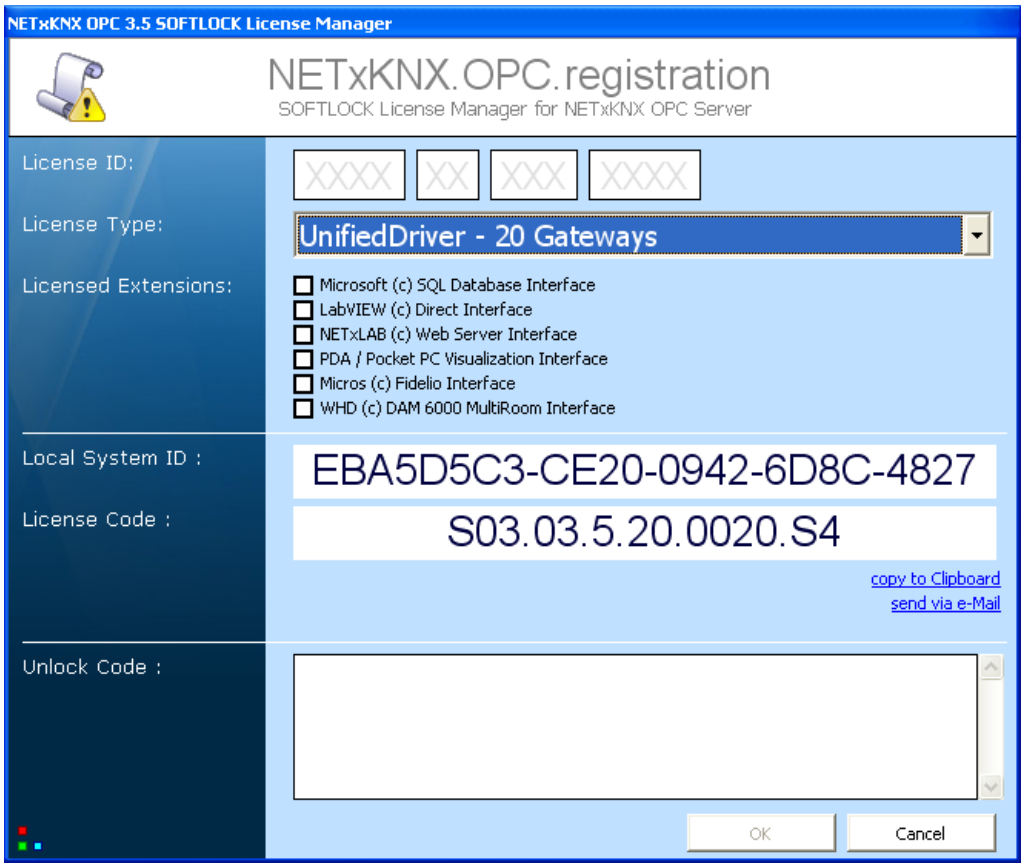
Two different solutions are available:

Soft lock – Software based security system
Hard lock – Hardware based (Dongle) security system

Soft lock

Software registration:

The release of the software is fixed to a software license which is connected to the local hardware.
The licensing process is done during the set-up by starting the "register system" tool.



The licence consists out of four parts:

- License ID – the name of the license (see at your invoice)
- License Type – is graduated after the number of gateways
- Licensed Extensions – Auxiliary modules to be licensed

Following both entries must be passed on at NETxAutomation, as answer the local code is supplied:

- License ID – the name of the license (see at your invoice)
- Local System ID – the local hardware key (it is generated automatically)
- License Code – Licese code (it is generated automatically)

- Unlock Code – enter here Unlock Code provided by NETxAutomation

If you receive an error message after the process, you have to check the given data and you have to try the registration again.

! The user has to have administrator rights.

Hard lock

The second solution is the hard lock.

!The hard lock key (USB dongle) has always be connected, if it is disconnected, the correct work of the server will stop after two warnings at about 1,5 minutes.

! The hard lock key (USB dongle) has to be connected during the installation.

Gateways

The NETxKNX OPC Server 3.5 supports several forms of gateways.

- ABB IG/S 1.1
- KNX NetIP (Tunneling) gateways like
  - ABB IPR/S 2.1,
  - Siemens 5WG1 146-1AB01
  - Siemens 5WG1 148-1AB21
  - Merten IP Router
  - Gira IP Router
  - Berker IP Router
- b.a.b-tec eibNode

! It is possible to use different equipment types in one workspace.

The Ethernet Network

The NETxKNX OPC System communicates with the KNX equipment via the Ethernet. This has to be configured in a way that the telegrams can be sent and received. Please note that especially the firewall, the switches, the ports and the router have to be released and the support for Multicast and Broadcast has to be activated.

System requirements

Following operating systems are supported:

OPC Server Direct(KNX):

Microsoft Windows XP Professional 32bit
Microsoft Windows Vista 32bit
Microsoft Windows 7 32bit | 64
Microsoft Windows 2008 Server 64 bit

OPC Server UnifiedDriver:

Microsoft Windows XP Professional 32bit
Microsoft Windows 7 32bit und 64bit
Microsoft Windows 8 64 bit

Microsoft Windows 2008 Server 32bit and 64 bit
Microsoft Windows 2012 Server 64bit

Hardware requirements:

Minimum
PC with INTEL or AMD Processor
512 MB Ram
Ethernet-card 10/100 MBit
Screen with 800x600 resolution

The best would be
PC with INTEL or AMD Processor
2048 MB Ram or more
Ethernet-card 1000 MBit
Screen with 1280x1024 resolution

Multicore Processors are recommended.

Support and contact

Please send all your support questions to:

support@NETxAutomation.com

If you have general questions regarding the product please send your email to:

info@NETxAutomation.com

All gateways, which are to be managed in the system, must be registered in the gateway definition table. Telegrams (also the "BROADCAST" telegrams) of unknown (registered) gateways are not accepted by the system.

eibNode:
The NetID parameter of attached eibNodes and the OPC server must agree.

IG/S:
The ProjektIDs and the multicast address of attached IG/S and the OPC server must agree.

The receiving of "BROADCAST" telegrams:
If a KNX group address is assigned only to the "BROADCAST" gateway, all telegrams with this KNX address, which come from any of the defined gateways (!),will be interpreted as "BROADCAST" telegrams.
But if this KNX group address is assigned to another gateway (e.g.: "192.168.1.2") as well and comes a telegram with this KNX address of this gateway, the telegram will be interpreted as "192.168.1.2"-Gateway-Telegramm.

Important Information

Upgrading from NETxEIB OPC Server 3.0 to NETxKNX OPC Server 3.5

A workspace from NETxEIB OPC Server 3.0 cannot be used without manual changes within NETxKNX OPC Server 3.5 workspaces directory. Filenames and Parameter names have changed. So best way to upgrade is to create a new workspace, open each configuration file with an Editor and fill out manually each definition using copy and paste.

Also the OPC ItemIDs will change from NETxEIB… to NETxKNX!
(With alias feature in NETxKNX OPC Server 3.5 old itemID definitions can be kept.)