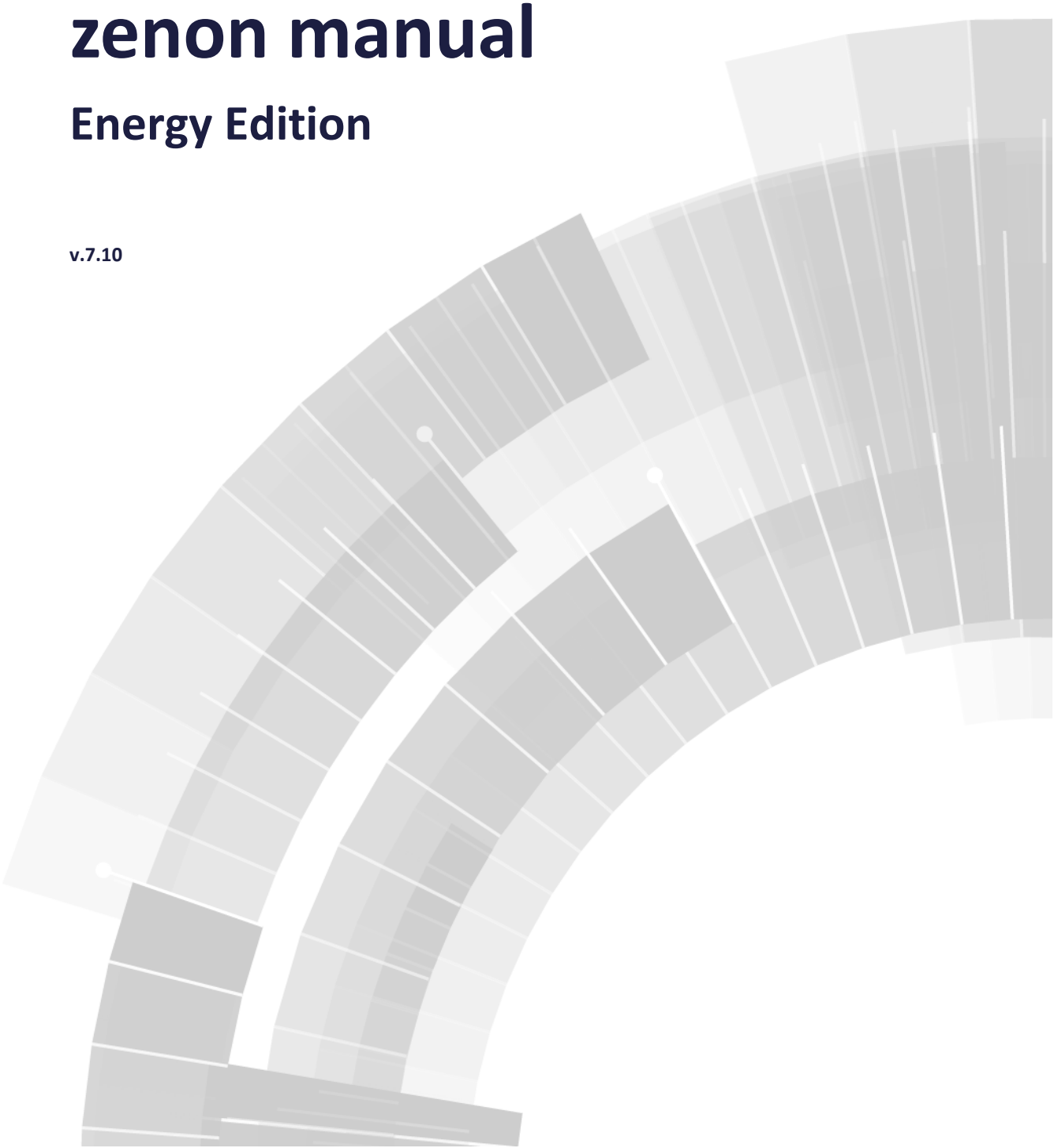
COPA-DATA
do it your way

zenon manual

Energy Edition

v.7.10



Contents

1. Welcome to COPA-DATA help

GENERAL HELP

If you cannot find any information you require in this help chapter or can think of anything that you would like added, please send an email to documentation@copadata.com (mailto:documentation@copadata.com).

PROJECT SUPPORT

You can receive support for any real project you may have from our Support Team, who you can contact via email at support@copadata.com (mailto:support@copadata.com).

LICENSES AND MODULES

If you find that you need other modules or licenses, our staff will be happy to help you. Email sales@copadata.com (mailto:sales@copadata.com).

2. Energy Edition

zenon Energy Edition is a package with special functionality for the energy sector and the procedural technology. The user benefits from easy-to-implement functions that allow for an individual adjustment of the application to the physical environment.

License information

Must be licensed in Editor and Runtime.

The following is also available for the Energy Edition:

- ALC (Automatic Line Coloring): Already included in the license for Energy Edition, provides basic properties for line coloring.
- Topological element transformer
- Topology package: Requires additional licensing on the server (not on the client) and expands ALC by:
  - Multiple supplies
  - Secured supply
  - Topological interlockings
  - Topological element disconnector
  - Error detection and ground fault search

2.1 Automatic Line Coloring (ALC) - Topology

The topological coloring of lines allows easy automatic dynamizing of tubes in technology (for media) as well as in the energy distribution (for electricity). So process controlled coloring of topological nets can easily be realized.

Because the tube structure is designed in the screen with all its technological elements (e.g. tanks and valves, or generators, switches and consumers), it is internally emulated as a model and the media flow is displayed in the Runtime.

In order to allow screen-overlapping models the entire design and configuration is always project-wide. You therefore have one entire topological model per project, which is used for the calculation of the tube statuses and ultimately for the coloring of the tubes.

The whole topology is created automatically from the graphic design. No other engineering actions are necessary.

> **Info**
>
> *The ALC algorithm only runs through once from a source starting from each switch.*

DETAIL SCREENS

To display individual screens, a partial area can be taken from the topological network and displayed individually by means of alias. A detail screen (on page 36) can be displayed with the data from different equipment parts, for instance outputs or partial networks.

LICENSING

Must be licensed for Editor and Runtime (single-user, server, standby). No need to be licensed for Runtime client. Licensing is carried out using the zenon Energy Edition.

- ALC: Included in the license for Energy Edition; provides basic properties for line coloring.
- Topology package: Requires additional licensing on the server (not on the client) and expands ALC by:
  - Multiple supplies
  - Secured supply
  - Topological interlockings
  - Transformer and separator topological elements

- Error detection (version 6.50 and above)

2.1.1 ALC elements

Automatic Line Coloring (ALC) makes it possible to color lines regardless of the process status. The combined element is used as the process element. Automatic line coloring allows easy automatic dynamizing of tubes in technology (for media) as well as in the topological networks (for electricity).

ENGINEERING

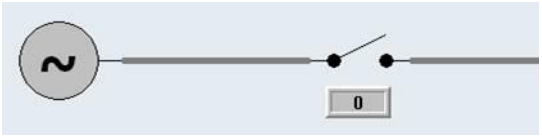
For the design two types of screen elements with different functions are distinguished. On the one hand these are procedural elements (on page 8) (source, switch/disconnector, drain, transformer or link) and on the other hand lines (on page 18).

In doing so, the technical elements have a function and a color (source and transformer). If the procedural elements are active, the connected lines take on the color of these elements at the source and transformer or they take on the color of the element's input line for the switch and the link. If the procedural elements are inactive, the color of the lines is taken from the definition in the editor.

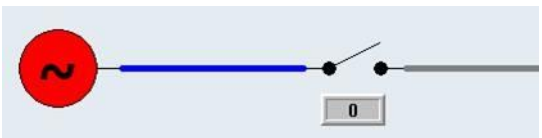
The different functions of the elements are assigned in the properties of the combined element.

EXAMPLE

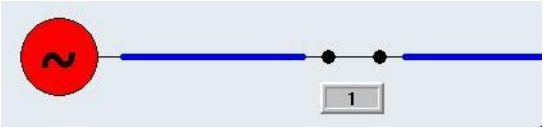
A source has a connected line. A switch is connected to the line. And a second line is connected there. If the source is active, the first line is colored with the color of the Automatic Line Coloring defined in the source up to the valve. The other line is not colored before the switch is closed.



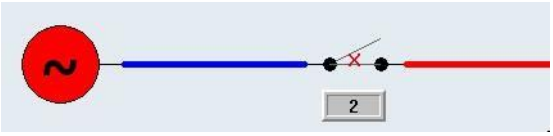
Source inactive



Source active



Switch closed



Undefined or invalid

> **Info**
>
> *If the procedural element status is `undefined` or `malfunction`, this is automatically detected. All connected lines and all further elements are displayed in the color of the predefined source `undefined`' for both states.*

NUMBER OF CLOSED SWITCHES IN A SERIES

For the correct functioning of the ALC algorithm, the number of connected switches in a series plays a role.

Recommendation: Arrange a maximum of 256 closed switches in a series between the source and the drain.

Procedural elements

Procedural elements are created in zenon with a combined Element. Their state determines the coloring of the connected line.

The following settings are available:

Property	Description
`Function type`	Defines the technological type of the Combined element.
▸ `Conclusion`	For bus bar ends. Blocks the error message "**`Line only connected on one side`**" when being compiled in the Editor.

Source	Passes on its color. If the source is active (value: 1), all connected lines that have Color from ALC option set in the element properties are allocated the color of the source. The color is defined in the project properties as the source color. (e.g. tanks or generators). A source is a single pole with a static source number assigned to it. The source is switchable over the state of its main variable. Generally, sources are considered as net-synchronous and detachable.
Generator	A generator generally behaves like a source, but it is considered as independent an not net-synchronous.
Switch	With this lines can be split. If the switch is closed/active (value: 1), then the connection between the two lines is closed and the line is colored up to the next switch with the defined source color. In this case a switch forwards the source color of the input line to the output line. If the status of the switch is malfunction, undefined or INVALID, the color of the line turns into the color undefined from the ALC configuration in the project properties. A switch thus delivers source number 0 (undefined) to its output (connection 2) instead of the incoming source number. Example: see **Switch example - colors from ALC** (on page 11) section.
Disconnector	A disconnector generally behaves like a switch. Nevertheless, a disconnector may not be connected in the topological model. A status (on, off, intermediate position, malfunction) is determined via its main variable.
Slider	A slider (a valve) acts in a similar manner to a switch, but it is used for water and gas lines. Value of the main variable: Switch OFF: Value 0 -> Slider closed-> No water flow Slider ON: Value 1 -> Slider closed completely-> No water flow Slider DIF: Value 2 -> Slider partially open-> Water flow Slider STO: Value 3 -> Slider malfunction
Drain	This defines the end of the line. The drain does not influence the coloring; it is only used so that the model can be displayed in full. If an external program (e.g. VBA) should access the model, then the drain probably is needed for further calculations, and so has to be inserted. In Energy projects, the drain is used for representing consumers. These customers are considered for the calculation of the ALC interlockings (command groups) '**Consumer is undersupplied**'.
Transformer	A transformer is a drain and a source at the same time. SO with a transformer the input color (input source) can be transformed to a new output color (transformer

	source color). The output connection is only active, if the transformer is switched active. But the output line does not get the color of the input line as with a switch, but the source color of the transformer. So a source has to be defined for each transformer. A transformer cannot be switched active or inactive, it always is active. `Transformer capable of reverse feed:` To have a transformer capable of reverse feed, you must select, for `Source for reverse feed`, a different source than `UNDEFINED [0]`. This means that the transformer behaves the same for both directions - from the input to the output (forward) and also from the output to the input (backward). The only difference is that the `Source for reverse feed` property and not the `Source` property is used for further distribution of the source number . **Note:** Defective network statuses or missing configurations, such as a feed from the input and output at the same time or a short circuit from input and output are not specially colored. This means that the transformer capable of taking a reverse feed behaves like two transformers switched to run antiparallel that are not capable of taking a reverse feed.
▸ `Link`	With a link a line can be continued on some other place. If a link is supplied by a line, all other links with the same link number also are supplied by this line. Here it does not matter, whether the links are in the same screen or on different screens in the project. So screen independent lines can be defined. It is possible to have more than two links with the same link number in one project. Links can be supplied by several lines at the same time or can themselves supply several lines. In principle there is no difference between inputs and outputs. The source information is passed on to all connected lines. **Attention:** Two link elements cannot be connected directly to one line. In between, there has to be at least one other procedural element (switch/disconnector or transformer). A link cannot be switched active or inactive, it always is active.
`Link number`	Only the link number is entered for a `link` function. All identical link numbers in a project correlate with each other. Detailed description in the function type Link. This property is only active, if the function type link has been selected.
`Source`	Here a source is assigned to an element. In this selection box all sources defined in the ALC configuration (in the project properties) are available. All source names are listed. This property is only active if the function type '`source`', '`transformer`' or '`generator`' has been selected.

A variable of the IEC type BOOL or integer has to be linked to the element as the main variable, so that the switch can get the status (open, closed, invalid). In the same way, the source gets its status (active/inactive) from the linked main variable.

For the function types source and transformer the defined source number is forwarded to the consumers (drains) over open/closed switches. The statuses and colors of all connected lines are calculated from the superposed sum of the supplying source numbers and procedural elements.

> **Info**
>
> *Only the first two bits are considered for the switching. The first bit stands for the actual switching. 0 equals off and 1 equals 1.*
>
> *The second bit is the error bit. There is no error only if it is 0.*

STATES

- A switch and a source are switched on if the value of the linked variable is `1`.
- A switch is invalid if the value of the linked variable is `>1` or has an `INVALID` status.
  An invalid switch provides the source number `0` (undefined) at its exit (connection 2) instead of the source number entering. In the direction towards the input the switch behaves as normal.

Note: if the (acknowledgment) variable has the status `INVALID`, the whole subsequent network is `INVALID`, because the status of the network is not known. The status `INVALID` is forwarded (routed) using subsequent closed switches.

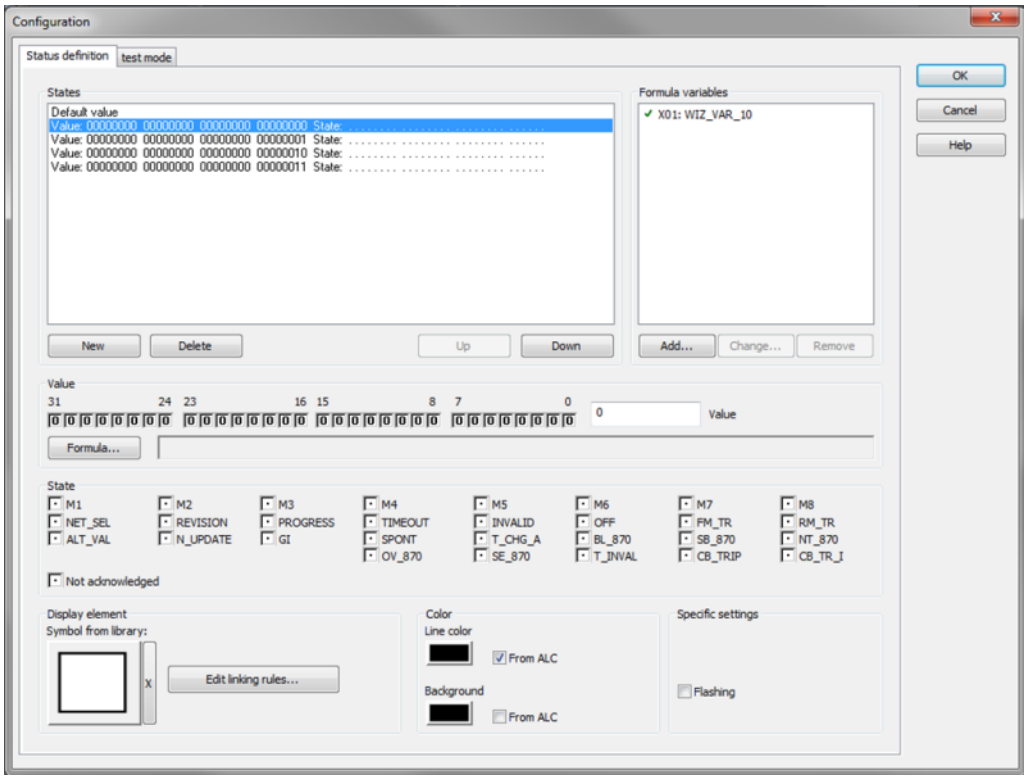
> **Attention**
>
> *If in the single status the color and the filling color from the ALC is activated, also the procedural elements are colored by the status of the connected lines in the Runtime.*

Switch example - colors from ALC

EXAMPLE 1

Combined element with value status `00` and line color from ALC:

1. Configuration in the Editor:
    - Combined element with value status `00`
    - Line color from ALC active



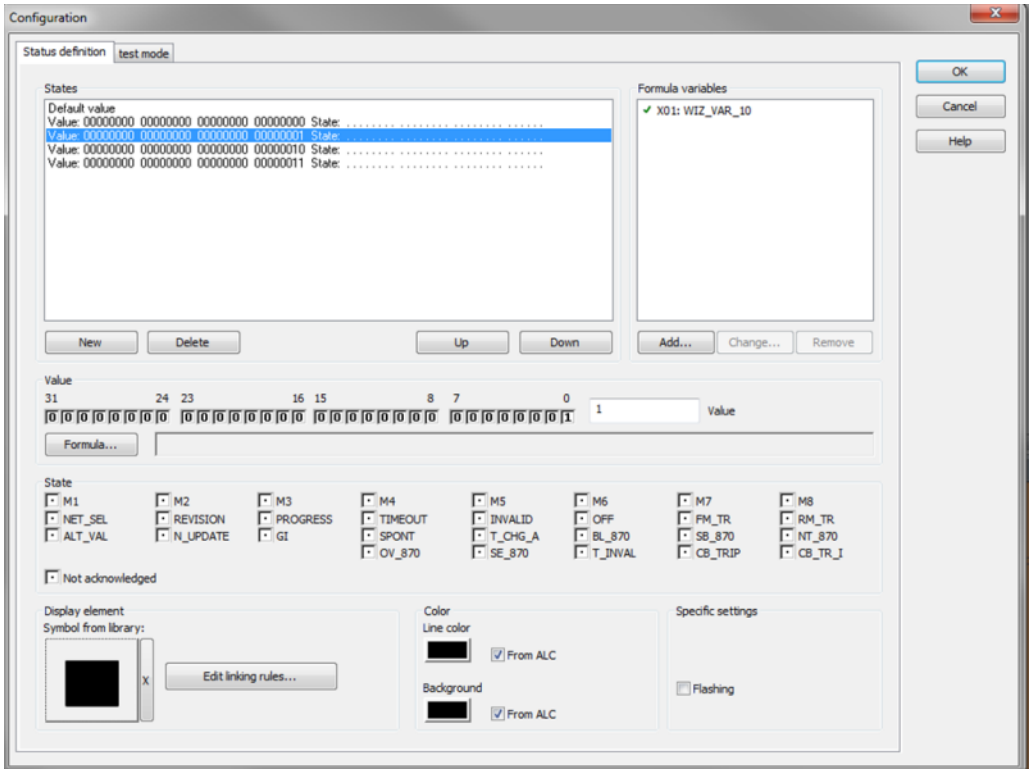
2. Results in the following in Runtime:
    - Source color: green
    - Color without voltage: white
    - Switch status: `off/open` (value `0`)

EXAMPLE 2

Combined element with value status `01` and colors from ALC:

1. Engineering in the Editor
    - Combined element with value status `01`
    - Line color from ALC active

- Fill color from ALC active



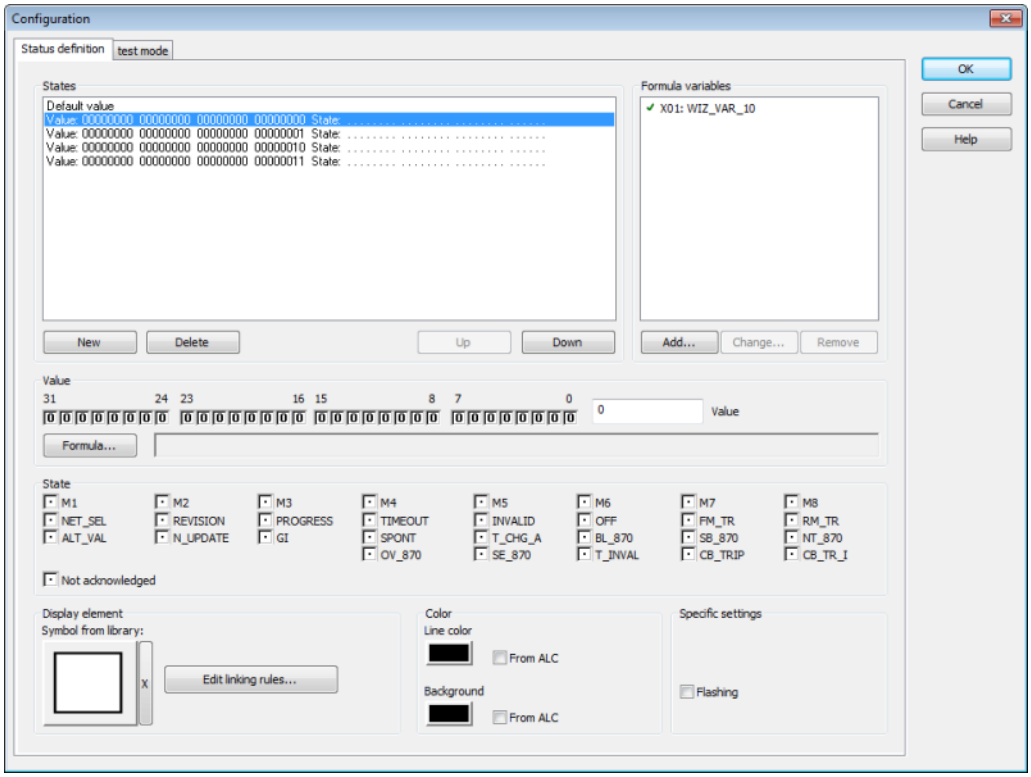
2. Results in the following in Runtime:
   - Source color: `Green`
   - Color without voltage: `White`
   - Switch status: `on/closed` (value`1`)

EXAMPLE 3

Combined element with value status `00` without colors from ALC:

1. Configuration in the Editor:
   - Combined element with value status `00`

- Line color from ALC not active



2. Results in the following in Runtime:

   - Source color: `Green`
   - Color not energized and construction color of the line: `White`
   - Defined line and fill color of the combined element: `black`
   - Switch status: `off/open` (value `0`)

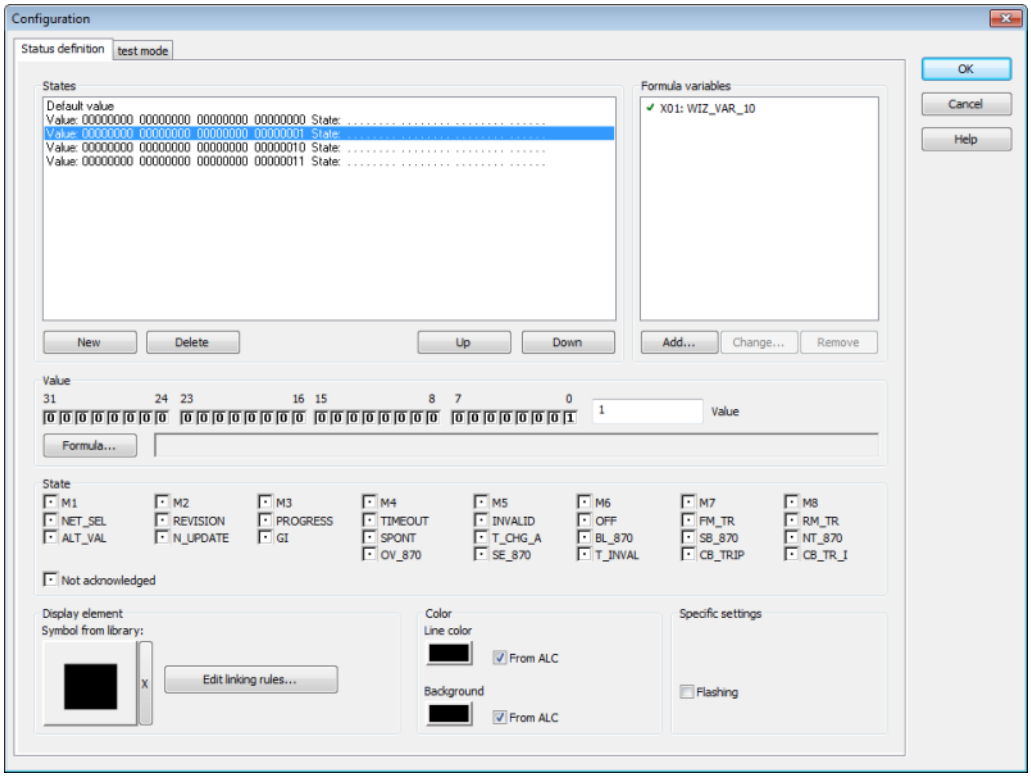
EXAMPLE 4

Combined element with value status `01` without colors from ALC:

1. Engineering in the Editor

   - Combined element with value status `01`
   - Line color from ALC inactive

- Fill color from ALC inactive



2. Results in the following in Runtime:

- Source color = green
- Color not energized and construction color of the line: `White`
- Defined line and fill color of the combined element: `black`
- Switch status: `on/closed` (value`1`)

Connection points of procedural elements

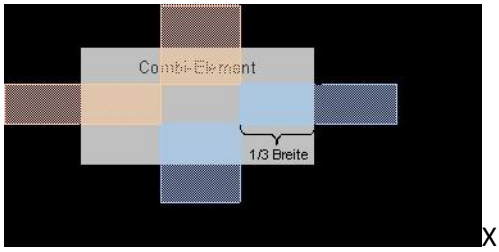
When configuring, a line is connected to a procedural element (combined element) by overlapping drawings in the screen at connection points of the combined element. Only one line can be connected to the same connection point at the same time. All lines that start within the area defined below, are connected (Topology from the graphic).

Attention

Use ALC elements only in un-rotated state because:

The calculation for the topological model for the ALC in the Editor is based on the position of the elements in un-rotated state and without considering any dynamics.

All possible connection points are shown in detail in the following illustration:



X = 1/3 width

Y = 1/3 width (max. 10 pixels)

Z = 1/3 height

W = 1/3 height (max. 10 pixels)

Info

If a line is outside the area shown above, there is no connection and thus no coloring. So there will also be no coloring for further lines.

With sources, drains and links, all described connection points can be generally used.

Attention

With sources and drains only one connection point must be used at the same time. It does not matter which connection point. If different connection points are used at the same time, undefined states can occur.

Elements of the type link can also use several connection points at the same time. The incoming color information is passed on to all lines.

In switches/disconnectors/sliders and transformers, the connection 1 (input) is on the left or on the top and connection 2 (outputs) are on the right or on the bottom. This background color can be changed with the `Switch input/output` property.

Info

At switches and transformers it has to be cared, that only one input connection and one output connection is used. The simultaneous use of several input or output connection points results in inconsistencies and is therefore not reliable.

Info

For all procedural elements the following is true: Only one line can be connected to a connection point. Junctions cannot be realized directly on an element but must be drawn with lines.

Switch input/output

If a transformer, disconnector or switch is configured, the input and output can be switched. To do this:

1. Select either transformer, disconnector or switch as a `Function type`
2. activate the `Switch input/output` check box

The input is then placed at the right or at the bottom and the output is placed left or at the top.

OVERVIEW

Configuration device	Input	Output
normal	Left	Right
normal	top	below
swapped	Right	Left
swapped	below	top

Lines

Lines are represented by vector elements Line, Polylines and Tube.

If the option `Color from ALC` is activated for a line, the coloring is defined by the ALC configuration. Lines are automatically colored by the system depending on the status of the procedural elements and the ALC settings.

Here the color usually comes from the highest priority source number of the media flowing through the line, or stays "empty/not energized" just as defined in the screen with static or dynamic colors.

You define the display type by means of drop-down lists:

- Priority for display
- Display multiple supply
- Display secured supply

The following options are available in the properties of the lines:

Parameters	Description
Color from ALC	Activates the automatic line coloring for these vector elements. That means: If the source for the line is active and all switches/valves leading from the source to the line are closed/open, the line is accordingly colored. If the line is fed by a single source, the defined source color is used for coloring the line. The line width is not changed.
Priority for display	Defines if multiple supply, secured supply or both are displayed. Default: Multiple supplies
Secured supply	The element is displayed according to the rules of the secured supply. A line is then considered to have a secure supply if it is supplied by at least two different switches or transformers with a non-system source. System sources do not contribute to secured supply, but do not exclude it.
Multiple supplies	The element is displayed according to the rules of the multiple supply. A line is considered to have multiple supplies if it is supplied by at least two different sources. In doing do, it does not matter if they are system or user sources and from which side the line is supplied by the sources.
No priority	The coloring rules for multiple supply and for secured supply are applied at the same time if both criteria are met. That means: If a line ▸ has multiple supplies and a secured supply, ▸ The priority is set to No priority, ▸ The display for multiple supply is set to two sources with highest priority, ▸ The display for secured supply is set to double width, then the line is twice as wide and displayed as a dashed line in two colors.

display multiple supplies	Multiple supply means that a line is supplied by multiple sources at the same time. Here you can define how lines with multiple supply are displayed. Default: highest priority source
highest priority source	The line gets the color of the source with the highest priority. **Note:** Priorities correspond to the sequence chosen in the ALC configuration.
two highest priority sources	Applies for lines fed by two or more different sources. The two sources with the highest priorities define the coloring. The line is displayed with the these two colors (dashed). The dash length can be changed using the Dashing length supplied multiple times property. System sources apply fir multiple supplies just as with genuine sources and color lines in two colors it they are configured accordingly.
Alternative color	The color defined in the Alternative color property is used.
Dashing length supplied multiple times	Defines the dash length (in pixels) of lines, polylines or tubes for the dashed ALC coloring for two sources with the highest priority for display multiple supplies. ▸ Minimal: 0 (automatic dash length) ▸ Maximum: 32767 ▸ Default: 0
Alternative color	Alternative color for the ALC coloring of lines, polylines or tubes with multiple supplies.
display secured supply	Secured supply means that a line gets multiple supply from one source (parallel). Here you can define how 'secured supply' is displayed. A line is always displayed as having a secure supply if it is supplied by at least two switches with a genuine source (not system source).

	Default: normal
double width	Relevant for lines fed in parallel by the same source. If this is the case, the line is displayed with double the configured width. (Example: A line with line width 5 pixels is displayed with 10 pixels if secure-fed.) If this line is fed by two or more different sources (multi-supply), the line width does not change! The color is always defined by the source with the highest priority!
double brightness	Relevant for lines fed in parallel by the same source. The line is displayed with double the original brightness. If this line is fed by two or more different sources (multi-supply), the line color does not change! If this line is multi-fed from one source (secure supply), the line is displayed with double the original brightness. Formula for the calculation of the double brightness: 1. The defined RGB color is transformed to the HLS system. 2. L (luminance = brightness) is recalculated with NewLuminance = 240*3/4 + L/4 3. The color value is recalculated to the RGB system with the new brightness. The color is always defined by the source with the highest priority!
normal	The element is displayed in the color of the source and with the configured width.
Use alias	Active: Alias is used.
Alias	Opens the dialog (on page 36) for selecting a model.

Info

The source color and the priorities of the sources are defined in the project properties.

User-defined sources must have a higher ID than 9. IDs up to 9 are reserved for system sources.

Info

The calculation of the color of a line in the Runtime is done with the following priority list:

1. Automatic Line Coloring (highest priority, overrules all other settings)
2. Dynamic colors
3. Static colors

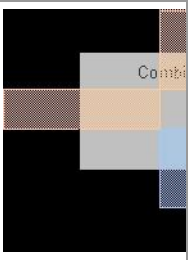
Example

In the following example Source 0 has the color blue and Source 1 has the color red. And Source 0 is the source with the highest priority.

Source 0

Source 1

This results in the following displays for the different options:

	Line / Polyline	Tube
highest priority source		
two highest priority sources		
double width		
double brightness		

Connection points of lines

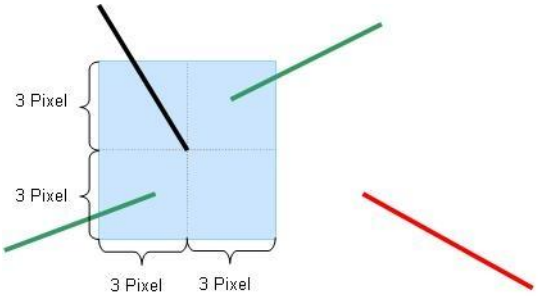
The connection of one line (line, polyline or tube) to another line is done with overlapping drawing in the screen at connection points. The connection points - either connection areas - are at the start and the end of each line and are around 3 pixels large.

Example

The start point of a line has the coordinates (start point x/start point y): 150/100 pixels. This results in a connection area (x / y): 147 - 153 / 97 - 103 pixels.

If the line start or end of this line and that of one or more other lines is within this area, the lines are automatically connected without any further engineering. A mere overlapping of the connection areas of the single lines is not sufficient!

In the following illustration the connection area is displayed graphically (the green lines are connected to the black one, the red line not.



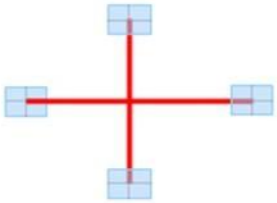
Info

Any number of lines can be connected in a connection area.

Attention

If a line is outside the connection area (e.g. the red line in the illustration), no connection is established and there is no coloring of the line. So there will also be no coloring for further lines.

Line crossings can easily be realized, if the ends of the lines are not in the connection area.



Attention

Use ALC elements only in un-rotated state because:

The calculation for the topological model for the ALC in the Editor is based on the position of the elements in un-rotated state and without considering any dynamics.

Checking the project

Engineer the desired procedural elements and lines in one or more screens and save these screens. Then you can check via **`Create all Runtime files`** or **`Create changed Runtime files`** whether there are any errors or conflicts in the screens. If error and/or conflicts should exist, corresponding error messages or warnings are displayed in the output window.

Info

Double click the corresponding line in the output window. The screen with the erroneous screen element will be opened automatically. If the erroneous screen element is part of a symbol, the corresponding symbol is automatically selected.

The following error message can be displayed.

- ALC: Screen '%s' - Two Link elements with different Link number are connected to line '%s' . (Double click opens the screen and selects the line.)
- ALC: Screen '%s' - More than two connection points are used at element '%s'. For each element only one input and one output may be used. (Double click opens the screen and selects the element.)

The following warnings can be displayed.

- ALC: Screen '%s' - Alias line '%s' is connected to a no-alias line. (Double click opens the screen and selects the line.)
- ALC: Screen '%s' - Alias element '%s' is connected to a no-alias line. (Double click opens the screen and selects the element)
- ALC: Screen '%s' - No-alias element '%s' is connected to an alias line. (Double click opens the screen and selects the element)
- ALC: Screen '%s' - Line '%s' is only connected on one side. (Double click opens the screen and selects the line.)
- ALC: Screen '%s' - Element '%s' is not connected. (Double click opens the screen and selects the element)
- ALC: Screen '%s' - Element '%s' is only connected on one side. (Double click opens the screen and selects the element)

In the error messages or warnings the corresponding elements are identified using the element reference. This reference also serves as the link key for ALC aliases.

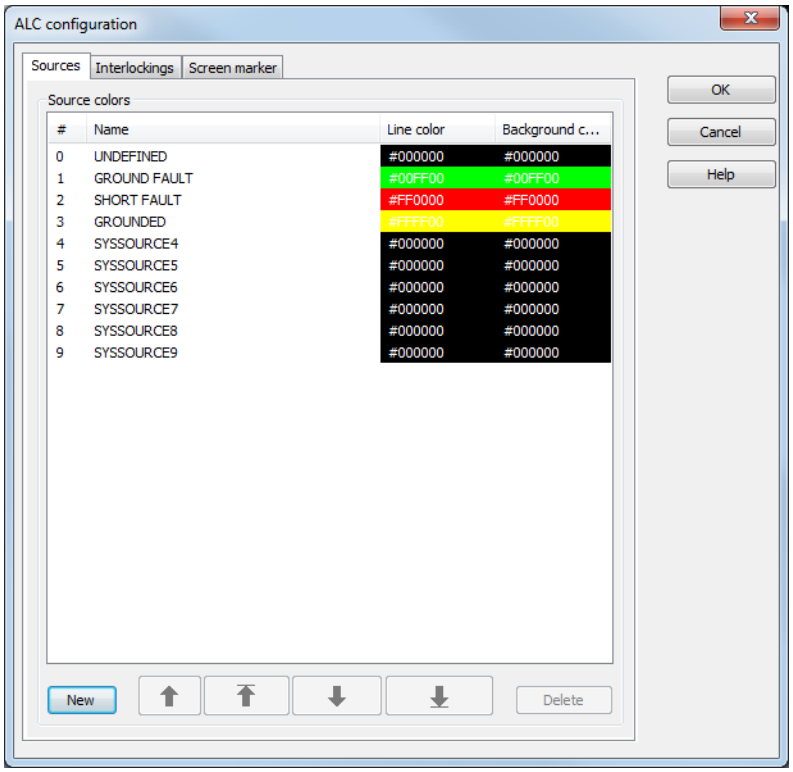
2.1.2 Configuration

To configure ALC:

1. In project properties, select `ALC configuration` the property in the `Automatic Line Coloring` group
2. Click on the `...` button
3. The dialog for configuration is opened
4. Configure the desired properties for:
   - Sources (on page 26)
     (note also the principles for Coloring for UNDEFINED (on page 28).)
   - Interlockings (on page 30)
   - Screen marker (on page 33)

Configuration of the sources

The sources, e.g. their names and colors (sequence and priority), are configured project-specifically within the project properties under 'ALC configuration'.



Parameters	Description
Number	Internal unique consecutive number, so that the source can be identified. This number is given by the system automatically and cannot be changed. **Attention:** IDs 0 to 9 are reserved for the system sources and must not be used user-specific.
Name	Logical name for the source (e.g.: 'water' or 'grounded'). This name is also used when selecting the source number for Combined elements. You can change the name by clicking it with the left mouse button. With this edit mode is switched on. The changes are accepted with Enter or by selecting another source. **Note:** The labels are not language switchable.
Foreground	Foreground color of the source. This color is used for coloring lines, polylines and as the outside color of tubes.
Background	Background color of the source. This is used as the background color for tubes and procedural elements (Combined element).
New	Adds a new color.
Delete	Deletes the selected color.

The colors can be configured directly by entering the corresponding hexadecimal code or by using a color palette.

For direct input:

1. Click on the color description with the left mouse button
2. The field is switched to editing mode
3. Enter the code
4. Press the input key or select another source in oder to accept the change

To select via a color palette:

1. highlight the desired line
2. click on the ... button behind the color
3. Online help is opened.
4. select the desired color

The hexadecimal code describes the RGB color value and consists of the following. #RRGGBB.

Element	Meaning
#	Identifier to indicate that a hexadecimal color code is used.
RR	2 digits are the red value of the color in hexadecimal system. 0-255 is 0-FF
GG	2 digits are the green value of the color in hexadecimal system. 0-255 is 0-FF
BB	2 digits are the blue value of the color in hexadecimal system. 0-255 is 0-FF

Attention

Limitations for deleting sources:

The sources 0 to 9 are reserved for system sources and cannot be deleted.

Only the source with the highest ID can be deleted.

Info

The sequence in this list represents the priority of the sources, with the first element having the highest priority.

To change the priorities of the single sources, they can be moved upwards or downwards using the arrow buttons

Coloring mode for UNDEFINED

Coloring in the network can be implemented in two modes with the UNDEFINED status:

- Standard
- Input takes priority

This setting is made using the Automatic Line Coloring/Mode for coloring project property.

STANDARD

The graph search starts with a source and goes through the whole network, so that each closed switch (switch variable has the value `1`) per direction is only gone through once, so no cycles occur. In doing so, each node visited (=line segment) is colored with the source color. The directly-related lines are marked as a node.

If the search finds a switch that has a switch variable with the following status, the UNDEFINED color is used for coloring from this point onwards:

- INVALID [values: any desired],
- is invalid [value: 3] `3`]
- is in intermediate position [value: `2`])

The graph search is now continued in the same form. Each switch is gone through just once per direction with the UNDEFINED color. Therefore each switch can be gone through a maximum of four times per source:

1. with source number in forwards direction,
2. with source number in backwards direction,
3. with UNDEFINED in forwards direction,
4. with UNDEFINED in backwards direction,

INPUT TAKES PRIORITY

With the `Supply takes priority` setting, only lines that have a supply from at least one source but not clearly from any one source are colored as UNDEFINED. If a line is supplied with at least one source, it can no longer receive an UNDEFINED color from another source.

This search is a two-stage search:

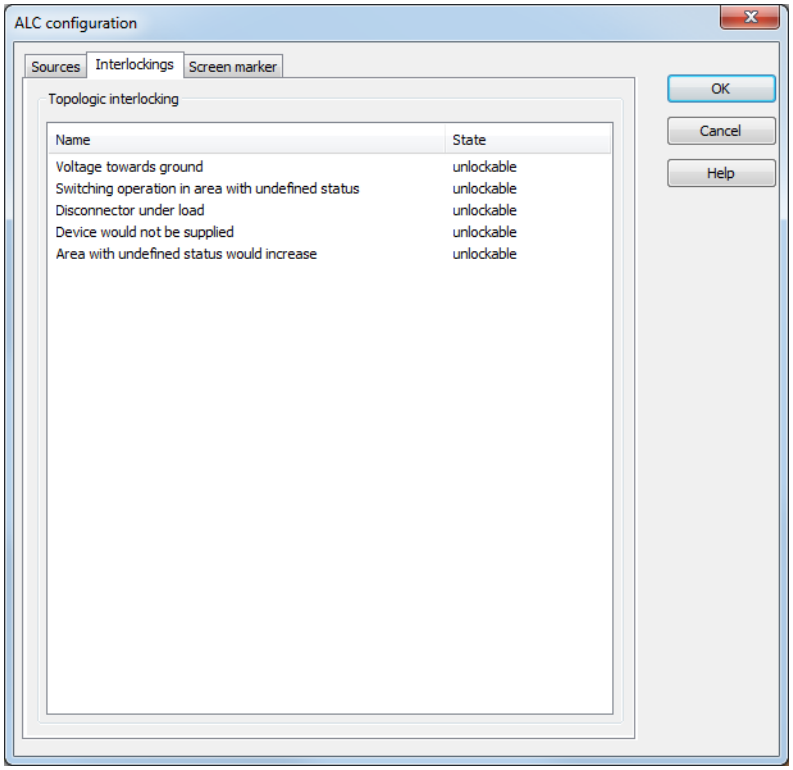
- In the first stage, as with `Standard`, the source color is distributed in the network from each switched source, as long as the next switch is closed. The search is ended if the switch is open or invalid/undefined.
- In the second stage, the search is started at each invalid/undefined switch that receives a supply from one side and the UNDEFINED color is distributed to the unsupplied side. This search also considers the switches that are invalid/undefined as closed and thus distributes the UNDEFINED

color in the network until it meets a clearly open switch. In addition, a search is ended if a line element is reached that is already supplied.

Configuration of topological interlockings

topological interlockings from the ALC for commands can be configured here.

> **Info**
>
> *This dialog is only available when both the "Energy Edition" and the "Automatic Line Coloring" modules are licensed.*



The following conditions are available: The settings made here apply globally, for the whole Topological Model:

Parameter	Description
Voltage towards ground	Interlocking is active if a switch/disconnector is to be closed and a grounded potential is connected to its first connector and its other connector is connected or undefined.
Switching action in an area with an undefined status	Interlocking is active if a switch disconnector is to be closed and both of its connectors are 'undefined' or 'disturbed'.
Disconnector under load	Interlocking is active if certain conditions have been met for switching on or off. Conditions: See "Disconnector under load - interlocking conditions (on page 32)" section.
Device would no longer be supplied	Interlocking is active, when a consumer, which was supplied before, would be unsupplied after the switching action (by switch or disconnector).
Area with undefined status would increase	Interlocking is active if a switch disconnector is to be closed and one connector is 'undefined' or 'disturbed' and the other not.

If you click in the **Status** column in one of these interlockings, a drop-down list opens with three choices:

Parameters	Description
do not check	The selected condition is not considered in this project (topological model).
unlockable	The selected condition is considered in this project. If the condition applies, the user can unlock it with a command (**Command** screen). This unlocking action is logged in the Chronological Event List.
not unlockable	The selected condition is considered in this project. The user cannot unlock it.

EXCEPTION TOPOLOGICAL INTERLOCKING

The topological interlocking is not carried out if:

- the variable of a switch has the status Revision
  or
- the variable is manually corrects or set to `Alternate value` and with this is set to the same variable value as the initial value; in other words if the switch:
  - is set to OFF and then it is manually corrected to OFF or replaced.
  - is set to ON and then it is manually corrected to ON or replaced.

Disconnector under load - interlocking conditions

For the `disconnector under load` topological interlocking, a disconnector can be switched if one of the following conditions is met:

WHEN BEING SWITCHED ON:

Before being switched:

- The left and the right line segment receive energy from the same source
- If the left line segment does not receive any voltage, the right line segment is grounded
- If the left line segment is grounded, the right line segment does not receive any voltage
- If the left line segment is not under load
- If the right line segment is not under load

WHEN BEING SWITCHED OFF:

After being switched:

- The left and the right line segment would receive energy from the same source
- If the left line segment does not receive any voltage and the right line segment is grounded
- If the left line segment were grounded, the right line segment would not receive any voltage
- If the left line segment were not under load
- If the right line segment were not under load

Info

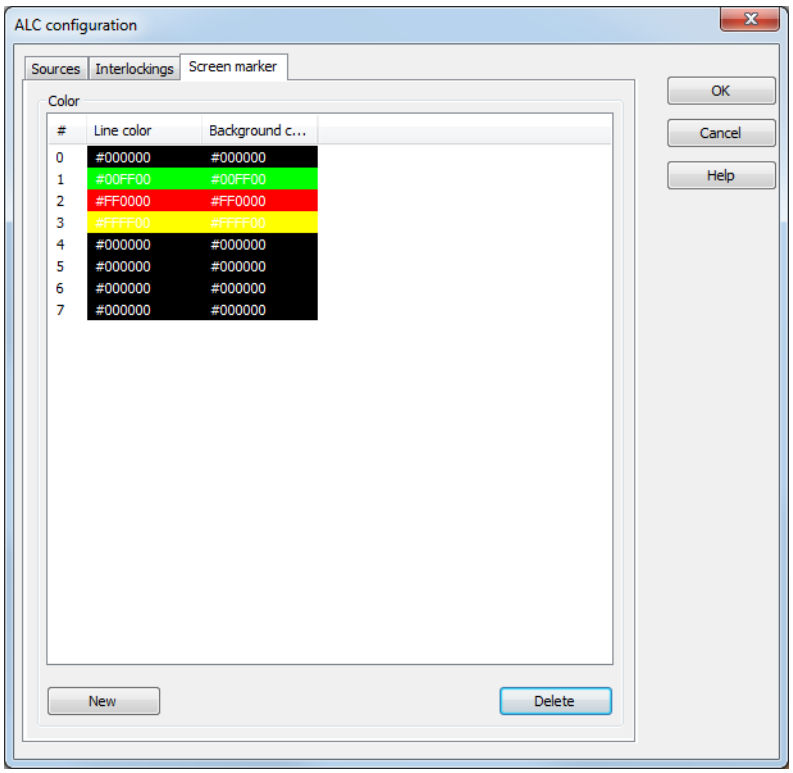
Meaning of "not under load"

All of the following conditions must be met for the status of `not under load`:

- All switches and disconnectors connected to the line segment are open.
- All sources and consuming devices connected to the line segment are switched off.
- No transformer may be connected to the line segment.
- It must not be a line that is only connected to this disconnector (open line).

Configuration of the screen marker

Here you configure the color table for the color marker for the impedance-based error detection and calculation of load distribution (on page 55). See also: AddMarker



Parameters	Description
Number	Unique internal serial number for clear assignment. This number is given by the system automatically and cannot be changed.
Line color	Line color of the screen marker.
Filling color	Filling color of the screen marker.
New	Adds a new color.
Delete	Deletes the selected color.

The colors can be configured directly by entering the corresponding hexadecimal code or by using a color palette.

For direct input:

1. Click on the color description with the left mouse button
2. The field is switched to editing mode
3. Enter the code
4. Press the input key or select another source in oder to accept the change

To select via a color palette:

1. highlight the desired line
2. click on the ... button behind the color
3. Online help is opened.
4. select the desired color

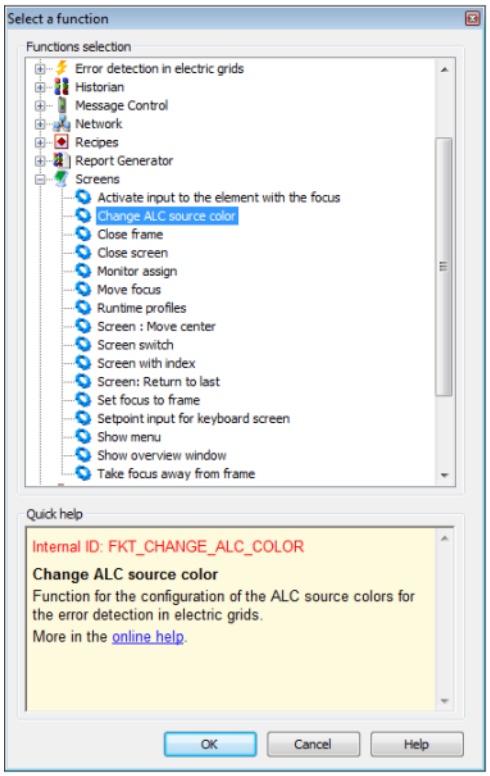
The hexadecimal code describes the RGB color value and consists of the following. #RRGGBB.

Element	Meaning
#	Identifier to indicate that a hexadecimal color code is used.
RR	2 digits are the red value of the color in hexadecimal system. 0-255 is 0-FF
GG	2 digits are the green value of the color in hexadecimal system. 0-255 is 0-FF
BB	2 digits are the blue value of the color in hexadecimal system. 0-255 is 0-FF

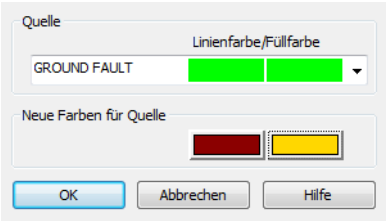
2.1.3 Change ALC source color

The foreground and background color of an ALC source can be temporarily changed for the coloring in Runtime using the `Change ALC source color` function. The change remains until Runtime is ended, reloaded or the function is executed again. To create the function:

- select New Function
- navigate to the screens node
- select `Change ALC source color`



- The dialog to define line colors and filling colors opens
- define the desired color



Property	Function
Source	Drop-down list to select the source and display the colors currently assigned. These colors cannot be changed here.
New color for source	Click on the color and a dialog opens to select a color.

2.1.4 Detail screens

To display individual screens, a partial area can be taken from the topological network and displayed individually by means of alias. The screen elements in the detail screen are not included in the topological model, but do however get their ALC colors from the model. They relate to an alias of the screen elements in the overall screen.

CREATE ALIAS

Aliases can be created for the elements:

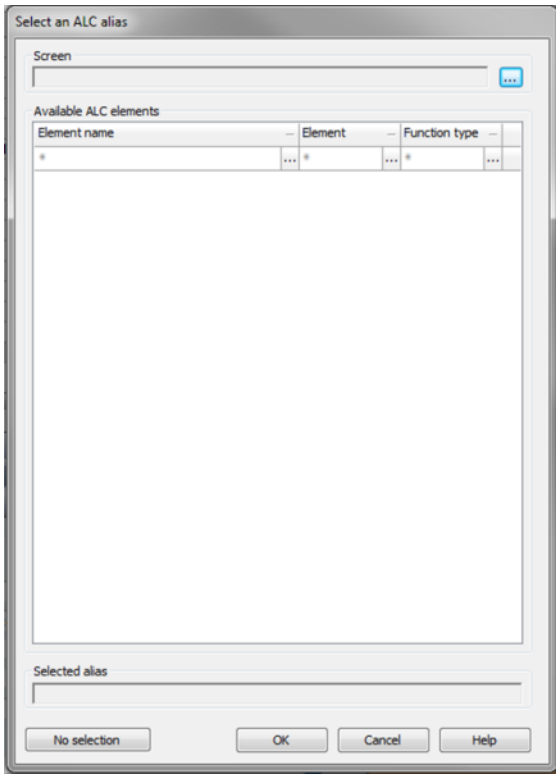
- Line
- Polyline
- Tube
- Combined element

To create a source element as an alias:

- Activate it in the element's properties `Use alias`

(to do this, ALC must be licensed and the `Color from ALC` property active)

- Click on the ... button in the `Alias` property
- the dialog to select elements opens



Parameter	Description
Screen	Click the ... button and a dialog opens to select a screen.
Available ALC elements	Shows the elements that belong to a screen with the element name, type of element and function type. Clicking on an element selects an alias. **Filter** The elements can be sorted according to all columns. When setting a filter, the options offered from all other filters are reduced to values that can be sensibly combined. ▸ **Name**: Input of a standard search term with wild cards (*). The last 12 search terms are offered in the list until the Editor is ended. ▸ **Element**: Select from drop-down list. ▸ **Function type**: Select from drop-down list. Clicking on **...** opens saved search or drop-down list. If a filter is active, clicking on the **x** deletes the filter.
Selected alias	Shows the selected element in the field of **Available ALC elements**.
No selection	Removes selected element.
OK	Saves selection and closes dialog.
Cancel	Discards changes and closes dialog.
Help	Opens online help.

Info

When selecting an element for a new alias, only elements and screens from the same project that the alias was defined in can be selected. Elements from subprojects or parallel projects are not available.

REPLACING ALIAS NAMES

Aliases can be be changed when switching screens with Replace link. A detail screen can therefore be displayed with the data from different equipment parts, for instance lines or partial networks. Alias names are replaced along the lines of variables and functions. It is also possible to replace in elements that are used in symbols. The same dialog as is opened for the target as the `Alias` property.

Note: Substitution using index variables is not possible.

2.1.5 Error detection in electric grids

Error detection marks network parts that are subject to ground faults or short circuits by means of special colors in ALC. Sources for error detection are what are called ground fault or short circuit reporters that are assigned to a circuit breaker. Ground fault and short circuit reporters are always at the output of a circuit breaker element. Error messages are fixed in the screen and must be reset manually.

> **Info**
>
> *This function is only available when both the "Energy Edition" and the "Automatic Line Coloring" modules are licensed.*

ERROR DETECTION

Error detection runs locally. Each client in the network has its own independent model and can therefore search for ground faults and short circuits in different parts of the network.

Error detection in the electrical network is divided into:

- Ground fault search (on page 41)
- Short circuit search (on page 51)

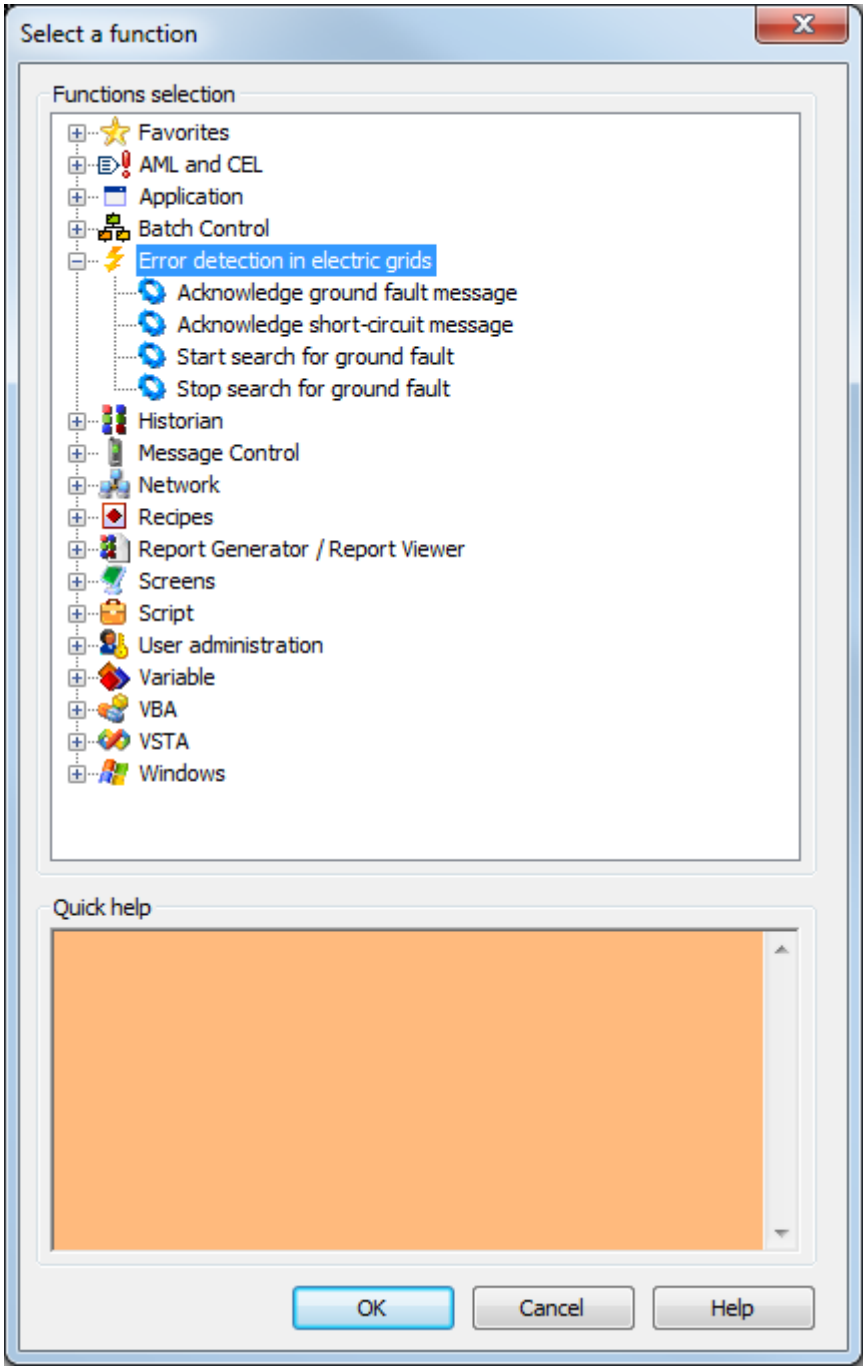
To configure error detection

- You require a license for ALC and zenon Energy Edition
- configure the appropriate screens
- Configure (on page 8) ALC to the corresponding combined elements (switch, transformer, disconnector, slider)
- configure (on page 18) the lines so that they are colored by ALC

Special functions are available in Runtime for error detection:

- Start ground fault search (on page 45)
- Acknowledge (on page 47) ground fault message (on page 47)

- End ground fault search (on page 49)
- Acknowledge ground fault message (on page 53)



COLORINGS

Errors can be shown by a special coloring for the lines in ALC. In Runtime, the color assigned by ALC changes automatically as soon as the status of the line changes. The colorings configured can be changed in Runtime via the change ALC source color (on page 35) function.

Messages are processed in the order in which they arrive. In the event of conflicts

- The colors for displaying errors take priority
- short circuit messages have priority over ground fault messages

Ground fault search

The ground fault search serves to highlight the network parts hat potentially have a ground fault by coloring these. The color is taken from the configuration of ALC source colors (on page 25) for the GROUND FAULT source.

Which network parts potentially have a ground fault can be deduced from the ground fault messages from ground fault identification devices (ground fault indicators, protective device with ground fault recording). The following applies for ground faults:

- Each device can have one to three ground fault messages.
- Ground faults are either dealt with by permanent message processing or by transient message processing.
- For directional ground fault devices, the direction can be lagging or leading in relation to triggering.
  - leading: First the message, then the transient bit.
  - lagging: First the transient bit, then the message.

> **Info**
>
> *A network component that potentially has a ground fault is then no longer considered to have a ground fault if this has been successfully connected.*

CONFIGURATION

To configure the ground fault search:

1. assign the combined element that represents the switching element to the `Function type` switch (on page 43)
2. define the ground fault search mode (on page 42), fault display (on page 44) and ground fault indication triggering (on page 44)
3. set up the functions for start ground fault search (on page 45), acknowledge ground fault search (on page 47) and end ground fault search (on page 49)

> **Info**
>
> *In order to also be able to set limits in intermeshed networks, only one area subject to a ground fault per path is searched for a fault.*

Mode of the search for ground faults

The short circuit search can either:

- color the network part potentially subject to a short circuit
  or
- the whole network where the short circuit is located

The coloring mode is defined via the `Mode of the search for ground faults` property.

To configure the property:

- navigate to the `Automatic Line Coloring` node in properties
- select the desired mode in the `Mode of the search for ground faults` property drop-down list
  - Color network part: colors only the network parts that are potentially subject to a short circuit
  - Color whole network: colors in the whole linked network where the short circuit is located

This setting can be changed in Runtime via the zenon API object model. In doing so, the short circuit search is recalculated once again.

Earth Fault Identification Type

The direction and type of information processing for the switch type combined element are determined by the `Type` setting. to configure:

1. navigate to the `Automatic Line Coloring` node in the combined element properties
2. open the `Ground fault recognition` node
3. select the desired type with the direction and type of alarm processing in the `Type` property
    - Direction:
      indicates if the raising edge of trip alarm or if the raising edge of a direction comes before
    - leading: The current direction status is used for the raising edge of the trip alarm
    - lagging: after a raising edge of the trip alarm, the first raising edge of a direction is waited on; if this does not occur within 2 seconds, the earth fault device is considered non-directional
    - Information processing:
      states which information can be processed
    - none: normal switch; information is not processed
    - Permanent message processing: Newly received messages are considered a new ground fault trip
    - Transient message processing: Messages that are received during a current Search (on page 45) are suppressed

Note: The distinction between permanent message processing and transient message processing only relates to processing the message, not to the type. Transient bit message processing need not therefore relate to a transient bit.

> **Attention**
>
> *To suppress intermittent ground faults, ground fault messages that are received in intervals shorter than 2 seconds are ignored.*

Ground fault display

The variable linked at `Display` is an output variable for error detection and displays the recorded status of the ground fault identification device. This is necessary because all messages remain saved internally until they are acknowledged, i.e. they do not necessarily conform to the current status of the message variables.

Each time a recording is made, a set value is sent to this variable. In doing so, the values are as follows:

Value	Meaning
0	no ground fault
1	ground fault forwards
2	Ground fault backwards
3	non-directional ground fault
4	Error status - > both directions have activated

Info

To reduce problems in network operation, the variable linked here should be a linked variable.

Earth fault triggering

The alarm to report an earth fault is defined by the `Triggering` variable It can contain information on the presence of an earth fault and the direction of the earth fault from the point of view of the earth fault recognition device. In doing so, a distinction is made between:

- non-directional earth fault alarms
- Directional earth fault alarms with a trip alarm
- Directional earth fault alarms with a trip alarm

To configure the variable for the `Triggering`:

1. navigate to the `Automatic Line Coloring` node in the combined element properties

2. open the `Ground fault recognition` node

   a) for non-directional earth fault alarms

   Click on the ... button in the `Triggering` property

   select the variable you wish to import in the dialog that opens

   The properties for the direction remain empty

   b) for directional earth fault alarms with a trip alarm

   link the variable with `Triggering` and add the appropriate direction:

   Forwards: link a variable to the `Forwards` property

   Backwards: link a variable to the `Backwards` property

   c) for directional earth fault alarms without a trip alarm

   Link the variable with the corresponding direction:

   Forwards: link a variable to the `Forwards` property

   Backwards: link a variable to the `Backwards` property

   The `Triggering` property remains empty

Note: If you address a directional identification device with `Forwards` in both directions, this is then considered erroneous and ignored.

Start ground fault search

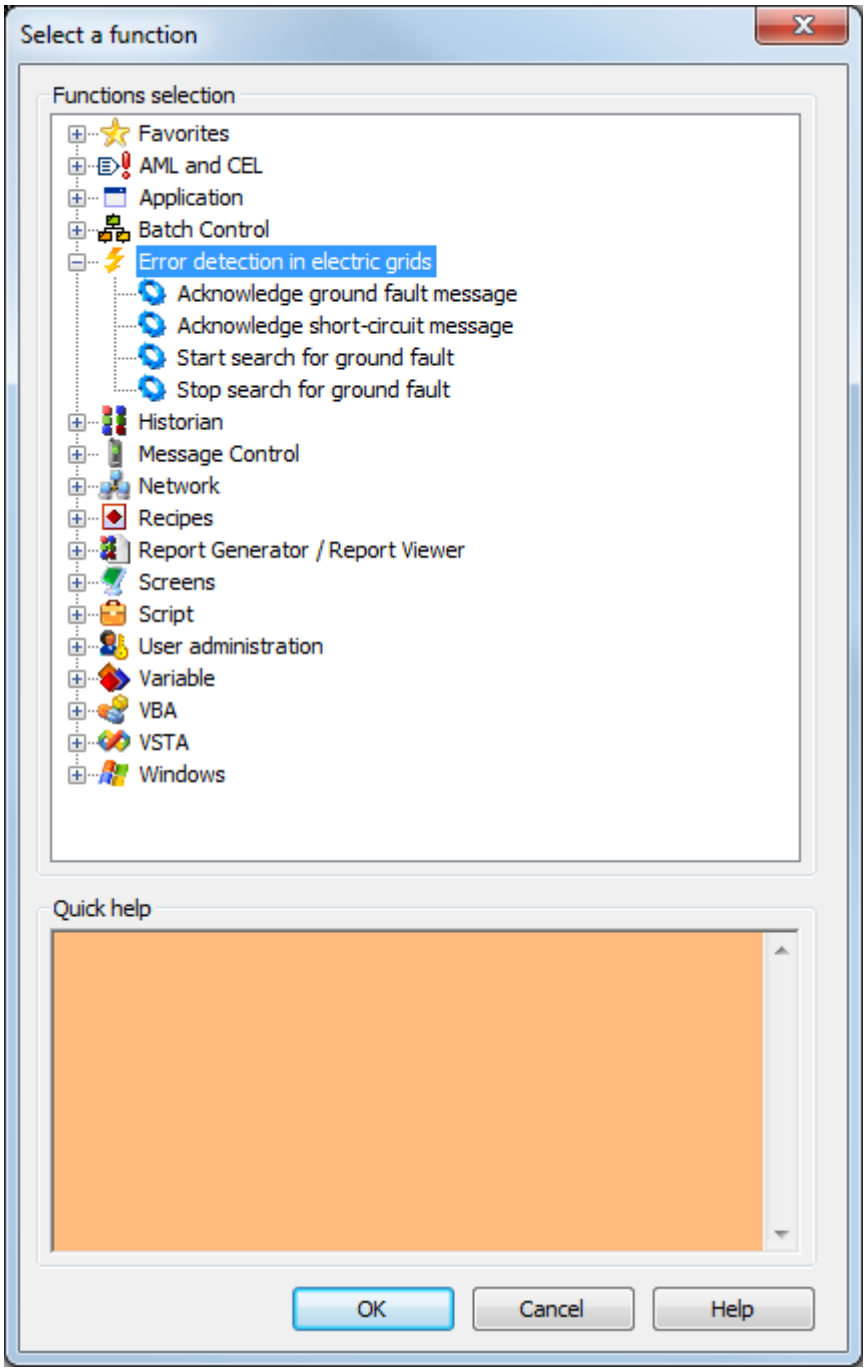
The function `Start search for ground fault` serves to localize a ground fault and has two effects in Runtime:

1. Fault reports from all ground fault identification devices that were configured with wiper message processing are ignored.
2. The search algorithm is changed: Switch actions can only reduce the area subject to a ground fault further. Newly received messages do not therefore increase the area potentially subject to a ground fault.

To configure the `Start search for ground fault` function:

- create a new function
- navigate to the error detection node in the electrical network

- Select the `Start search for ground fault` function



- link the function to a button

Acknowledge ground fault message

With the `Acknowledge ground fault message` function, an internally recorded ground fault from a ground fault indication device can be acknowledged. In doing so, the internally-latched ground fault status is reset if the status is still pending, or highlighted as acknowledged. A recorded ground fault message is only deleted internally if this has been acknowledged and is no longer pending.

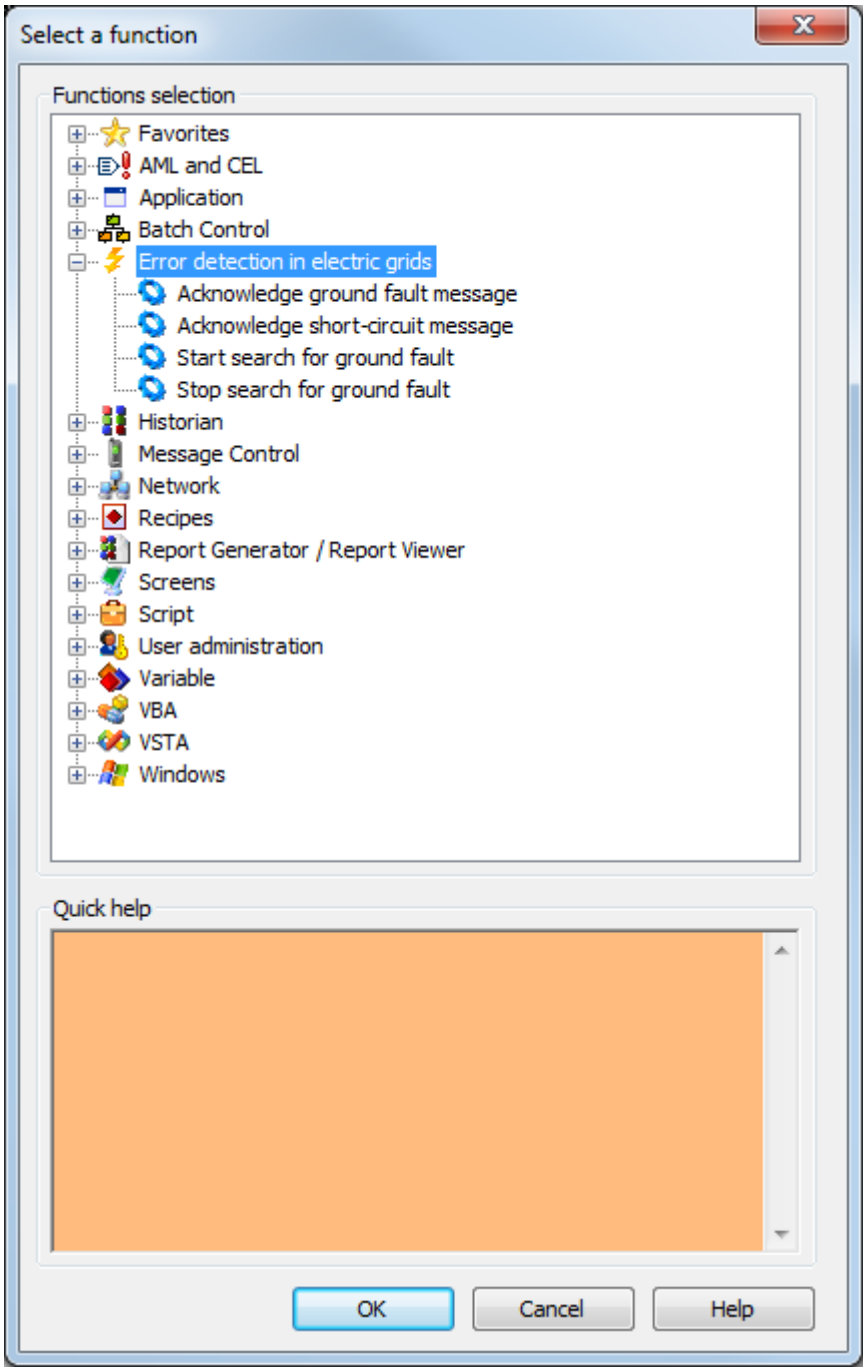
Rules when acknowledging:

- If a variable that corresponds to a triggering or direction variable of a ground fault recognition device is linked, this special ground fault message is acknowledged.
- If no variable has been linked, all ground fault messages are acknowledged.
- Acknowledgment can also take place via the zenon API object model.

To configure the `Acknowledge ground fault message` function:

- create a new function
- navigate to the error detection node in the electrical network

- Select the `Acknowledge ground fault message` function



- the dialog to select a variable opens
- link the desired variable to the function
- link the function to a button

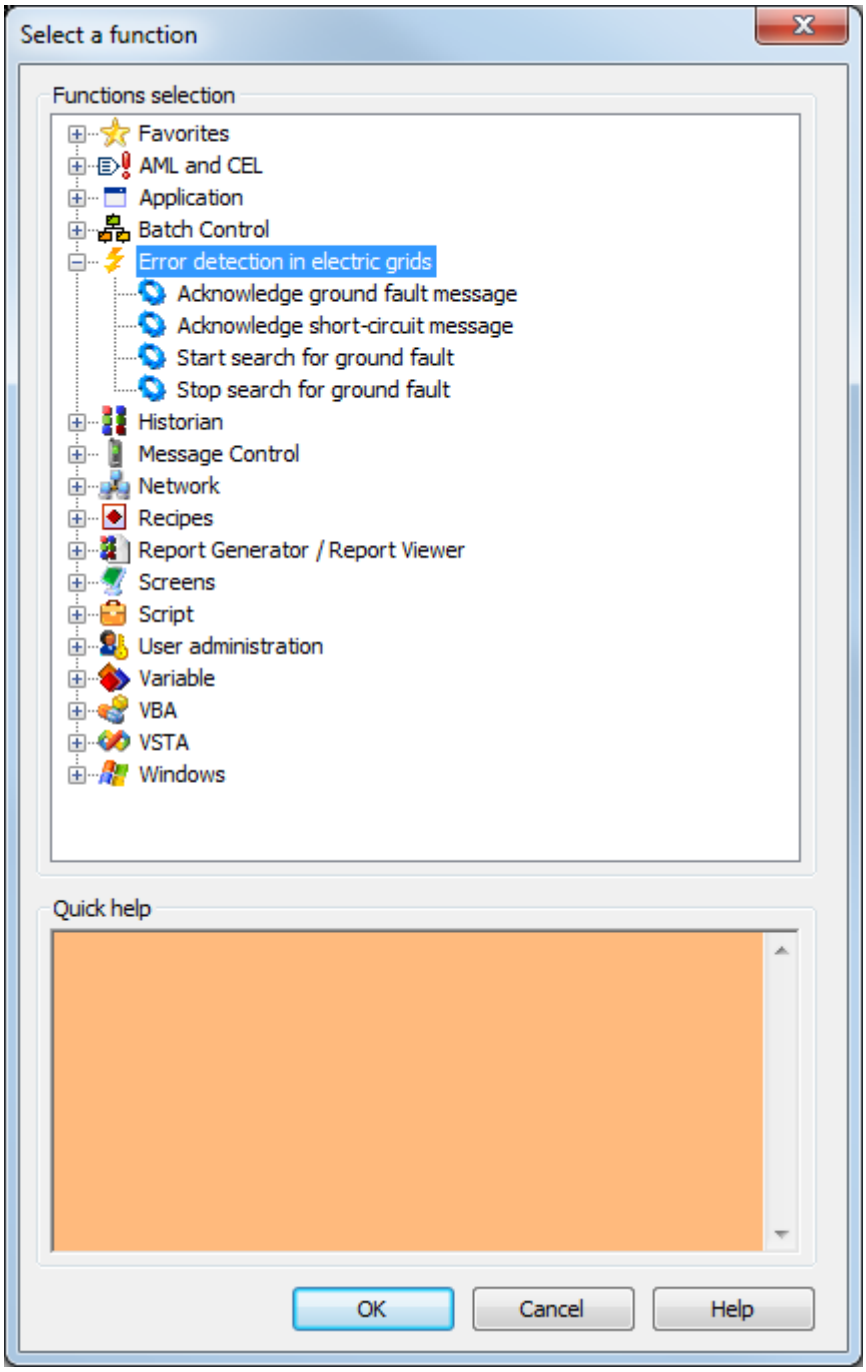
End ground fault search

You end the ground fault search with the `Stop search for ground fault` function in Runtime.

To configure the function:

- create a new function
- navigate to the error detection node in the electrical network

- Select the `Stop search for ground fault` function



- link the function to a button

Short circuit search

The short circuit search serves to highlight the network parts that potentially have a short circuit by coloring these. The color is taken from the configuration of ALC source colors for the `SHORT FAULT` source.

The network parts that are potentially subject to short circuits are deduced from short circuit reports. A short circuit identification device (short circuit indicator, protective device) can have one to three short circuit messages. For directional short circuit indication devices, the direction can be lagging or leading in relation to triggering. A network component that potentially has a short circuit is then no longer considered to have a ground fault if this has been successfully connected.

ENGINEERING

To configure the short circuit search:

1. assign the combined element that represents the switching element to the `Function type` switch (on page 51)
2. define Short circuit display (on page 52) andShort circuit identification triggering (on page 52)
3. Set up the Acknowledge short circuit message (on page 53) function

Short circuit identification type

The direction and type of information processing for the switch type combined element are determined by the `Type` setting. to configure:

1. navigate to the `Automatic Line Coloring` node in the combined element properties
2. open the `Short-circuit detection` node
3. select the desired type at the `Type` property
   - `Direction:`
     indicates if the raising edge of trip alarm or if the raising edge of a direction comes before
   - `leading:`
     for the raising edge of the trip alarm, the current direction status is used
   - `lagging:`
     after a raising edge of the trip alarm, the first raising edge of a direction is waited on;

if this does not occur within 2 seconds, the short circuit device is considered non-directional

- `Information processing:`
  states which information can be processed
- `none:`
  normal switch; information is not processed
- `Permanent message processing:`
  Newly received messages are considered a new ground fault trip

Short circuit display

The variable linked at `Display` is an output variable for error detection and displays the recorded status of the short circuit identification device. This is necessary because all messages remain saved internally until they are acknowledged, i.e. they do not necessarily conform to the current status of the message variables.

Each time a recording is made, a set value is sent to this variable. In doing so, the values are as follows:

Value	Meaning
0	no short circuit
1	Short circuit forwards
2	Short circuit backwards
3	Non-directional short circuit

Short circuit identification triggering

The variable for the message from the short circuit identification device is defined by the `Triggering` variable You can receive information about the presence of a short circuit and the direction of the short circuit from the point of view of the short circuit identification device.In doing so, a distinction is made between:

- non-directional short circuit reporters
- directional short circuit reporters with a trip alarm
- directional short circuit alarms with a trip alarm

To configure the variables for:

1. navigate to the `Automatic Line Coloring` node in the combined element properties
2. open the `Short-circuit detection` node
   a) for non-directional short circuit detection devices

   Click on the ... button in the `Triggering` property

   select the variable you wish to import in the dialog that opens

   The properties for the direction remain empty

   b) for directional short circuit detection devices with a trip alarm

   link the variable with `Triggering` and add the appropriate direction:

   Forwards: link a variable to the `Forwards` property

   Backwards: link a variable to the `Backwards` property

   c) for directional short circuit detection devices without a trip alarm

   Link the variable with the corresponding direction:

   Forwards: link a variable to the `Forwards` property

   Backwards: link a variable to the `Backwards` property

   The `Triggering` property remains empty

Acknowledge short circuit message

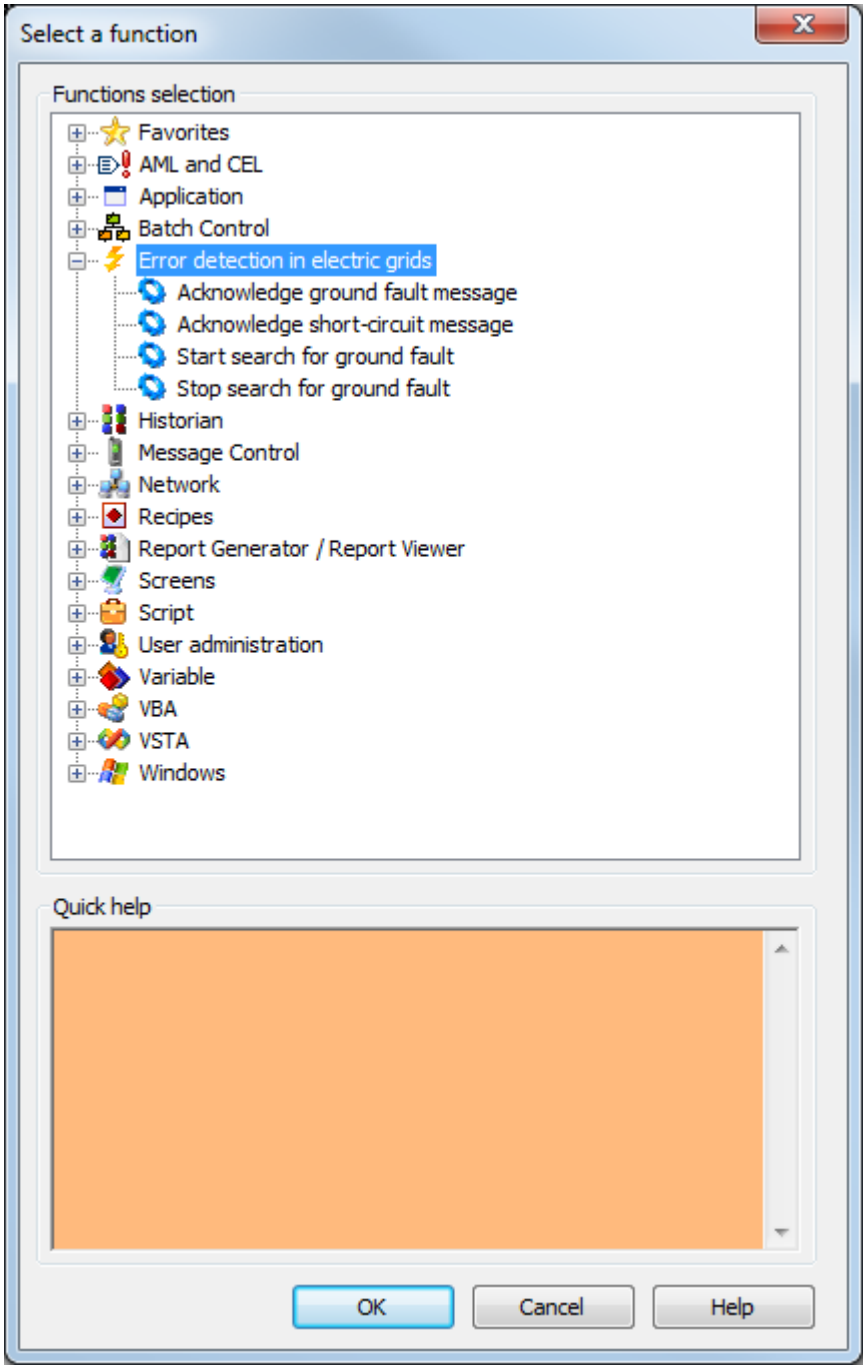
With the `Acknowledge short-circuit message` function, an internally recorded short circuit from a short circuit indication device can be acknowledged. In doing so, the internally-latched ground fault status is reset if the status is still pending, or highlighted as acknowledged. A recorded short circuit message is only deleted internally if this has been acknowledged and is no longer pending.

Rules when acknowledging:

- If a variable that corresponds to a triggering or direction variable of a short circuit recognition device is linked, this special short circuit message is acknowledged.
- If no variable has been linked, all short circuit messages are acknowledged.
- Acknowledgment can also take place via the zenon API object model.

TO CONFIGURE THE ACKNOWLEDGE SHORT-CIRCUIT MESSAGE FUNCTION:

- create a new function
- navigate to the error detection node in the electrical network
- Select the `Acknowledge short-circuit message` function



- select the variable you wish to import in the dialog that opens
- link the function to a button

2.1.6 impedance-based error detection and calculation of load distribution

Impedance based error detection and calculation of load distribution expands ALC. Whereas ALC identifies nodes and beams, this model also detects lines and their parameters. The model is not used internally in zenon, but provides properties and methods for external evaluation.

PROPERTIES FOR ALC AND THE EXTENDED TOPOLOGICAL MODEL

The ALC elements combined element and line (line, polyline, tube) have special properties for error detection for protection and to calculate the load distribution. These properties are not evaluated in zenon, but are available via the zenon API algorithms to be created by users.

The simple topological model for the coloring was supplements by an expanded topological model that includes all lines as separate beams. The extended topological model is stored as **`ALC.xml`** and can be read by external applications this way. **`ALC.xml`** contains two sections:

- **`GraphElements`**: contains the extended topological model without aliases
- **`GraphAliases`**: contains only the aliases

Each object has a unique ID, via which it is referenced in the file. The attributes correspond to a subset of the zenon screen elements that have created the elements.

GRAPHELEMENT

ID	Description
Picture	Screen name
ElementID	Screen element ID
ElementRef	Screen element reference
Type	Screen element -type (see "element")
SourceID	Source number
ReverseSourceID	Source name in reverse direction
Variable	Status variable
VarProtReact	Reactance variable
MaxIType	Type of maximum current
MaxIVal	Maximum current constant value
VarMaxI	Maximum current variable
VarCurI	Instantaneous current variable
VarCalcI	Calculated current variable
VarCurP	Instantaneous power variable
LoadType	Type of load
LoadVal	Load constant value
VarLoad	Load variable
React	Reactance
Resist	Resistance
Length	Line length
Node1IDs	List of all element IDs connected with Node1
Node2IDs	List of all element IDs connected with Node2

GRAPHALIAS

ID	Description
Picture	Screen name
ElementID	Screen element ID
ElementRef	Screen element reference
Type	Screen element type (see "element")
OrigElemRef	Screen element - reference to the original screen element
OrigGraphElemID	ID of the original elements in "GraphElements"

API

In the object model of the zenon API, the objects **ALCGraphElement** and **ALCGraphAlias** are available for the model. These contain the same information as the XML file. These objects can be accessed in the ALC engine via:

- **GraphElemCount()**
- **GraphAliasCount()**
- **GraphElemItem()**
- **GraphAliasItem()**

USER-SPECIFIC TOPOLOGICAL INTERLOCKINGS

If a topological interlocking is checked, the following event is called up at the ALC engine:

- **void CheckInterlocking(IALCEdge* pALCEdge, long nNewState, tpLockResult* LockResult, BSTR* bsText, VARIANT_BOOL* bUnlockable);**

The switch/disconnector to be switched and the new status is transferred. The event can fill **LockResult**, **bUnlockable** and **bsText** in order to display a breached interlocking condition. If the event handler returns **tpBusy** in **LockResult**, the event handler is queried until it no longer provides **tpBusy**, however for a maximum of 10 seconds. The interlocking is active after 10 seconds. The interlocking text and unlockability are reported back in **bsText** and **bUnlockable**.

SCREEN MARKER

Marker elements can be inserted into screen s via the zenon API.

These are added or deleted via the API functions in **DynPictures**:

- `BSTR AddMarker(BSTR bsScreenName, long nElementID, short nPosition, short nLineColorIndex, short nFillColorIndex);`
- `VARIANT_BOOL DelMarker(BSTR bsID);`

The GUID of the marker, which is supplied by AddMarker(), identifies the marker uniquely and serves as both the element name (with the prefix "`$MARKER_`") as well as the key for deletion via DelMarker(). The markers inserted via API are saved in the project according to the screen. **Attention:** Saving is not remanant, i.e. only until Runtime is restarted.

The markers set there are displayed regardless of the monitor on which the screen is opened. The markers are treated internally as normally operable screen elements. Mouse events are called up for this.

The appearance of the markers is set using the project settings in the `Automatic Line Coloring` area of the project configuration:

- Display type of the screen marker: Triangle, circle, square, cross
- Screen marker size: Size in pixels:
- Line width of the screen marker: Width in pixels
- Marker color: is defined via the index in the marker color table (on page 33), that is located in the properties of the screen elements in the `Automatic Line Coloring` group

2.2 Command input

Command input serves primarily for the secured switching of variables in energy technology. 'Secured' means that there is a check whether the switching operation is allowed, according to the configured interlocking condition and the dynamically updated topology (current physical state of the topological network). The configuration of the topology and the topological commands is done via the ALC (Automatic Line Coloring).

A variable point for the command input always consists of 2 physical variables: a response variable and a command variable. Depending on the action to be executed, they are executed on one of the two variables.

Example

`Double command On`: Sends the command/the new value to the command variable. By way of the response variable, the success of the executed action can be checked.

The synchronization of the actions from the command input is done via a communication object that is runtime-monitored and updated cyclically in the driver. This object is automatically assigned to the response variable. The activation of such an object is evaluated in the system by the activation of the "Select" status bit.

> *`Status input off`: Resets all the active states of the response variable; the command variable is not affected by this action.*

Two-step command operations are usually performed via a context menu and the screen type „Command input“. Specific control elements are provided for this. They enable an individual optical and functional design of the command input. Thus you can for example assign individual actions to button **`Actions`**. After that you can access them directly. This screen type also provides the necessary requirements in order to carry out functions such as unlocking, two-step execution, two-hand control, locking etc. Such a screen can be loaded over a function directly at the element (instead of the Set value dialog), or via a context menu.

Whether an action (switching action) at a specific moment is allowed (no interlocking conditions apply) or forbidden (a non-unlockable condition applies) or whether it can only be executed after unlocking (an unlockable condition applies), is determined from the command groups and the current state of the topological model in the Runtime.

PROJECT MANAGER CONTEXT MENU

Menu item	Action
`New command input`	Creates a new command input.
`Export XML all`	Exports all entries as an XML file.
`Import XML`	Imports units from an XML file.
Editor profile	Opens the drop-down list for selecting a Editor profile.
`Help`	Opens online help.

Info

The interlockings can be exported and imported as well as exchanged and duplicated on the interlocking level.

Attention

All the following functions are only available if the 'Energy Edition' was licensed.

2.2.1 Command input detail view toolbar and context menu

CONTEXT MENU COMMAND INPUT AND INTERLOCKING

Menu item	Action
New command group	Creates a new interlocking and opens the dialog for selecting variables.
Export XML all	Exports all entries as an XML file.
Export selected XML	Exports selected entries as an XML file.
Import XML	Imports from an XML file.
Copy	Copies the selected condition.
Paste	Pastes the condition from the clipboard.
Delete	Deletes selected condition.
Rename	Enables the element to be renamed.
Properties	Opens the property window for the selected element.
Help	Opens online help.

CONTEXT MENU GROUP ACTIONS

Menu item	Action
Command new	Creates a new command and opens the properties.
New set value input	Creates a new set value input and opens the properties.
New status input	Creates a new status input and opens the properties.
New replace	Creates a new replace and opens the properties.
New release	Creates a new manual correction and opens the properties.
New manual correction	Creates a new manual correction and opens the properties.
New block	Creates a new block and opens the properties.
New lock	Creates a new lock and opens the properties.
New revision	Creates a new revision and opens the properties.

Paste	Pastes the condition from the clipboard.
Help	Opens online help.

CONTEXT MENU INDIVIDUAL ACTION

Properties	Opens the property window for the selected element.
Menu item	**Action**
Add condition	Creates a new condition.
Copy	Copies the selected condition.
Paste	Pastes the condition from the clipboard.
Delete	Deletes selected condition.
Help	Opens online help.

CONTEXT MENU CONDITION

Menu item	Action
Copy	Copies the selected condition.
Paste	Pastes the condition from the clipboard.
Delete	Deletes selected condition.
Properties	Opens the property window for the selected element.
Help	Opens online help.

CONTEXT MENU GROUP VARIABLES

Menu item	Action
Add variable	Opens the dialog for selecting variables.
Paste	Pastes the condition from the clipboard.
Help	Opens online help.

CONTEXT MENU INDIVIDUAL VARIABLE

Menu item	Action
Linked elements	Opens drop-down list with linked elements.
Copy	Copies selected variable
Paste	Pastes the variables from the clipboard.
Delete	Deletes selected variable.
Properties	Opens the property window for the selected element.
Help	Opens online help.

2.2.2 Engineering in the Editor

You can learn how to configure and execute a command input in the Tutorial Command input.

> **Info**
>
> *Please use the IEC-60870 101/104 driver also for tests if possible. This driver fully supports the COT (cause of transmission) evaluation. This is an extended functionality for communication monitoring. The communication can be evaluated in the* ***multi-numerical*** *and* ***multi-binary*** *reaction matrices.*

Creating a screen of the type command input

The creation of the command input screen in the editor is done by the definition of a new screen of the type **command input**. (You will find more information on the pre-defined screen types in the manual Screens/Pre-defined screen types'.)

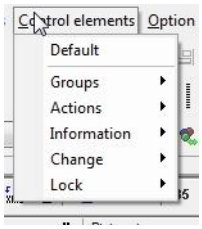
The screen 'Command input' is used for user interaction via command input during the runtime (one and two-step command input) It allows the user to perfrom all activities that are necessary for command execution. This can be e.g. the unlocking of an upcoming command group or the confirmaton of the execution of a two-step command input.

Info

When using the one-step command input, you can also use a context menu or a standard function. The screen type command input is then not required in the project.

You can use specific control elements (on page 115) for this screen type, which allow all user actions necessary for command input and which visualize current information about the status of the action to be executed. (e.g. display of the switching direction)

After the screen is opened an empty screen is displayed. You add the default control elements via menu **`Control elements/Add template`**.



Info

Beneath the groups you can find a collection of control elements which you need for special actions. E.g. all control elements for locking elements . Thus saving the time for manually searching for all needed control elements.

Limits and reaction matrices for switching direction texts

In the first step of the two-step command input, user-definable switching direction texts are displayed (e.g. in the context menu) for actions of type 'Command'. These texts can be defined via the limits or via the states of the reaction matrices.

These texts give the user a better understanding and a better overview of the actions that are available in the Runtime (e.g. 'Command: Open disconnector')

This also allows you to use different texts for every variable that uses the same interlocking, without needing to make adjustments on the user interface.

Info

As the switching direction texts are read out from the limit settings, they are completely language switchable.

STANDARD TEXTS FOR SWITCHING DIRECTION

Limit	Standard text
Off	@OFF
On	@ON
Diff	@INTER
Fault	@FAULT
None	@NONE
Direct	@DIR

Creating variables

Variables must be linked both with 'Switching direction texts (on page 64)' and with the corresponding 'Command groups (on page 92)'. If one of these two settings was not made, no actions are offered for selection during runtime.

Info

Because of the different use of limits/reaction matrices for the command/response variable, individual switching direction texts can be displayed for the actions. Always depending on the variable, on which the action is executed.

Project overlapping variables

> **Attention**
>
> *The variables used in the command groups must be in the same project in order for the command input to work properly.*

If you do use a variable from another project (e.g. subordinate project in multi-project administration), the command group, the response variable, the action variable and the action-specific screen ('command input' screen) is expected to also exist in the other project.

> Info
>
> *You can also use project-overlapping variables for the interlockings by the process. The above limitations apply only to the variables of the command group.*

Define Command

Select the entry **`Command input`** in the project tree and open the context menu. Select `New command group`.

After creating a new command group, it is added to the detail view of the project manager with standard name "Command group + index". The index is replaced by a consecutive number. This name serves for the unique identification of the interlocking in the system.

> **Info**
>
> *You can assign any name you like to the command groups. However, the names must be unique within the project (applies for standard interlockings and command groups).*

The following parameters are available for command groups:

Parameter	Description
Name	Name of the command group. Must be unique among all interlockings in the project. This name is used later with the variable which uses this interlocking. The actual allocation is done with a unique consecutive numerical ID. The name is only used for GUI, Export and Import.
Variable name of response	This is the variable name or the mask for the replacement of the response variable. The placeholder for the replacement text is the character sequence ‚*' within a name. Only one placeholder can be used in a name. When entering the mask, it is important to take care that this name results in an existing variable name after replacement. If the variable that is used here (replaced or absolute) does not exist during compiling, the command group is not available in the RT. An according message announces this error during compiling.
Display of desired direction	If activated, status bit In progress (PROGRESS) is written at actions command and Manual correction. The value that the status bit is set to depends on the switching direction of the action. The status bit is set to 1 if: ▶ the Return state/switching direction of the action is ON or OFF. ▶ The response variable does not already have the value of the set switching direction. The status bit is set during the check of the interlockings and it remains on that value until the screen returns to step 1 or until it is closed. This also implies that the status remains while the watchdog timer and/or the edge delay is active. If the execution of an action is triggered by a context menu or if it is a one-step action, the status bit is set appropriately.
Watchdog timer	There is the following setting for this drop-down list: ▶ none: The watchdog timer (on page 114) is deactivated. ▶ Response variable: The value of the response variable is used to determine if the command was successful. ▶ COT : Watchdog timer in its initial form. Cause of Transmission (COT) is used to determine if the operation was successful.

Screen modal	If this is active, the screen is displayed modally, independent of the setting 'Modal dialog' in the screen settings.
Screen titel from response variable	The identification of the response variable is shown in the screen title. This only happens when there a title was configured for the screen at the frame. Text is online language switchable.
Command screen	Name of the screen to be loaded if no action specific screen is defined and if the screen is not opened via the function 'Screen : Switch to'.
Breaker tripping detection	Only available if property Display of desired direction is activated. Active: Response variable is monitored for an unexpected change from <>0 to 0. With active recognition all variable whose status or value are used in the formula for the breaker tripping detection, are activated for reading at the program start after loading all projects and stay this way as long as the Runtime runs. With this variables of all projects which are loaded in the Runtime can be used independent of the loading order.
Suppress detection	Entering the formula with which the detection of a breaker tripping can be suppressed. A click on button . . . opens the formula editor. For the formula all variables of the interlocking can be used. Name replacements as for example at the definition of the Interlocking condition of an action (on page 68) are also possible. **Note:** Variables which are used in the formula cannot be deleted at the interlocking.

Actions

Command groups always contain a set of predefined actions, which are usually adjusted to a specific variable (a specific device) . For example, different command groups can be defined individually and centrally for different topological elements (switch / disconnector etc.) .

A defined command variable is assigned to every action inside a command group. The response variable is defined centrally for the whole command group.

NAME REPLACEMENT

To simplify and to generalize the definition of the variables, these variable references (for command and response variables) can be defined over a name replacement. At this wild cards can be used. (Notice: Wildcards are only allowed as prefix or suffix; e.g. *xxx or xxx*. With this flexible definition, general interlocking conditions can be defined very simply. With this the number of the command groups which must be defined is reduced considerably.

Example

- Definition of the command variables '*_BE'
- Definition of the response variables '*_RM'

In the Runtime the command input automatically adds the name of the response variable, which is shown/selected in the process screen, to the name of the command variable. The names of both variables differ only in their endings.

ACTION TYPES

The action types are the actually available switching commands. According to the command, different activities are performed.

The system provides a variety of actions. The following action types can be defined for the command groups:

Action type	Comment
Command	
Set point default	
Status default	
Release	Can only be configured once per command group.
Manual correction	
Replace	
Lock	Can only be configured once per command group.

Revision	
Block	Can only be configured once per command group.

In the detail view of the command input the actions in the tree are shown with the corresponding switching direction and at direct write set value with the selected set value.

> **Attention**
>
> *The identification of the action types in the Menu ID must be clear, so that they are clearly identifiable in the context menu (on page 82). If two actions have the same ID, they are tagged with the special symbol M in the action tree.*

ACTION TYPE COMMAND

According to the command type, this is used as 'Single command' or 'Double command' in the system.

When a command is executed, the configured Command status (0 or 1) is written to the command variable. The value which is expected from the response variable as a result of the command is defined under the property Return state/switching direction '(on / off / none)'.

Switching direction	Value of the response variable
None	Will not change
Off	Value will be 0
On	Value will be 1

For single commands, there is an automatic reset of the variable to 0 or 1 (depending on the switching direction), after the engineered 'edge delay'. This does not apply if SBO is activated for the command variable.

> **Info**
>
> *If during the action execution the current value of the response variable is different to the one defined in the switching direction or if the switching direction was defined to be On or Off, the status bit In progress (PROGRESS) is set.*

ACTION TYPE SET POINT INPUT

Offers the possibility to write any numerical value to the selected command variable. The command window offers special control elements for that, which allow the definition of the set value. With the

help of property `Return state/switching direction` you can define how the set value should be written:

Switching direction	Value of the response variable
DIR	Set value is written directly. You define the value which should be written with the help of function `Set value`. The text which should be displayed can be engineered using a limit/rema for the state/value 5. If this is not the case, a standard text (on page 64) is used. Nominal/actual value comparison is not supported. The action button of an action whose value equals the actual value is not locked. The action can be carried out several times in a row.
Set value	Value of the control element `Set value` is written to the response variable.

If set values are set via command input and a response variable is set in the combined element dynamic element, it can be set regardless of the setting of the `Setting values active` property. All action buttons in the command input screen that trigger a direct modification of the response variable are then set to invisible.

Attention

When writing the set value directly neither the limits of the linked variable are checked nor is it checked if the write set value is allowed for this variable.

ACTION TYPE STATUS INPUT

Depending on the definition in the 'Switching direction', the following is executed:

Switching direction	Action
Off	The states configured in the list 'Modifiable states' are all reset to 0.
On	The states configured in the list 'Modifiable states' are all set to 1 (active).
None	The states configured in the list 'Modifiable states' must be defined in the Runtime with the help of the control element 'Set status'. Every status is defined individually.

If you change a status in the Runtime, that change is logged in the Chronological Event List (status incl. value). You have the possibility to switch between languages in the Runtime.

Info

For all status defaults, there is always a write to the response variable.

ACTION TYPE RELEASE

Note: The action can only be executed in Runtime, if the replacement value (ALT_VAL) (value: 1). is active for the selected response variable. 1).

The **Release** actions resets the replacement value (ALT_VAL) status bit to 0 (inactive). If the Switched off (OFF) status bit is also active, it is also set to 0 (inactive). runtime receives the current value from the driver for the response variable once the **Release** action has been carried out.

Note: Can only be configured once per command group.

ACTION TYPE MANUAL CORRECTION

Info

Manual correction means:

Correction of a non-remote-controlled switch in zenon.

A marker variable is usually corrected (no connection to the process). There should never really be an I-bit pending for marker variables. It is possible, but makes no sense, to have a variable with a connection to the process.

Behavior:

Correction is completely normal values setting from the perspective of the driver.

The opposite of that: Replace

The process value of a remote-controlled switch is temporarily replaced with a replacement value (due to revision, for example).

The `correct` action sets the value of the selected response variable according to the setting of the switching direction:

Switching direction	Action
`Off`	0
`Diff`	2
`DIR`	Set value is written directly. You define the value which should be written with the help of function `Set value`. The text which should be displayed can be engineered using a limit/rema for the state/value 5. If this is not the case, a standard text (on page 64) is used. Nominal/actual value comparison is not supported. The action button of an action whose value equals the actual value is not locked. The action can be carried out several times in a row.
`On`	1
`Set value`	Value of the control element `Set value` is written to the response variable.
`Fault`	3

Attention

When writing the set value directly neither the limits of the linked variable are checked nor is it checked if the write set value is allowed for this variable.

Info

The `In progress` (PROGRESS) status bit is set if:

- When the action is carried out, the current value of the response variable is different to the value set for the switching direction
  and
- The switching direction was defined as `on` or `off`

ACTION TYPE REPLACE

The response variable is set to the status alternative value `Alternate value` (ALT_VAL). Additionally, the value defined by the 'Switching direction' is placed on the response variable.

Desired switching direction	Status
`Off`	0
`On`	1
`Diff`	2
`Fault`	3
`None`	4

ACTION TYPE LOCK

Enables the lock of a response variable for the actions of the command input.

Info

If a switch is locked using action `Lock`, status bit M1 is set.

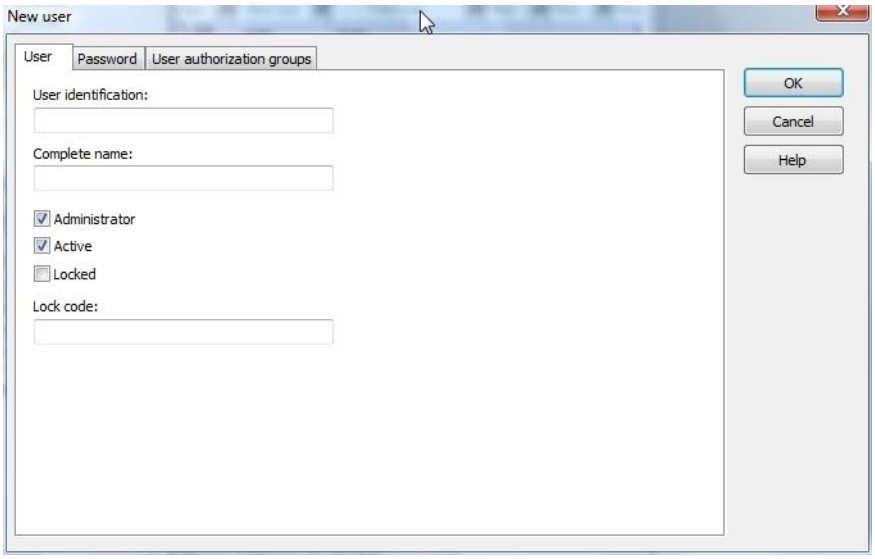
Prerequisites for this is that there are users present in the system who have a configured 'Lock code'. Locking/unlocking a response variable can only be done with the correct input of a 'Lock code'.

The same variable can be locked by multiple users in parallel. Actions for the response variable are possible only after alls locks have been unlocked by entering the lock code.

There can be no actions executed if

- the locked variable is used as an action variable
- actions of the command variable use the locked variable as response variable

The lock code can be defined individually for every user. This parameterization is done directly for an already existing user with the property 'lock code'



You can also set the lock code during Runtime/Online for an existing user.

In the Runtime you cannot delete users who still have an active command lock.

Attention

Users can be deleted in the development environment. This causes the loss of the defined locks after restarting or reloading.

Users locked (deactivated) in the administration cannot activate or deactivate command locks.

Info

Information about active locks are also synchronized in the redundant network and therefore are available after a server switch.

A list of the currently active locks can be shown in the command screen over a special control element ('lock list').

Note: Can only be configured once per command group.

ACTION TYPE REVISION

Sets the status bit Revision of the response variable to the value configured in property `Return state/switching direction`.

Desired switching direction	Status
`Off`	Set to 0
`On`	Set to 1

ACTION TYPE BLOCK

By executing this action, the response variable is switched off.

Sets the status bit OFF of the response variable.

Note: Can only be configured once per command group.

Create action

Actions define the switching commands that are possible for command groups. By selecting the element 'Action' in the detail view of the command group, you can define a new action with a right mouse click. Details about the defined action are also shown in the detail view after creation (e.g. 'Double command: *_BE [ON,1].

All further settings for the actions are made in the properties window. Some of the properties are inactive, depending on the action type.

Available properties:

Property	Description
Variable	Variables on which is written. For some actions, this is the response variable. In this case, the field is locked. The placeholder for the replacement text is the character sequence ‚*' within a name. Only one placeholder can be used in a name. If the variable that is used here (replaced or absolute) does not exist during compiling, the action is not available in the RT. An according message announces this error during compiling. Initialization: No Allocation
Type	Determines the type of the command. Options are: single or double command. Double command Writes the value of the setting 'Command setting status'. Single command: Like a double command, but after the edge delay, a rewrite with the value 0 is performed automatically. This rewrite is no longer runtime-monitored. Locked for all other actions. Initialization: Single command
Desired switching direction	Defines the expected value or the status of the response variable after action execution. Locked for the actions Block, Set value, Lock and Release. Initialization: Off
Command setting status	Defines the value that is written to the command variable during the action 'Command'. Locked for all actions except 'Single command' and 'Double command'. Initialization: 0
Suppress CEL entry	If this is active, no entry in the CEL will be made when executing an action. Initialization: Inactive
Timeout	Timeout used for watchdog timer. Only available for the actions Command and Setpoint input. Unit is seconds

	Initialization: 30
Edge delay	Time in milliseconds, with which the writing of 0 is delayed for a single command. Only available for the action 'Single command'. The system does not wait until the watchdog timer is ended. Initialization: 1000 ms
Modifyable states	List of the states which can be modified with the action 'Set status'. Only available for the action 'Status input' . Initialization: None modifiable
Nominal/actual value comparison	If this is active, there will be a check whether the value of the response variable already matches the switching direction. If this is true, an unlockable interlocking variable is shown. Initialization: Inactive
Menu spec. screen	Screen of type 'Power' which is activated when the action is activated over the context menu of the element. If no screen is entered here, the screen entered for the command group is used. A screen that was configured, but that does not exist, causes an error message during compiling and the action is not taken over. Initialization: No Allocation
Command button	Allocation of the action to an action button in the screen, defined at the command group. If the command group is used for another screen (e.g. via function), the allocation to the action button remains nevertheless. In other words, the action is always placed on the button with the allocated action ID. If such a button is missing, the action is not available in the screen. Only the action buttons that were not allocated yet are provided in the selection list. This setting is locked for the action type "Lock", or if no screen was allocated to the command group. Initialization: No Allocation
Close automatically	If this is active then the screen is closed automatically after action execution. Initialization: Inactive
Two-hand operation	If this is active, the control 'Execute 2' is only active when you hold the key 'Ctrl'. Only available for two-stage execution.

	Initialization: Inactive
`two-stage`	If this is active, an action is executed only after operating the control 'Execute 2'. If not active, the action is executed after releasing the last interlocking or, if there is no upcoming interlocking, immediately. Locked for the action 'Lock'. Initialization: Active

Info

By selecting single properties, you receive additional information about functionality in the Property Help.

Info

Actions and command conditions, once defined, can be exported / imported in XML. This allows for easy archiving and reusing in other applications.

Info

The status can be set using the command `status input`.

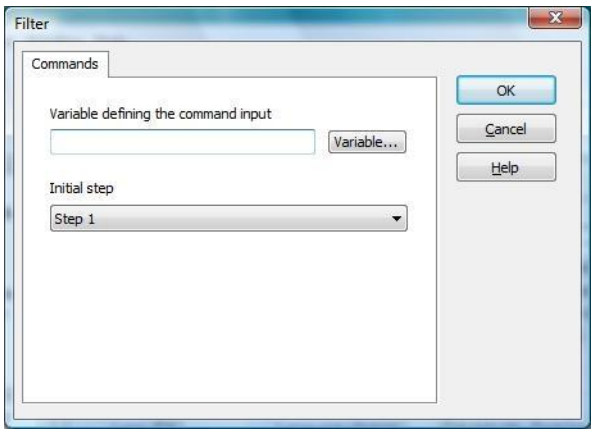
Execute actions

Command input in the Energy Edition can be used in different situations. The user can choose the variant he prefers. It is also possible to use different methods simultaneously (element-related).

1	Calling up a **screen switching** function on a **command** screen.
2	Calling up a **bar graph**, **time**, **universal slider**, **numerical value**, **indicating instrument** or **status element** screen element. To activate, the Set value via element property must be Command input.
3	Calling up a **bar graph**, **time**, **universal slider**, **numerical value**, **indicating instrument** or **status element** screen element. To activate, the Set value via element property must be input field or element (for example, command input as replacement for setting values).
4	Call via a context menu if command input was set for the action type property.
5	Call via a context menu if Set value was set for the action type property.

Screen switch to screen of type Command

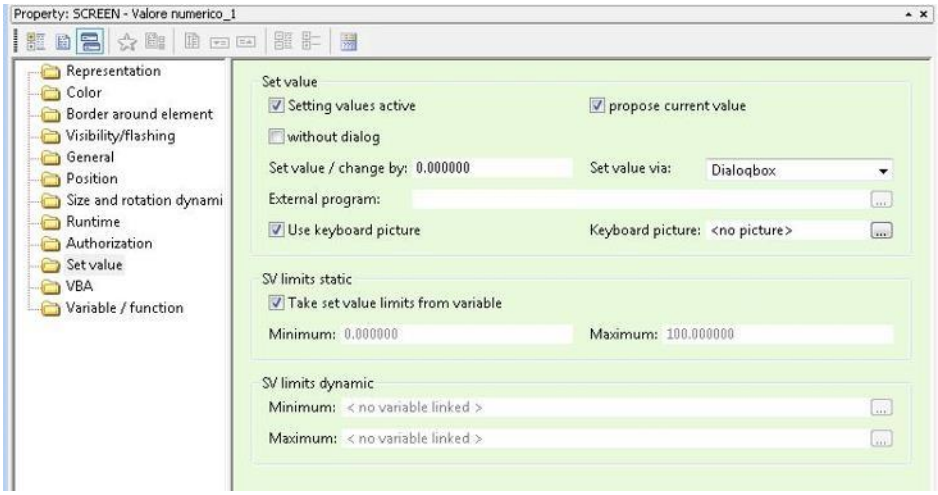
If a screen of type 'Power' is selected during the function 'Screen switch', the following settings page is displayed in the configuration dialog for the function:



Parameter	Description
Variable defining the command input	The variable configured here defines the command group to be used. The screen determines the appropriate response variable and the associated action variable via the name of the selected variable.
Initial step	Defines the step (status) in which the command input screen is loaded. Step 1: The screen is loaded and waits for action definition and action execution. Action executions must be performed manually by the user. Lock: The screen is opened in the command step for the action lock.

Screen element Command input

If the option 'Command input' is defined for the setting 'Set value via' in the properties of a screen element, the command window can be opened during Runtime by left-clicking on the element. The screen to be opened is defined at the command group of the variable linked to the element. The response variable is determined automatically from the command group.



Info

If the option 'Command' was selected, the command window is loaded instead of the 'Set value' dialog.

The following screen element types support Command:

1	Numerical value
2	Text element
3	Bar Graph
4	Indicating instrument
5	Status element
6	Clock
7	Universal slider

If no command group was defined for the variable assigned to the element, or if the response variable of the command group does not exist, an error entry for the diagnosis viewer is generated and the screen is not opened.

Screen element Set value

If the option 'Dialog box' or 'Element' is set for the setting 'Set value via' of an element, and if the variable assigned to the element has a command group, all changes to the set value are also performed via the command group. The following requirements must be met, so that a value can be written:

- The action 'Set value' must be configured for the command group and the resulting action variable is the variable of the selected screen element.
- There must be no command interlocking preventing the execution of the action.

> **Info**
>
> *If one of the above requirements is not met, there is no write and no set value is written to the variable.*

Context menu Command

The command input can also be activated via the context menu of the element. This is the most frequently used method. In this regard, the context menu is already the first step of the two-stage command input.

The menu must have an entry of the **command** action type. The display of the single action is defined automatically by the menu. The display of the actions can be influenced selectively, depending on the '**names**' of the menu entry.

When creating a new action in the command input (on page 58), a menu ID corresponding to the action type and the switching direction for the `Action type` property is created and offered in the drop-down list. If the content corresponds to an ID defined as standard text for the action type and switching direction, the content is adapted if the action type or switching direction change.

To create a context menu for the command input:

1. Create the desired actions in the command input (on page 58)
2. Select `Action type` in the properties of the context menu item as **`command input`**
3. Select the desired action and switching direction via the drop-down menu with the `Menu ID` property
4. Give it a clear label in the `Text` property
   Note: If no entry is defined for `Text`, the field is automatically filled with the "**`command input`**" label.

> **Attention**
>
> *The name of the `Text` property must be unique. If two names the same are issued, further menu items with the same name are not displayed.*
>
> *Because automatically created menu items with the same action result in the same text, there are macros (on page 87) available for these.*
>
> *The character sequence ID_CMD_AUTO is reserved for automatically created menu items. These must always be used with macros, because otherwise only the menu item is inserted.*

When checking for duplicate entries the following rules apply:

- Manual menu points have priority over automatic ones.
- If it is the same type then the last entry has twice the priority.
- If a duplicate entry is found, a warning is set off in the log. This includes the menu ID and description. Automatically expandable entries have **<auto>** added to the ID.

ACTIONS FOR ACTION TYPE COMMAND INPUT

Action	Switching direction	Menu ID
ID_CMD_AUTO		This menu entry automatically shows all possible actions for an element, if no direct menu entry from the list is used already.
Single command	ON (1)	ID_CMD_EBEF_ON
Single command	OFF (1)	ID_CMD_EBEF_OFF
Single command	NONE	ID_CMD_EBEF_NONE
Double command	ON (1)	ID_CMD_DBEF_ON
Double command	OFF (2)	ID_CMD_DBEF_OFF
Double command	NONE	ID_CMD_DBEF_NONE
Set value	NONE	ID_CMD_SVALUE
Set value	DIRECT	ID_CMD_SVALUE_DIR
Status default	NONE	ID_CMD_STATE
Status default	ON (1)	ID_CMD_STATE_ON
Status default	OFF (0)	ID_CMD_STATE_OFF
Replace	NONE	ID_CMD_REPL_NONE
Replace	ON (1)	ID_CMD_REPL_ON
Replace	OFF (0)	ID_CMD_REPL_OFF
Replace	FAULT	ID_CMD_REPL_DEF
Replace	DIFF	ID_CMD_REPL_DIFF
Manual correction	NONE	ID_CMD_UPD_NONE
Manual correction	ON (1)	ID_CMD_UPD_ON
Manual correction	OFF (0)	ID_CMD_UPD_OFF
Manual correction	DIFF	ID_CMD_UPD_DIFF
Manual correction	FAULT	ID_CMD_UPD_DEF
Manual correction	DIRECT	ID_CMD_UPD_DIR
Block	NONE	ID_CMD_BLOCK

Release	NONE	ID_CMD_UNLOCK
Lock	NONE	ID_CMD_LOCK
Revision	OFF (0)	ID_CMD_REV_OFF
Revision	ON (1)	**ID_CMD_REV_ON**

NAME OF THE MENU ITEMS OF THE CONTEXT MENU

1. AUTOMATIC CREATION

Entries which were created via ID_CMD_AUTO automatically receive a name after the following pattern: 'Action name' plus 'Limit text of the switching direction'.

2. MANUAL CREATION FROM TABLE

If the menu entries are created from the table, for every action under 'Display - Text' a text must be defined for the entry in the context menu.

Names for the menu entries:

Command, Set value, Status, Replace, Release, Manual correction, Block, Lock, Revision

ACTION TEXTS

Action	Text
Single command Double command	Text from the limit text, according to the switching direction.
Manual correction Replace	If a switching direction (other than 'None') is defined, the text from the limit text according to the switching direction is displayed.
Status	'OFF' or 'ON', depending on the current switching direction
Revision	Text from the limit text, according to the switching direction.
Others	No special action text is displayed.

Example

Displayed text for a double command with defined limit:

'Command: switching direction ON'

Info

- All displayed texts are language switchable with the standard mechanisms.
  See also: Which texts are language switchable?
- All displayed menu entries are automatically sorted alphabetically.

The currently used command group is determined via the variable which is linked with the screen element. If no command group is assigned to the variable or if there is no response variable, the context menu is not displayed in the Runtime (an according error message is transferred to the diagnosis server).

Info

The menu entries of the command input are displayed depending on the command group. The menu entry is showed only when the connected action exists. Consequently, if the variable of the element is the command variable, only **the** *actions for the command variable plus the action 'Lock' can be displayed. Actions for the response variable are hidden automatically.*

CONDITIONS

The menu entries are only released when the corresponding actions are executable. The following conditions are requirements:

- All menu entries are locked, when the status bit SELECT(10) of the response variable is active.
- All menu entries are locked, when the response variable could not be determined.
- All menu entries are locked, when the response variable has no value and could not get a value within 30 seconds.
- All menu entries are locked on an Internet Client without write access.
- Menu entries are locked when there is no connection to the server.

- The menu entry connected to the action 'Release' is locked, when the status bit ALTERNATEVALUE(27) of the action variable is not active.
- The menu entry connected with the action 'Replace', whose switching direction matches the value of the action variable, is locked.
- All menu entries, except the one which is connected with the action 'Lock', are locked, when a change lock is active for the response variable.
- When the status bit REVISION(9) of the response variable is active, the actions 'Set value', 'Replace', 'Correct', and 'Command' are locked.
- As long as a watchdog timer, an edge generation or an SBO is active for the command group, all menu entries are locked. This results from the fact that the status bit SELECT also stays active.
- The menu entry connected with the action 'Revision', whose switching direction matches the value of the action variable, is locked.

Macros for the context menu

A macro is a defined character sequence that is replaced by another text when menu items are created in Runtime. Virtually all macros can occur more than once per menu item. They can also contain further macros as a result. In doing so, the expansion sequence must be considered. Macros are not case sensitive when configuring menus. If macros contain a macro as a result, the macro must be contained in capitals in the result. The entry is made with `$` as a prefix and suffix.

The sequence of the expansion is from left to right in the following priority.

1. $NOTE$
2. TAG
3. $REMA<Status>$
4. $RDIR$
5. ALL
6. DIR
7. ACT
8. $NOTE$

Macro	Description
$NOTE$	The whole text including the macro is interpreted as a note. If the resulting text is empty, the **ALL** macro is used.
TAG	Is replaced by the identification of the action variable. The identification can be translated by the online language translation function. If no translation character (@) is contained, the whole identification is highlighted for translation.
$REMA<Status>$	**<Status>** is a Rema or limit value state, the text of which is used as a replacement. If the status is not present, the menu item is not displayed. The limit value text is translated linguistically according to the placement of @ . The status can be a number between -2^{31} and 2^{31-1}. Leading characters and a prefix are permitted. If characters are contained that cannot be converted to a number, or the number is outside the given area, the menu item is not displayed.
$RDIR$	Text for the switching direction from reaction matrix/limit value as in **DIR macro, with the exception of:** ▸ Action `Write set value direct` The text is taken from the rema/limit value of the status, which corresponds to the value of the set point to be set. ▸ Action `Status on` and `Status off` Text is taken from the rema/limit value for the `on` or `off` statuses. ▸ Action `Correct direct` The text is taken from the rema/limit value of the status, which corresponds to the value of the set point to be set.
ALL	Results in **Action naming**: **Switching direction**. Corresponds to the combination of the **ACT:** macro **DIR**
DIR	Switching direction of the action.
ACT	Action naming of the action.
$NOTE$	For the last macro, the note macro is again checked and the text to the right of this including the macro is deleted. If the resulting text is empty or only consists of spaces, the menu item is not

	inserted.

AUTOMATICALLY CREATED MENU ITEMS

Automatically created menu items are created as a menu ID with ID_CMD_AUTO. In this case, macros must always be used, because otherwise only a menu item would be inserted.

COMPATIBILITY

Previous to version 6.51 text at automatic menu items was ignored. When converting projects that were created with versions earlier than 6.51, the macros $ALL$$NOTE$ are automatically inserted before the configured text. Therefore the menu items behave as before.

ONLINE LANGUAGE SWITCH

The labeling for the menu item in the `Text` property is translated linguistically before macro expansion from the character `@`.

Note: If, for the **`$TAGS$`** macro, no translation indicator (`@`) is contained, the complete text is translated.

Error messages

When menus are loaded in the runtime environment, their content is checked for consistency. Errors cause error messages for the online diagnosis tool. The following messages can appear:

Parameter	Description
Menu entry for command input suppressed, because name is several times in the menu!	The menu already contains a menu entry with the name used in the command input. Do not use that name for any other menu entries.
Menu entry for command input suppressed, because description is several times in the menu!	There is already a menu entry with the same description in the menu. Automatically created menu entries are not added, when a menu entry with the same description is already there.
Text for menu entry cannot be detected!	The description of an automatically created menu entry could not be determined. This most probably indicates a missing limit text.
No command group linked to variable of the screen element!	The variable associated with the screen element has no command group or a no longer valid command group. According error messages are given during compiling.
Response variable does not exist!	The response variable used in the command group does not exist.
Select cannot be activated!	Statusbit SELECT(10) could not be activated within the timeout.

Executing actions

After activating a menu entry for command input, the associated action is executed. Execution over the menu causes setting of the Status bit 'SELECT'. Only if this was successful, the execution of the actual action is started (e.g. started 'double command').

After that the command windows opens if one of the following criteria are fulfilled:

- If the action to be executed is 'Set value', 'Set status' with input or 'Manual Correction', the screen associated to the action is opened in 'Step 1'. The desired value / status can then be defined in the screen.
- If the action to be executed is 'Lock', the action-specific screen is called in the step 'Lock'
- If an upcoming interlocking condition prevents the execution, the screen engineered for the action is called in the step 'Unlocking'. This also happens if the SBO could not be activated without errors.
- If a two-layered execution is engineered for the action, the action-specific screen is called in 'Step 2'.

- If no specific screen was engineered for the action, the screen that was engineered centrally for the command group is opened.

The internal Select and the SBO are passed on to the screen.

> **Info**
>
> *If none of the above conditions apply, the action is executed immediately, without any further user actions.*

Context menu Set value

If the variable associated to a screen element is linked to a command group, the writing of a set value is also handled via the command input. For this, an action 'Set value' must present for the command input. If this action is missing, the setting of the set value cannot be performed.

> **Info**
>
> *An upcoming interlocking condition prevents the setting of a set value.*

Command conditions

Command groups contain the definition of the switching actions as well as the definition of the command conditions. Command conditions are optional parameters that can be defined application-specifically.

Every action within a command group can be extended with 'Command conditions'. These process-controlled interlockings prevent an unwanted execution of actions, depending on the current process status.

The command groups consist of 3 essential parts:

Parameters	Description
Actions	These define which command is executed and on which variables these actions should be applied and they parameterize the internal interlockings.
Condition variables	These define which variables can be used in the command conditions.
Command conditions	These conditions make the execution of commands dependent on the current process status.

See also

Interlockings

Define command conditions

Any number of command conditions can be defined for every action. These conditions allow for an additional restriction of the executability of an action. These conditions are defined with formulas in which you can use the variables from the active projects. The formula addresses the linked variable via the index in the condition.

Info

The condition variable is automatically replaced if you use a '' in the definition.*

Define condition variable

First we have to define variables which can be used later for the formulas of the command conditions. If the defined conditions are fulfilled by the linked process variables later, during Runtime, the user has the respective actions available.

Info

If a variable which was used in a command condition is deleted later on, the index within the condition is adjusted. The succeeding variables are put forward and the formulas are adjusted automatically.

The following procedure is recommended for defining a command condition:

Example

1. Select the node Variable in the detail view of the command input and select the option 'New' in the context menu.
2. In the selection dialog, select a process variable, which serves as the base for the formulas of the command conditions. You can also abort the variable selection dialog, which leads to an empty definition. You can define an automatic replacement for this empty link with a '*'.
3. Select an already existing action and the node 'Conditions' in the detail view. With 'Condition new', you can define any number of conditions for every action. The definition is not performed with formulas; rather, non-fulfilled conditions cause a lock of the associated actions in the Runtime.

All further settings (e.g. the allocation of an interlocking to a variable) can be made in the properties window. The properties are described in detail in the properties help section of the Editor.

Interlocking condition

Any number of command conditions can be defined for every action. Theses conditions are checked before execution of the respective action. If a check fails, the respective action cannot be used during runtime.

The conditions are defined as formulas. The syntax is analogous to the definition of the formulas in the Formula Editor.

Additionally, the following interlocking types (in addition to the command groups) are checked before action execution:

INTERNAL INTERLOCKING CONDITIONS

These conditions are checked automatically before every action execution; the engineer cannot influence this. These Internal interlocking conditions (on page 94) are predefined by the system and serve as plausibility checks.

Example

Select could not be activated.

TOPOLOGICAL INTERLOCKING CONDITIONS

These conditions result from the current topological status during Runtime. The definition of these conditions is done in the 'Configuration of the topological interlockings (on page 30)' settings.

Internal interlocking conditions

With the help of the internal interlocking conditions the basic requirements for the action are checked (plausibility check). The results or the activation of an interlocking are displayed during runtime in the command input window as interlocking condition.

Parameter	Description
Status already exists	The state which should be set equals the current value of the response variable. This check is only active if the command group of the 'Nominal/actual value comparison' is active. This interlocking is unlockable.
Internal error occurred	Command cannot execute the check. This happens when the data type of the action variable is not allowed for this action. Example: Action 'Single command On' for string variables This interlocking is not unlockable.
no interlocking object	Command group could not be determined. This interlocking is not unlockable.
Action not defined	Action to be executed could not be determined. This interlocking is not unlockable.
Differences between local and global interlocking	Single command parameter not consistent. Parameterizing error This interlocking is not unlockable.
One or more values are not available	Value of condition variable not available. Lock code: 14 Value of condition variable disturbed. Lock code: 15 This interlocking is not unlockable.
Locking administration not valid	The administration of the lockings could not be loaded or is invalid. This interlocking is not unlockable.
Variable locked for changes	Command locked by response variable (congestion). This interlocking is not unlockable.
SBO rejected	The activation of the Select was rejected by the PLC. This interlocking is not unlockable.
Timeout for SBO activation	No confirmation for the activation (positive or negative) was received within the timeout. This interlocking is not unlockable.
Timeout for SBO deactivation	No confirmation for the deactivation (positive or negative) was received within the timeout. This interlocking is not unlockable.
Timeout for execution	There was no notice for finishing the action execution within the timeout. This interlocking is not unlockable.
SBO expired	The PLC has reported the expiration of the SBO activation.

	This interlocking is not unlockable.

Command input in Distributed Engineering

> **Info**
>
> *Because the command conditions and the general interlockings (standard functionality / without energy edition) are deposited in the same structure in the editor, the checkout symbol is set to identical for both nodes in the project tree (enable changes). All actions on the command conditions also apply to the general interlockings and vice versa.*

Variables marked for deletion are considered as not existent for the compilation of the command conditions. During compiling, the respective error messages are displayed in the output window.

Formula editor

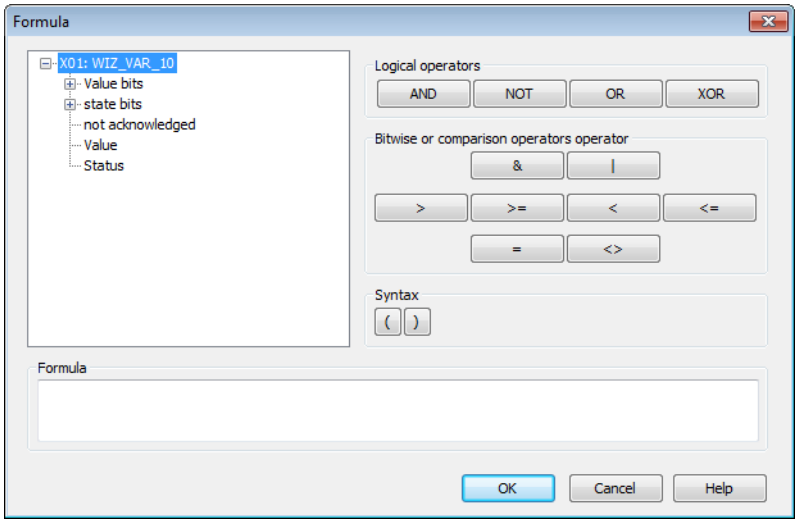
The formula editor provides support when creating formulas with logical or comparative operators with a combined element, for interlockings and commands. If additional variables are required for a formula, create these in the **formula variables** area of the status window by clicking on the **Add** button. existing formulas are displayed in the status list with the letters**F** .

Note on the input of decimal points:

- Decimal separator: Comma (**,**) is automatically converted into a dot (**.**):
- Zero as a decimal point is removed automatically; 23,000 automatically becomes 23

CREATING A FORMULA

Click on the **Formula** button in the status window The formula editor opens



You select the bits for your formula in the left screen.

On the right, you find the operators for logical and comparative operations.

The formula created is displayed in the **Formula** area.

Info

Up to 99 variables can be linked in one formula. X01 to X99. The length of the formula may not exceed 4096 characters.

THE MEANING OF THE BITS:

Parameter	Description
Value bits	32 value bits (von 0 -31) are available. They describe the variable value bit by bit. For binary variables, only bit 0 is of importance, for SINT and USINT only the bits from 0–7, etc. Note: The value refers to the raw value (signal range) of the variables and not to the converted measuring range.
State bits	Here you find the most commonly used status bits. You find the exact definition and use of the status bits in the Status Bits List (on page 100).
unreceipted	Not acknowledged is treated like a usual status bit. But here it is listed separately, because it does not belong to the classical variable statuses.
value and status	In the formulas, all values (value bits and status bits) are treated as binary values and can be logically linked with AND, OR, etc. The total value and overall status are an exception to this. In order to get a Boolean result this total value has to be ORed with a constant bitwise (on page 104). For this, we use the operator &. For the result 0 (false) of this logical ORing we get the binary value 0 (false), otherwise 1 (true). Example: see chapter Example bit by bit ORing (on page 104)

Info

The status bits NORM and N_NORM are only available in the formula editor and cannot be engineered via the status.

If other settings outside the formula are set for the current status, they are combined with the formula with a logical AND.

Refer to the examples (on page 105) section for examples.

Info

Formulas with binary x values and bit-by-bit link can be used with a maximum of 2 binary values. If more values are necessary the link must be done without binary x values.

Example:

`X01.Value & X02.Value` -> works

`X01.Value & X02.Value & X03.Value` -> does not work

But:

`X01.00 AND X02.00 AND X03.00 AND X04.00 AND X05.00` -> works

List of status bits

Bit number	Short term	Long name	zenon Logic label
0	M1	User status1	_VSB_ST_M1
1	M2	User status2	_VSB_ST_M2
2	M3	User status3	_VSB_ST_M3
3	M4	User status4	_VSB_ST_M4
4	M5	User status5	_VSB_ST_M5
5	M6	User status6	_VSB_ST_M6
6	M7	User status7	_VSB_ST_M7
7	M8	User status8	_VSB_ST_M8
8	NET_SEL	Select in the network	_VSB_SELEC
9	REVISION	Revision	_VSB_REV
10	PROGRESS	In operation	_VSB_DIREC
11	TIMEOUT	Runtime exceedance	_VSB_RTE
12	MAN_VAL	Manual value	_VSB_MVALUE
13	M14	User status14	_VSB_ST_14
14	M15	User status15	_VSB_ST_15
15	M16	User status16	_VSB_ST_16
16	GI	General query	_VSB_GR
17	SPONT	Spontaneous	_VSB_SPONT
18	INVALID	Invalid	_VSB_I_BIT
19	T_CHG_A	Daylight saving time/winter time announcement	_VSB_SUWI
20	OFF	Switched off	_VSB_N_UPD
21	T_EXTERN	Real time external	_VSB_RT_E
22	T_INTERN	Realtime internal	_VSB_RT_I
23	N_SORTAB	Not sortable	_VSB_NSORT
24	FM_TR	Error message transformer value	_VSB_DM_TR

25	RM_TR	Working message transformer value	_VSB_RM_TR
26	INFO	Information for the variable	_VSB_INFO
27	ALT_VAL	Alternate value If no value was transferred, the defined alternate value is used otherwise the last valid value is used.	_VSB_AVALUE
28	RES28	Reserved for internal use (alarm flashing)	_VSB_RES28
29	N_UPDATE	Not updated	_VSB_ACTUAL
30	T_STD	Standard time	_VSB_WINTER
31	RES31	Reserved for internal use (alarm flashing)	_VSB_RES31
32	COT0	Cause of transmission bit 1	_VSB_TCB0
33	COT1	Cause of transmission bit 2	_VSB_TCB1
34	COT2	Cause of transmission bit 3	_VSB_TCB2
35	COT3	Cause of transmission bit 4	_VSB_TCB3
36	COT4	Cause of transmission bit 5	_VSB_TCB4
37	COT5	Cause of transmission bit 6	_VSB_TCB5
38	N_CONF	Negative acceptance of Select by device (IEC 60870)	_VSB_PN_BIT
39	TEST	Test bit (IEC870 [T])	_VSB_T_BIT
40	WR_ACK	Writing acknowledged	_VSB_WR_ACK
41	WR_SUC	Writing successful	_VSB_WR_SUC
42	NORM	Normal status	_VSB_NORM
43	N_NORM	Deviation normal status	_VSB_ABNORM
44	BL_870	IEC 60870 Status: blocked	_VSB_BL_BIT
45	SB_870	IEC 60870 Status: substituted	_VSB_SP_BIT
46	NT_870	IEC 60870 Status: not topical	_VSB_NT_BIT

47	OV_870	IEC 60870 Status: overflow	_VSB_OV_BIT
48	SE_870	IEC 60870 Status: select	_VSB_SE_BIT
49	T_INVAL	Time invalid	not defined
50	CB_TRIP	Breaker tripping detected	not defined
51	CB_TR_I	Breaker tripping detection inactive	not defined
52	RES52	reserved	not defined
53	RES53	reserved	not defined
54	RES54	reserved	not defined
55	RES55	reserved	not defined
56	RES56	reserved	not defined
57	RES57	reserved	not defined
58	RES58	reserved	not defined
59	RES59	reserved	not defined
60	RES60	reserved	not defined
61	RES61	reserved	not defined
62	RES62	reserved	not defined
63	RES63	reserved	not defined

Info

In formulas all status bits are available. For other use the availability can be reduced.

You can read details on status processing in the Status processing chapter.

Logical Operators

Logical links: Variables will only be checked for the logical value '0'; if the value does not equal '0', it will be considered as '1'.

In contrast to bit formulas, the technical range can be modified by a stretch factor -> (not equal '0' or '1').

Operator	Meaning
AND	logical 'AND'
NOT	Negation
OR	logical 'OR'
XOR	logical 'EXCLUSIVE OR'

The operators have the following priority in the formula calculation:

Priority	Operator
1	& (operator for bit formulas (on page 103))
2	NOT
3	AND
4	XOR/OR

Info

Up to 99 variables can be linked in one formula. X01 to X99.

Info

The status bits NORM and N_NORM are only available in the formula editor and cannot be engineered via the status.

Bit formulas

Bit formulas only have a logical high or low state. In contrast to logical formulas, the raw value is already predefined (0,1).

Operator	Description
&	AND
|	OR

Example: ORing bitwise

You want to find out if one of the user status bits 1-8 (M1 ... M8) of the variable X01 is set.

USUAL FORMULA:

```
X01.M1 OR X01.M2 OR X01.M3 OR X01.M4 OR X01.M5 OR X01.M6 OR X01.M7 OR X01.M8
```

This query can be made much easier by the logical ORing of the overall status.

LOGICAL ORING:

```
X01.Status & 0xFF
```

The constant can be entered in hexadecimals, as described above:

`0xFF` corresponds to decimal `256`; these are the first eight status bits (binary 11111111). If one of these bit is set to `1`, the result of this bitwise ORing is `1` (true), otherwise it is `0` (false).

If, for example, all user status bits except the user status bit M7 should be queried, the binary statement for this would be: 10111111. Bit 7 is not of interest and is thus set to `0`. This corresponds to 0xBF in hexadecimal. The expression for the formula is then: **`X01.Status & 0xBF`**.

Instead of ORing bitwise with a constant, the value can also be directly compared to a decimal number. If the comparison is wrong, the binary value is `0` (false) otherwise it is `1` (true).

Example:

You want to find out if the value is equal to the constant `202`: The formula is:

```
X01.value = 202
```

If the value is equal to the constant `202`, the result of the comparison is `1` (true) otherwise it is `0` (false).

Note: The bitwise ORing works with the OR character (|) in a similar manner to this example.

Comparison operators

Comparison operators serve for the direct comparison of two numeric values. The result of this comparison is a binary value. "`0`“ if the condition is not fulfilled and „`1`“ if the condition is fulfilled.

Operator	Description
<	less
>	greater
<=	Less then or equal
>=	Greater than or equal
=	Equal
<>	unequal

To the left and to the right of the comparison operator, there has to be a (total) value or a (total) status, single bits cannot be used with these comparison operators.

There can also be a constant to the right of the comparison operator. (the constants can only be integers; a comparison to a floating point number is not possible.)
These constants are entered as hexadecimal values or decimal values in the combined element. Hexadecimal figures are automatically converted to decimal values by clicking on ox (for example, 0x64 is in decimal figures 100).

Example

X01.value >= X02.value
The result is 1, if the value of X01 is higher than or equal to the value of X02

X01.value = 0x64
The result is 1, if the value of X01 is exactly equal to the numeric value 100 (= hex 0x64)

(X01.value = 0x64) OR (X01.value = 0x65)
The result is 1, if the value of X01 is exactly equal to the numeric value 100 or 101 (= hex 0x64 and hex 0x65)

Examples for formulas

SIMPLE LOGICAL AND LINKING BETWEEN TWO BIT VALUES

Example

Formula: X01.03 AND X02.03

This formula has the status TRUE, if both **bit 3** of variable 1 and **bit 3** of variable 2 both have the value 1.

COMPARISON OF AN ANALOGUE VALUE OR STATUS OF A VARIABLE

Example

(X01.Value> X02.Value)

COMPARE ANALOG VALUES WITH EACH OTHER ON A LOGICAL BASIS

Example

(X01.Value> X02.Value) AND (X01.Value = X02.Value)

COMPARE WITH VALUE BITS AND STATUS BITS

Example

(X01.Value> X02.Value) AND (X01.Value = X02.Value) OR (X01.03 = X02.03)

COMPARE A VALUE WITH A DECIMAL OR HEXADECIMAL VALUE

Example

Formula: (X01.Value = 111)

Formula: (X01.Value = 0x6F)

If a hexadecimal values is used, this is later transferred to decimal by clicking on OK. If a decimal value is entered and confirmed, the value continues to be displayed as a decimal value after reopening.

Info

It is not possible to use a comma or a period when entering values.

Create menu

Command input can also be activated via a context menu. Context menus are created in the Editor using node **Menus** and are defined in the properties of the element they concern.

Generally there are three types of menu entries:

Parameters	Description
Action type	Sets out which type of action is to be carried out via the corresponding menu item in Runtime. Not all action types are available in the main menu, some are only available via the context menu. ▸ Acknowledge alarm (context menu only) ▸ Command input (context menu only) ▸ Acknowledge flashing (context menu only) ▸ Function ▸ Help ▸ No action ▸ Send value to hardware ▸ VBA macro (context menu only)
Submenu	Opens a sub-menu in Runtime.
Separator	A horizontal line divides menu entries.

Underline text: Entering a & causes the following characters to be displayed as underlined.

Plan entries

To configure a menu item in the main menu or context menu:

1. Activate the corresponding menu cell
2. In properties, select:
   - Action type: depending on menu type
     see also: Main menu action types and Context menu action types
   - Menu ID: Entry ID
     Note: There are pre-defined types with a fixed ID available in the command input at Command input

- `Text`: clear labeling of the menu cells

> **Attention**
>
> *The name of the `Text` property must be unique. If two names the same are issued, further menu items with the same name are not displayed.*

You can find details on the definition on context menus for command in chapter menusCommand.

Import and Export

> **See also**
>
> *See chapter Import and Export / Command groups.*

> **Info**
>
> **COPY TO CLIPBOARD**
>
> With the clipboard, you can only copy and paste on the level of the complete command group. You cannot copy single actions.
>
> *Existing command groups get a new name when pasted.*

Create Runtime files - start the Runtime

When creating Runtime files for the command groups, a check for engineering errors and correct replacement is performed.

For every variable which has a command group allocated to it, a specific version of the command group for zenon is created. This version only contains **these** actions which can be triggered over this variable.

Example

The command group for the command variable only has actions on this command variable anymore. Except for the action 'Lock'. This action is also available for the command variable.

Info

The compiling of the command input must also be triggered after changes to the variables.

Replacing links

To raise the reusability of the command group, it is possible to replace variable references. Replacement is possible for response, command and condition variables.

When replacing, the placeholder '*' is automatically replaced by name of the variable associated with the command input.

Example

Suppose we have the variables xyz_RM, abcRM and bool_RM.

Our mask is '*_RM'.

Variable name	Replacement text	Result	Comment
xyz_RM	xyz	xyz_RM	Variable exists, allocation successful
abcRM	<empty>	_RM	The masks is not correct, because _ is missing. Variable does not exist.
bool_RM	bool	bool_RM	Variable exists, allocation successful

Info

If the variable for which the command group is to be compiled, matches the response variable, Thus the replacement text is determined by the response variable. Otherwise, the replacement text is determined by the action variable oft the first action that fits the variable.

If the replacement text was successfully determined, the place holder '' is replaced by this text.*

This is why the following points should be considered for the names.

- The names of the variables and the mask should allow for a clear allocation.
- The names of the variables used for the response/command/condition variables, should be producible over the same replacement text.
- If the response variable is replaced but the command variable is not, it is important to consider that the command group created for the command variable has to use the expected response variable.
- An additional test run makes sure that the command group of the response variable contains only actions with action variables using the same response variable in their compiled interlocking. Actions violating this rule will cause a warning and will be removed.

Error while creating Runtime data

At the creation of the Runtime data for the command input, an extensive validation is carried out concerning wrong engineering and not-available references.

Info

After an `Error` the object the caused the error is not available during runtime.

If the command group has an `Error`, no command group is assigned to the variable. Consequently, during the Runtime, all user operations are locked.

A `Warning` is generated when the project would cause a problem but runs error-free.

In the error messages, the following placeholders are used:

<VERNAME>	Placeholder is replaced in the error message by the name of the command group.
<VERRM>	Placeholder is replaced in the error message by the name of the response variable.
<AUFVAR>	Placeholder is replaced in the error message by the name of the variable to which the command group is assigned.
<ACTVAR>	Placeholder is replaced in the error message by the name of the variable of the action.
<Actionname>	Placeholder is replaced in the error message by the description of the action.
<VARNAME>	Placeholder is replaced by the variable in the visualization.

The following error messages can occur during the creation of the Runtime files:

Message text	Description
<VERNAME>: Interlocking PV <VERRM> does not exist!	Condition variable for general interlocking not available.
Variable '<AUFVAR>' uses not existing command input!	Variable uses a non-existing command input.
(<AUFVAR>) command input '<VERNAME>' contains no actions!	Command groups without action are not considered by the Runtime. This message can also be a follow-up error.
(<AUFVAR>) response variable '<VARRM>' does not use the command group '<VERNAME>'	A response variable using another command group is used. The response variable always has to be linked with the interlocking, which uses it as response variable.
(<AUFVAR>) response varibale '<VARRM>' for command '<VERNAME>' uses a driver without process linking!	The response variable must lie on a driver with process connection.
(<AUFVAR>Command '<VERNAME>' contains no actions after compiling!	A command group without actions does not make sense.
(<AUFVAR>) response variable '<VARRM>' of command '<VERNAME>' not available!	The used response variable is not present or marked as deleted.
(<AUFVAR>) command '<VERNAME>' uses screen '<Bild GUID>'(<BILDNAME>) which is not of the type Power!	This message is a warning. If a user action becomes necessary during execution, it cannot be performed.
(<AUFVAR>) command '<VERNAME>' uses not available screen '<Bild GUID>'!	The screen assigned to the command group does not exist.
(<AUFVAR>) Replaced action variable '<ACTVAR>' for action '<Action name>' of command '<VERNAME>' not available!	The action variable, after a replacement, is not present or marked as deleted.
(<AUFVAR>) action '<actionname>' of the command input '<VERNAME>' uses the not existing variable '<ACTVAR>'	The action uses a varibale which is not present in the project or marked as deleted.
(<AUFVAR>) action variable '<ACTVAR>' for action '<Actionname>' of command '<VERNAME>' uses a driver without process connection!	The variable assigned to an action must not lie on an internal driver.
<VERNAME>(<AUFVAR>): Aktion '<Actionname>' already exists!	The following actions may only be configured once per action variable and command group: Double command with the same command status.

	Correction with the same switching direction. Replacing with the same switching direction. Revision with the same switching direction. Single command Set point default Release Block Lock
<VERNAME>(<AUFVAR>): Action '<Actionname>': Single and double command for the same command variable not possible!	Single and double command must not be used in parallel.
(<AUFVAR>) Action '<Actionname>' of the command input '<VERNAME>' uses screen '<Bild GUID>('<Bildname>') which is not of type Power!	This message is a warning. No user actions will be possible.
(<AUFVAR>) Action '<Actionname>' of the command input '<VERNAME>' uses not existing screen '<Bild GUID>'!	The action is assigned to a non-exising screen.
<VERNAME>(<AUFVAR>): Interlocking PV '<VARNAME>' does not exist!	Replaced condition variable does not exist.
(<AUFVAR>) Variable '<VARNAME>' of action interlocking condition of the command input '<VERNAME>' does not exist!	Variable of the interlocking condition does not exist.
(<AUFVAR>) action variable '<VARNAME>' for action '<Actionname>' of command group '<VERNAME>' uses another command group!	The action variables used for a command group may only be connected to no command group or to the command group in which they are used.
(<AUFVAR>) command variable <ACTVAR> does not have a validly compiled interlocking! Action <Actionname> removed.	The action variable used in the action has no compiled command group. This message can also be a follow-up error.
(<AUFVAR>) command variable <ACTVAR> uses response variable <VARNAME>! Action <Actionname> removed.	The compiled command group of the response variable contains actions with action variables which do not use the same response variable as <AUFVAR>. Note: There must not be any actions of response variables changing other response variables.

2.2.3 Operating during Runtime

A watchdog timer is automatically carried out in the background if a used enters commands in Runtime.

Cause of Transmission (COT) process

The Cause of Transmission (COT) informs zenon whether the variable can be written to and whether the writing was successful. The action variable receives a COT corresponding to the level of the command. In the background, the command module then checks to see if the response variable changes its value and the COT according to the command.

The process in the background:

1. The value and COT_act (6) are sent to the action variable.
2. The PROGRESS status bit is sent to the response variable.
3. If the PLC receives the value COT_act, the subsequent value COT_actcon (7) or COT_actterm (10) is awaited.
4. End of the process:
   a) The process is ended if both conditions have been met:
   - COT_actterm was received

   and

   - The value of the response variables corresponds to the switching direction (`Return state/switching direction` property).
     It does not matter which of the two conditions is met first. As soon as both of them are fulfilled, the procedure will be terminated.

   a) If only one or none of the above conditions from item 5a is met within the configured timeout, then:
   - the process is ended and will be terminated and
   - the TIMEOUT status bit of the response variable is activated.
5. The status bit PROGRESS wird zurückgesetzt.

Note: Value changes of the response variable will only be accepted after receiving COT_act.

Info

COT can be evaluated in Runtime - just like all other status bits - using multi-numeric or multi-binary reaction matrices.

Note: *COT is supported not only by IEC870, but also by some other Energy drivers - different versions thereof. Some drivers support COT although the protocol itself does not contain COT (e.g. DNP3). You can find details in the corresponding driver documentation.*

Command input screen control elements

The following control elements are available in Runtime:

Name	Control type		Default
Action buttons	Text	Buttons, which can have an action assigned to them. By clicking in the screen, the assigned action is activated and the screen changes to the step "Release" The button is not shown when: - No action is assigned to the button in the current command group. - The variable, with which the screen was loaded, is the command variable, and the action assigned to the button does not use the command variable as action variable. However, if the action 'Lock' was assigned to the button, it is visible. The button is shown as locked when: - The screen is not in 'Step 1'. - The response variable has set one of the status bits I_KENNUNG(18), OFF(20) or NICHT_AKTUELL(29) and writes the assigned action to the command variable. - The response variable has the status REVISION(9) active and the assigned action writes to the command variable. - The response variable has the status REVISION(9) active and the assigned action is 'Correct'. - The assigned action is 'Release' and the response variable does not have the status Alternativevalue(27) active. - The assigned action is 'Correct' and the value of the response variable matches the switching direction. - The assigned action is 'Replace' and the value of the response variable matches the switching direction. - The response variable has the status	Action1 Action2

		REVISION(9) active and the assigned action is 'Replace'. - The assigned action is 'Revision' and the value of the response variable matches the switching direction.	
RV TTA	Text	Name of the response variable	X
RV identification	Text	Name of the response variable	X
Action variable unit	Text	Unit of the current action variable.	X
Action variable set status	List	Defines the status to be set for the action 'Status default' for the switching direction 'None'. The statuses are set to the current status and updated when changes occur. Is locked when the active action is not 'Set status'.	
Switching direction	Text	The switching direction configured for the active action. The texts are documented with the setting 'Switching direction'. Depending on the active action, the following text is shown: Command, revision, correction, replace: Text from limit value, depending on switching direction. Status: On or Off Other: empty	X
Execute Step 2	Button	Delivers the actions to execution. This control is visible only when the screen is in 'Step 2'. The Control is locked when: - Two handed operation was configured and the key 'Ctrl' is not pressed. - The status REVISION(9) of the response variable is set and the assigned action is 'Command', 'Set value', 'Replace' or	X

		'Correct'. The button was already clicked.	
Action variable minimum	Numerical	Minimum value of the action variable. Not visible if the action variable is of data type 'String'.	
Action variable maximum	Numerical	Minimum value of the action variable. Not visible if the action variable is of data type 'String'.	
Scrollbars	Numerical	Setpoint input with scroll bar Sets the value in the control 'Set value' or is set by this value. Not visible if the action variable is of data type 'String'. The Control is locked when: - No action is active. - The screen is not in 'Step 1'.	
Set value	Numerical, Text	Allows the input of the set value. By clicking the Control, it is switched to edit mode and the setpoint input is possible. The edit mode can be left again with "Enter". The new value is set only after clicking the control 'Execute'. The desired value for the action 'Set value' is provided with this control. The Control is locked when: - The status REVISION(9) of the response variable is set. - No action is active. - The screen is not in 'Step 1'.	
RV value	Text	Value of the response variable	X

RV status	Text	Contains the status of the response variable in the short form.	X
RV unit	Text	Unit of the response variable	X
Interlocking text	Text	Text of the upcoming interlocking. Text is online language switchable	X
Unlocking		If an unlockable interlocking is upcoming, it can be unlocked with this button. This control is shown only when the screen is in the step 'Unlock'. The Control is locked when the upcoming interlocking is not unlockable.	X
Exit	Button	Closes the screen without action execution. The button is only visible in a modal screen. This button is important for modal screens, because it is required to leave the screen in case of an error!	
Cancel	Button	Aborts the execution of the command input and returns to 'Step 1'. The button is locked when the screen is in 'Step 1'.	X
Lock list	List	Contains the locks that were activated at the response variable. Is locked when no action 'Lock' was configured for the command group. Text is online language switchable	
User identification	Input field	For entering the user identification for the lock. Is locked when no action 'Lock' was configured for the command group.	
Lock code	Input field	For entering the user-specific lock code. Is locked when no action 'Lock' was configured for the command group.	
Execute lock	Button	Activates a lock for the user entered in the	

		Control 'User identification'. Is locked when no action 'Lock' was configured for the command group. This user action is logged in the CEL, if not suppressed by the engineering.	
Unlock	Button	Removes the lock by the user entered in the user identification. Is locked when no action 'Lock' was configured for the command group. This user action is logged in the CEL, if not suppressed by the engineering.	
Execute	Button	Takes over the value of the Control 'Set value' or 'Set status' This Control is visible only when the screen is in 'Step 1'. The Control is locked additionally to the general lock, when: - The active action is not 'Set status', 'Set value' or 'Correct set value'. - The value in the control 'Set value' for the action variable is invalid.	
Comment	Input field	Comment about the lock.	
Action variable Status	Text	Status of the active action variable in short form.	X
Action variable Name	Text	Name of the active action variable.	X
Action variable Identification	Text	Identification of the active action variable.	X
Action variable value	Text	Value of the active action variable.	X
Active action	Text	Name of the active action.	X

Blocked or locked elements

GENERAL LOCK OF THE CONTROLS

Some requirements must be met in order to unlock the controls in the screen. Since these requirements usually concern several controls, they are not listed with the control each time, but they are documented here.

1. All Controls except **Exit** are locked when:

- the screen is not the owner of the active Select
- no interlocking was configured for the add-on variable
- the response variable does not exist
- The response variable has not received a value yet
- the status 'Select' of the response variable was not set
- an action for the action variable is running
- an action for the action variable is running
- the system is waiting for the SBO confirmation
- the data of the lock are being transmitted
- the data of the lock are invalid
- the INVALID bit status is active for the selected variable
- the currently-registered user does not have the necessary authorization levels

1. All Controls except **Exit** and the controls for the lock are locked when:

- one of the locking conditions of point 1 apply
- the response variable was locked against command inputs
- the status bit **S_MERKER_1(0)**, i.e. the command lock, of the response variable was set

COMMAND INPUT

- Action buttons: Action buttons are locked if the authorization level of the registered user forbids execution.

- Unlocking: Unlocking is only possible if the user does not have the necessary authorization levels for this.
- Context menu: Menu items that are assigned to a command action can only be selected if the registered user has the necessary authorizations.

LOADING A SCREEN WITH INITIAL STEP 'LOCK'

If the screen is loaded with the initial step 'Lock', all but the following controls are hidden in the screen:

- RV TTA
- RV identification
- RV value
- RV status
- RV measuring unit
- Lock code
- User identification
- Execute lock

Reload

If an online reload is started, you have to take the following possible effects into consideration:

If a watchdog timer, an edge generation or an SBO is currently active, the reload is delayed until they are closed.

An open screen of type command input is closed and the procedure is restarted after the reload, depending on the current step.

Step before Reload	Behavior after Reload
Step 1	Screen is loaded for Step 1 again
Unlocking or Step 2	Step 'Unlocking level' is activated. The interlocking check is re-executed.
Lock	Lock is re-activated

Before loading, the add-on variable, response variable, command variable, command group and action are re-determined. If one of the objects does not exist anymore, the controls are locked.

If the command group of the add-on variable was removed or exchanged, the screen is loaded with locked controls. The screen has to be reloaded afterwards or the command input has to be re-executed.

If the command group of the response variable was removed or exchanged, all locks against command input are removed from the variable.

If a user, for whom a command lock was activated, does not exist anymore, the lock is removed. The status bit S_MERKER_1(0) is updated accordingly.

Logging in the CEL

Besides the switching actions, the following user actions in the CEL are logged.

Parameters	Description
Unlocking	The unlocking of an interlocking is logged in the CEL.
Execute action	If the action setting 'Suppress CEL entry' is not active, the action exection is logged in the CEL.

Lock return variable

For a locked response variable, the statusbit S_MERKER_1(0) is set. The lock can be activated or deactivated by entering the username and the lock code defined during the user definition. One lock can be activated per user. The active locks are remanent and are also considered after a system restart.

> **Info**
>
> *The locks are automatically synchronized in the network; therefore, they can also be used in redundant operation.*

Server change in redundant operation

If the process handling server is changed, the select object is lost and the command input must be executed again. The same applies for the SBO.

Exit Runtime

As long as there are still active actions in the system, the proper exiting of the runtime (e.g. over a function call) is delayed.

Exiting is also delayed while the SBO procedure is active. If SBO is active, it will be deactivated.

> **Info**
>
> *This situation can arise especially for the action 'Single command' with watchdog and/or edge generation with one-step execution. The runtime is exited after the action was finished.*