ESFVISU
process model editor, version 1.2

Content:

Address

ESF Software GmbH
Europaallee 14 + 16
D-67657 Kaiserslautern
Federal Republic of Germany

Telefon: +49 (0) 631 / 303200 - 0
Telefax: +49 (0) 631 / 303200 - 9

office@esf-software.com

www.esf-software.com

Copyright

Trademarks

1 Technical requirements

1.1 Hardware

Processor	Pentium IV or equal, running at 1200 MHz.
Main memory	256 MB
Free disk space (additional disc space requirements depend on the configured archives)	40 GB
Screen resolution	1024 x 768 Pixel
Colors	Color depth min. 16 Bit per Pixel.
Interfaces	Serial or USB interface to connect EIB with EIBA FALCON driver.

1.2 Software

WINDOWS 95	NO
WINDOWS 98, First Edition	NO
WINDOWS 98, Second Edition	YES
WINDOWS ME	YES
WINDOWS NT	NO
WINDOWS 2000, all versions	YES
WINDOWS XP, all versions	YES

1.3 Process interface

For the purpose of European Installation Bus (EIB), the FALCON driver of the EIBA (EIB Association) is used.

Optional the ESFVISU is equipped with an OPC (OLE for Process Control)- client, so that instead of EIB, or additonal to EIB, OPC- servers, that are available for a multitude of automation systems, can be used for the processing connection.

Attention: Under Windows 2000 the FALCON driver has to be installed manually from the setup CD !

1.4 Import from ETS

ETS 2 version 1.3	YES, use ETS, "OPC - Export"
ETS 2 versions 1.1, 1.2 and 1.3	YES, use print report redirected to file.
Older ETS – versions	NO

1.5 Microsoft Internet Explorer, Version 6; DirectX, Version 9.0b

These applications are required; they can be installed or updated from the setup CD.

2 First steps

A process model describes technical processes from an application point of view.

Whereas the process interfaces implement access to connected devices, the logical process model describes the desired functionality based on parameters given by these devices. At this level, additional parameters may be defined and automatic control functions may be set up to supplement the devices behavior.

For example the process interface may provide access to a room's temperature and heating control, the logical process model will define archives of the temperatures, measured over a time period, and relate the desired room's temperature to the occupancy state, perhaps planned in a calendar, by setting the control value for the heating device.

Several process interfaces may be combined in one logical process model.

This is also the level to define alarm and warning ranges for parameters and to set up the alarm behavior, which may comprise sending notifications with email.

In order to clearly distinguish different applications, we use different colors for the application's icons, **green** for the **process model editor**.
To start the process model editor use the WINDOWS program manager or just double click on the editor's icon in the ESFVISU control panel.

The simplest process model:

The simplest process model just imports a process interface project, i.e. an EIB or OPC project, and supplements no additional functions.

The process model will be created, connected to a process interface project, and then saved. Now it already can be used by other applications relying on a process model: currently the visualization editor and the calendar program. More functions can be added later to the process model.

Proceeding

The process interface (i.e. the EIB Editor) provides the raw data, that now are used for different purposes.
Some values shall become archivated, others are to be monitored and eventually produce emails. Other types of values are used in scenes or sequences, being evaluated or serve for the visualization of a building or a process to enable the user to encroach upon the process.

Therefore it is necessary to extend the EIB group addresses by archives, logical functions, calculation formulas or email advices. The process model editor provides these functions.

The process model that results from the extension of the group addresses provides the visualization basis.

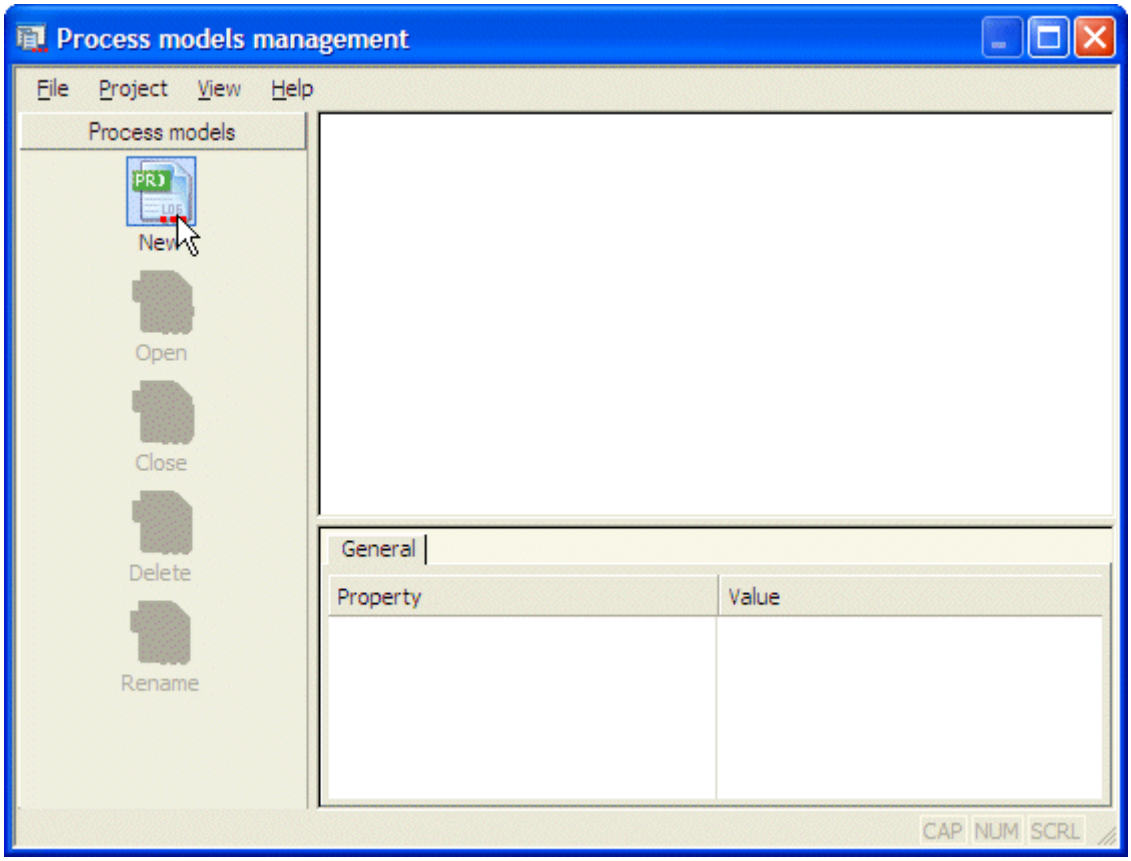
The construction of a process model follows 3 steps:

1. Import of an EIB or an OPC project.
2. Generate the archives, the calculation formulas or email notifications.
3. Save the project.

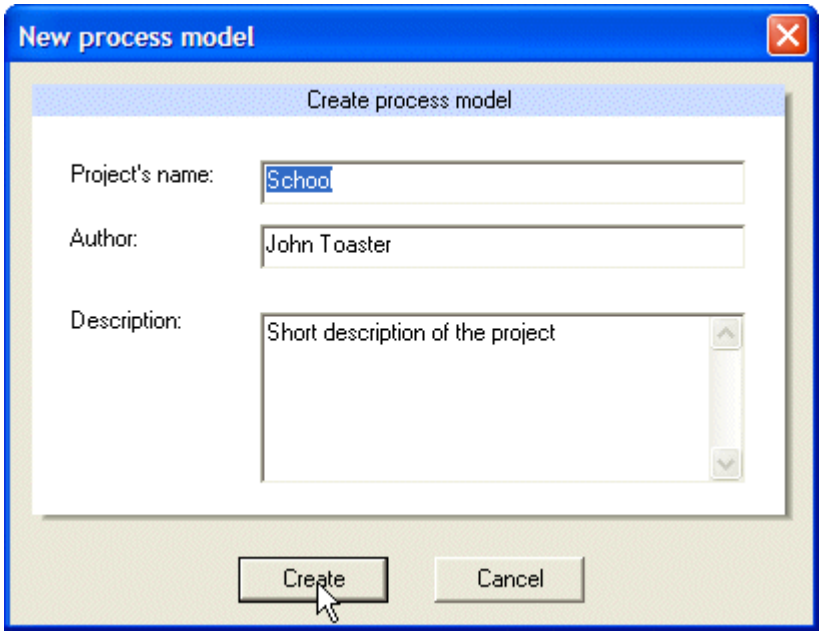
Step 2 can be omitted if neither formulas nor email notifications shall be generated.
An existing process model can be changed in the process model editor at any time.

2.1 Step 1: Create process model

Menu option **File – Projects..** opens the projects management dialog. Alternatively click on the symbol in the toolbar. The projects manager is used to create new projects, open, close, rename or delete existing projects.



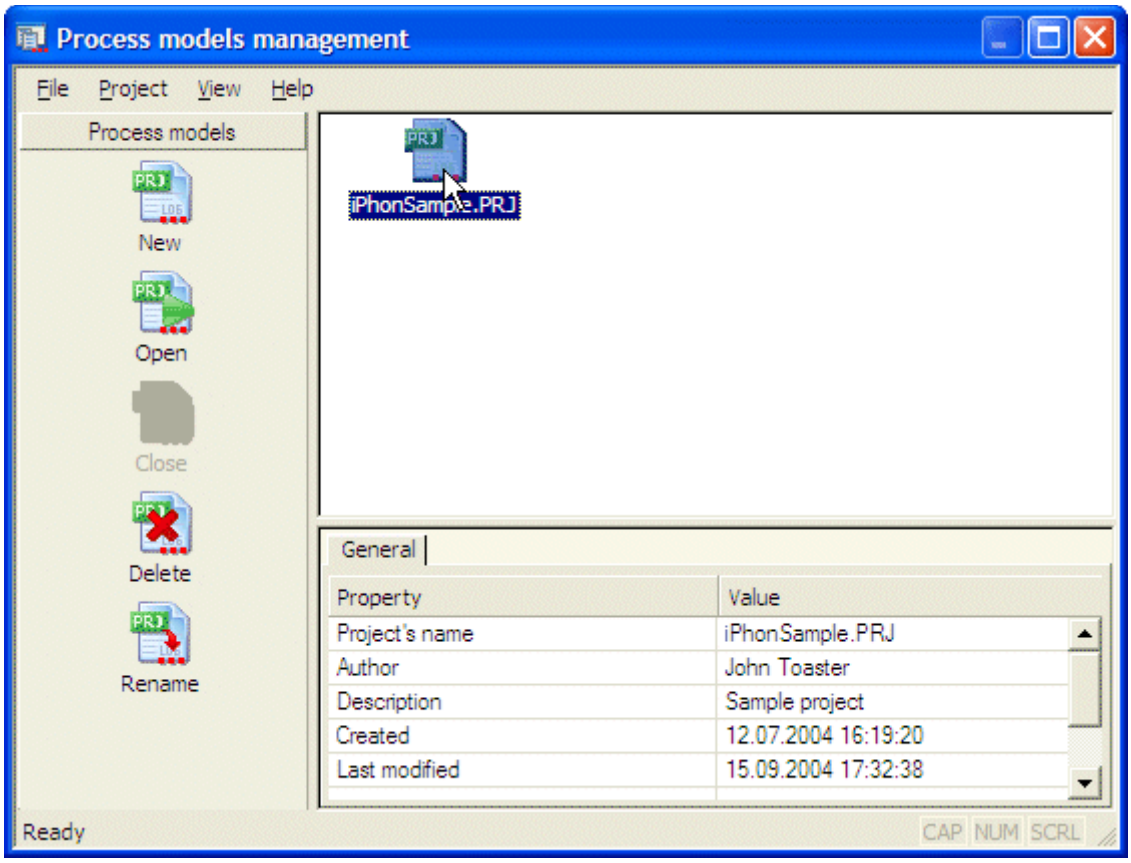
New: Click on the symbol **New** opens a dialog to define a new process model project.



Project's name: Name of the visualization project

Author and description: It is useful but not mandatory to enter the author's name or the project description. These entries can be changed later.

Create: After entering at least the name of the project click button **Create** to confirm the creation of the project. The project will be available in the project manager's projects list.

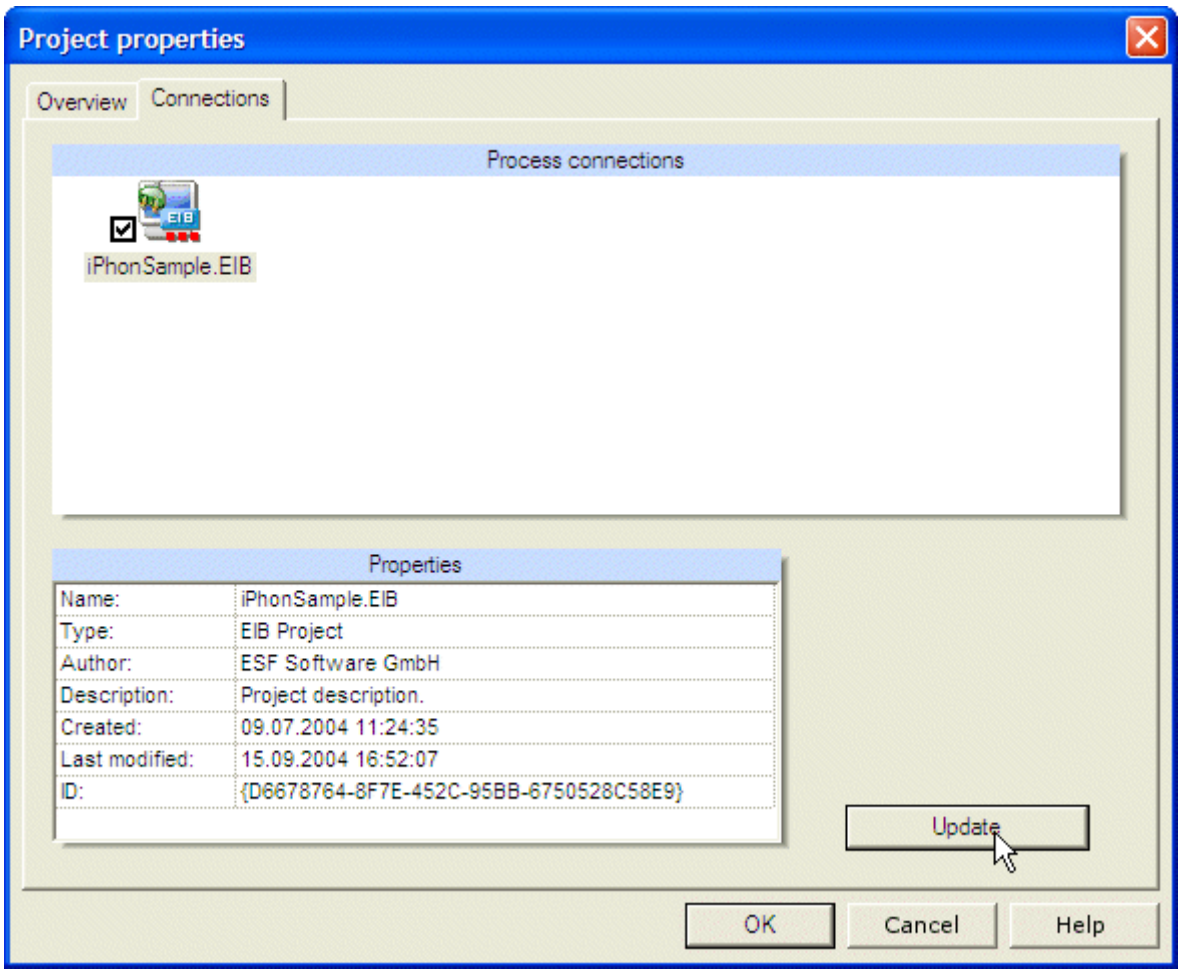


Open: Select a project with a click on the project's symbol. Click on symbol **Open** to open the selected project. Alternatively, you may just double click on the project's symbol.

2.2 Step 2: Import from process interfaces

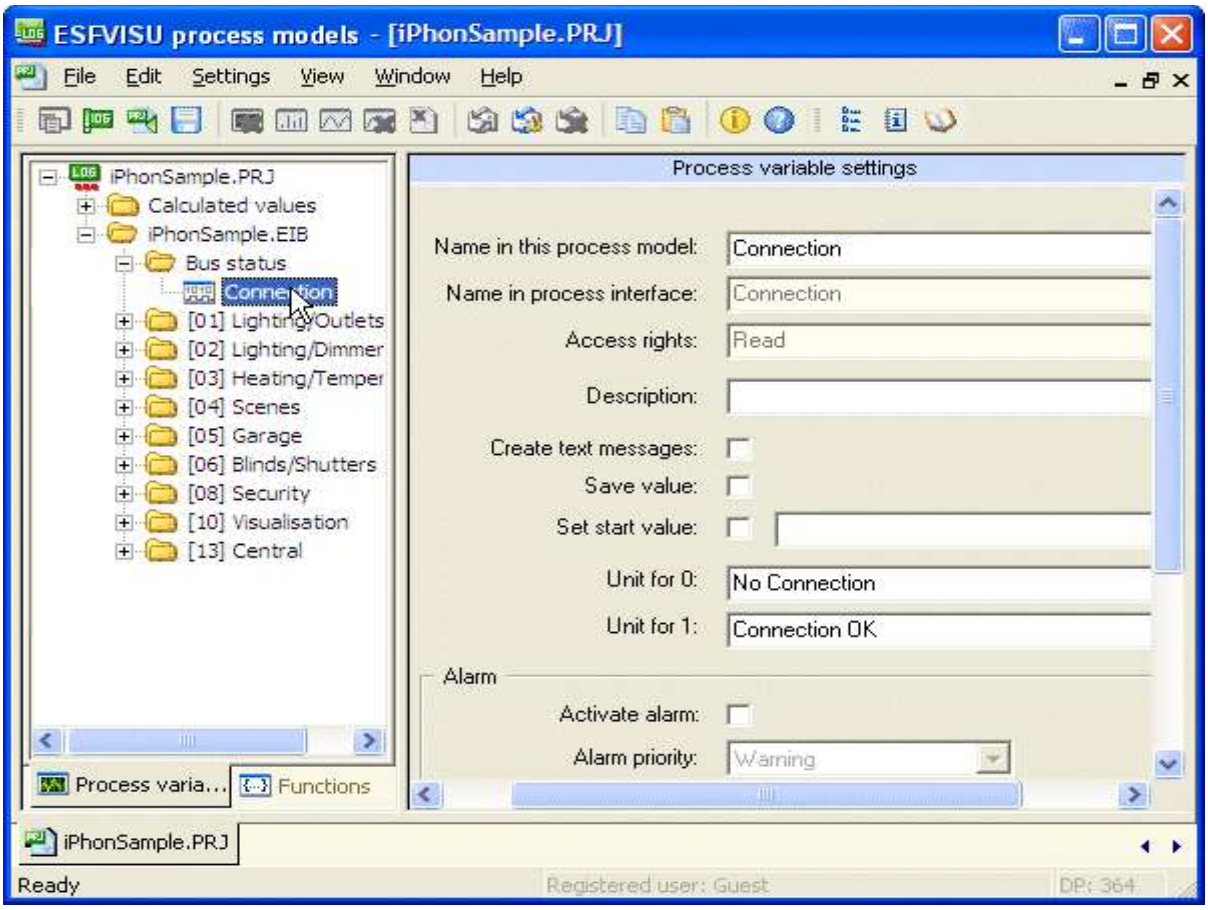
The process model must be connected to one more process interface projects, currently this can be EIB projects or OPC projects.

Select the menu option **File – Project properties..** or select the symbol **Properties** in the tasks tool window to open the project properties dialog.



Select the tab **Process interfaces** to get a list of all available process interface projects. Check the symbol of the process interface project you wish to use with the process model. Then press button **Update** to open a dialog to import process interface data into the process model.

The process interface project, i.e. an EIB project, has been connected. For each data point in the process interface project a corresponding process variable has been created in the process model. Click on a process variable in the process variable's structure tree at the left to open a window with the process variable's properties.



Refer to the chapter ***Functions*** to see how calculated process variables and various optional functions can be added. They are not required for a minimal process model.

2.3 Step 3: Save process model

Use the menu option **File – Save project** to save the project. Alternatively click on the symbol in the in the toolbar. The project will also be saved when the process model or the editor is closed. After the project has been saved, it is ready to use i.e. by the visualization editor.

2.4 Step 4: Update from process interfaces.

When a connected process interface project, i.e. an EIB project or an OPC project, has been changed, the process model must be updated.

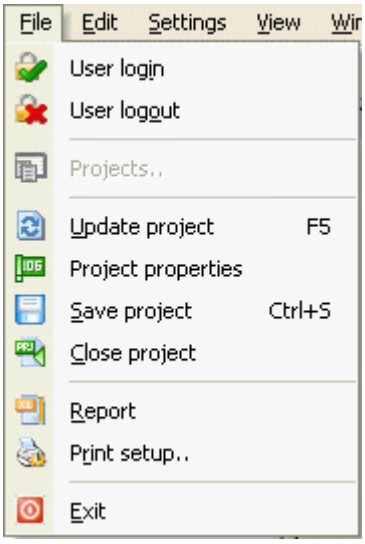
Use menu option **File – Update project** or function key F5 to update the process model. Properties of process variables, i.e. archives for the process variable's values, will always be preserved.

If data points have been deleted in the process interface project, the corresponding process variables in the process model won't be automatically deleted, but delete marks will be assigned. Then it is up to the user to actually delete process variables with delete marks.

To delete a process variable, click on the process variable in the process variables structure tree and press the keyboard **Delete** button.

3 User interface

3.1 Menu



User login: Opens a dialog to login a user to the system.

User logout: Logout current user from the system.

Projects..: Opens the project management dialog. In this dialog projects can be created and maintained.

Update project: Updates data imported from the logical process model. Shortcut F5.

Project properties: Opens a dialog to define or update general project properties.

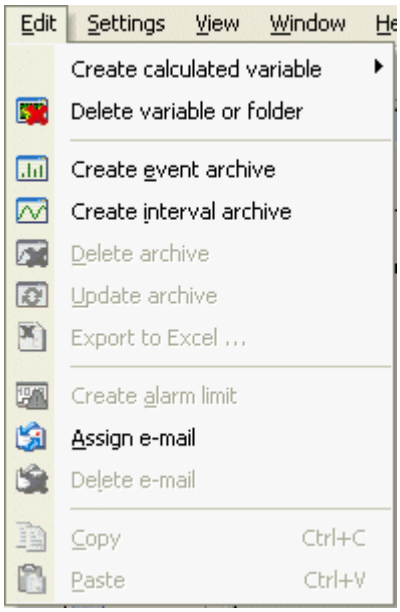
Save project: Saves the current project.

Close project: Closes the process model project.

Report: Generates report for the project.

Print setup: Select printer and printer properties.

Exit: Closes the editor. Shortcut ALT + F4.



Create calculated variable: Create a process variable calculated by others. The process variable may be a binary (shortcut CTRL + B), analog (shortcut CTRL + A) or text string (shortcut CTRL + T) variable.

Delete variable of folder: Deletes the selected process variable or selected folder of the process variables.

Create event or interval archive: Creates a new event or interval archive for the selected process variable.

Delete archive: Delete the selected archive.

Update archive: Update the display of the selected archive.

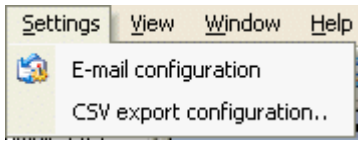
Export to Excel: Exports the data in the archive data viewer to a Microsoft XLS file.

Assign e- mail: Creates an email notification for the selected process variable. The email will be sent on certain conditions of the process variable.

Delete e-mail: Deletes email notification.

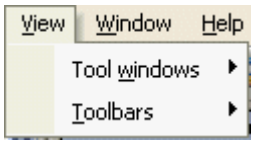
Copy: Copy selected items to clipboard.

Paste: Paste items from clipboard.



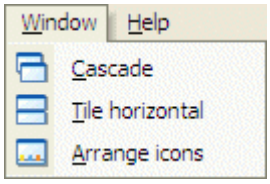
E-mail configuration: Configure the email configuration.

CSV-Export settings: Opens dialog for edit CSV export options.



Tool windows: Used to toggle the visibility of different tool windows.

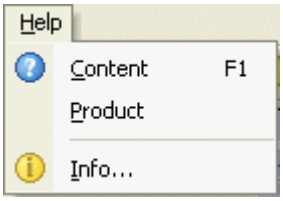
Toolbars: Used to toggle the visibility of different toolbars.



Cascade: The worksheet windows overlap.

Tile horizontal: The worksheet windows are tiled horizontally.

Arrange icons: Arranges symbols of minimized windows.



Content: Opens the help file. In addition to the help file the info tool window also provides advise.

Product: Shows product and license information.

Info: Shows version and copyright information.

3.2 Toolbar

Projects management: Opens the project management dialog. In this dialog projects can be created and maintained.

Project Properties: Opens a dialog to define or update general project properties.

Close project: Closes the process model project.

Save project: Saves the current project.

Delete variable of folder:
Deletes the selected process variable or the selected folder of process variables.

Create event archive:
Creates a new event archive for the selected process variable.

Create interval archive:
Creates a new interval archive for the selected process variable.

Delete archive: Deletes the selected archive.

Export to Excel:
Exports the data in the archive data viewer to a Microsoft XLS file.

Create E-Mail: Creates an email notification for the selected process variable. The email will be sent on certain conditions of the process variable.

E-Mail configuration: Configure the email configuration.

Delete E-Mail: Deletes the email notification.

Copy: Copy selected items to clipboard.

Paste: Paste items from clipboard.

Info: Shows version and copyright information.

Content: Opens the help file. In addition to the help file the info tool window also provides advise.

Tasks tool window: Toggles the visibility of the tasks tool window.

Info tool window: Toggles the visibility of the info tool window.

Catalog tool window: Toggles the visibility of the catalog tool window.

3.3 Tool windows

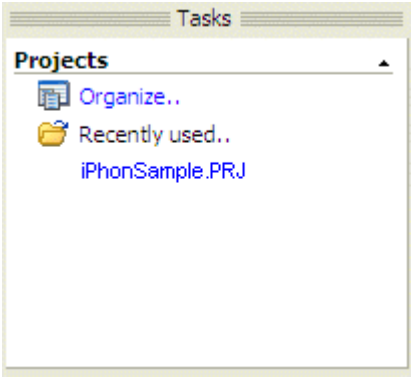
The editor comprises tool windows, which provide overviews, assistance and easy access to common actions. According to the current situation, the contents of the tool windows changes automatically.

Click the menu option **View - Tool windows** to show or hide the individual tool windows.

Alternatively, the visibility of the individual tool windows can be changed by clicking on the respective icons in a toolbar. With the menu option **View – Toolbars** this toolbar can be turned on or off.

3.3.1 Tasks tool window

The tasks tool window shows frequent tasks, which can be selected with a single click. Its contents depend on the current state.



In case no project has been loaded yet:

Start of the project's management; open a project out of the most recent used projects.

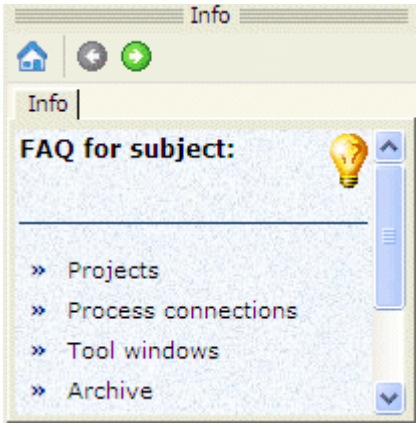
In case a project has been opened:

Close project.

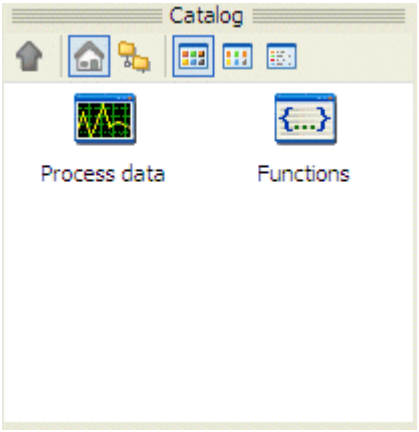
3.3.2 Info tool window

This window provides information similar to a FAQ (Frequently Asked Question) list.

The contents change according to the current situation in the editor and can be navigated with the built- in Internet Explorer.

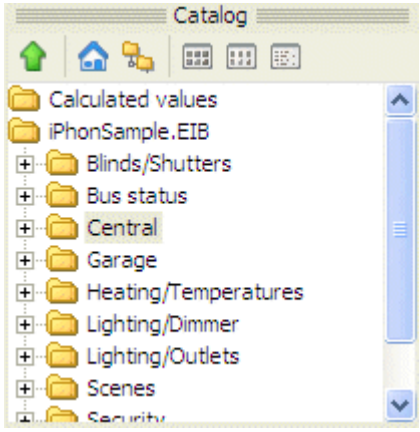


3.3.3 Catalog tool window



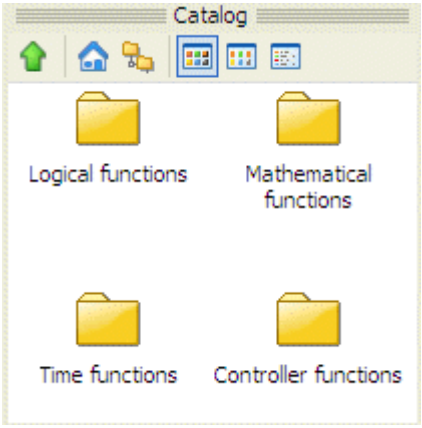
Catalog:

Catalogs of different categories. Currently these categories are display items for the process variables and the configurable functions of the process model.



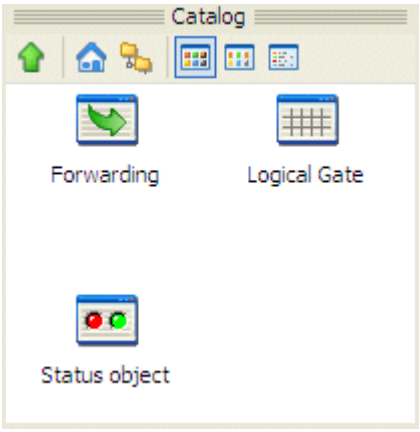
Process interfaces and data points:

The data points can be dragged to configuration windows of the process model's functions.



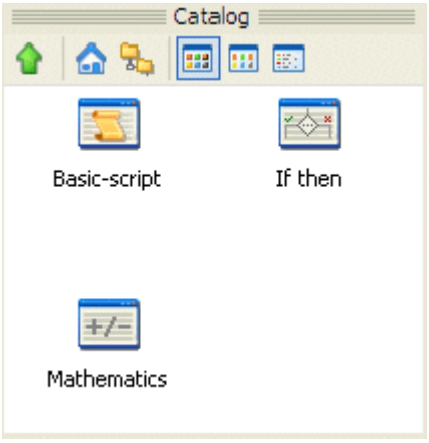
Functions:

Overview of different categories of functions, which can be accomplished with the process model.



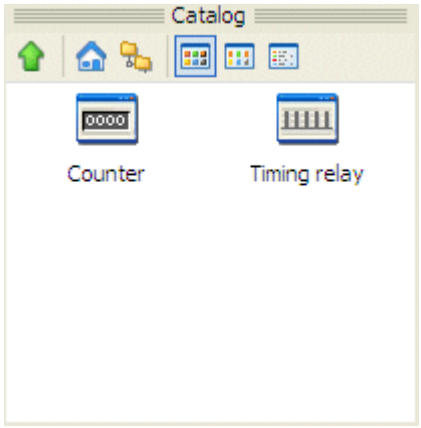
Logical functions:

The functions can be dragged to main window of the process model editor to add in the functions overview.



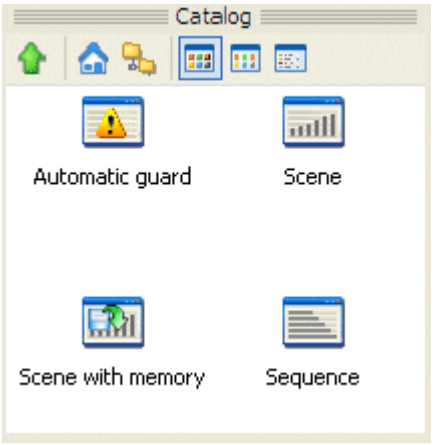
Mathematical functions:

The functions can be dragged to main window of the process model editor to add in the functions overview.



Time functions:

The functions can be dragged to main window of the process model editor to add in the functions overview.



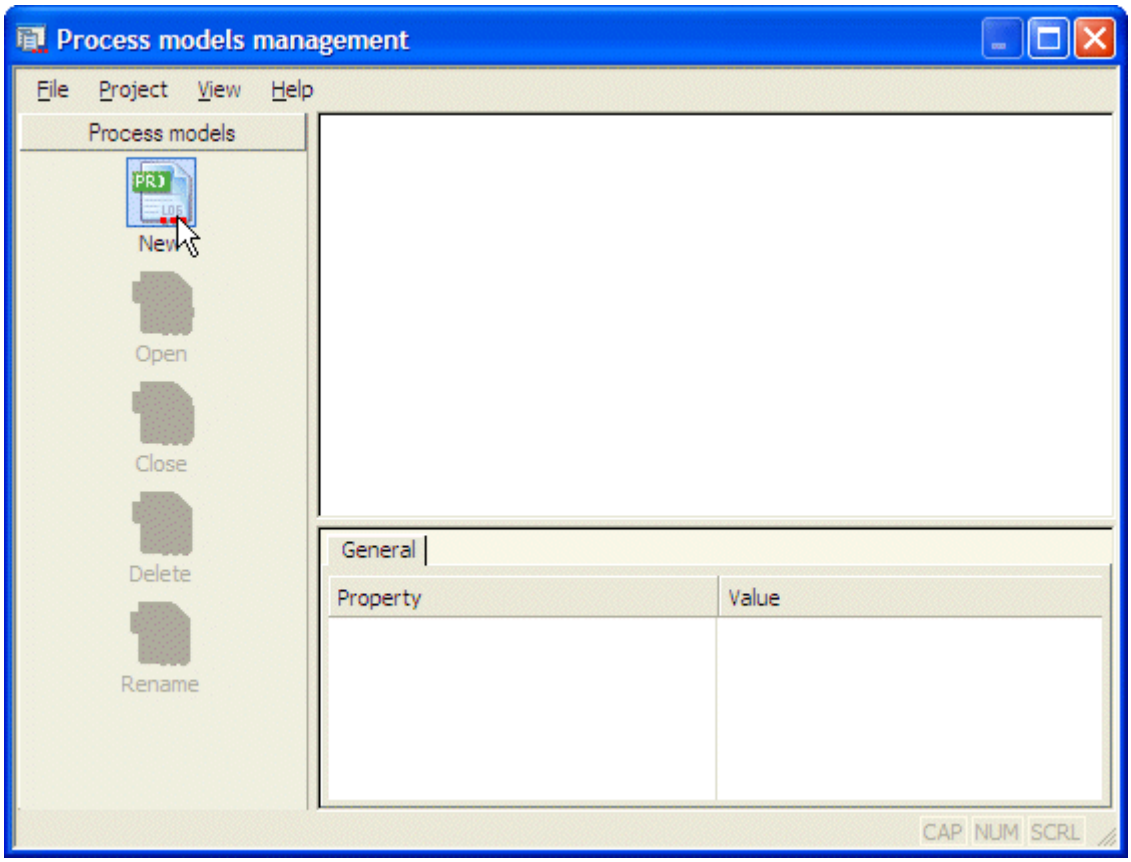
Controller functions:

The functions can be dragged to main window of the process model editor to add in the functions overview.

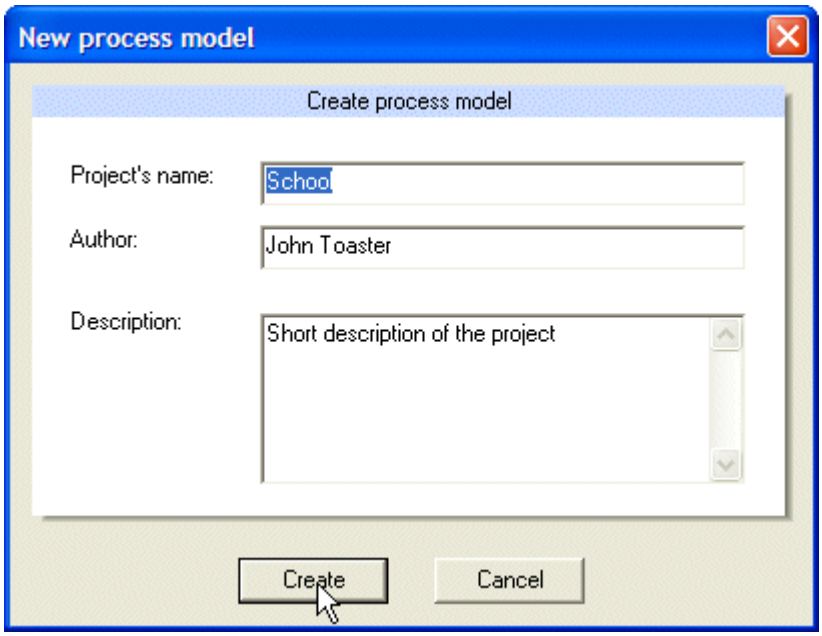
4 Functions

4.1 Create and maintain process models

Menu option **File – Projects..** opens the process model projects management dialog. Alternatively, click on **Oganize..** in the tasks tool window. In this dialog you may create, open, delete or rename process model projects.



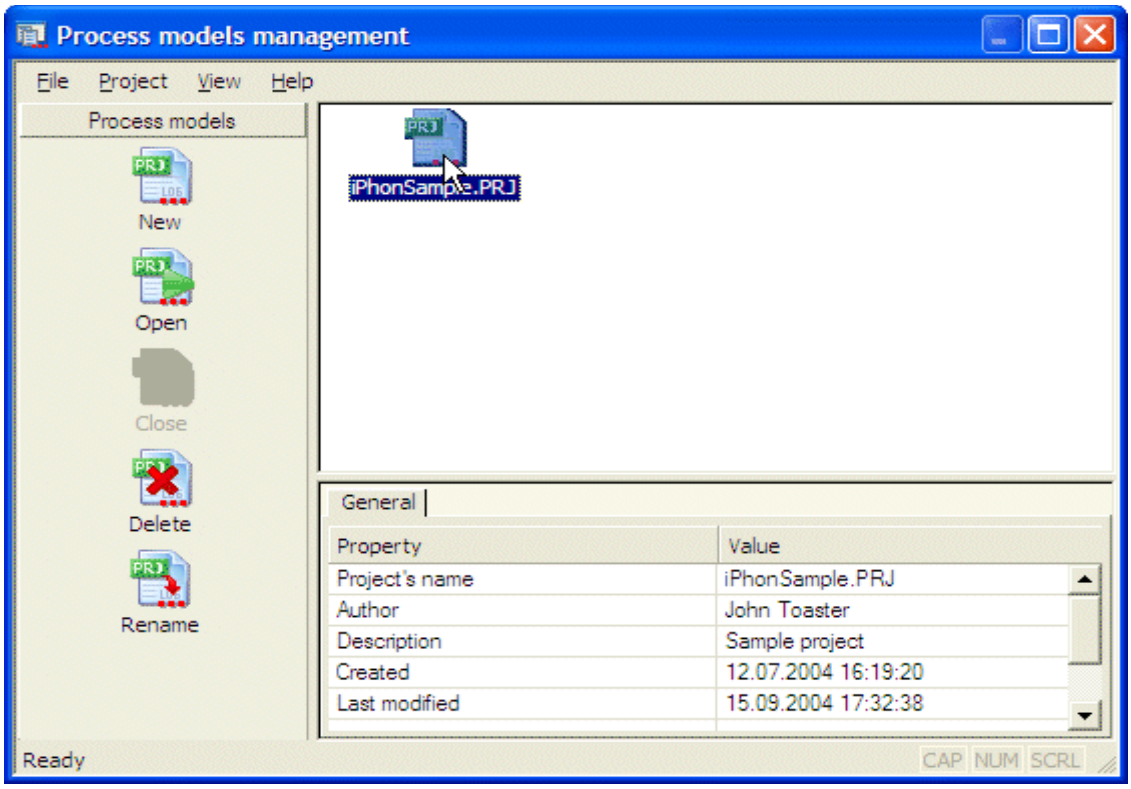
New: Click on button **New** opens a dialog to create a new calendar project.



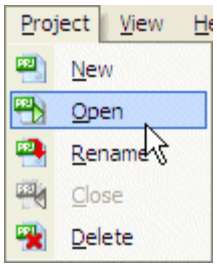
Project's name: Unique name of the process model project.

Author and description: It is useful but not mandatory to enter the author's name or the project description. These entries can be changed later.

Create: After entering at least a name for the project, pressing the **Create** button will create a new process model project. The new project appears in the project's list.



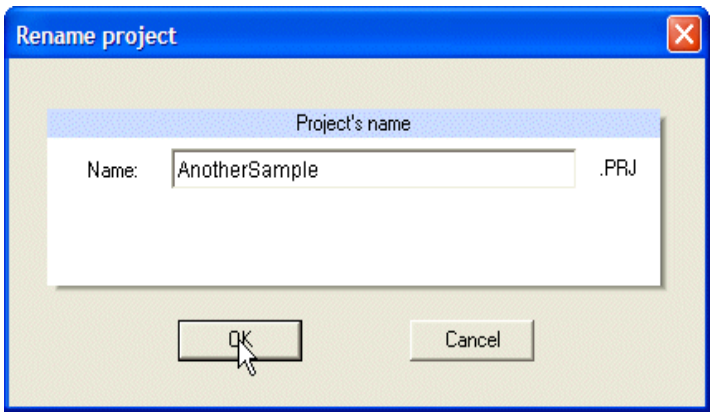
Open: Press the button **Open** to open the selected project. Alternatively double click the project's icon or use the menu option **Project – Open**.



Alternatively, the selected project can be opened with the menu option **Project – Open** or just a double click on the project's symbol.

Close: Closes the current project.

Delete: Deletes the selected project.

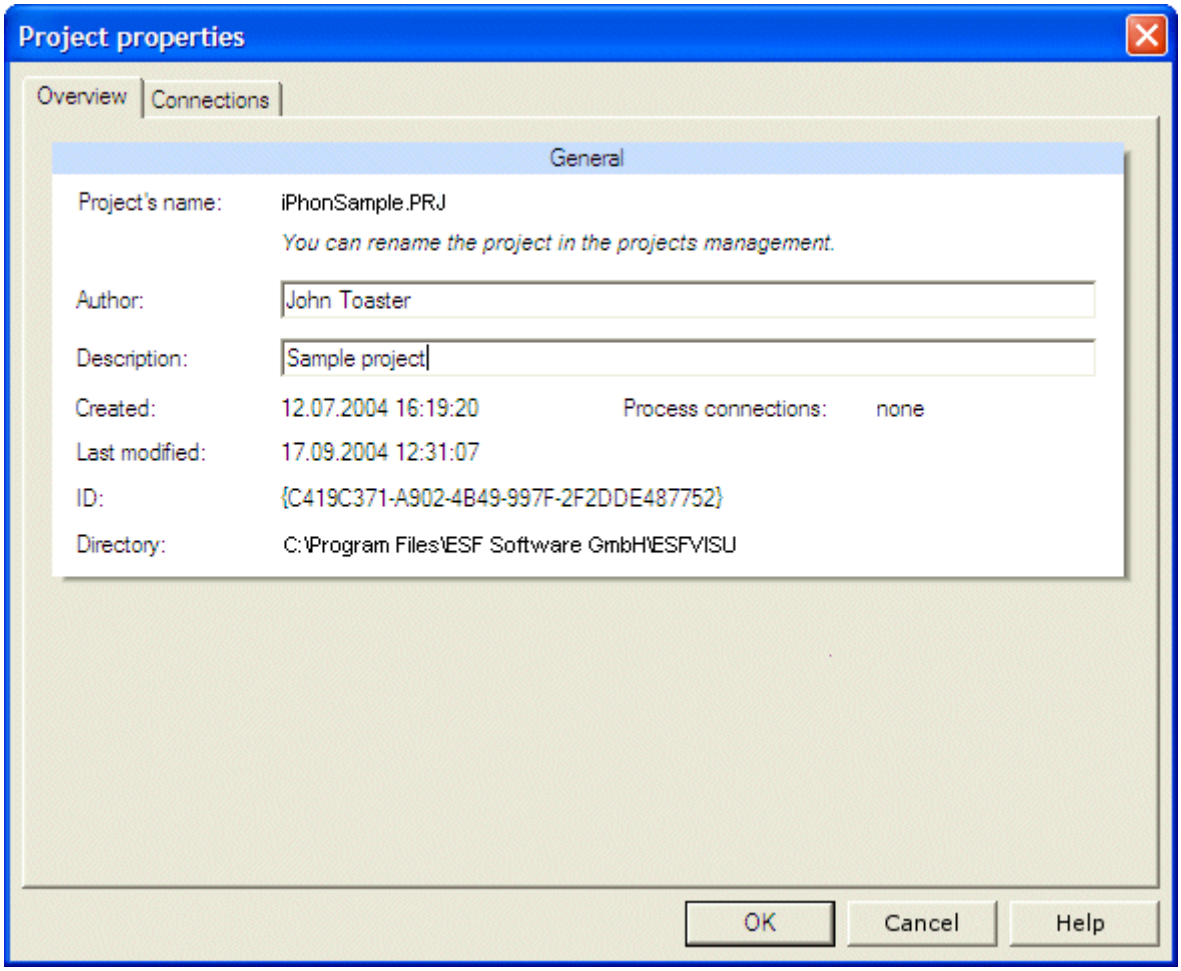


Rename: Opens a dialog to rename the selected project. After the project's name has been changed, press **OK** to submit the change and close the dialog. The extension "**.prj**" will be added automatically.

***Note** that a project can only be renamed if it is not currently opened.*

4.2 Project properties

The Menu option **Project – Properties** opens the window of the project's properties. Alternatively, click the symbol in the toolbar.



Tab **Overview** shows the general properties of the project loaded. Author and description of the project may be changed here.

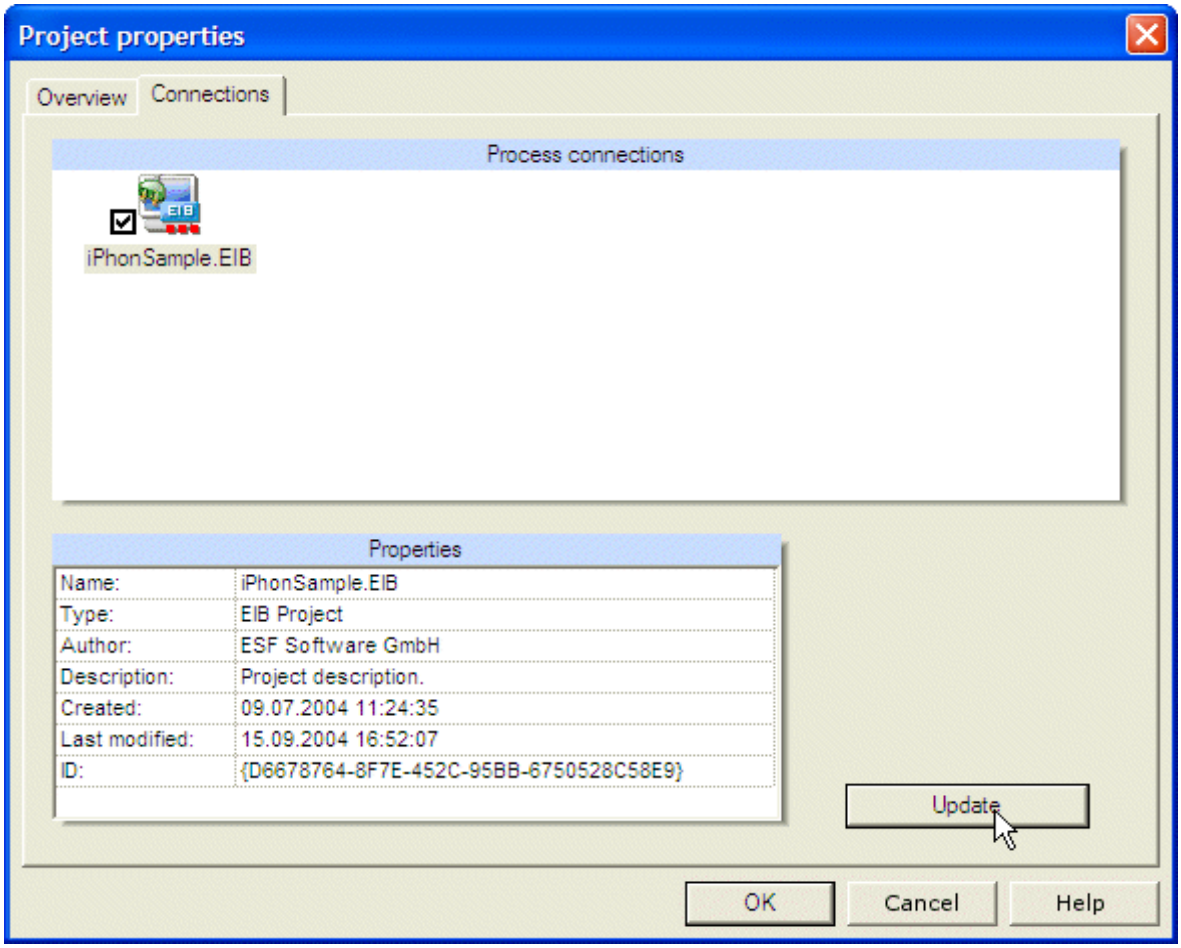
> ***Note*** *that the project's name can be changed in the project's management window.*

4.3 Import from process interface projects

The process model must be connected to one or more process interface projects, i.e. EIB projects and OPC projects. The process variables in the process model will be created from the data points of the process interface projects.

Each process variable's name, type and description is initialized from the respective data point in the process interface project. For example an EIB group address of the type "Boolean" will result in a binary process variable, other EIB group addresses are mapped to analog or string process variables.

The menu option **Project – Properties** opens the window of the project's properties. Alternatively, click the symbol in the toolbar.



The tab **Process interfaces** lists all available process interface projects.

The process interface project models used with the process model are marked with a small hook. Click on the control box at the left of a process interfaces project's symbol to toggle the hook.

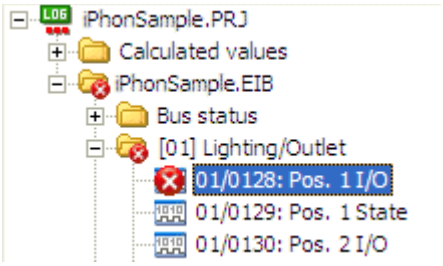
> ***Note:*** *You may mix EIB and OPC projects in one process model.*

Press the button **Update** to update data in the process model from the selected process interface project.

Press the button **OK** to commit your selection and to close the dialog.

4.4 Update from process interface projects

The process model must be updated when a connected process interface project, i.e. an EIB project, has been changed. Use the menu option **File – Update project** or press the function key F5 to perform the update. Alternatively, you may update the process model from process interface projects in the project properties dialog.



If a data point has been deleted in the process interface project, i.e. an EIB group address has been deleted in the EIB editor, the respective process variable will be displayed with a deletion mark.

It is up to the user to actually delete the process variable. Before, the user may check the results of the deletion.

To delete a process variable click on the variable in the process variable's structure tree and press the keyboard **Delete** key.

Important:

When the type of a data point has been changed in the process interface, i.e. an EIB data point has been changed from "Boolean" to "2 Octet Float", the process variable of the old type will be marked as deleted and a new process variable with the new type will be created.

The process variable's name, description, access rights, unit and description will always be updated from the respective data point in the process interface project.

New data points in the process interface project will result in new process variables of the process model.

There is a special case with EIB projects: When the EIB group address scheme has been changed from 2- level group addresses to 3- level group addresses or vice versa, the structure tree of the process variables will be changed accordingly. However, the process variables will be retained.

4.5 Properties of process variables

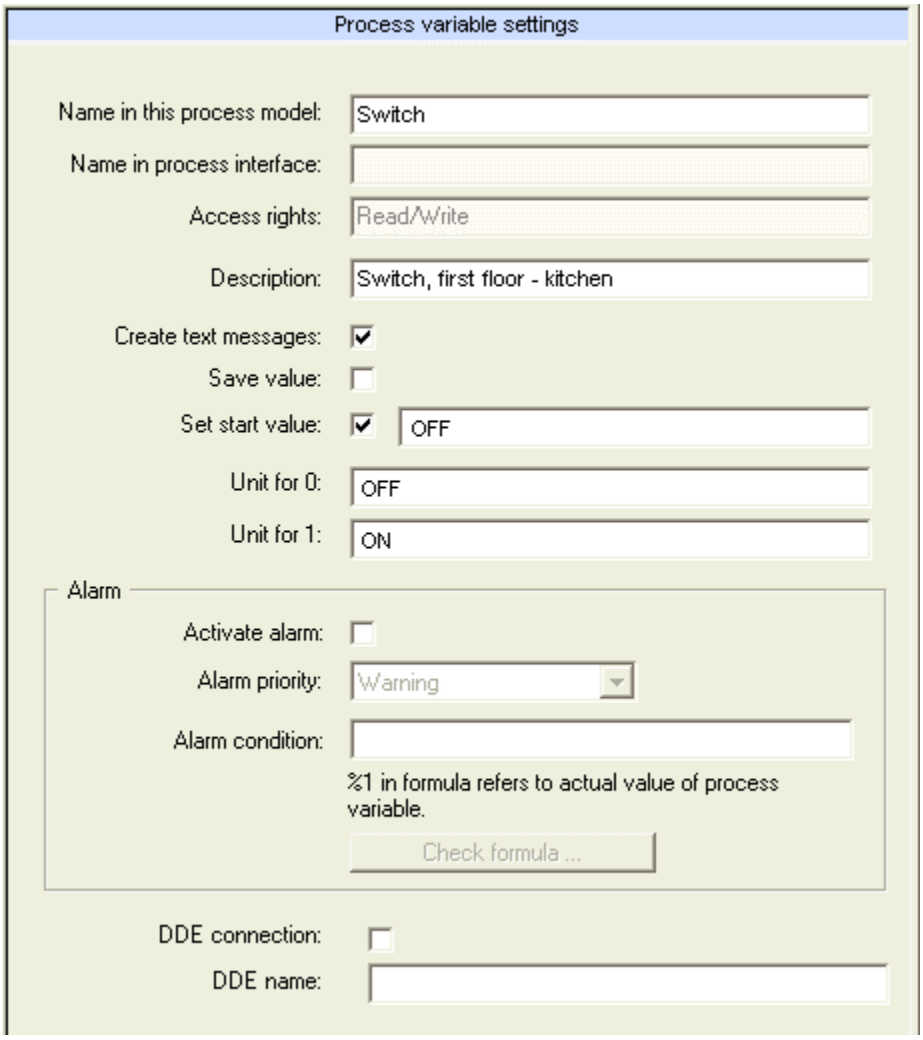
For each data point imported from a process interface project, i.e. an EIB project, a process variable in the process model is created.

The process variable in the process model is initialized from the data point in the respective process interface project. In addition to the properties copied from the data point, email notifications, archives, and other properties can be added to the process variable.

To edit the properties of a process variable click left on the process variable in the process variables structure tree.

Properties of binary process variables:

> ***Note:*** *The mandatory properties of process variables are already initialized when the process interface project has been imported. All changes to the process variable's properties are optional.*



Name in the process model: The name of the process variable is initialized with the name of the data point in the process interface project, i.e. EIB group address in an EIB project. Here it can be changed to a more meaningful name in the context of the application.

Description: The description is initialized with the description of the data point in the process interface project. Here it may be changed.

Create text messages: If on, for each change of the process variable's value a text message will be created, which will be displayed in the visualization player's message window. Otherwise the generation of a text message will be suppressed.

Save value: If on, on termination of the process model the value of the process value will be saved and the process variable will be initialized with this value when the process model will be restarted.

Set start value: If on, the variable will be initialized with the defined value, when the process model is started, unless a previously saved value for the process variable is available. A previously saved value will override the start value.

Unit for 0: Internally, binary values are encoded with the values 0 and 1. It is possible to assign a text to each numerical values, such that in plain messages the text will be used instead.

Unit for 1: See "Unit for 0".

Activate alarm: When the value of a process variable is changing, the process model will check an alarm condition if "Activate alarm" has been turned on. If the alarm condition holds true, a specified alarm handling will be started. In turn, the alarm will be signaled and for example an email message will be sent, if configured. All alarms and warnings are inserted into the system log file.

Alarm priority: Warnings are less severe alarms, depending on the application.

Alarm condition: The alarm condition is defined as a formula. Use "%1" (without quotes) in the formula to refer to the actual value, which is either 0 or 1 for binary variables.

Example: *Signal alarm when the value is 1*

Formula: %1 = 1

DDE connection: If turned on, a WINDOWS DDE (Dynamic Data Exchange) connection with read access will be set up for the process variable.

DDE name: DDE item name used to read the process variable's value from other applications, i.e. Microsoft Excel.

To address the process variable from DDE, a complete specification of the process variable is composed of the name of the DDE server ("iPhon DDE Server"), the process model's name and the DDE item name of the process variable.

Excel example:

DDE name (DDE item):	room_temperature
Process model (DDE topic):	school.prj
iPhon DDE Server (DDE server):	iPhon DDE Server

The DDE variable's name in Excel:

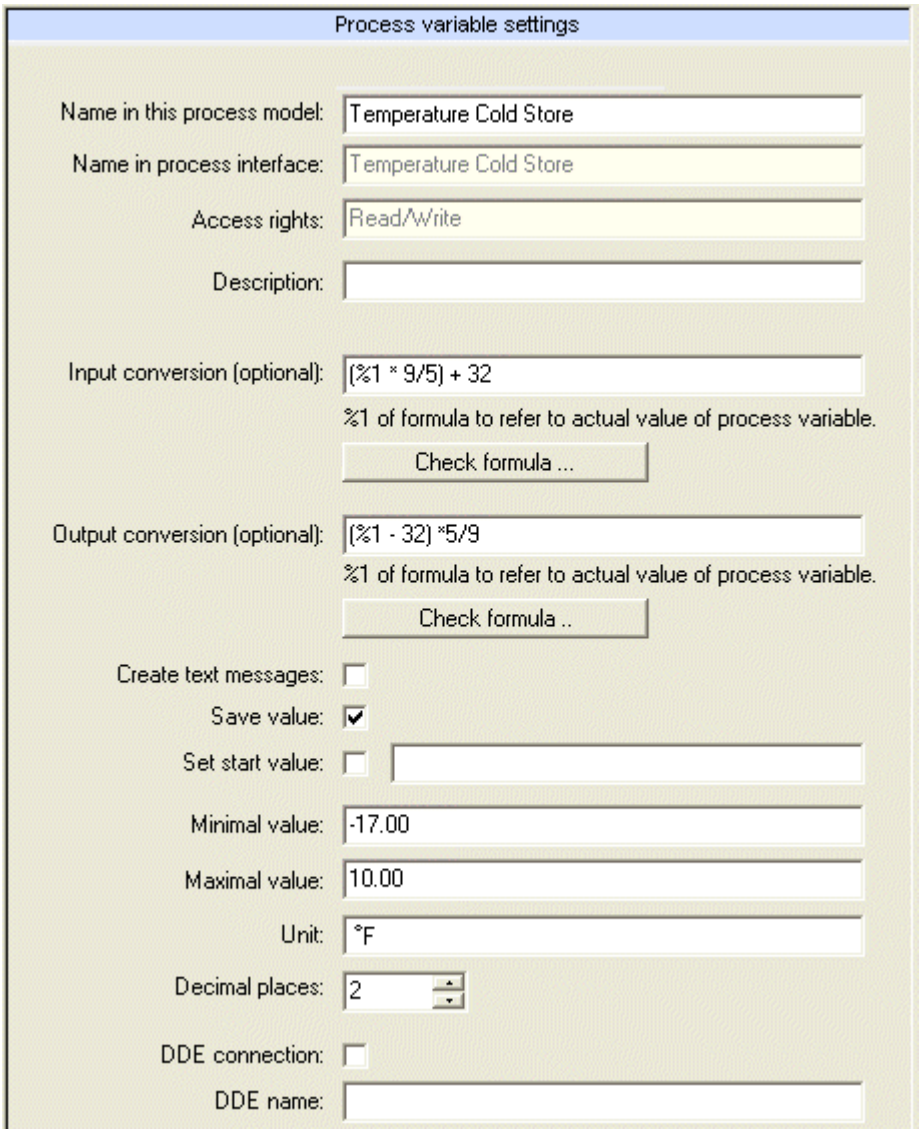
'iPhon DDE Server'|school.prj!room_temperature

Excel requires names quoted with single quotation marks, if spaces are used.

Access rights and name in the process interface project must be modified in the process interface project and cannot be edited here.

Properties of analog process variables:

> ***Note:*** *The mandatory properties of process variables are already initialized when the process interface project has been imported. All changes to the process variable's properties are optional.*



Name in the process model: The name of the process variable is initialized with the name of the data point in the process interface project, i.e. EIB group address in an EIB project. Here it can be changed to a more meaningful name in the context of the application.

Description: The description is initialized with the description of the data point in the process interface project. Here it may be changed.

Input conversion: The value received from the process interface is not always of the proper type from an application's point of view. For example a temperature sensor may return a voltage, which must be converted to a temperature.

The conversion can be described with a formula. Use "%1" (without quotes) in the formula to refer to the actual value.

Operators: +, -, ^, *, /, \, Mod, +, -, &, =, <>, <, >, <=, >=, Not, And, Or, Xor

You may also use parenthesis.

> ***Example:*** *The sensor sends a voltage instead of a temperature. The range 0...10 V must be depicted to 0...15 °C.*
>
> Formula: %1 * 1.5

Output conversion: The value can be converted before it will be sent, similar to the conversion applied to values received.

Check formula: Button **Check formula** opens a dialog, which will immediately show the result of a formula applied to different input values.

Minimal value: The process model will not send values below the minimal value. This option can be used to ensure proper operation of devices.

Maximal value: The process model will not send values above the maximal value. This option can be used to ensure proper operation of devices.

Unit: If defined, the unit's text will be used in plain messages.

Decimal places: Number of decimal places used for plain messages.

DDE connection: If turned on, a WINDOWS DDE (Dynamic Data Exchange) connection with read access will be set up for the process variable.

DDE name: DDE item name used to read the process variable's value from other applications, i.e. Microsoft Excel. See binary process variables for an example.

4.6 Calculated process variables

Function description:

In addition to process variables created from data points in process interface projects, i.e. EIB or OPC projects, new process variables can also be defined by the user. The values of self-defined process variables usually are calculated from other already existing process variables. As an exception, the value of a self-defined process variable may not be calculated based on others, but for example controlled by a calendar program. This is regarded to as a special case of calculated process variables.

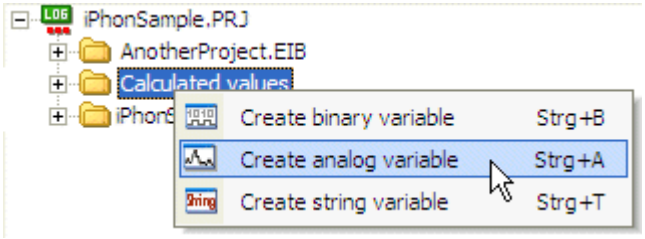
Example: *Counter*

The process interface returns pulse information, i.e. from an electricity counter, each pulse representing a certain energy consumption. The process model provides functions, which allow to count the pulses and to add up the consumption values. In this case a new process variable will be defined to hold the total consumption.

The process model provides the same features to self-defined process variables as to process variables derived from data points in process interface projects. For example, you may create archives of self- defined variable's values and the alarm handling applies.

Creation of self-defined process variables:

To define a process variable first right click on folder **Calculated values** to open the folder's context menu. Then select the type of process variable to be created.



The editor will open a window to define the process variable's properties.

4.7 Alarm limits

Function description:

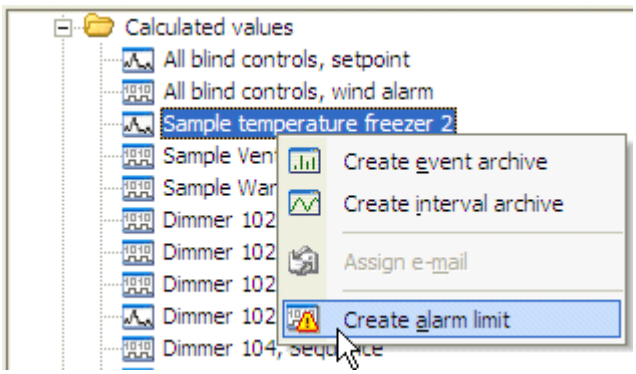
Binary process variables can simply define alarms with an optional alarm condition. The alarm condition is defined with a formula based on the binary process variable's value. Since a binary variable's value is either 0 or 1 there is at most one alarm condition.

Analog process variables may require several warnings and alarms, i.e. a warning when the temperature reaches 5° Celsius and an alarm when the temperature reaches 10° Celsius.

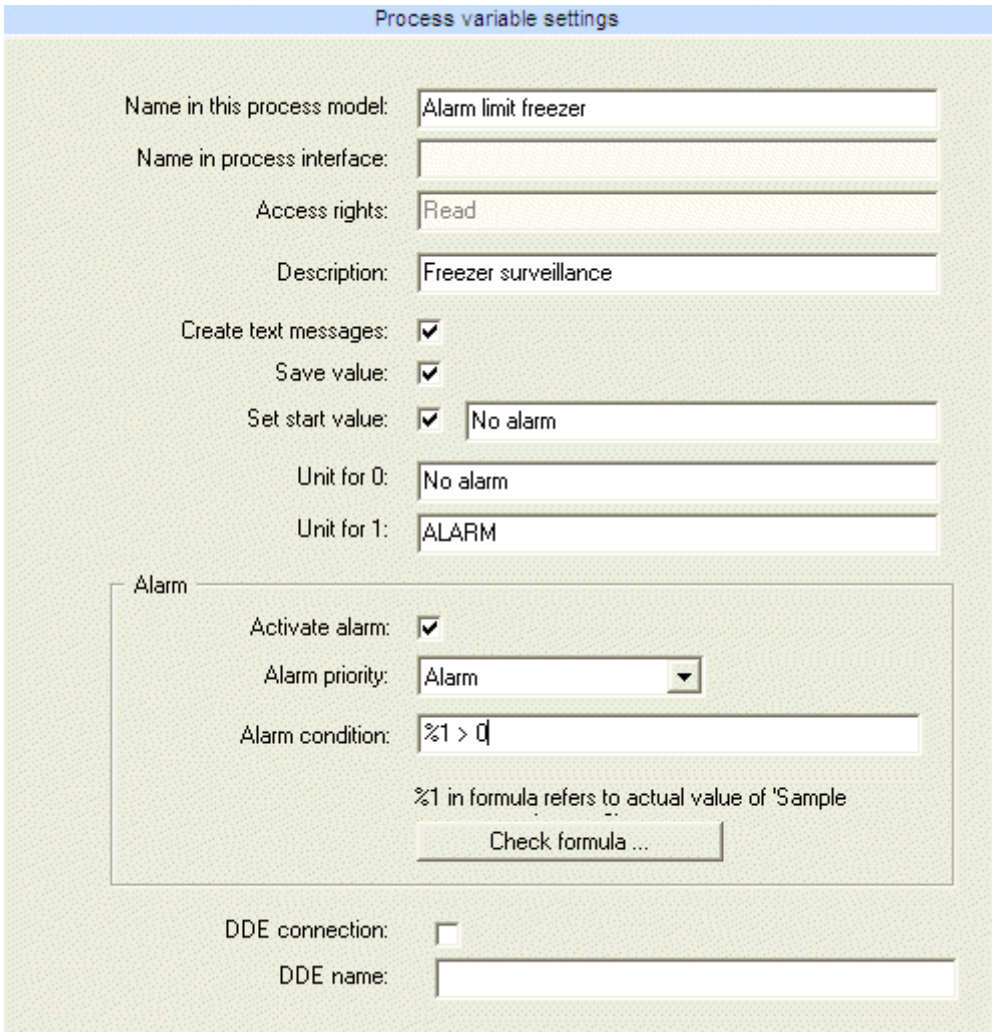
For each alarm condition an alarm limit can be defined, which has value 1 if the condition on the analog variable holds true and value 0 otherwise. The condition is defined with a formula. These alarm limits are quite similar to regular binary variables, the most important difference is the alarm condition: The alarm condition does not refer to the alarm limit's value but to the analog variable to which it has been attached.

Create alarm limit:

Click right on an analog process variable to open it's context menu. Then select **Create alarm limit**. Beneath the analog process variable an **Alarm limits** subfolder will be created which comprises already a first alarm limit.



Select the alarm limit to define the alarm limit's properties.



Name in this process model: The name of the alarm limit.

Description: The description of the alarm limit.

Create text messages: If on, for each change of the alarm limit's value a text message will be created, which will be displayed in the visualization player's message window. Otherwise the generation of a text message will be suppressed.

Save value: If on, on termination of the process model the value of the process value will be saved and the process variable will be initialized with this value when the process model will be restarted.

Set start value: If on, the variable will be initialized with the defined value, when the process model is started, unless a previously saved value for the process variable is available. A previously saved value will override the start value.

Unit for 0: Internally, binary values are encoded with the values 0 and 1. It is possible to assign a text to each numerical values, such that in plain messages the text will be used instead.

Unit for 1: See "Unit for 0".

Activate alarm: When the value of the alarm limit is changing, the process model will check an alarm condition if **Activate alarm** has been turned on. If the alarm condition holds true, a specified alarm handling will be started. In turn, the alarm will be signaled and for example an email message will be sent, if configured. All alarms and warnings are inserted into the system log file.

Alarm priority: Warnings are less severe alarms, depending on the application.

Alarm condition: The alarm condition is defined as a formula. Use "%1" (without quotes) in the formula to refer to the value of the analog variable.

> ***Example:*** *Signal alarm when the analog variable's value is above 0.*
>
> Formula: %1 > 1

DDE connection: If turned on, a WINDOWS DDE (Dynamic Data Exchange) connection with read access will be set up for the alarm limit.

DDE name: DDE item name used to read the alarm limit's value from other applications, i.e. Microsoft Excel.

To address the alarm limit from DDE, a complete specification of the alarm limit is composed of the name of the DDE server ("iPhon DDE Server"), the process model's name and the DDE item name of the alarm limit.
See properties of binary process variables for an example.

4.8 Email notifications

Email notifications can be assigned to binary process variables and to alarm limits.

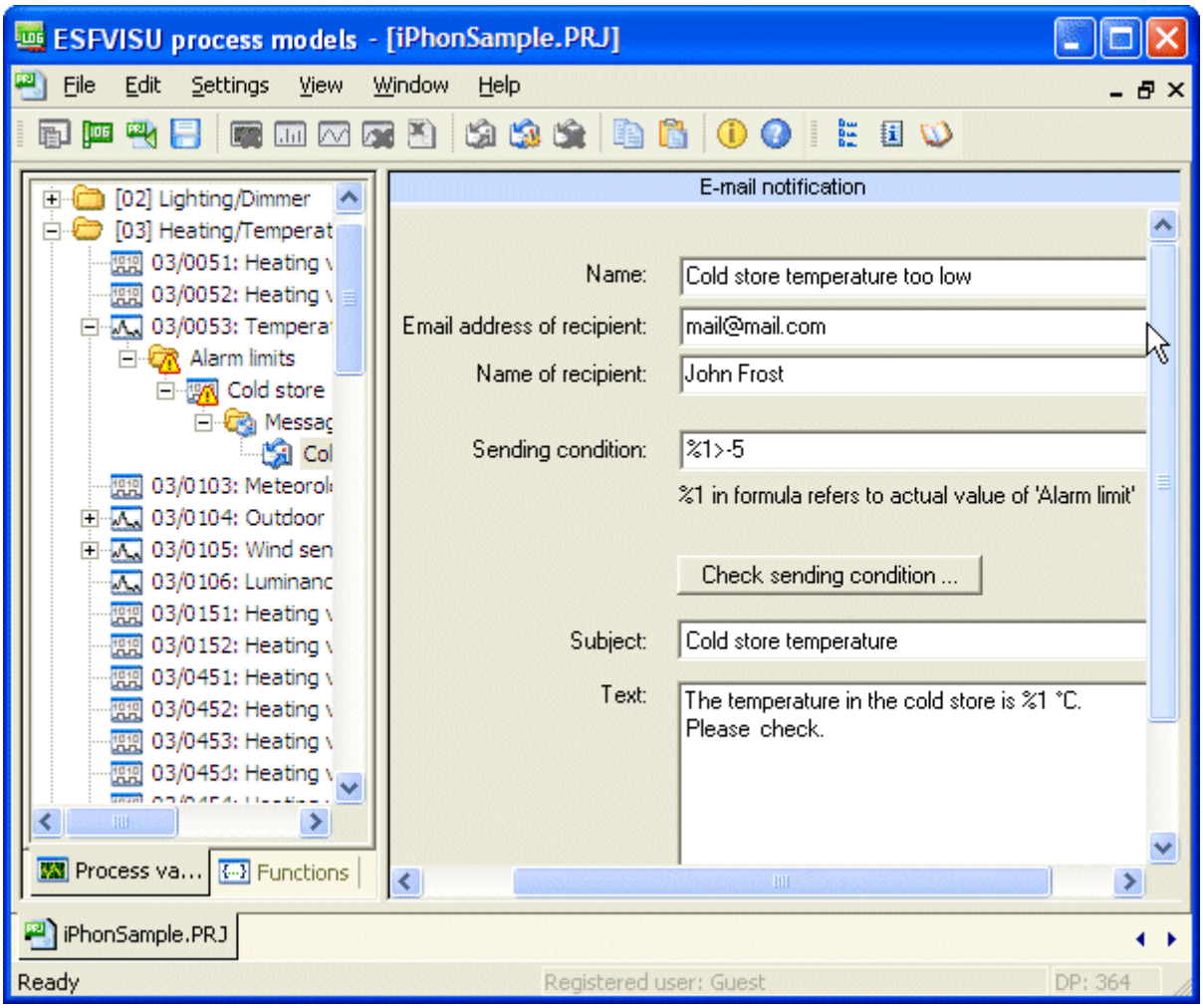
Whenever the value of a binary process variable or an alarm limit is about to change, the process models check whether an email notification is to be sent. Each email notification comprises it's own condition. In particular the condition to send an email notification is not identical with an alarm condition: Alarm personnel may not only be informed when an alarm condition becomes true, but also when later the alarm condition becomes false.

> ***Note:***
> *Email notifications can only be used if the computer is configured to send emails.*

4.8.1 Create email notification

Left click on a process variable and use the menu option **Edit – Assign email** or click the toolbar symbol to create a new email notification for the process variable.

The process variable will comprise a subfolder **Notifications** with an empty email notification already defined. Click on the newly created email notification to configure it's properties.



Name: Name of the email notification. This name will be used in the structure tree to identify the email notification.

Email address of recipient: Enter the recipient's email address.

Name of recipient: Enter the recipient's name.

Sending condition: A formula is used to specify the sending condition. The email will be sent when the formula returns true.

Use "%1" (without quotes) in the formula to refer to the actual value of the process variable respectively the alarm limit.

Operators: +, -, ^, *, /, \, Mod, +, -, &, =, <>, <, >, <=, >=, Not, And, Or, Xor

You may also use parenthesis.

Subject: Enter the email's subject.

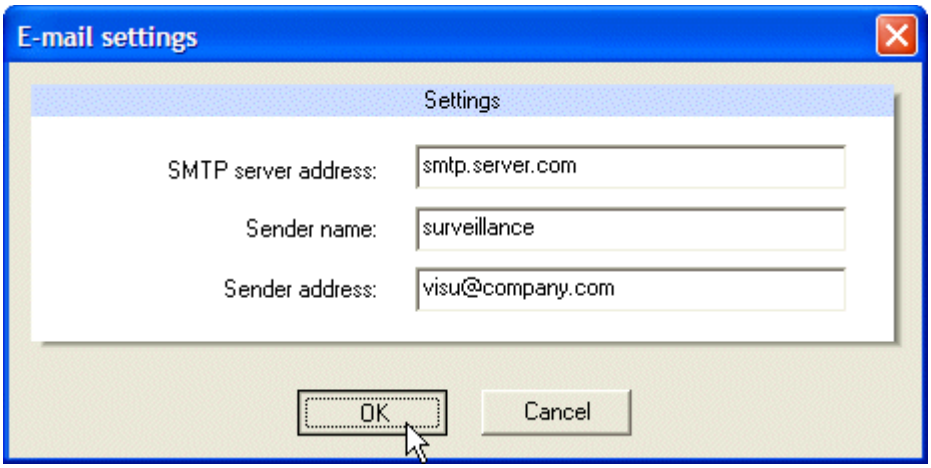
Text: The email's text. Use "%1" (without quotes) in the formula to refer to the actual value of the process variable respectively the alarm limit.

4.8.2 Delete email notification

Use the menu option **Edit - Delete email** or click on the toolbar symbol to delete the email notification. Alternatively press the keyboard **Delete** key.

4.8.3 General email settings

In order to send emails, the process model requires some general information.
Use the menu option **Settings - Email configuration** or click the toolbar symbol to configure general email settings.



Address of SMTP server: Enter the address of the SMTP server.

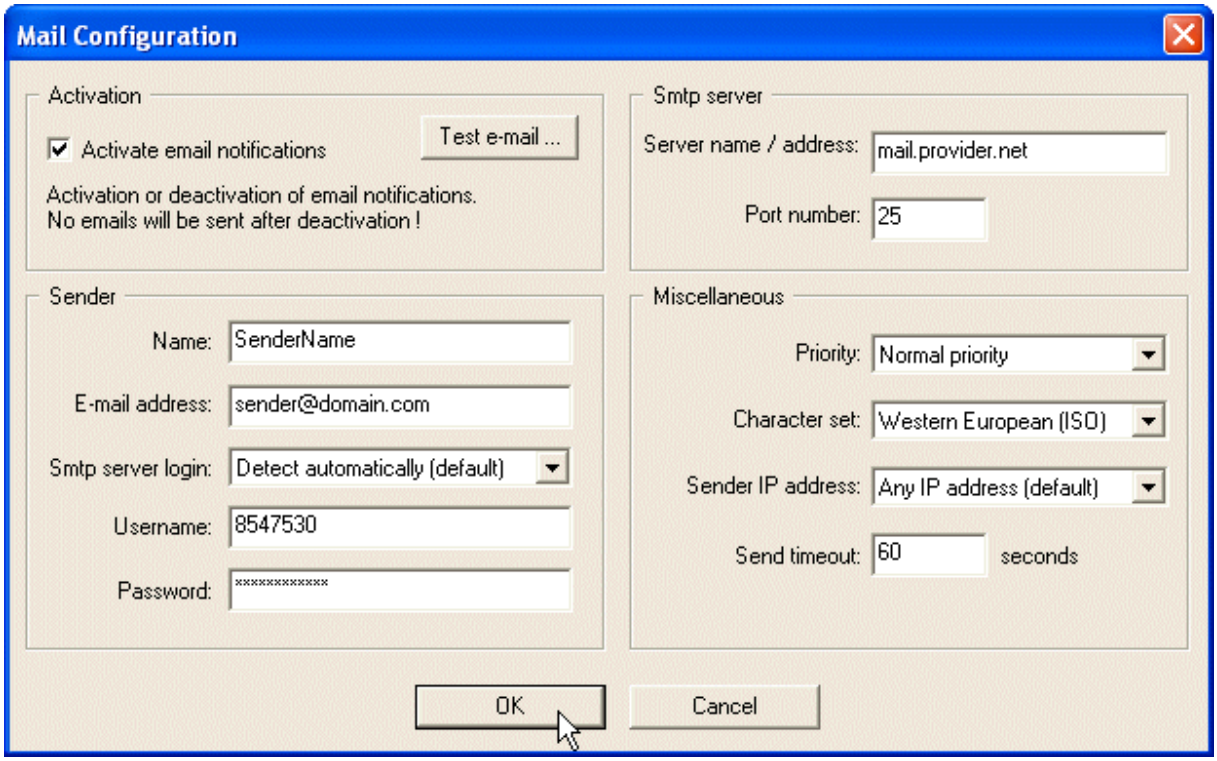
Sender's name: Enter the sender's name.

Sender's email address: Enter the sender's email address.

4.8.4 Extended E-Mail Settings:

The configuration of the e-mail client can be done directly in the user interface of the e-mail client. The e-mail client is visible in the Windows status line as a small icon:

Double click on the icon with the white picture opens the dialog with the e-mail client settings:



Activation:

Activate email notifications: Activates or deactivates the e-mail client.

Note: *No e-mails will be sent if the e-mail client is inactive !*

All changes of the activation state will be recorded in the system logfile. On system start the activation state of the e-mail client will be recorded in the logfile.

Test e-mail: Opens the dialog to send an e-mail for testing.

To send the test mail the e-mail client uses the same settings, which are visible in the dialog.

Sender:

Name: Name of the sender.

E-mail address: e-mail address of the sender. This field may not be empty.

On e-mail reception the sender identification will be composed of the sender name and the sender's e-mail address, e.g.

John Testman <j.testman@domain.com>

If the name of the sender is empty, the e-mail address will be used as sender identification.

Smtp server login: specifies the authentication mechanism to log in to the Smtp server.

The following options are availabe:

***Detect automatically (default)*:** ESFVISU automatically selects an appropriate mechanism to log in to the smtp server of the e-mail provider. ESFVISU supports the methods AUTH CRAM-MD5, AUTH LOGIN und AUTH PLAIN.

***Authentication: AUTH CRAM-MD5*:** AUTH CRAM-MD5 will be used to log in to the Smtp server of the e-mail provider.

***Authentication: AUTH LOGIN*:** AUTH LOGIN will be used to log in to the Smtp server of the e-mail provider.

***Authentication: AUTH PLAIN*:** AUTH PLAIN will be used to log in to the Smtp server of the e-mail provider.

None: No authentication mechanism will be used to log in to the Smtp server of the e-mail provider.

Username: The user name, which will be used by the e-mail client to login on the smtp server. The user name will be provided by the e-mail service provider.

Password: The password, which will be used by the e-mail client to login on the smtp server. The password will be provided by the e-mail service provider.

Smtp server:

Server name / address: Address of the smtp server.

Port number: Port number of the smtp service of the smtp server. The default port number is 25.

Miscellaneous:

Priority: Priority of the e-mail.

Character set: specifies the character set to encode the e-mail message.

Sender IP address: specifies the IP address, which shall be used by the e-mail client to connect with the server. Usually this address will be assigned automatically (default: any IP address).

Send timeout: specifies the timespan the client shall wait for answers from the mail server while establishing the connection.

The last 25 send attempts will be archived in protocol files. These send logs will be stored in the zip-archive '*CallServerLog.zip*' in the 'Temp'-directory of the visualization.

4.9 Checking formulas and conditions

Formulas are used as general means to define conditions.

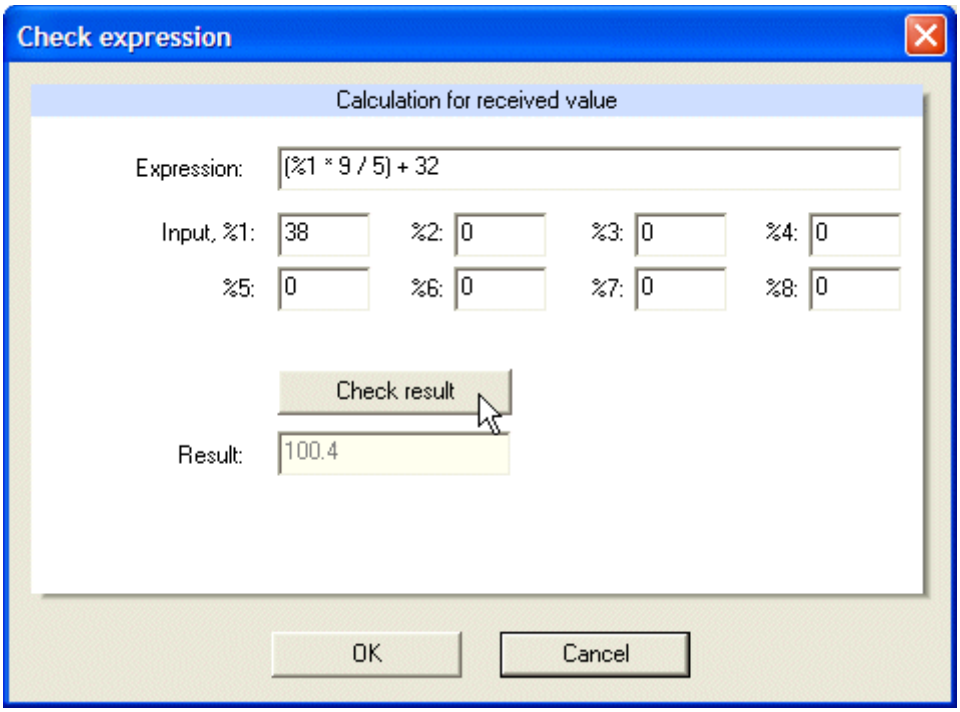
Operators: +, -, ^, *, /, \, Mod, +, -, &, =, <>, <, >, <=, >=, Not, And, Or, Xor

You may also use parenthesis.

Different dialogs allow to enter formulas and also provide the possibility to check the formula before it will be used.

The formula checker dialog allows to enter a mathematical expression, which refers to one or more input variables via "%1", "%2",.., the sign "%" followed by the number of the input variable.

To check the formula, arbitrary values can be tested.



For real numbers, the period is used to separate the decimal places.

4.10 Archives

For each process variable one or more archives can be created.

Event archive: In an event archive each change of the process variable's value will be stored.

Interval archive: In an interval archive the archived values are computed based on a defined time interval, i.e. 5 minutes, 1 hour,..
For each interval the interval archive will archive several computed values:

- Average of all values received.
- Minimum of all values received.
- Maximum of all values received.
- Total sum of all values received (useful for calculation of consumption values).
- Difference of total sum to respective value of previous interval (useful for calculation of consumption values).

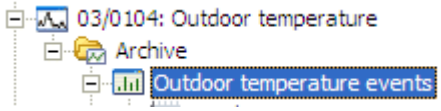
Limits:

The number of archives is not limited by the process model editor. However there are reasonable limits based on the available disk space, frequency of value changes and the system's computation power.

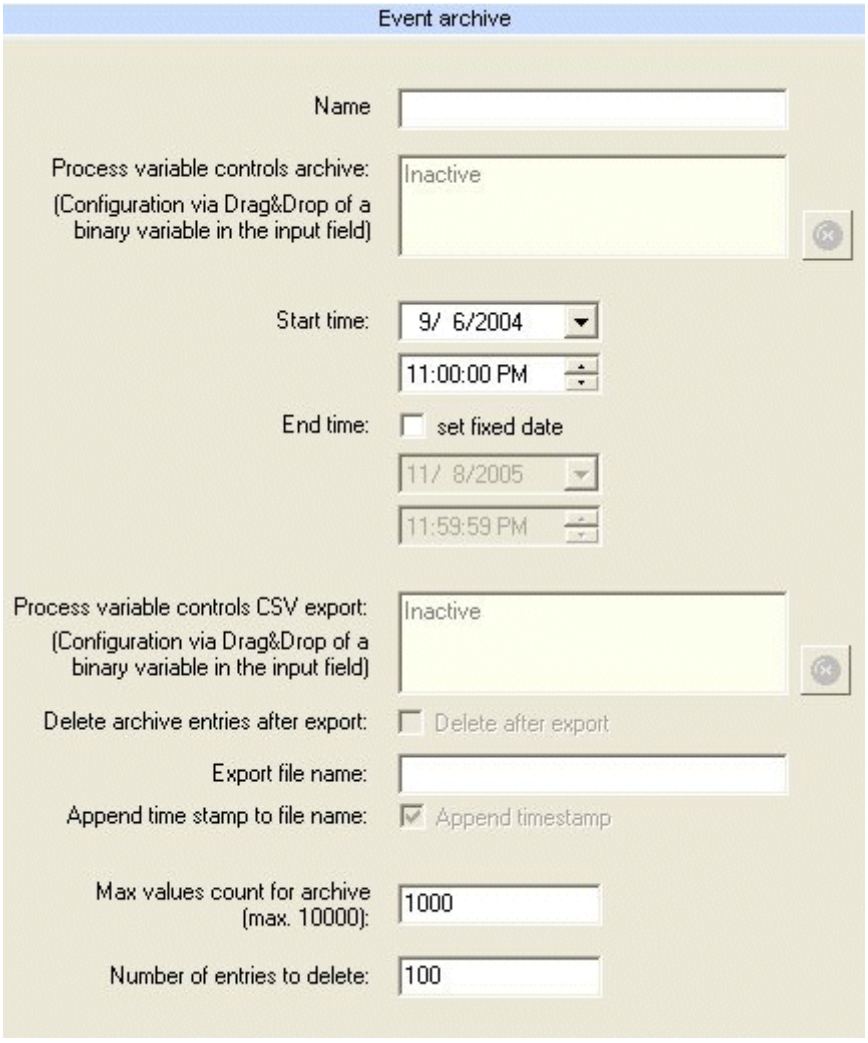
4.10.1 Event archives

Jeder für die Prozessvariable empfangene Wert wird in das Archiv eingetragen.

Create event archive:



Click left on a process variable, then use the menu option **Edit – Create event archive** or click the toolbar symbol . The newly created archive will be assigned to the selected process variable.



Name: Enter the archive's name.

Start time: Enter the date and the time at which archiving values will start.

End time: Enter the date and the time at which archiving values will stop.

If no end time has been specified, the archive will be organized as a first- in- first- out storage. If the maximal number of archived values has been reached, the oldest entries will be deleted to reclaim space in the archive.

Maximal numbers of entries: Maximal number of values in this archive.

Numbers of entries to delete: Deleting entries in the archive to reclaim storage will be more efficient, if quite a number of entries can be deleted at once. Enter the maximal number of entries the archive is allowed to delete to reclaim storage.

Process variable controls archive:

When this field contains a process variable, then the variable controls the activation of the archive. If the value of the process variable is 1, then the archive is active and values will be stored in the archive. If the value is 0, then the archive is inactive and no values will be stored in the archive.
That way time controlled archives can be configured, which are able to collect values only at weekends (using the ESFVISU calendar application) or archives, which will be active on various conditions. Important: the trigger event will not be stored in the archive.

Process variable controls CSV data export:

When this field contains a process variable, then the variable controls the creation of a CSV formatted file containing the archived values. If the value of this process variable changes from 0 to 1, then all archived values will be written into a CSV file.
That way it is possible to weekly or monthly create CSV files with the archived values.

Delete archive entries after export:

Specifies whether the archived data shall be deleted after creation of the export file.

Export file name:

Name of the CSV file to create.

Append time stamp to file name:

Specifies whether the name of the CSV file shall be extended by a timestamp (e.g. to prevent an existing CSV file from overwriting).

Delete event archive

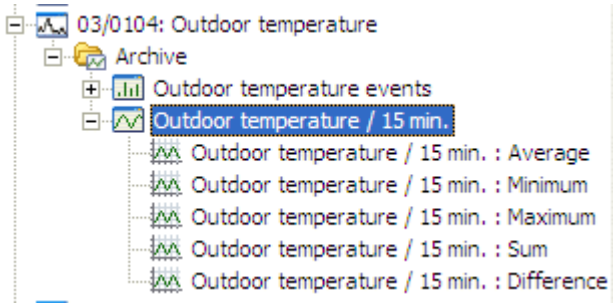
To delete an event archive first select the archive in the structure tree. Then use the menu option **Edit – Delete archive** or click on the toolbar symbol . Alternatively you may press the keyboard **Delete** key.

4.10.2 Interval archives

In an interval archive the archived values are computed based on a defined time interval, i.e. 5 minutes, 1 hour,..

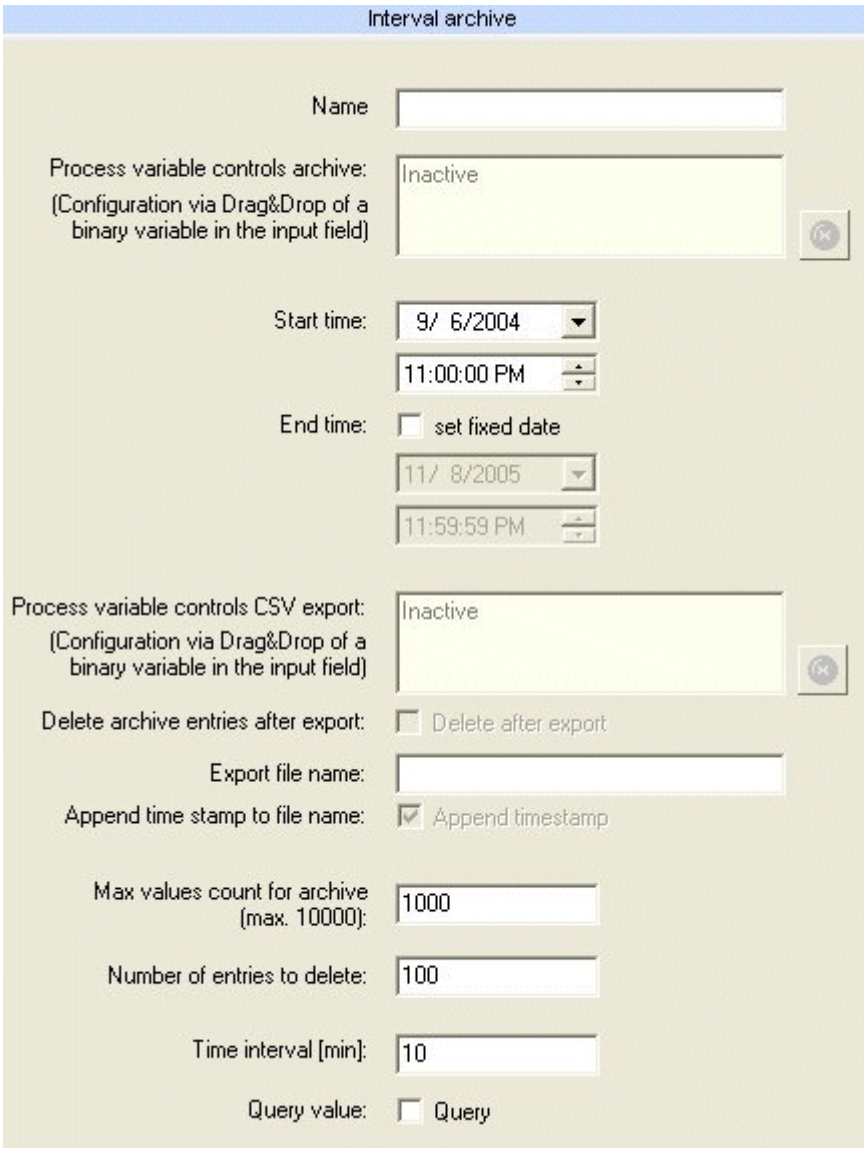
For each interval the interval archive will archive several computed values:

- Average of all values received.
- Minimum of all values received.
- Maximum of all values received.
- Total sum of all values received (useful for calculation of consumption values).
- Difference of total sum to respective value of previous interval (useful for calculation of consumption values).



Create interval archive:

Click left on a process variable, then use the menu option **Edit – Create interval archive** or click the toolbar symbol . The newly created archive will be assigned to the selected process variable.



Name: Enter the archive's name.

Start time: Enter the date and the time at which archiving values will start.

End time: Enter the date and the time at which archiving values will stop.

If no end time has been specified, the archive will be organized as a first- in- first- out storage. If the maximal number of archived values has been reached, the oldest entries will be deleted to reclaim space in the archive.

Maximal number of entries: Maximal number of values in this archive.

Number of entries to delete: Deleting entries in the archive to reclaim storage will be more efficient, because quite a number of entries can be deleted at once. Enter the maximal number of entries the archive is allowed to delete to reclaim storage.

Time interval: Enter the interval in minutes.

Query value: If enabled, within each interval the value of the process variable will be queried. This options avoids intervals with no value.

Note: *You may define several archives with different time intervals for one process variable.*

Example: Archiving temperatures

There are two interval archives for a temperature process variable:

One archive with

- Time interval: 10 min
- Maximum entries: 144 entries
- Query values: Yes

This archive will store 10 minutes average values for the last 24 hours.

One archive with

- Time interval: 60 min
- Maximum entries: 720
- Query values: No (values are already queried)

This archive will store hourly average values for the last 30 days.

Example: Archiving consumption values

There are two archives for a process variable receiving pulses, each pulse representing a certain consumption value.

One archive with

- Time interval: 15 min
- Maximum entries: 960
- Query values: No

This archive will count 15 minutes sums for the last 10 days.

One archive with

- Time interval: 60 min
- Maximum entries: 720
- Query values: No

This archive will count hourly sums for the last 30 days.

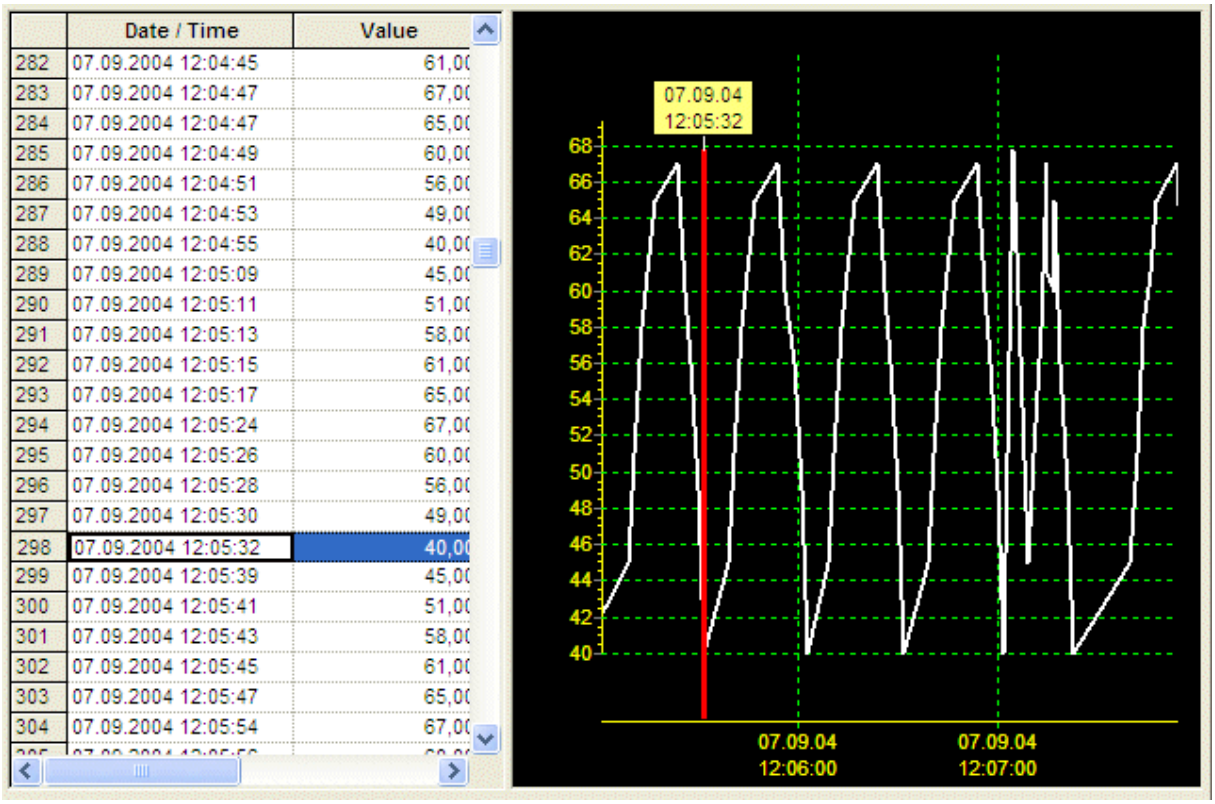
Delete interval archive

To delete an interval archive first select the archive in the structure tree. Then use the menu option **Edit – Delete archive** or click on the toolbar symbol  . Alternatively you may press the keyboard **Delete** key.

4.10.3 View archives

The process model editor provides a viewer for archived values.

Double click on the **archive** in the process variables structure tree to view the archive's values.



The viewer provides both the **data tables** and a **graphical representation**.

Double click on the **graphical representation** to modify the graph's layout. The modified layout will be saved for future use.

Zoom:

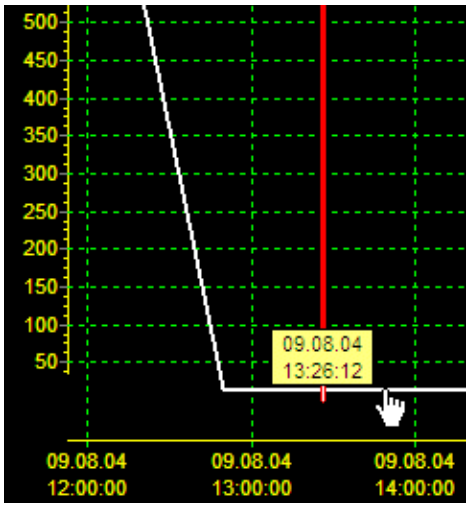
To enlarge an area of the graph click left on the top left corner of the area's rectangle, then move the mouse to the opposite corner (bottom right) while the mouse buttons remains pressed. Then release the mouse button.

The area will be enlarged and centered in the window.

To reset zoom mode click on the graph and move the mouse left while the mouse button remains pressed. Then release the mouse button.

Move the graph's cursor:

Move the mouse and the mouse cursor will change it's shape to a hand. Then left click the mouse at the desired position and the graph's cursor will jump near to the mouse position.



or:

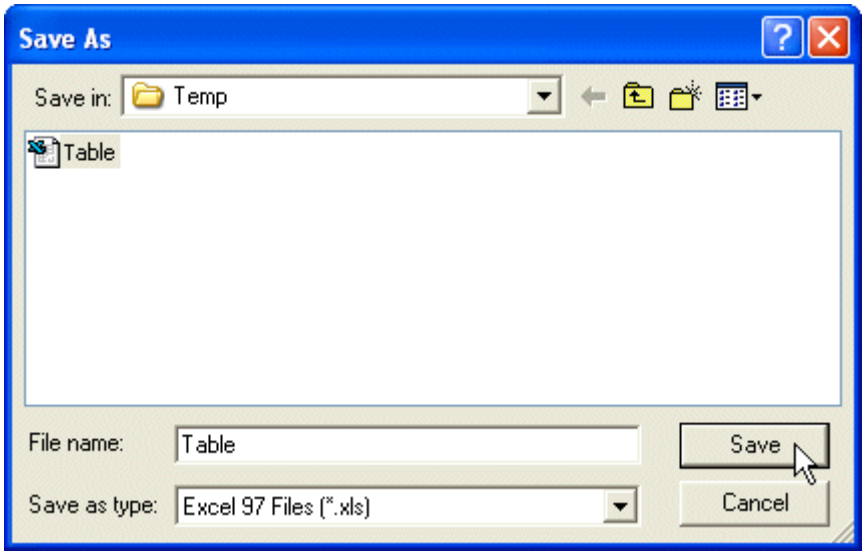
Move display rectangle:

The visible part of the display area can be moved horizontally. Click right on the graph and move the mouse while the right mouse button remains pressed.

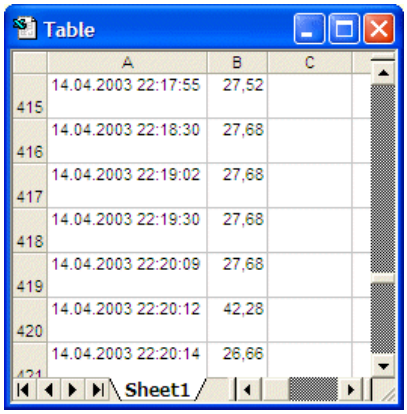
4.10.4 Export to Excel or as CSV-File

The archived values can be exported to Microsoft Excel.

Select a range of values in the data table. Then use the menu option **Edit – Export to Excel** or click the toolbar symbol . A file dialog will open to save an Excel (***.xls**) – file with the exported values.



If text file (*.csv) selected, the table will be saved in CSV-format.

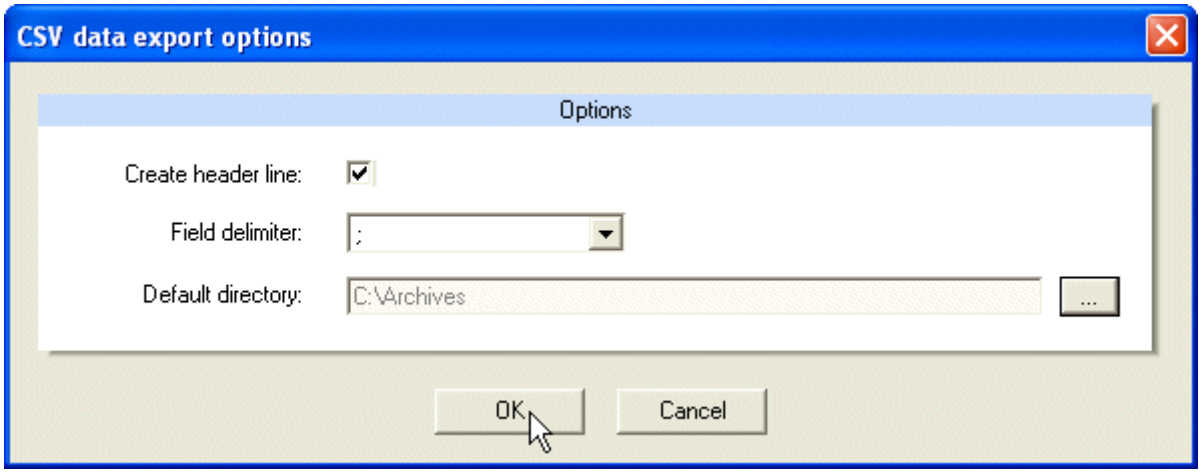


Microsoft Excel file:

After the file with extension (***.xls**) has been saved, you may evaluate the archived values with Microsoft Excel.

4.10.5 Configuration of CSV export

Menu option **Settings – CSV export configuration** opens the CSV data export options dialog.



Create header line: Specifies whether a CSV file shall contain a header line or not.

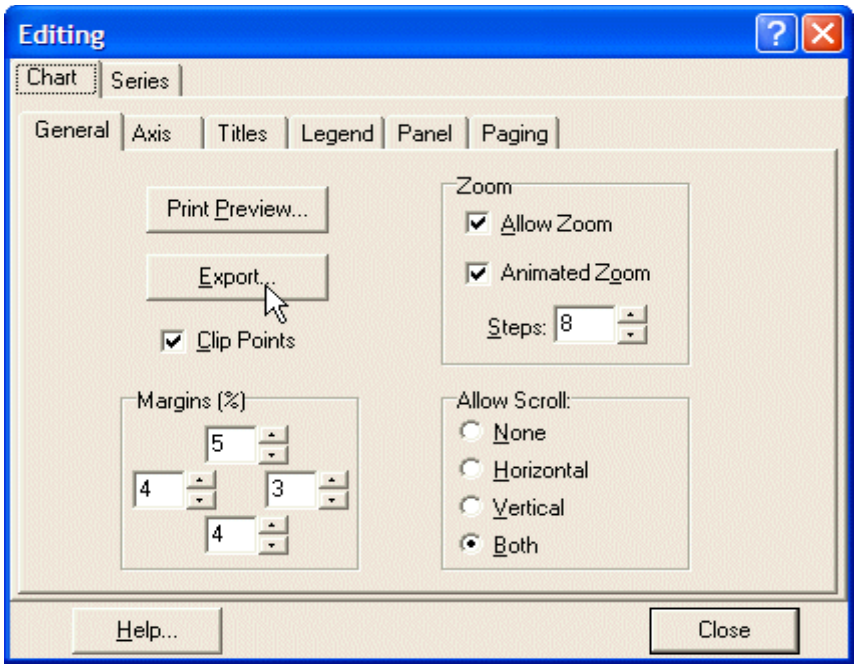
Field delimiter: Selection of the CSV field delimiter (; , TAB). Default: ;

Default directory: Selection of the directory for the CSV file export.

4.10.6 Graph export and printing

The graph can be edited, printed, saved as BMP/JPG file or copied to the clipboard.

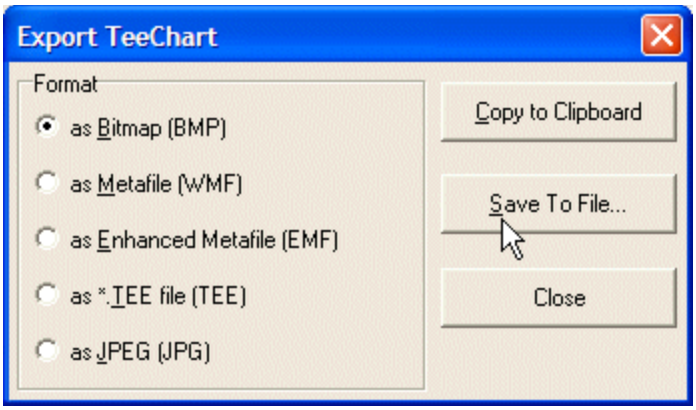
Double click the graph to change it's properties.



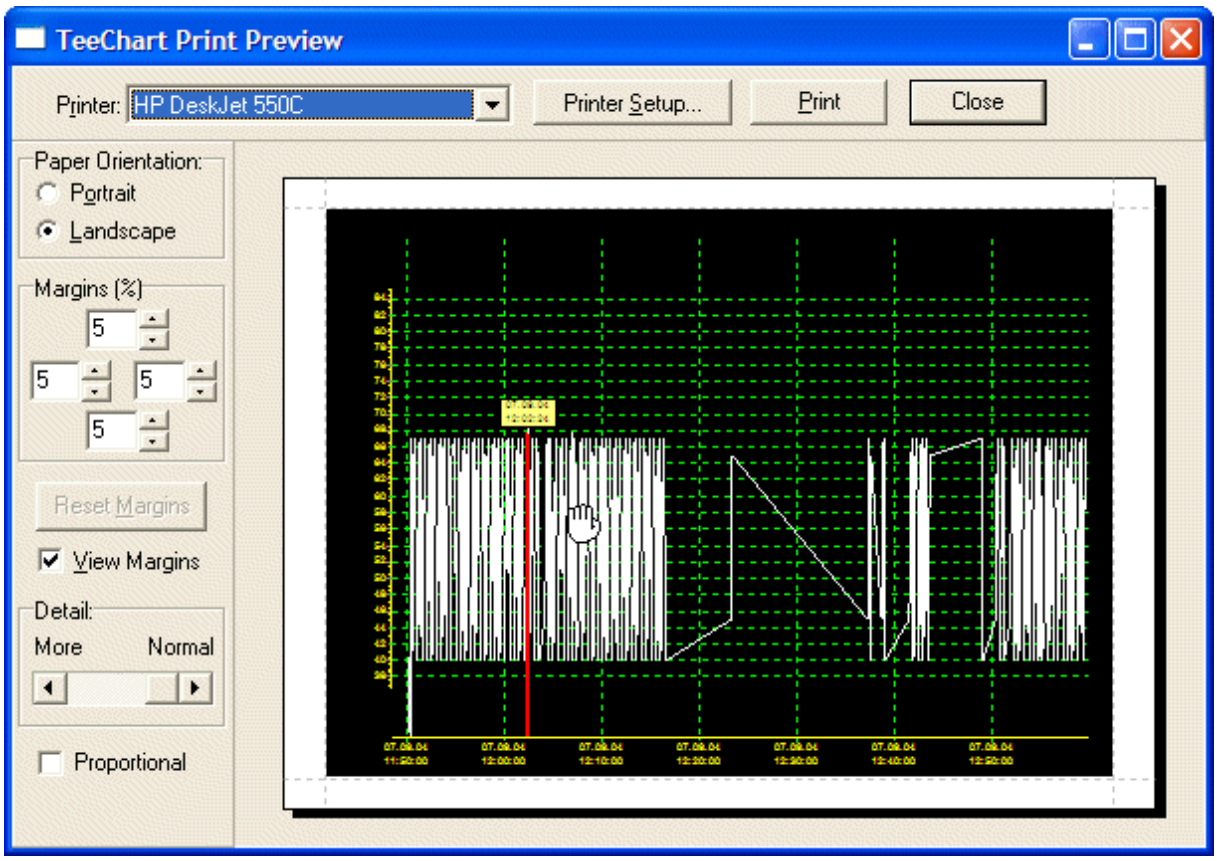
Save graphic to file: Press button **Export..** to start the export dialog.

Copy to Clipboard: The graphic will be available to other programs via the clipboard.

Save to File: The graphic can be saved as WMF-, EMF- or JPG- file.



Print Preview: Show print preview of the graphic, possibly print the graphic.



Select printing properties and press button **Print**.

4.11 Controller

The running process model is an active controller of the connected technical processes and supplements the existing installation with additional functions.

These functions include logical functions (AND, OR, XOR), status objects, forwarding, mathematical functions, time functions, counting functions (i.e. for consumption counters), sequences, scenes (i.e. light scenes), and guard functions (to secure a certain operating condition).

The evaluation of the process model runs in cycles:

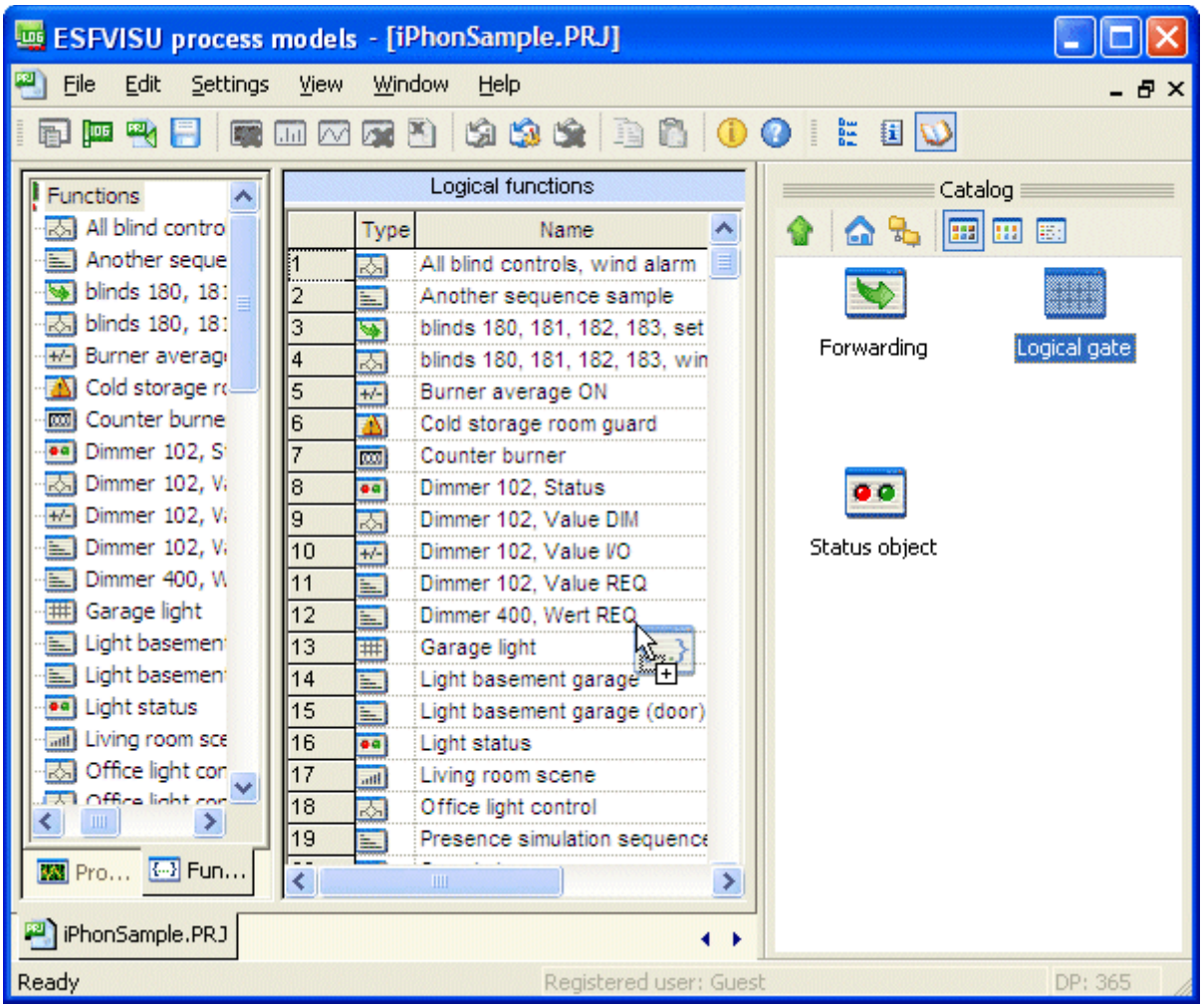
- **Fetch**
  For all functions copy variables to input parameters.
- **Execute**
  For all functions evaluate output parameters.
- **Update**
  For all functions copy output parameters to the variables and if configured send values to the process interface.
- Continue with step a)

In particular the order of the function's evaluation is not significant, since during one cycle the functions are evaluated independently.

4.11.1 Edit controller functions

To add a controller function first select the tab **Functions** in the structure tree at the left. The main window will show a grid with all controller functions configured so far.

Then in the catalog window select **Functions** and the respective function's category. Drag the symbol of the desired function in the catalog window to the function's grid in the main window.



To edit a controller function click left on the function in the function's structure tree at the left hand. Alternatively in the function's list double click in the function's row. The main window will show the configuration of the function.

You may edit the name of the description of the function directly in the grid.

4.11.2 Logical gate

Function description:

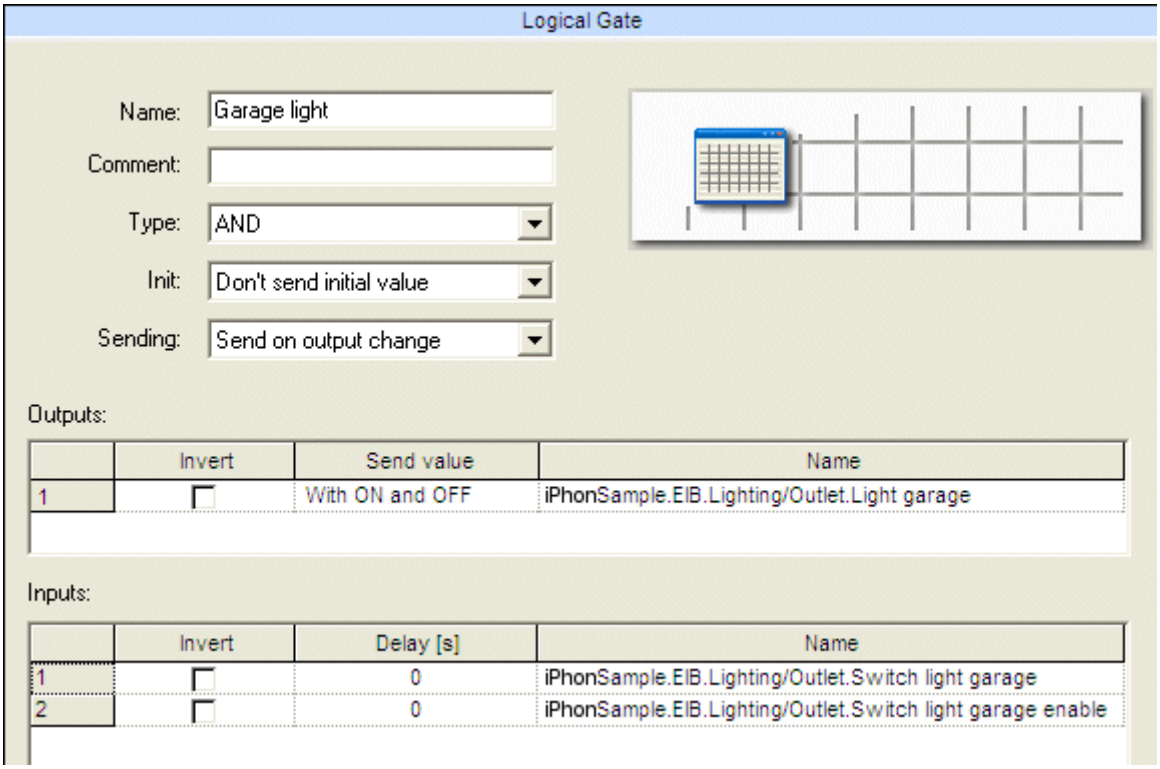
The logical gate is used to configure logical connections. It may be a AND, OR, or XOR (logical exclusive or)– gate.

The gate's logical function will be applied to the input variables, the result will be assigned to the output variable. Whether the output variable's value will be sent to the process interface (i.e. the EIB group address will be sent) depends on the configuration.

Configuration:

From the catalog window, category process data, drag the process variable to the grid of input variables respectively output variables.

To delete a variable from the grid first mark the row with left click on the first column in the row. Then press the keyboard's **Delete** key.



Function settings:

Name: The name of the function

Comment: A short description of the function

Type: Select the logical function, may be AND, OR, or XOR.

Init: Allow or disallow sending the function's result when the function is evaluated the first time, i.e. after the process model has been started.

Sending: Decide when to send the function's result.

On changing output: The function's result will only be sent when the function's result has been changed.

On input update: The function's result will always be sent when an input has been updated, regardless of the input's value.

Outputs: Process variables, which will hold the function's result.

Inputs: Process variables as the inputs of the function.

Settings for each output:

Invert: Decide whether the output should be inverted.

Sending: Choose a condition when to send the output value to the process interface.

Settings for each input:

Invert: Decide whether the input should be inverted.

Delay: Define a delay in seconds for the evaluation of changed input values. Changes will be ignored, if they are not stable for at least the delay period.

4.11.3 Basic script function

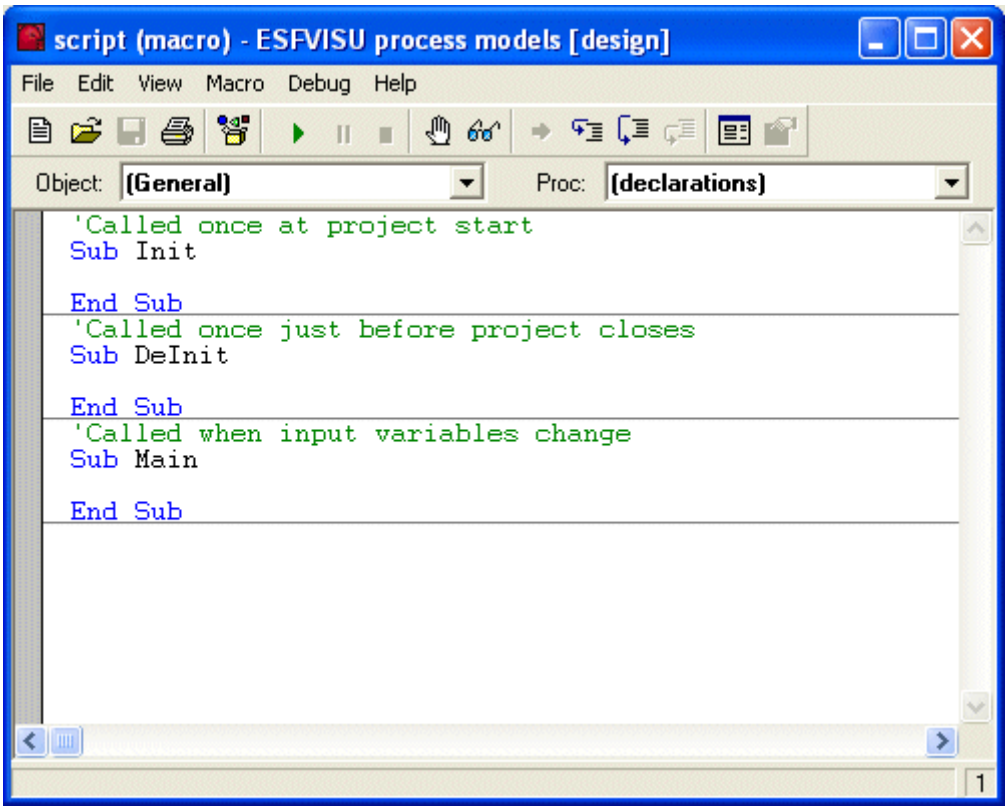
Function description:

The 'basic script function' enables to proceed self- defined scripts if the value of the process variables changes.

Create basic script:

To create a script, label a script data file (without the ending '.bas') and then press the button 'edit script...'.

The script editor will open and a default script, including 3 functions, will be shown:



Meaning of the script functions 'Init', 'DeInit' und 'Main':

Init: This function will be executed once when the process model starts. It can be used to intitialize process variables (i.e. set initial values, open data,...).

DeInit: This function will be executed once when the process model is terminated. It can be used i.e. to close opened data files.

Main: This function will always be executed if the value of one or more than one initial variables has changed.
Initial variables are variables that are included in the table 'Process variables that can be used in basic script' and who's check boxes are activated in the column 'input'.

Execution of the scripts by the process model:

The process modell executes all functions, including the basic scripts, in a cyclical way. During one execution cycle the values of all the input variables are tested. If they have changed, the script will be started and the function 'main' will be called. Afterwards the values of the outputs variables (not as 'input' activated variables) will be activated. If the output variables are connected to the process and their values have changed, then the changes of value will be sent to the process.

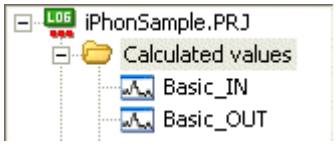
> ***Important hint:*** *No functions, waiting for events and user inputs (i.e. dialog windows) may be used in the basic scripts! These functions stop the cyclical execution of all the functions of the process model!*

Access to the process variables in the script:

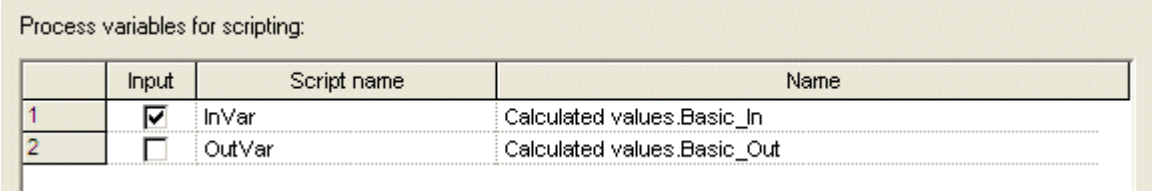
For the access to the process varibles in the script specify the script name in the variable table. The name can be used directly for reading access in the script. For writing access to the output variables the value has to be assigned to the 'value' adjunct of the variable.

Example:

In the process model two calculated values are applied: Basic_IN and Basic_OUT.



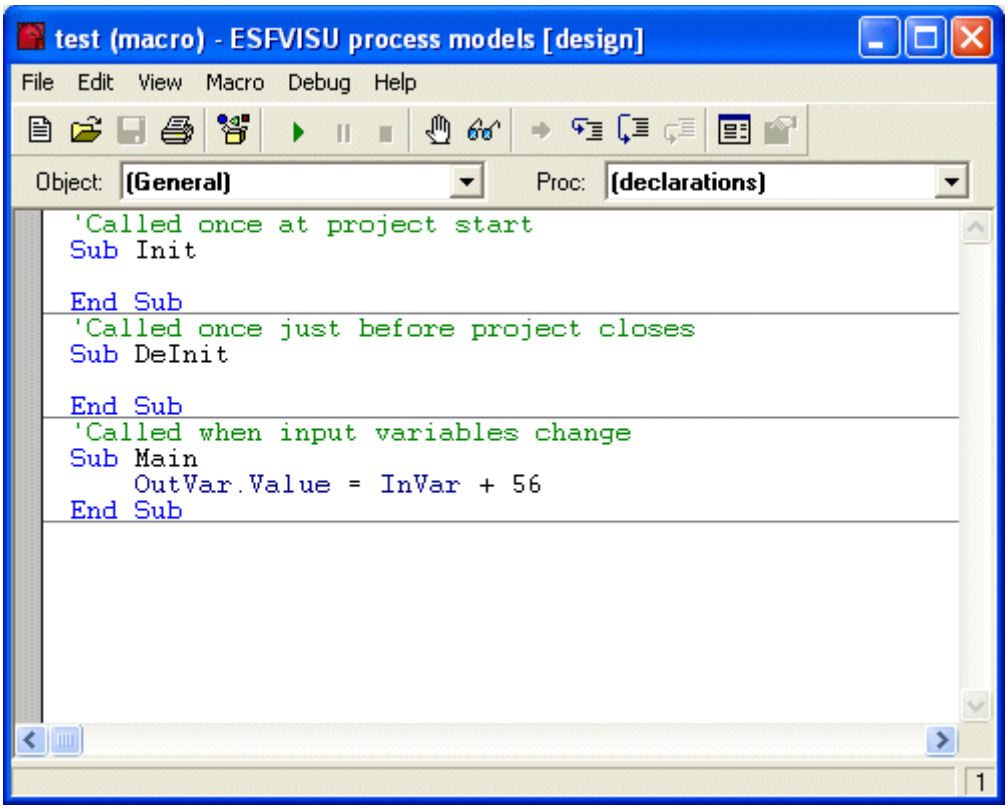
Both the variables are dragged to the variable list of a basic script. The 'Basic_IN' variable has been configuered as input variable, the 'Basic_Out' variable as output variable. The following script names are chosen:



InVar for 'Calculated value.Basic_IN'
OutVar for 'Calculated value.Basic_OUT'

Both the process variables are now available in the basic script under the terms 'InVar' and 'OutVar' (without apostrophe).

The following example function is very simple: OutVar is calculated as the addition of InVar with a fixed value.



Attention: *The value of the output variable must be changed by using the 'value' attribute.*

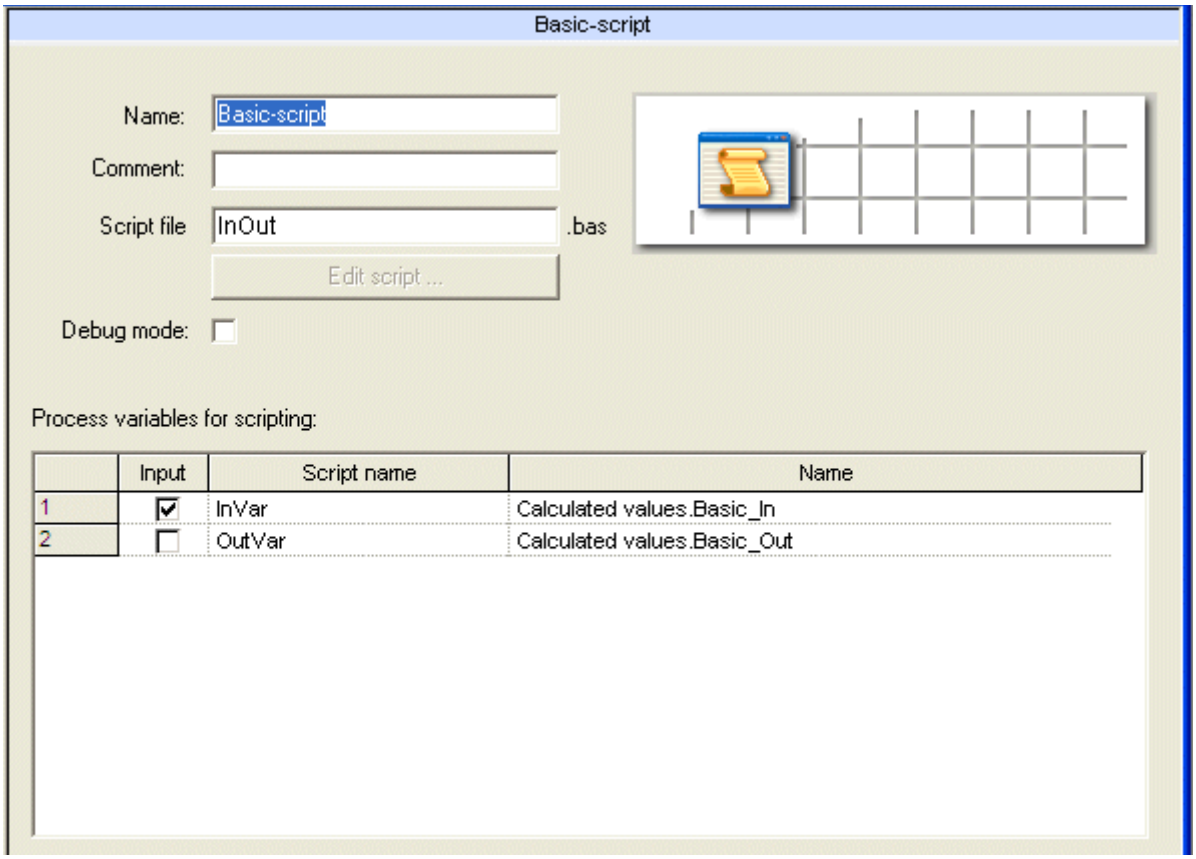
The script function 'main' will be evaluated after each changing of the input variable 'InVar' (which is the script name for 'Calculated values.Basic_IN'). After the script evaluation the value of the output variable 'OutVar' (which is the script name for 'Calculated values.Basic_Out') is the sum of 'InVar' and 56.

Basic language description:

The script language is a Basic programming environment integrated in the visualization. It is a Basic dialet, based on SaxBasic, and is widely compatible to Microsoft Visual Basic for Applications (VBA).
The language description is part of the documentation and can be found in the chapter 'Documents' in the visualizations control panel application.

Configuration:

To use a process variable in the script, drag the variable from the process variable's catalog in the table 'Process variables for scripting'. Then enter the script name and set the input/output flag for the variable. After that the variable can be used in the Basic script with its script name.
To delete the variable, select the line in the process variable table and press the 'Del' key.



Function settings:

Name: Name of the script.

Comment: Short description of the script functionality.

Skript file: Name of the script file. The system will create automatically a file with the extension '.bas'.

Edit script...: Opens the script editor.

Debug mode: If checked, the script will run in debug mode.

Process variables for scripting:
List of process variables, which can be used in the script. Each variable must be assigned a script name to access the variable in the script. The checkbox 'Input' specifies whether the variable is used as input (checked) or as output variable (unchecked).

Important notice:

Script programming requires care and attention. A script provides features, which can impair functioning of the ESFVISU visualization package, if improperly used.
Please read this description carefully and perform careful testing.
We can't guarantee the completeness and accuracy of this description and cannot take any responsibility for the result of scripts. Script programming is always at the users own risk.

4.11.4 Mathematics

Function description:

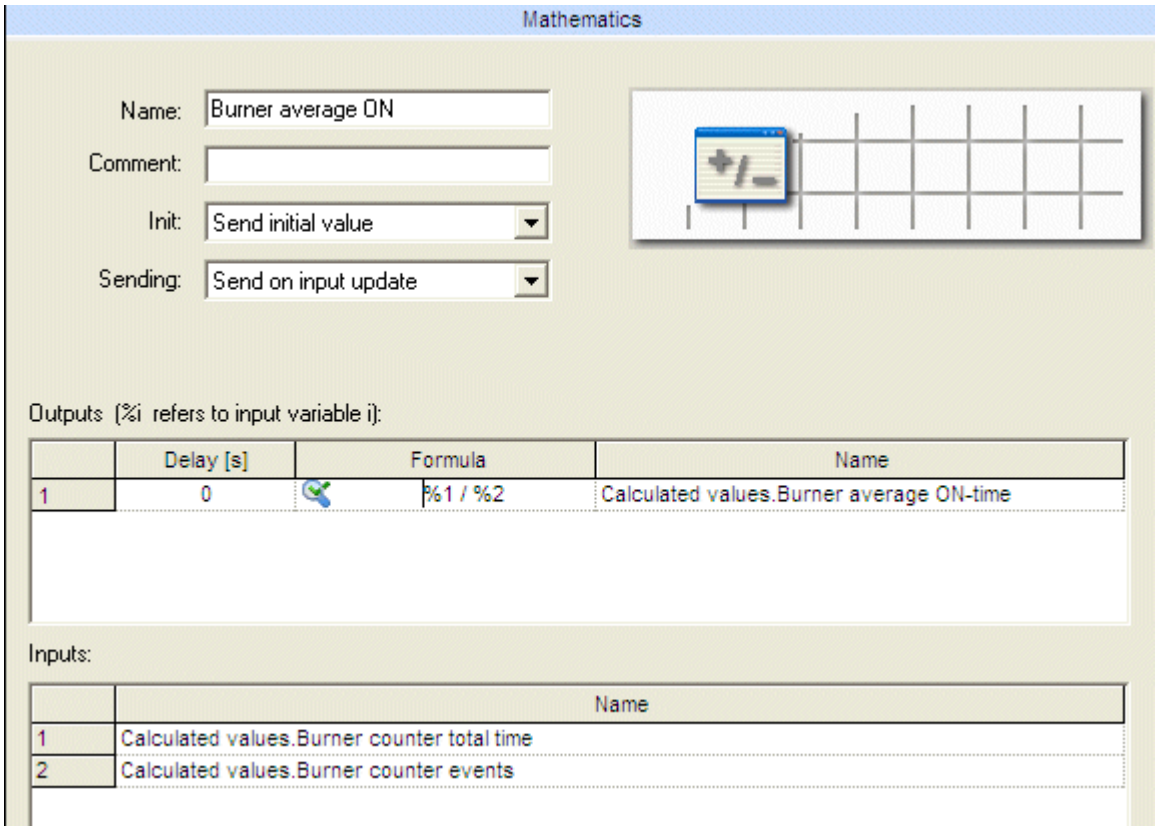
Matematical functions are used to calculate the value of an output variable based on a formula and one or more input variables.

Whether the output variable's value will be sent to the process interface (i.e. the EIB group address will be sent) depends on the configuration.

Configuration:

From the catalog window, category process data, drag the process variable to the grid of input variables respectively output variables.

To delete a variable from the grid first mark the row with left click on the first column in the row. Then press the keyboard's **Delete** key.



Function settings:

Name: The name of the function

Comment: A short description of the function

Init: Allow or disallow sending the function's result when the function is evaluated the first time, i.e. after the process model has been started.

Sending: Decide when to send the function's result.

On changing output: The function's result will only be sent when the function's result has been changed.

On input update: The function's result will always be sent when an input has been updated, regardless of the input's value.

Outputs: Process variables, which will hold the function's result.

Inputs: Process variables as the inputs of the function.

Settings for each output:

Delay: Define a delay in seconds for the evaluation of changed input values. Changes will be ignored, if they are not stable for at least the delay period.

Formula: Enter a formula to calculate the output's value. With "%1", "%2" .. (without quotes) refer to the first, second, .. input variable.

4.11.5 Sequence

Function description:

The sequence function is a time schedule of several commands to either set or query a process variable's value.
After each command the function will delay execution until a delay period has passed.

The delay period will be specified in seconds. There are two special cases for the delay period:

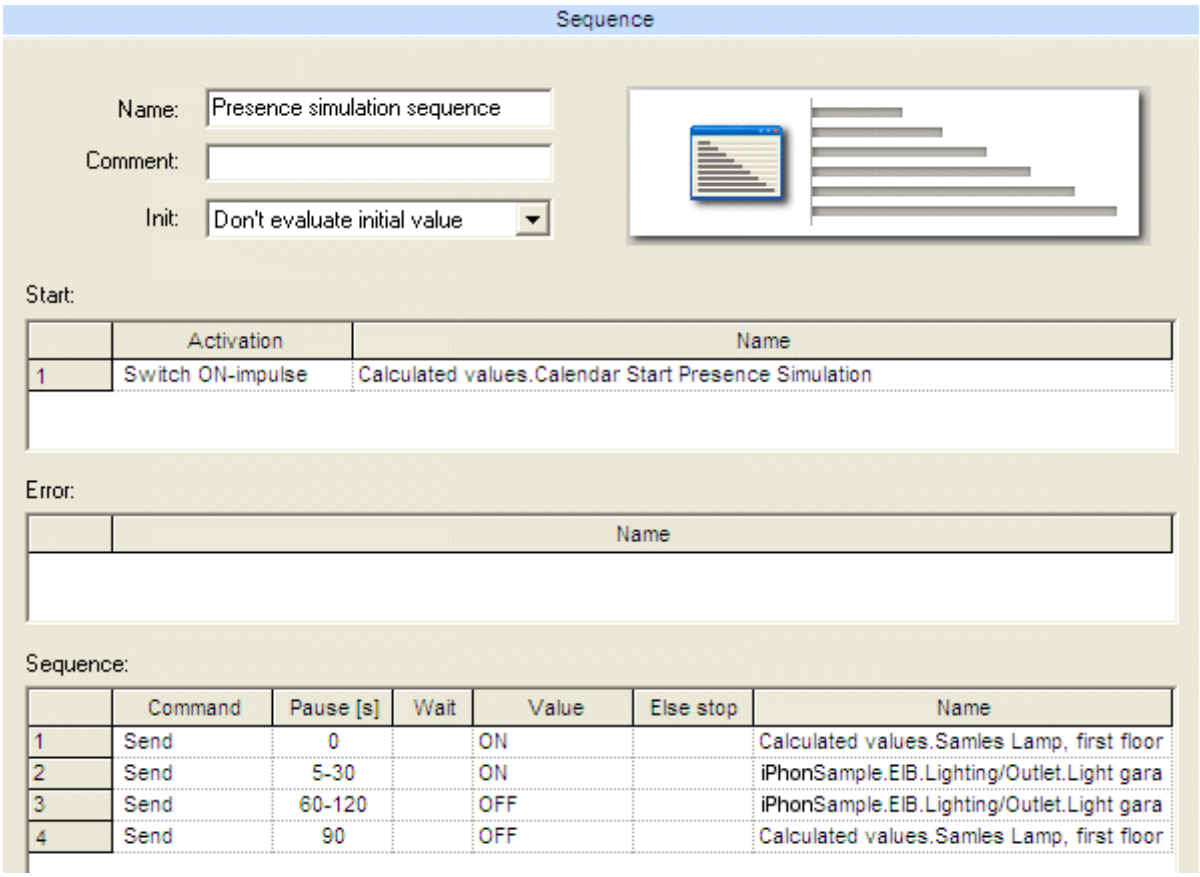
- 0 seconds: No delay
- An interval, i.e. 10 - 30: The delay will be a random value in this interval.

As an extension to simple sequences of read or write commands, the sequence may check the state of a variable. If the condition on the variable, given as a formula, has not been reached within a certain time period, or not been reached after a certain time period, the sequence may be stopped. If stopped, an error variable may be set accordingly.

A sequence is controlled by a binary process variable. It may be started when the variable is changing (activation by pulse) or may repeatedly run as long as the binary variable has a certain value.

Configuration:

From the catalog window, category process data, drag the process variable to the grid of input variables respectively output variables.
To delete a variable from the grid first mark the row with left click on the first column in the row. Then press the keyboard's **Delete** key.



Function settings:

Name: The name of the function

Comment: A short description of the function

Init: Allow or disallow sending the function's result when the function is evaluated the first time, i.e. after the process model has been started.

Start: Binary process variable controlling the sequence.

*In column **Activation** select the activation method for the sequence:*

- **"pulse off":** The sequence will be started when the value of the process variable changes from "on" (value 1) to "off" (value 0).
- **"pulse on":** The sequence will be started when the value of the process variable changes from "off" (value 0) to "on" (value 1).
- **"state off ":** The sequence will be repeatedly run while the process variable changes is "off" (value 0).
- **"state on ":** The sequence will be repeatedly run while the process variable changes is "on" (value 1).

Error: Optionally a binary process variable may be dragged to this field. If the sequence has been stopped because checking was not successful within a specified time period, the variable will be set to 1, otherwise 0.

Sequence: Commands on one or more process variables.

For each row in the sequence:

- **Command:** May be "send" or "check". Sending may be either sending a query telegram or a write telegram. When sending a write telegram, the column **Value** contains the value to be written. In case of checking, the column **Value** contains a formula. Checking is successful if the formula returns true.

- **Delay:** When sending, the command will be delayed for the specified number of seconds. Instead of the number of seconds an interval of seconds, i.e. 10 - 20 can be specified. In this case the sequence will use a random number of seconds in this interval. When delay is 0, the sequence will immediately execute the command. Checking expects that the formula in the column **Value** will return true within the delay time interval.

- **Wait:** This column applies to checking only. If "waiting" is turned on, the sequence will wait until the delay time has passed even when checking has been successful already before. Otherwise the sequence will continue as soon as the formula in the column **Value** returns true even if the time period has not passed completely.

- **Value:** When sending a write telegram this is the value to write.
  When checking, this field comprises a formula. Refer to the process variable in the row with "%1", without quotes. In case of a formula you may double- click the formula symbol to open a dialog to check the formula.

- **Else Stop:** This column applies to checking only. If enabled, the sequence will be stopped if checking has not been successful within the time period. The error variable – if specified – will be set to 1.
  Note: *the stop variable could be used to indicate a problem, i.e. could be an alarm variable.*

Notes:

- *If a check command has not been successful, a sequence started by a pulse will just terminate. A sequence controlled by a state will automatically restart with the first command. However, you can check the error variable at the beginning of the sequence to avoid execution of the following commands.*

- *Checking of variables can be useful to disable/enable a sequence. For example a variable could be checked, which is set by the calendar application.*

4.11.6 Status object

Function description:

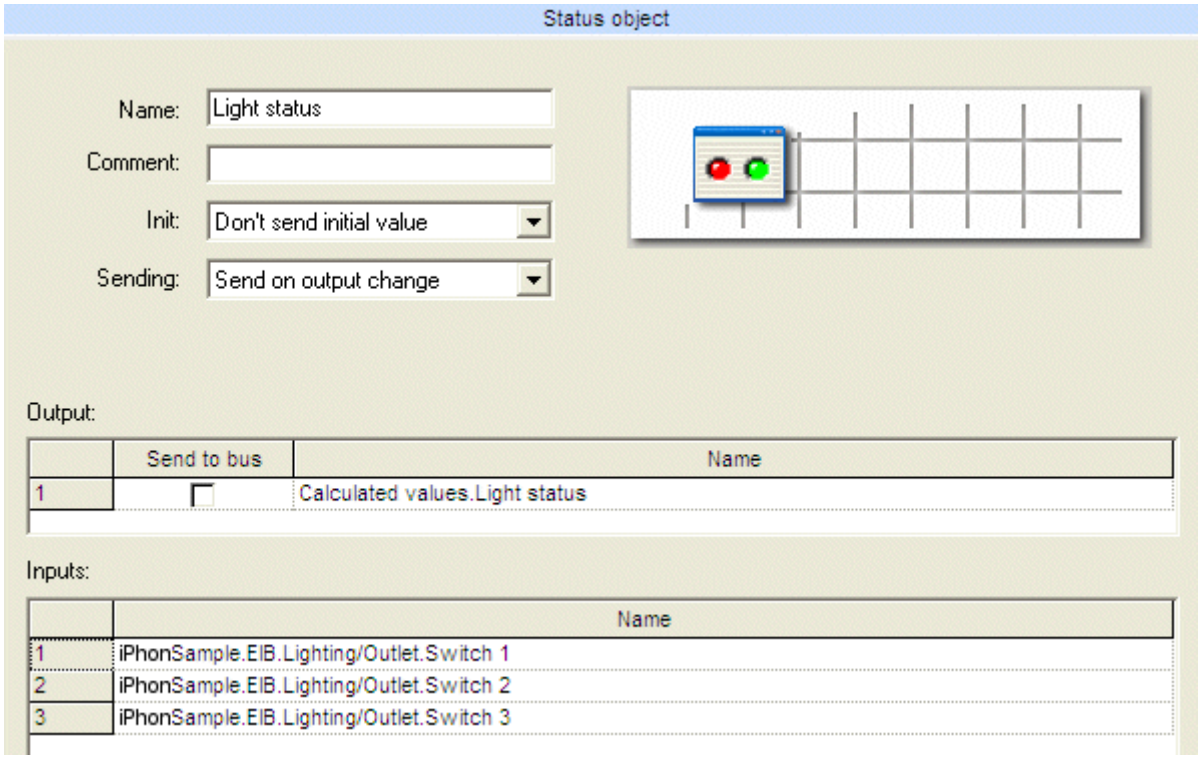
The status object updates the output process variable with one or more input process variables. The value of the output variable is always the value of the input variable, which has been changed at last.

Whether the output variable's value will be sent to the process interface (i.e. the EIB group address will be sent) depends on the configuration.

Configuration:

From the catalog window, category process data, drag the process variable to the grid of input variables respectively output variables.

To delete a variable from the grid first mark the row with left click on the first column in the row. Then press the keyboard's **Delete** key.



Function settings:

Name: The name of the function

Comment: A short description of the function

Init: Allow or disallow sending the function's result when the function is evaluated the first time, i.e. after the process model has been started.

Sending: Decide when to send the function's result.

On changing output: The function's result will only be sent when the function's result has been changed.

On input update: The function's result will always be sent when an input has been updated, regardless of the input's value.

Output: Process variable which will hold the function's result.

Inputs: Process variables as the inputs of the function.

Output settings:

Sending: Decide whether the output value will be sent to the process interface.

4.11.7 Scene

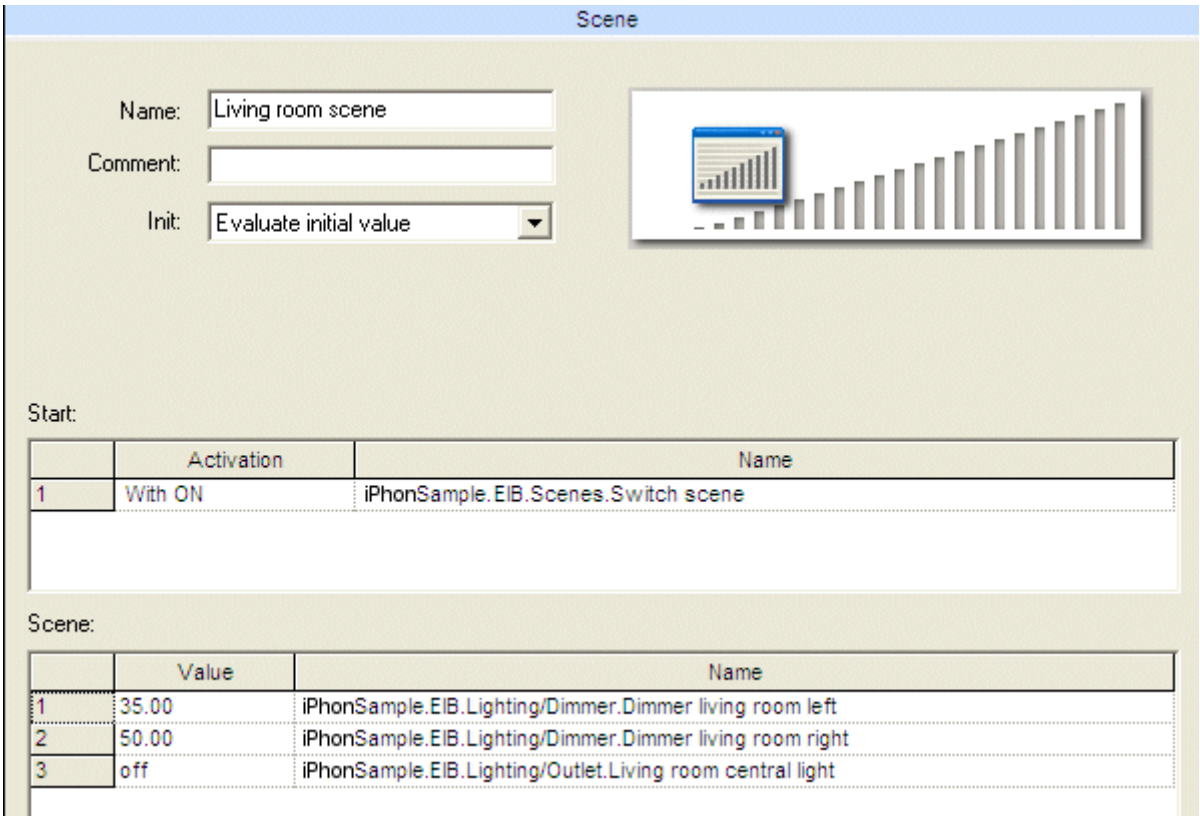
Function description:

The scene function is used to send write or read commands to several process variables at the same time (in practise one immediately after the other). It is controlled by a binary process variable.

Configuration:

From the catalog window, category process data, drag the process variable to the grid of input variables respectively output variables.

To delete a variable from the grid first mark the row with left click on the first column in the row. Then press the keyboard's **Delete** key.



Function settings:

Name: The name of the function

Comment: A short description of the function

Init: Allow or disallow sending the function's result when the function is evaluated the first time, i.e. after the process model has been started.

Start: Binary process variable controlling the sequence.

In column **Activation** select the activation method for the scene function:

- **"state off":** The scene will be started when the value of the variable changes from "on" (value 1) to "off" (value 0).
- **"state on":** The scene will be started when the value of the variable changes from "off" (value 0) to "on" (value 1).
- **"state on or off":** The scene will be started when the value of the variable changes.

Scene: Commands on one or more process variables.

For each row in the scene:

Value: Either the value to be written or a query command.

Binary process variable:	Select command from the combo box.
Analog process variable:	Directly enter the value or enter "read" (abbreviation "re"). When "read" has been entered, the scene function will send a read telegram instead of a write telegram.

4.11.8 Scene with memory

Function description:

Additional to the functionality of 'Scene', the 'Scene with memory' allows the user to save the **current** values of all contained process variables in a scene memory. The saved values will be sent to the process, if the user calls the scene. The writing of current values into the scene memory can be controlled by one or more process variables. The table 'Save' contains these control variables.

With this function, it is possible to predefine a scene, which can be dynamically changed by the user:

- Create the 'scene with memory' function in the process model.
- As in the standard scene, specify the variables for output and scene call.
- Additionally, specify one or more variables to control the saving of the current state of the output variables in the scene memory (table 'Save'). In this table, you can insert a calculated process variable, which is visible in the visualization only, or you can insert a process variable, which is connected with a 'real' switch.

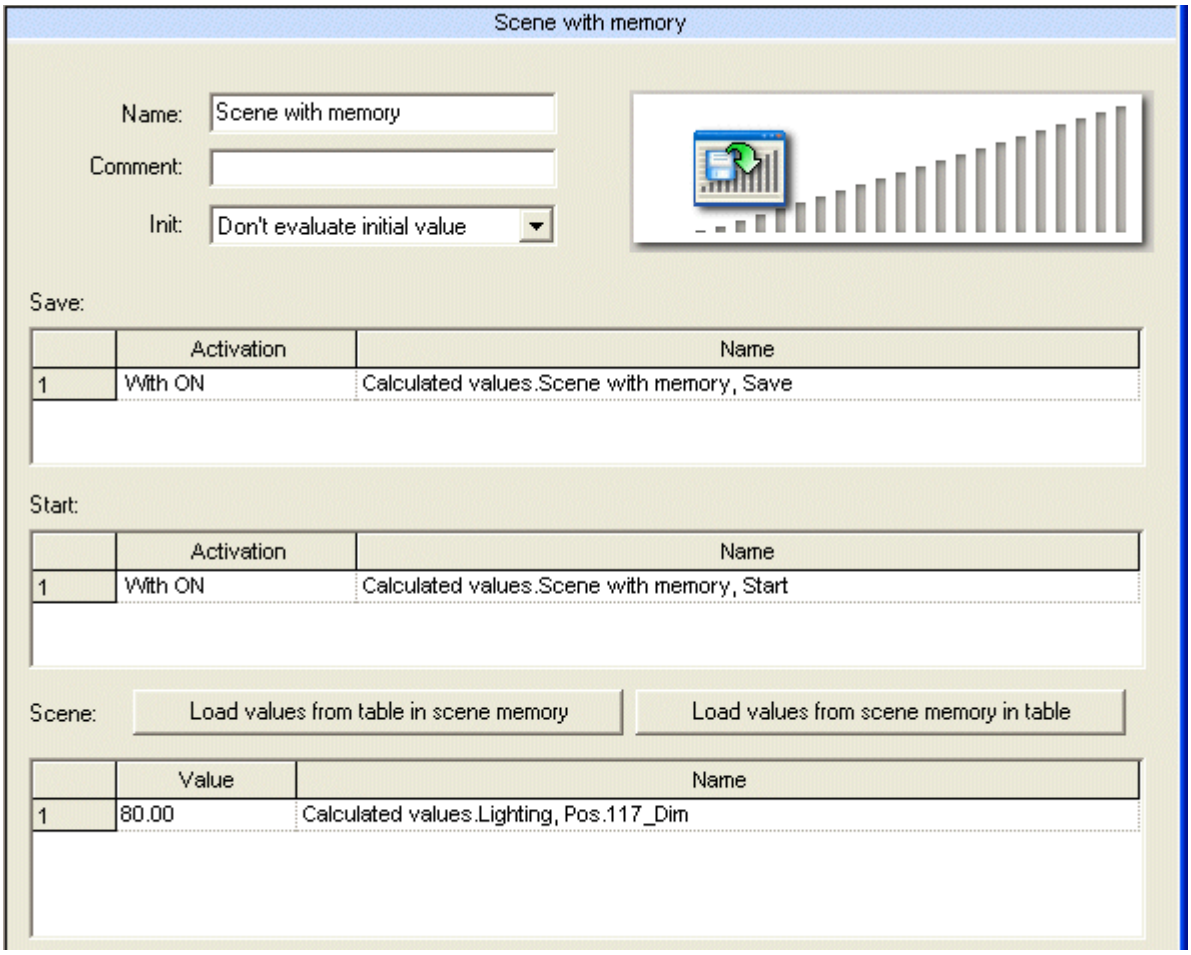
- The user may set the desired state of the scene's output variables in his installation (e.g. lights on/off, dimmer values, ...) and press the switch, which is connected to one of the saving control variables. The corresponding control variable will change its state and thus start the transfer of the current values into the scene memory.
- By the next call of the scene, the new values will be read from the scene memory and sent to the process.

The content of the scene memory can be loaded into the table 'Scene' by clicking the button 'Load values from scene memory in table'. In the table the values ca be changed and then saved by clicking the button 'Load values from table in scene memory'.

Configuration:

From the catalog window, category process data, drag the process variable to the grid of input variables respectively output variables.

To delete a variable from the grid first mark the row with left click on the first column in the row. Then press the keyboard's **Delete** key.



Function settings:

Name: Function name.

Comment: Short description of the functionality.

Init: Allow or disallow sending the function's result when the function is evaluated the first time, i.e. after the process model has been started.

Save: Table with one or more binary process variables to control the transfer of current values into the scene memory.

Start: Table with one or more binary process variables to call the scene.

In column **Activation** select the activation method for the scene function:

- **"state off":** The scene will be started when the value of the variable changes from "on" (value 1) to "off" (value 0).
- **"state on":** The scene will be started when the value of the variable changes from "off" (value 0) to "on" (value 1).
- **"state on or off":** The scene will be started when the value of the variable changes.

Scene: All process variable of the scene and the corresponding output values.

Load values from table in scene memory: Writes the output values from the table in the scene memory.

Load values from scene memory in table: Loads the output values from the scene memory in the table.

For each row in the scene:

Value: Either the value to be written or a query command.

Binary process variable: Select command from the combo box.
Analog process variable: Directly enter the value or enter "read" (abbreviation "re"). When "read" has been entered, the scene function will send a read telegram instead of a write telegram.

4.11.9 Forwarding

Function description:

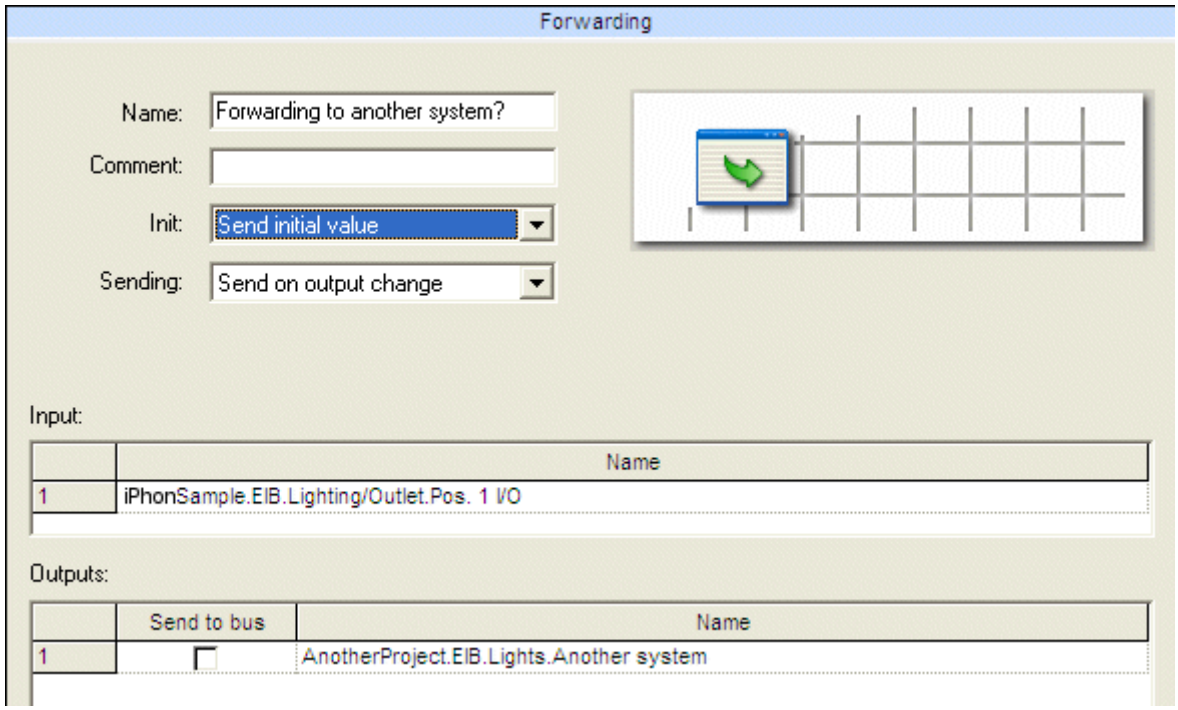
The function forwarding connects one input process variable to one or more output variables. The value of the input variable will be just copied to the output variables. Though no computation is implied, this function can be very useful to implement gateways from one technical subsystem (for instance connected with OPC) to another (for instance connected with EIB).

Whether the value of an output variable will be sent to the process interface (i.e. the EIB group address will be sent) depends on the configuration.

Configuration:

From the catalog window, category process data, drag the process variable to the grid of input variables respectively output variables.

To delete a variable from the grid first mark the row with left click on the first column in the row. Then press the keyboard's **Delete** key.



Function settings:

Name: The name of the function

Comment: A short description of the function

Init: Allow or disallow sending the function's result when the function is evaluated the first time, i.e. after the process model has been started.

Input: Process variable

Sending: Decide when to send the function's result.

On changing output: The function's result will only be sent when the function's result has been changed.

On input update: The function's result will always be sent when an input has been updated, regardless of the input's value.

Outputs: Process variables, which will hold the function's result.

Settings for each output:

Sending: Decide whether the output value will be sent to the process interface.

4.11.10 If then

Function description:

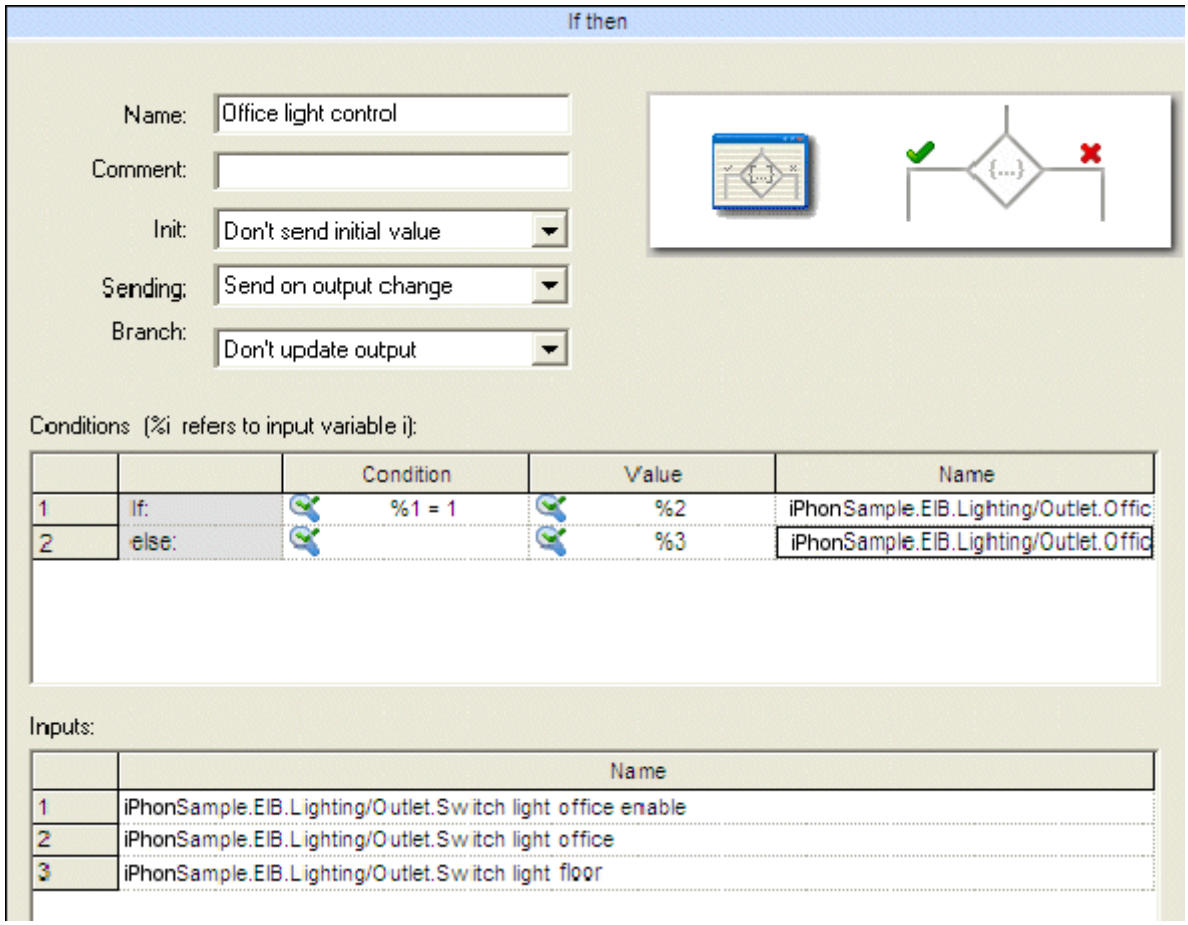
The if- then function is used to calculate the values of process variables depending on other process variables and conditions.

For example it can be used to configure a priority control: The value of a switch will only be sent to an actor, when the control of the actor by the switch has been allowed. The allowance may depend on another binary process variable.

Configuration:

From the catalog window, category process data, drag the process variable to the grid of input variables or the condition's grid. If dragged to the inputs grid, the process variable can be used in columns **Condition** and **Value**. If dragged to the conditions grid, a new condition row will be created and the process variable is used as an output variable.

To delete a condition from the grid first mark the row with left click on the first column in the row. Then press the keyboard's **Delete** key.



Function settings:

Name: The name of the function

Comment: A short description of the function

Init: Allow or disallow sending the function's result when the function is evaluated the first time, i.e. after the process model has been started.

Sending: Decide when to send the function's result.

On changing output: The function's result will only be sent when the function's result has been changed.

On input update: The function's result will always be sent when an input has been updated, regardless of the input's value.

Branch:

For an output variable there can be two reasons to calculate a new value:

- Due to the evaluation of conditions a different if- else branch (row) will be applied.
- Still in the same if- else branch the value changes.

Active branch: Always update output variables from the **Value** – column of the applicable if- else branch (row).

Passive branch: If for an output variable a new if- else branch will be applied (a different condition becomes valid), don't update the output variable's value. Next time the value of the row is recalculated due to a change of input variables, the output variable will be updated.

This option is important, if conditions are used to enable / disable a certain behavior but enabling/disabling is not supposed to send a value.

Conditions:

The condition's grid comprises of a sequence of if- else branches. When evaluating the conditions, the function will work down the rows, starting with the first row, and evaluate the **Condition** column until it finds a row with a valid condition. Then it will evaluate the formula in the column **Value**, assigns the value to the output variable in the same row, then stops.

For conditions and values left click on the formula symbol will open a dialog to check formulas. Refer to input variables with "%1", "%2",.. (without quotes) to refer to the input variable at the first, second, .. row in the input variable's grid.

The editor will check for erroneous entries and mark them red.

Inputs : The process variables used in **Condition** and **Value** formulas.

4.11.11 Counter

Function description:

The counter function can be used to implement counters for pulse or operating hours, Based on "on" and "off" messages from binary counters.

Counter for pulses:
The number of messages is to be counted.

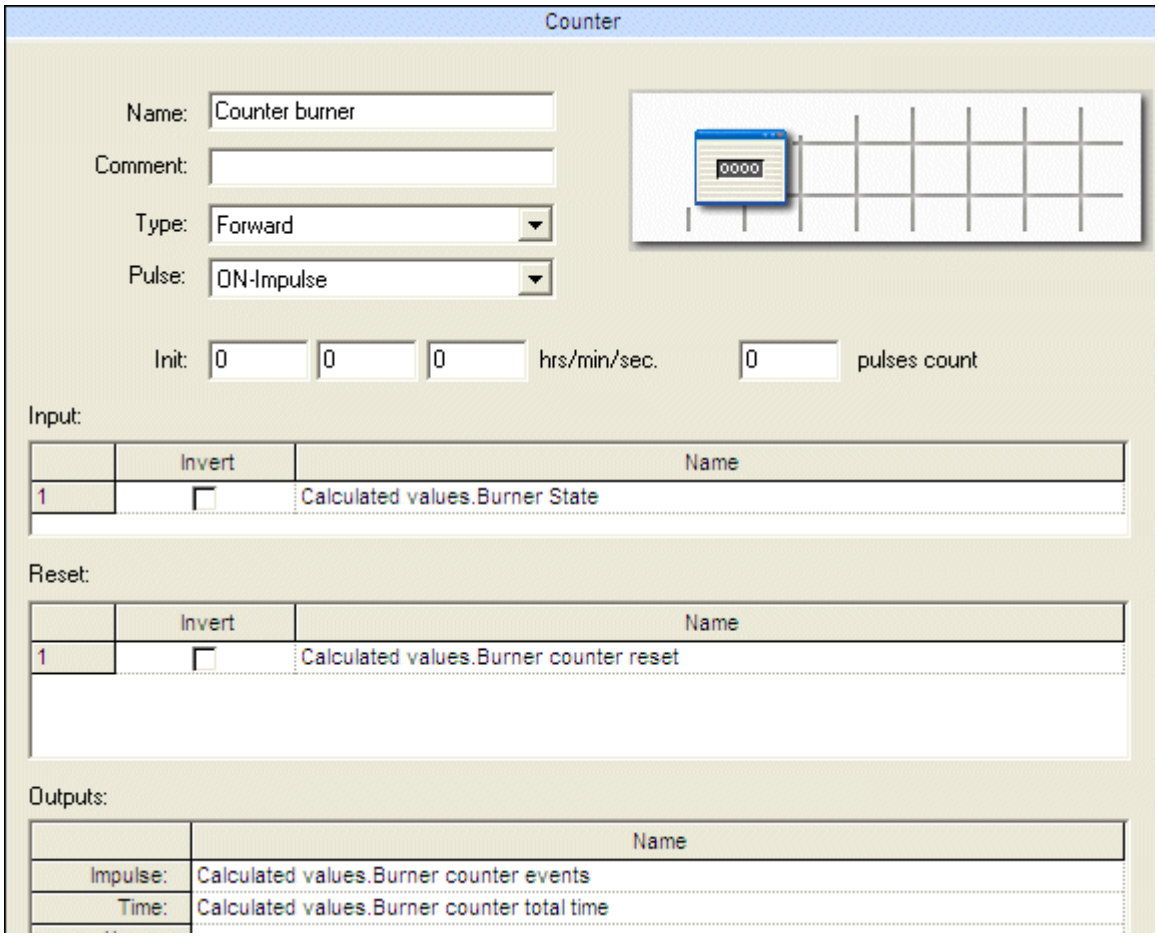
Counter for operating hours:
The times between "on" and "off" messages will be counted. The hours, minutes and seconds can be assigned to separate variables.

Configuration:

From the catalog window, category process data, drag the process variable to the grid of input variables respectively output variables.

To delete a variable from the grid first mark the row with left click on the first column in the row. Then press the keyboard's **Delete** key.

For time registration, the row **Time** must be connected to an analog process variable. Then additional rows will allow to connect separate variables for the hours, minutes and seconds.



Function settings:

Name: The name of the function

Comment: A short description of the function

Type: Select whether the counter should decrement or increment.

Pulse: Select whether to evaluate only "on" messages, only "off" messages or both.

Init: Initialize the function with a starting time (hours/minutes/seconds) and a start value for the pulse counter.

Input: The binary input variable.

Reset: Optionally a reset variable can be assigned. When the value of the reset variable is changing from "off" (value 0) to "on" (value 1) the counter will be reset to the starting values. If the reset variable is to be inverted, reset will be performed when the value is changing from "on" to "off".

Outputs:

Process variables, which will hold the results.

Pulses:	The number of pulses counted.
Time:	The total time in seconds.
Hours:	The hours fraction of the counted time.
Minutes:	The minutes fraction of the total time.
Seconds:	The seconds fraction of the total time.

4.11.12 Timing relay

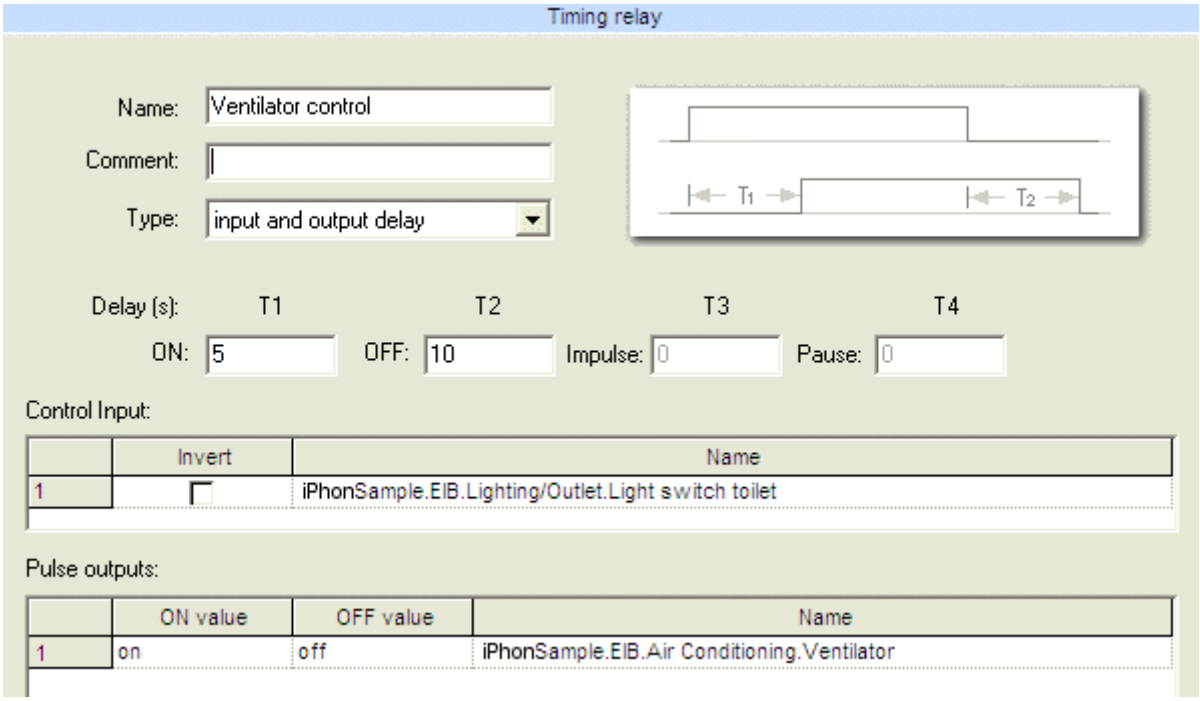
Function description:

The timing relay function can be used to cyclically send a telegram or to send an output pulse telegram with specified characteristics in return of controlling input pulse.

Configuration:

From the catalog window, category process data, drag the process variable to the grid of input variables respectively output variables.

To delete a variable from the grid first mark the row with left click on the first column in the row. Then press the keyboard's **Delete** key.



Function settings:

Name: The name of the function.

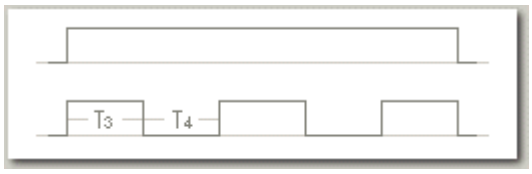
Comment: A short description of the function.

Type: Select the function type. This may be one of "pulse generator", "pulse shaper", "output delay", "input and output delay", "input wiper", "output wiper".

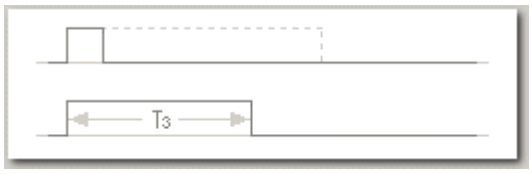
Delay: Specify timings T1 ("on"), T2 ("off"), T3 ("pulse") and T4 ("pause") in seconds

The following diagrams explain the behavior of different timing relay functions, dependent on timings.

pulse generator:

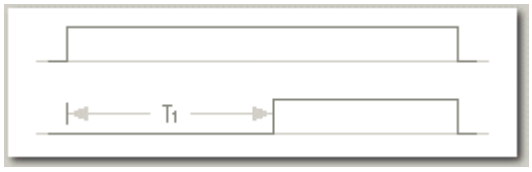


pulse shaper:



output delay:

input delay:



input and output delay:

output wiper:

input wiper:

Control input:

For function types "pulse shaper", "output delay", "input and output delay", "input wiper", "output wiper" a controlling input process variable is required. Changes on the controlling input variable will cause changes to the pulse output variables.

For function "pulse generator", the controlling input process variable is optional. If not defined, the pulse generator will be active all the time.

Pulse outputs:

All types of process variables can be used as outputs. Alternatively to sending write telegrams for "on" and "off" pulses, it is also possible to query values with read telegrams. For binary process variables the options are provided with a combo box. For querying analog process variables enter "read" or "re" as an abbreviation.

4.11.13 Automatic guard

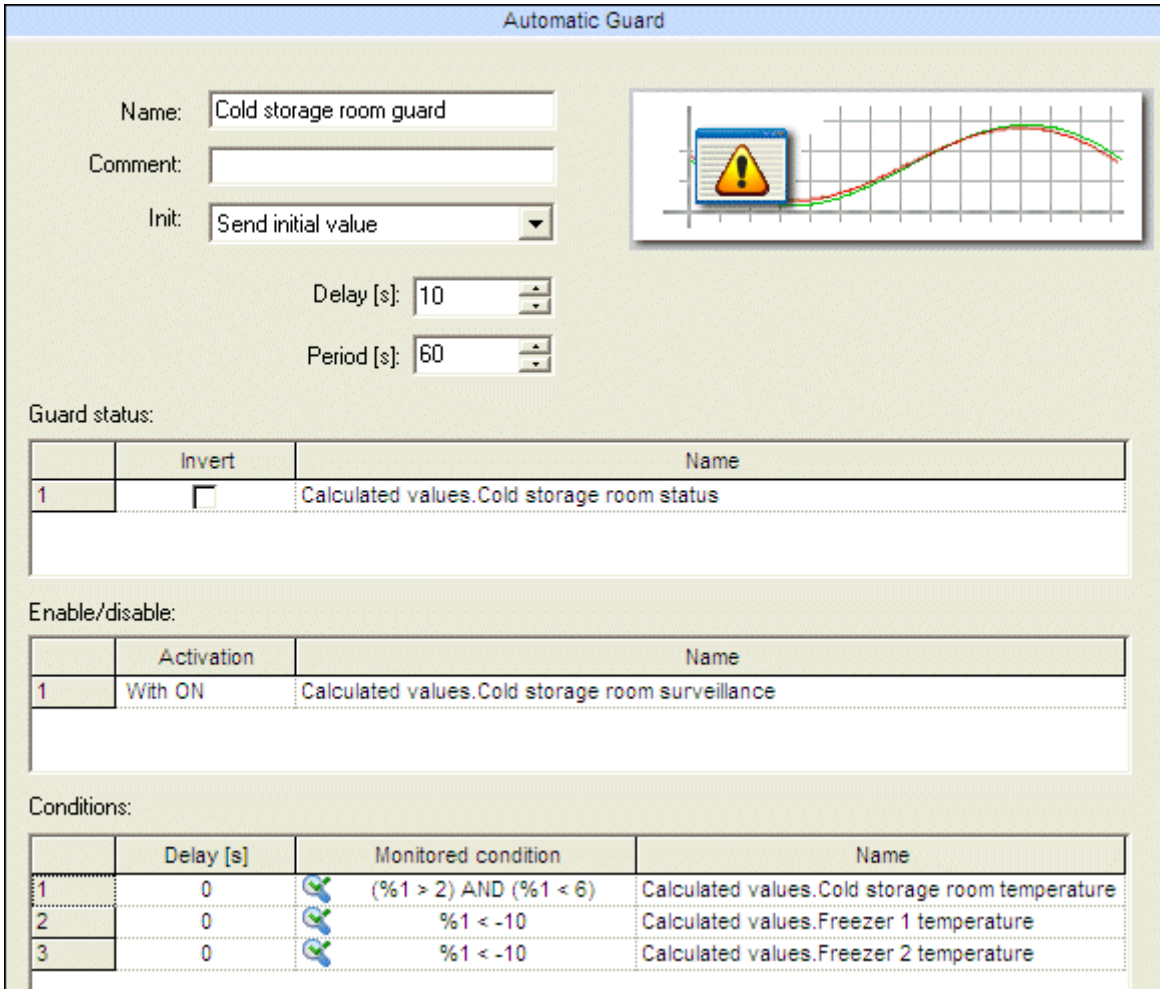
Function description:

The function automatic guard is used to keep a proper operating state under surveillance. The operating state is defined as a set of conditions on input variables, the conditions are defined as formulas.

Configuration:

From the catalog window, category process data, drag the process variable to the function's window.

To delete a variable from the grid first mark the row with left click on the first column in the row. Then press the keyboard's **Delete** key.



Function settings:

Name: The name of the function

Comment: A short description of the function

Init: Allow or disallow sending the function's result when the function is evaluated the first time, i.e. after the process model has been started.

Delay: Define a delay in seconds for the evaluation of changed input values. Changes will be ignored, if they are not stable for at least the delay period.

Period: Define after each time period in seconds the conditions will be evaluated.

Guard status: The binary process variable, which will receive the result of the check.
1: ALARM: At least one condition is not valid.
0: OK: All conditions are valid.

Enable/disable: Optionally defines a binary process variable, which will enable or disable the guard function. When "on" (value 1), the guard is active and the conditions will be evaluated, when "off" (value 0) the guard is deactivated and conditions will not be evaluated.

The process variable used for enabling/disabling the guard function may for instance be controlled by the calendar application or a switch.

Conditions:

Each row is used for one condition on one process variable. The Column **Delay** specifies how long the value of the input variable must be stable, until the formula in the column **Formula** will be evaluated. In the formula use "%1" (without quotes) to refer to the process variable in the row.

Examples:

Survey temperature of cooling room:	%1 < -5
Survey bus state:	%1 = 1
Survey room temperature:	(%1 >= 18) AND (%1 <= 21)

The state of the guard will be ALARM if at least one formula returns "false", will be OK if all formulas return "true".

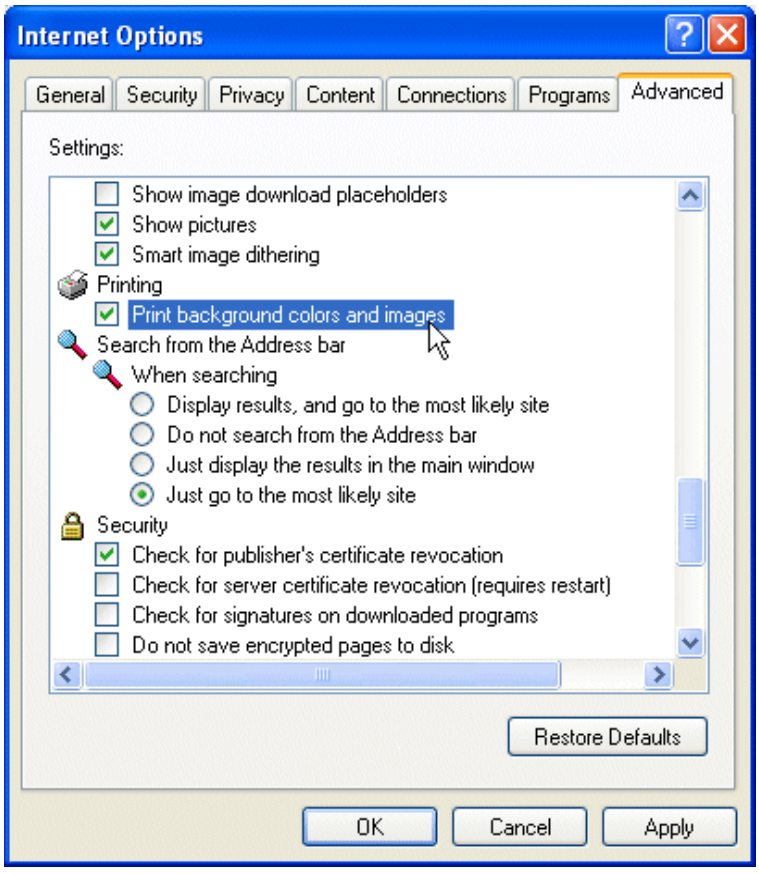
5 Reports

- Reports provide overviews of the project, including process variables and their properties (archives, arlarms, notifications,..).
- Reports may be printed (with print preview).

5.1 Create report

The Microsoft Internet Explorer is used to create and print reports.

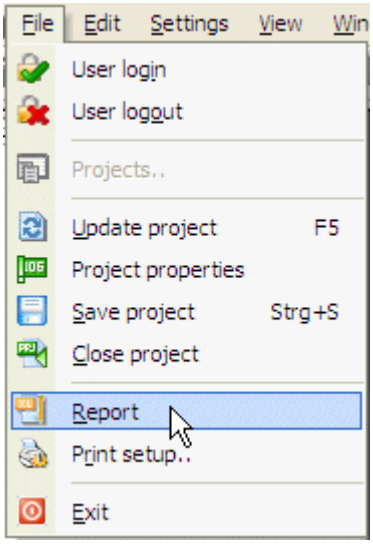
Internet options:



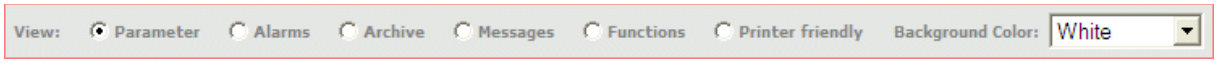
The reports background will be printed if the Internet Explorer has been configured accordingly:

- Select the menu option **Tools – Internet Options**.
- Select the tab **Advanced**.
- Under **Settings – Printing** check the option ***Print background colors and images***.

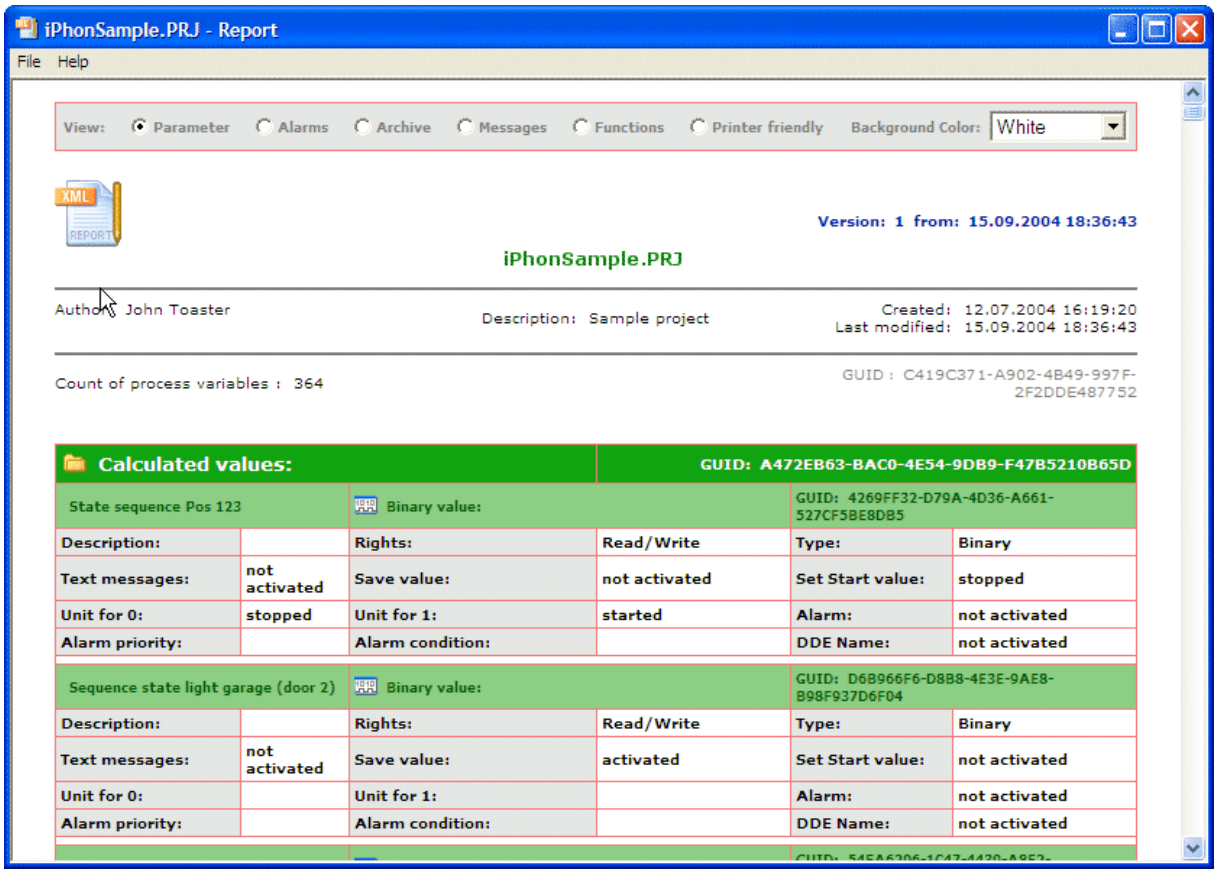
Use the menu option **File – Report** to create a report.



The menu at the top offers different options:



Variables: Shows all process variables.

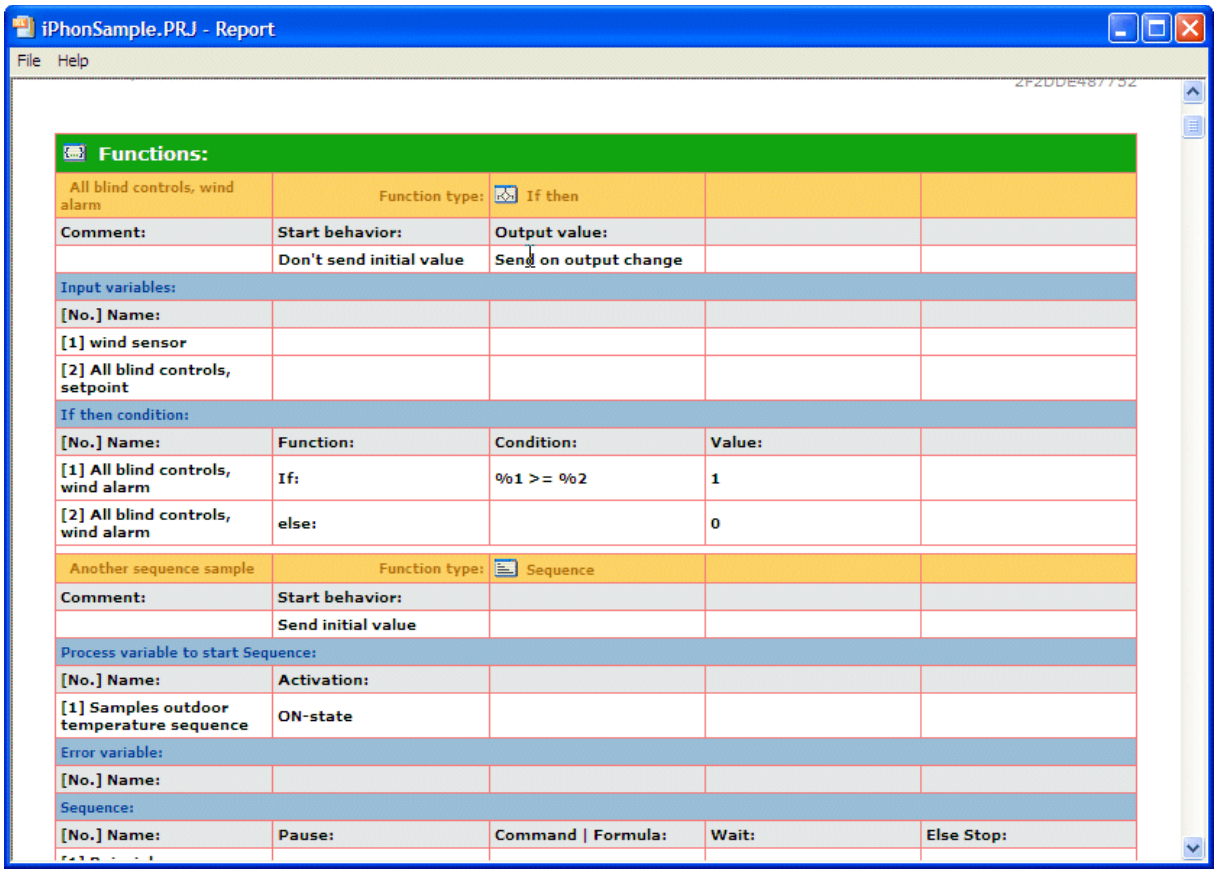


Alarms: Shows alarms and alarm conditions.

Archives: Shows archive definitions.

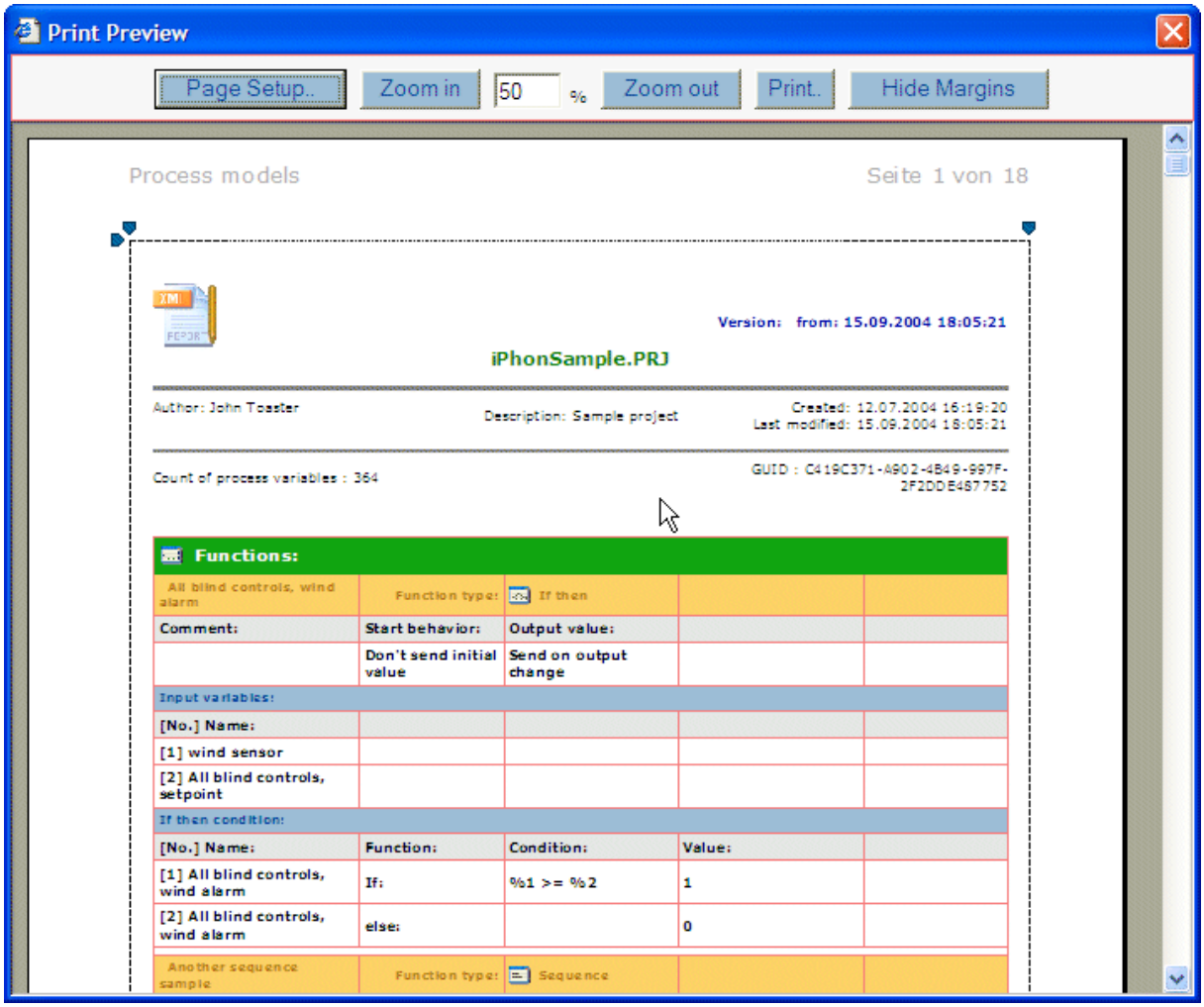
Messages: Shows configured notifications.

Functions: Shows configured controller functions.



Printer friendly: Black and white display of process variables.

5.2 Print preview



The print preview may be scaled.

Page Setup..	Opens the dialog "Page setup" to set the printer properties, page properties and the alignment.
Zoom in	Enlarges the print preview.
Zoom out	Reduces the print preview.
Hide Margins	Button to hide the margin markers.
Show Margins	Button to display the margin markers.

With these ◗,◗ markers the user has visual control over the margin settings.