zenon

by COPA-DATA

zenon manual

zenon Logic

v.8.20

Contents

1 Welcome to COPA-DATA help 6

2 zenon Logic 6

3 General information on zenon Logic help 7

4 Basics 7

4.1 zenon Logic General 7

4.2 zenon Logic projects in the zenon Editor 8

4.2.1 zenon Logic toolbar and context menu detail view 9

4.2.2 First steps with zenon Logic in zenon 10

4.2.3 Variable handling 18

4.2.4 Distributed engineering (multiuser) 27

4.2.5 Reusing projects 28

4.3 zenon Logic to zenon connection 29

4.3.1 zenon Logic to zenon functions 32

5 Driver and connection 34

6 Programmed driver simulation 35

7 zenon Logic Workbench 35

8 zenon Logic Runtime for Windows 35

8.1 User interface 36

8.1.1 Slider Allowed maximum 37

8.1.2 Settings 38

8.1.3 Messages 45

8.2 Licensing 45

8.2.1 TAG handling zenon - zenon Logic 46

8.2.2 Demo mode 47

8.3 Start the zenon Logic Runtime 48

8.4 zenon Logic Runtime cycle 53

8.4.1 Error message cycle time 55

8.5 Close the zenon Logic Runtime 56

8.6 Creating the hot restart file 56

8.7 Remote systems 57

8.7.1 zenon Runtime and zenon Logic Runtime on one remote PC 57
8.8 zenon Logic Runtime with Real Time Kernel (RTK) 58
8.8.1 Restrictions and requirements of the zenon Logic RTK 62
8.8.2 zenon Logic RTK configuration 63
8.8.3 Blue Screen Handling 70
8.8.4 Error codes at starting the zenon Logic RTK 71
8.9 Redundancy zenon Logic and zenon 73
8.9.1 Requirements 74
8.9.2 Configure redundancy 75
8.10 OEM specification 78
8.10.1 Support for printf function 81
8.11 Forwarding time stamps and the variable status information 82
8.12 zenon Logic I/O driver support 82
8.13 zenon Logic I/O driver: Hot-Restart/Online Change 84
8.14 Error message 86
9 zenon Logic Runtime Manager 89
9.1 Why should you use the zenon Logic Runtime Manager? 89
9.2 Main menu 90
9.3 Properties help 90
9.3.1 Configuration list 90
9.3.2 zenon Logic project folder 90
9.3.3 Name of the configuration 90
9.3.4 Port (Main) 91
9.3.5 Port (Binding) 91
9.3.6 Startup 91
9.3.7 Start in step mode 92
9.3.8 Windows real time priority 92
9.3.9 Hard real time 93
9.3.10 Report output 93
9.3.11 Auto start 93
9.3.12 Button "New" 94
9.3.13 Button "Edit" 94
9.3.14 Button "Save" 94
9.3.15 "Delete" button 94
9.3.16 Button "Start" 94
9.3.17 zenon Logic Runtime (settings) 95
9.3.18 Language selection (languages) 95
9.3.19 Show splash screen (setting) 95
9.3.20 Size of event queue 95

9.3.21 Maximum number of recordable events 95

1 Welcome to COPA-DATA help

ZENON VIDEO TUTORIALS

You can find practical examples for project configuration with zenon in our YouTube channel (https://www.copadata.com/tutorial_menu). The tutorials are grouped according to topics and give an initial insight into working with different zenon modules. All tutorials are available in English.

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.

PROJECT SUPPORT

You can receive support for any real project you may have from our customer service team, which you can contact via email at 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.

2 zenon Logic

zenon Logic is the programming environment integrated into zenon in accordance with IEC 61131. It is available as Editor (on page 8) and Runtime (Soft-PLC) (on page 35).

3 General information on zenon Logic help

This help offers you a quick introduction to working with zenon Logic. This section provides you with basic information on the elements, interfaces and settings of zenon Logic projects. Tips are given on the use of zenon Logic for the simulation of zenon direct driver variables.

The following explains the functionality and settings of the zenon Logic Runtime for Windows, a 61131-3 target system specifically designed for operation with the zenon Runtime. The core of this help serves as a comprehensive reference for zenon Logic Workbench.

This help can be called up in the zenon Logic Workbench either via the **Help** menu option. In order to access a help chapter about a certain topic (e.g. function block), select it with a mouse click and then press **F1**.

The in zenon integrated zenon Logic Workbench has not TAG limitation independent of I/O or TAG extension of the zenon Editor.

4 Basics

4.1 zenon Logic General

zenon Logic is an IEC 61131-3 programming environment for different target systems with zenon Logic Runtime kernel. In the case of zenon, the zenon Logic Runtime kernel is designed as a soft PLC for PC and CE platforms.

ZENON LOGIC WORKBENCH

The zenon Logic Workbench is the tool for programming the zenon Logic PLC. The zenon Logic Workbench is started in the zenon Editor. All five IEC 61131-3 predefined languages are available:

Abbreviation	Long form of the language
SFC	Sequential Function Chart - AS
FBD	Function Block Diagram - FUP
LD	Ladder Diagram - KOP
ST	Structured Text - ST
IL	Instruction List - AWL

IEC 61131-3

Part three of the IEC 61131 describes the syntax and semantic of computer languages for PLCs which have been defined in part 1 of the IEC 61131.

ZENON LOGIC RUNTIME

The zenon Logic Runtime is the target system and executes the compiled code of the zenon Logic Workbench. There is the zenon Logic Soft PLC for the PC and for Windows CE platforms, as well as selected embedded platforms.

> **Information**
>
> Further information on platforms can be found here: www.copadata.com/en/products/zenon-logic.html (https://www.copadata.com/en/hmi-scada-solutions/zenon-logic/).
>
> You can find more information about the operation of zenon Logic Runtime in the zenon Logic Runtime for Windows (on page 35) manual.

STRATON RUNTIME

In addition to the zenon Logic soft PLC, there is also the hardware PLC straton, which has implemented the zenon Logic runtime kernel as OEM software. The manufacturers of these hardware PLCs can be found on www.straton-plc.com (http://www.straton-plc.com).

4.2 zenon Logic projects in the zenon Editor

zenon Logic is an IEC 61131-3 programming environment for different target systems with zenon Logic Runtime kernel. In the case of zenon, the zenon Logic Runtime kernel is designed as a soft PLC for PC and CE platforms.
With the zenon Logic development environment - the Workbench - PLCs can be engineered and programmed in the five defined programming languages of IEC 61131-3.

PROJECT MANAGER CONTEXT MENU

Menu item	Action
zenon Logic New project	Opens the assistant to create a new zenon Logic project.
Import XML...	Imports entries from an XML file. **Attention:** At the XML import you must deactivate **online change** so that the reload of the zenon Logic variable works.
Import external zenon Logic	Opens the dialog to select a zenon Logic project folder.

Menu item	Action
project...	**Note:** not available for the **Smart Object** module.
Editor profiles	Opens the context menu with pre-defined Editor profiles.
Help	Opens online help.

4.2.1 zenon Logic toolbar and context menu detail view

TOOLBAR

Parameter	Description
zenon Logic New project	Opens the dialog to create a new project (on page 10), create it in the list and opens the zenon Logic Workbench with the new project.
zenon Logic Open project in Workbench	Opens the selected project in zenon Logic Workbench.
Delete	Deletes selected entries after a confirmation from list.
Import external zenon Logic project	Opens dialog to select the zenon Logic project files to be imported.
Import XML	Opens dialog to select the XML file to be imported. **Attention:** At the XML import you must deactivate **online change** so that the reload of the zenon Logic variable works.
Remove all filters	Removes all filter settings. **Note:** Only active if the current view is filtered.
Edit selected cell	Opens the selected cell for editing. The binocular symbol in the header shows which cell has been selected in a highlighted line. Only cells that can be edited can be selected.
Replace text in selected column	Opens the dialog to search and replace texts for the selected column.
Properties	Opens the **Properties** window.
Help	Opens online help.

4.2.2 First steps with zenon Logic in zenon

Here you can find out how you create, configure and edit a zenon Logic project in the zenon Editor.

4.2.2.1 Create a zenon Logic project in a zenon project

zenon Logic projects can be created both in the zenon Editor as well as in the zenon Logic Workbench.

As soon as a new zenon Logic project is created, this is visible:

- in the zenon Editor in the list of zenon Logic projects
- in the workspace of the zenon Logic Workbench

Note: All of the zenon Logic projects embedded in a zenon project are opened together via a project list in the zenon Logic Workbench. The zenon Logic project by means of which the zenon Logic Workbench was opened is set as the start project.

In the IEC 61131-3 documentation, you can find information on creating a zenon Logic project in the zenon Logic Workbench.

CREATING A NEW ZENON LOGIC PROJECT IN ZENON

Procedure:

1. right click on the **zenon Logic (IEC 61131-3)** node or in the detail view.
2. select in the context menu or in the toolbar the **New zenon Logic project...** command.

   The configuration dialog is opened.
3. Enter a valid name for the project.
4. Select the desired driver:
   - straton drivers are automatically configured
   - for IEC driver the configuration dialog is opened after the dialog is closed

   Optionally, you can enter additional information in **Description**.
5. Create the project by clicking on **Finish**.

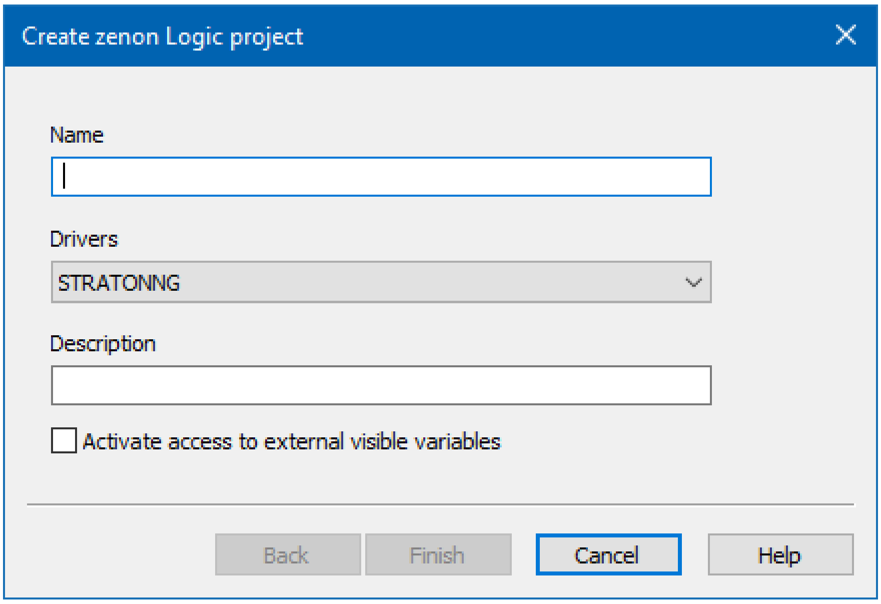
Information

You adjust the properties of the zenon Logic projects in the zenon project in the properties window of the zenon Editor. You will find a description of the single properties in the property help.

You can change the settings of the zenon Logic project in its project settings. In order to do this open the project settings in the Workbench under **Project/Settings...**.

NEW DIALOG PROJECT

If a new zenon Logic project is created in zenon, settings can be defined using the following dialog.



Create zenon Logic project

Name

Drivers: STRATONNG

Description

□ Activate access to external visible variables

Parameter	Description
Name	Freely definable name for the zenon Logic project. **Note:** The Name - consists of letters and numbers (the first character must be a letter) - must not exceed 15 characters - must not be empty **Note:** The only special character allowed is an underscore. Button **Finish** is only active if a valid name was entered.
Driver	Selection of the driver. You have the choice between: - **stratonNG** (Default):

Parameter	Description
	creates a new connection with the correct project name, the zenon Logic Runtime port and the IP address 127.0.0.1 - **straton32**: creates a new connection with the zenon Logic Runtime port and the IP address 127.0.0.1 - **IEC870:** opens configuration dialog - **IEC850**: opens configuration dialog This selection defines the eventual Runtime communication between zenon Logic and zenon. In addition the creation of the **operating and monitoring (O and M) variables** (on page 18) is based on this: - name based at the STRATON driver, - address based at the IEC driver
Description	Text field to enter additional information
Activate access to external visible variables	Default setting for the use of externally visible variables in Runtime. - *Active*: Project is created with default settings for the use of externally visible variables in Runtime: - **Embed symbols of all variables** compiler setting: *active* - **Retain capitalization of symbols** compiler setting: *active* - Additional creation of a **Logic-to-SCADA RT connection** driver - *Inactive*: No explicit default setting. The settings for the use of external variables in Runtime can be made manually in the existing project, if necessary. Default: *inactive*
Finish	Creates the new zenon Logic project.
Cancel	Cancels project creation
Help	Opens online help.

DRIVER SELECTION

Note: If a project has not been created in the zenon Editor but directly in the zenon Logic Workbench, a driver can be assigned later. For this, use the zenon Logic **Integrate project** dialog. In the <CA_PROJECTNAME> project properties, click on the drop-down lists in the **Driver** property in the **Workbench** properties group to open the dialog.

Decision aid for the driver selection:

Drivers	Properties
stratonNG driver:	Default driver for zenon Logic projects: - can be used cyclic or spontaneous (on change) - supports complex data types (structure data types or arrays) - enables several connections with one driver to several Runtimes at the same time (recommended for using with Windows CE))
straton32 driver:	Ethernet TCP/IP based. **Note:** Only included for compatibility reasons. Use of the **stratonNG driver** is recommended.
IEC870 driver:	It is mostly used in the infrastructure automation. Properties: - can be operated serial (-101) or over Ethernet TCP/IP (-104) - Data traffic is generally spontaneous (on change) - complex data types are not supported
IEC850 driver:	Successor of the IEC60870 standard and is completely object-orientated.

For information about the configuration of the driver see the respective help of the driver:

- straton32 driver: straton32
- stratonNG driver: stratonNG
- IEC870 driver: IEC870
- IEC850 driver: IEC850

> ### ⚠Attention
>
> If you have selected a driver once, you cannot change it afterwards.
>
> **Exception:** Changing from the **straton32** to the **stratonNG** driver is not possible.

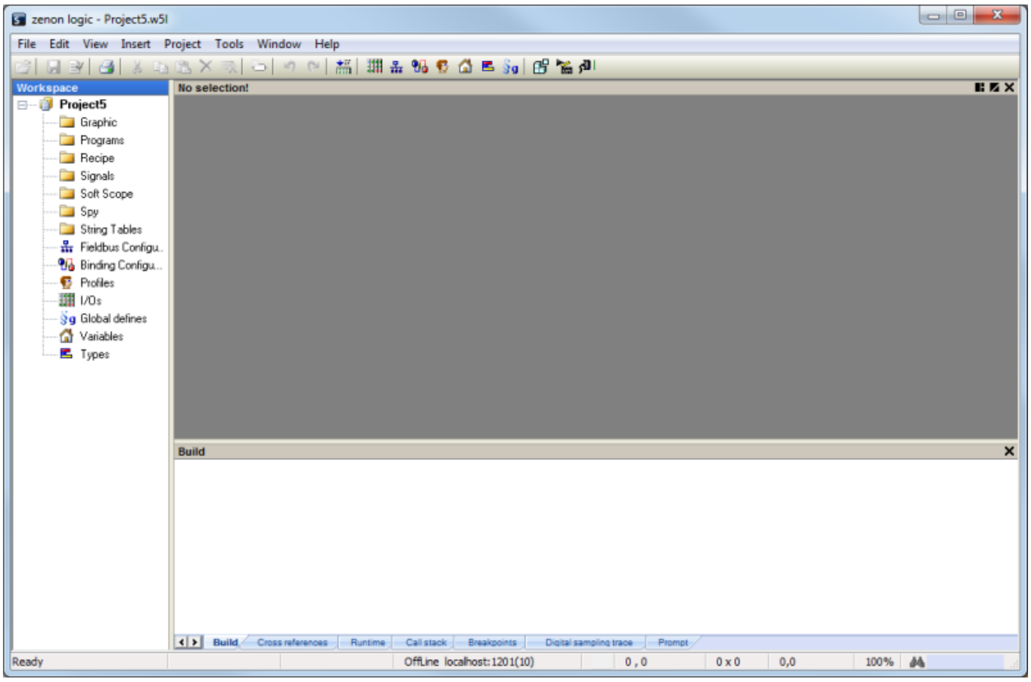
CHANGE DRIVER

To make a driver change from **starton32** to **stratonNG**:

- In the zenon Editor, select the project for which you want to replace the **straton32** driver.
- In the detail view, go to the **Variables/Driver** node.
- Select the **Driver** entry there.
- In the *detail view of the project manager*, select the **straton32** driver.
- select **Change driver** in the context menu
- The dialog for selecting the driver is opened.
- Select the driver.
- Close the dialog.
  The configuration is applied to the zenon Logic project.

4.2.2.2 Editing a zenon Logic project

With a double click on the project or the context menu, you can open an existing zenon Logic project for editing. Alternatively you can click the corresponding icon in the toolbar in the details window.



4.2.2.3 Deleting a zenon Logic project

To delete a zenon Logic project:

1. select menu item delete in the context menu of the project or in the toolbar

(Multi-select is possible)

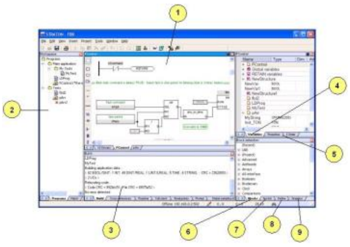
2. confirm the confirmation message whether you really want to delete the project
3. select whether you want to delete the driver of the project

   (When simultaneously deleting several projects, the deleting the driver is prompted for each project.)

> **⚠Attention**
>
> When you delete the driver, all variables linked to the driver in zenon are also deleted.

4.2.2.4Design of the Workbench

The zenon Logic Workbench can be divided in the following areas:



Range	Description
1	In this section documents are displayed and edited (programs, watch lists, ...).
2	The workspace: Documents are created and opened in this list.
3	The output window displays messages and provides diagnosis tools.
4	In the variable editor variables and instances are declared.
5	The properties tab is used for the configuration of the graphics documents.
6	The tab Libraries lists all available functions and function blocks
7	The tab Spy list displays the online values of selected variables during the debugging.
8	The tab Definitions displays all definitions.
9	The tab Graphics lists al graphical objects.

Each section (except section 1) can be shown and hidden with commands from the menu **Views**.

In the central area you will find the documents of the workspace. If several documents are opened at the same time, you can switch between the different documents using the tab at the bottom. With button **X** in the header the active document can be closed.

The variable editor and the document area in the center can be maximized at any time with the corresponding button (blue symbol with white line) or a double-click on the header.

If several documents are open in the central area, one of these can be fixed in the top or left part. In order to do this right click the corresponding tab and select **Fix**. With the same menu you can unfix the document at any time and fix another one:

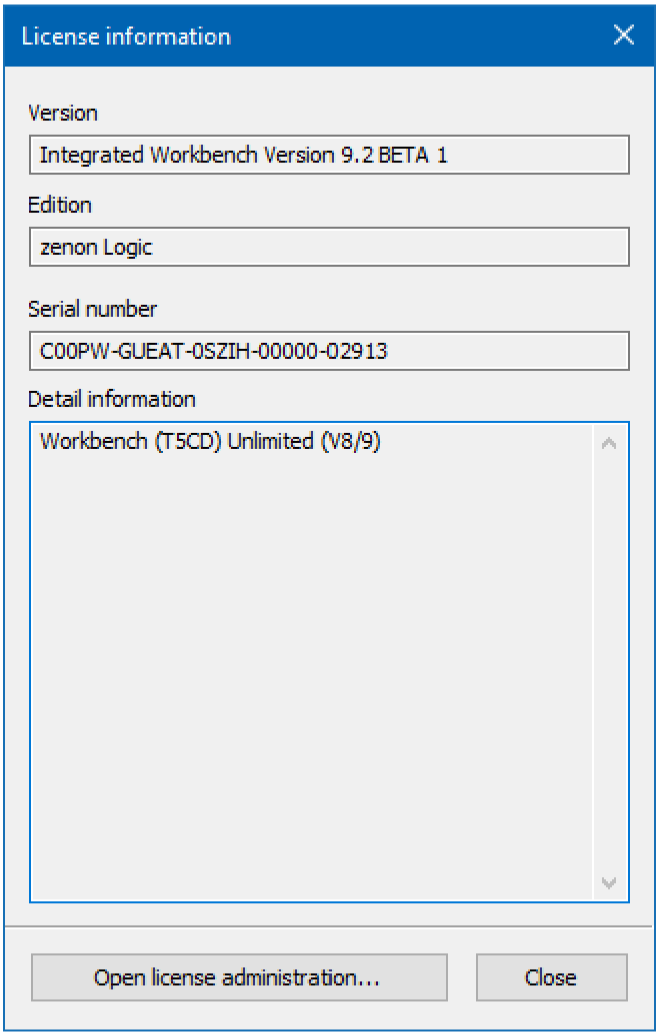
4.2.2.5 Show license information

To display license information in zenon Logic Workbench:

1. Open the menu **Help**.
2. Click on **About...** .

   The license information dialog is opened.

LICENSE INFORMATION DIALOG



License information

Version

Integrated Workbench Version 9.2 BETA 1

Edition

zenon Logic

Serial number

C00PW-GUEAT-0SZIH-00000-02913

Detail information

Workbench (T5CD) Unlimited (V8/9)

Parameters/buttons	Description
Version	Version number of the integrated workbench.
Edition	Edition used.
Serial number	License serial number.
Detail information	Shows details on the license.
Open license administration	Opens the **License Manager**. Licenses can be activated, returned and managed with this tool. The serial number is needed for this.
Close	Closes the dialog.

4.2.3 Variable handling

The integration of zenon Logic into zenon offers considerable advantages in the configuration of PLC and SCADA projects. The O&M variables (operating and monitoring) from the PLC system are available in zenon with just a few mouse clicks. When changing the variable (name, datatype, ...) these changes are immediately passed on the other system.

The advantage of the integration:

- Easy handling of O&M variables
- Immediate alignment of changes
- Saving of labor because of omission of import/export
- Increased quality by minimizing causes for errors

> ⚠**Attention**
>
> Instances of UDFBs are not available as B&B variables.

4.2.3.1 Create O&M variables in zenon Logic

All variables created in the zenon Logic Workbench can be marked as O&M (operating and monitoring) variables:

- Local program variables
- Global variables
- Retain variables (=remanent variables)
- Array variables
- Structure variables

Note: Variables of data type *TIME* cannot be declared as O&M variables.

HIGHLIGHT VARIABLES AS O&M

In order to do this the following steps are necessary:

(Refer to the online help of the zenon Logic Workbench for the exact steps to create a variable in zenon Logic.)

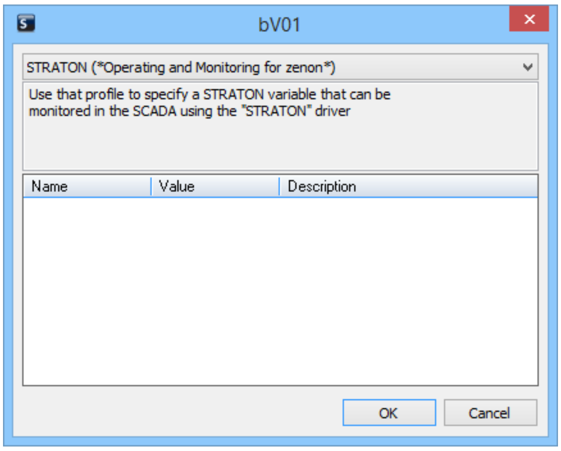
1. Right click on a variable in the variable list.
2. Select **Properties** from the context menu.
3. In the drop-down list, select *STRATON (*Operating and Monitoring for zenon*).*

4. Click on **OK**.

   A request for confirmation is opened.

5. Confirm the dialog by clicking on **Yes**.

6. Activate the checkbox of the variable in the *Symb* column.



Use that profile to specify a STRATON variable that can be monitored in the SCADA using the "STRATON" driver

Now the O&M variable has been entered in the zenon variable list.

⚠Attention

As soon as a zenon Logic variable is declared as O&M variable, you must not edit it as text in the Workbench!

This leads to the loss of variable declarations.

The O&M variable is displayed with a prefix in zenon. The prefix has the following structure:

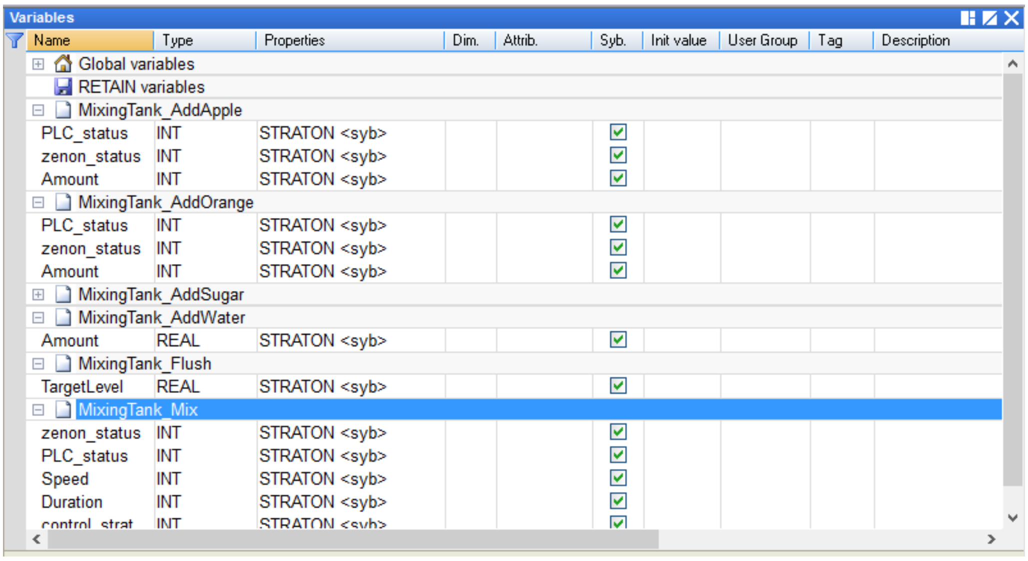
[zenon Logic project name >/zenon Logic program name or Global or Retain]/[variable name]

⚠Attention

Variables have a character length limitation. Including prefix the variable name must not be longer than 128 characters.

ACTIVATE IN VARIABLE LIST

You can update the profile and the checkbox directly in the variable list of zenon Logic so you do not have to go to the context menu and the profile dialog.



Variables

Name	Type	Properties	Dim.	Attrib.	Syb.	Init value	User Group	Tag	Description
⊞ Global variables									
RETAIN variables									
⊟ MixingTank_AddApple									
PLC_status	INT	STRATON <syb>			☑				
zenon_status	INT	STRATON <syb>			☑				
Amount	INT	STRATON <syb>			☑				
⊟ MixingTank_AddOrange									
PLC_status	INT	STRATON <syb>			☑				
zenon_status	INT	STRATON <syb>			☑				
Amount	INT	STRATON <syb>			☑				
⊞ MixingTank_AddSugar									
⊟ MixingTank_AddWater									
Amount	REAL	STRATON <syb>			☑				
⊟ MixingTank_Flush									
TargetLevel	REAL	STRATON <syb>			☑				
⊟ MixingTank_Mix									
zenon_status	INT	STRATON <syb>			☑				
PLC_status	INT	STRATON <syb>			☑				
Speed	INT	STRATON <syb>			☑				
Duration	INT	STRATON <syb>			☑				
control_strat	INT	STRATON <syb>			☑				

Mark the line **Properties** of the corresponding variable. Press **Ctrl+F12**. Thus the profile zenon Logic is displayed in this line. In addition you must activate the checkbox for embedding the symbol via double click or **Enter** in column **Symb.**.

> ### ⚠Attention
>
> The checkbox **Embed symbol** must be activated. Variables for which this checkbox is not active are not able to communicate with zenon using the STARTON driver and will not supply any values.

ADDITIONAL ATTRIBUTES OF THE VARIABLE

The additional attributes **Identification** and **Description** of a variable are also displayed in zenon when you select the profile STRATON (*Operating and Monitoring for zenon*). You can find the content of the identification and description in zenon in the **Identification** and **Resources label** variable properties.

ARRAYS AND STRUCTURE VARIABLES

If arrays or structure variables are marked with the profile STRATON (*Operating and Monitoring for zenon*), they are also displayed in the variable list in zenon. You must however activate the desired elements of the variable.

Attention

If a O&M variable is an array, you must take care when setting initial values.

Recommendation: Arrays with initial values should not be declared as O&M! If it is inevitable, the string describing the initial value must not be longer than 1024 characters.

UDFB INSTANCES

1. Input variables at UDFB instances are not available via the integrated solution in zenon.
2. In zenon variables of an UDFB instance cannot be created. This is only possible if an instance of the UDFB is created in zenon Logic and the zenon Logic profile is set.

   UDFB instances which are marked with profile zenon Logic (*Operating and Monitoring for zenon*), they are not automatically available in zenon. In case you want them to be available, you must activate this functionality manually with the help of the following entry in file K5DBXS.INI. In file K5DBXS.INI scroll to section [XS] and enter ShareUDFBDatatype=1 .

   If you deactivate this entry, existing UDFB data types in a project are still modified. However no new UDFBs are created.

HARDWARE IO DECLARATIONS

zenon Logic IO variables can be used zenon if in the zenon Logic Workbench they:

- have a profile
- have an alias name; this name is used in zenon instead of *%IX0.0* (board type, board index, bit number)
  An alias is not necessary if the zenon Logic profile is used and property **Embed symbol** was activated. In this case the variable is also available in zenon (Editor and Runtime).

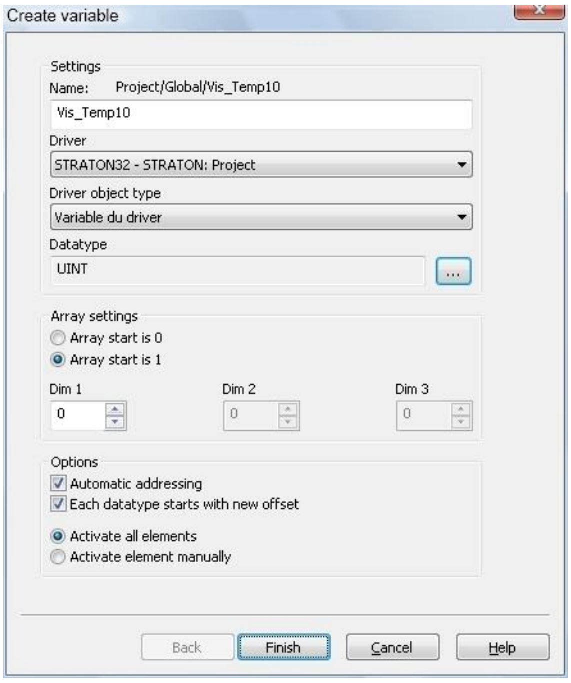
Note: If an I/O group is deleted in zenon Logic, the variables of the I/O board with zenon Logic profile are also deleted in zenon.

Attention:

- IO variables cannot be renamed in zenon.
- IO variables for zenon Logic IO cards cannot be created in zenon.
- The data type of IO variables should not be changed in zenon: Changes are not adopted in the zenon Logic Workbench.
- Changes to the hardware IO declaration should always be carried out in zenon Logic. In zenon no new variable for the straton32 driver can be created as hardware I/O variable for zenon Logic.

4.2.3.2 Create B&B variables in zenon

O&M (operating and monitoring) variables for the soft PLC zenon Logic can also be declared in zenon. In order to do this create a variable in zenon as usual:



Create variable

Settings

Name: Project/Global/Vis_Temp10

Vis_Temp10

Driver: STRATON32 - STRATON: Project

Driver object type: Variable du driver

Datatype: UINT

Array settings

○ Array start is 0

● Array start is 1

Dim 1: 0

Dim 2: 0

Dim 3: 0

Options

☑ Automatic addressing

☑ Each datatype starts with new offset

● Activate all elements

○ Activate element manually

Attention

Take care that the naming convention is IEC 61131-3 compatible (no special characters, ...); otherwise the button **Finish** will not be available in the dialog. See also: Variable names (on page 25)

Information

The variables declared in zenon are always entered in the Global area of the variable list in zenon Logic.

CHANGES TO VARIABLES OF A ZENON LOGIC DRIVER IN ZENON

If variables of a driver which is linked to a zenon Logic project are modified:

- a change recognition in the zenon Logic project is carried out
- the zenon Logic Workbench connection to the Runtime asks whether the project should be recompiled

This behavior is regardless of whether the changes affects the zenon Logic project.

4.2.3.3 Using zenon variables in zenon Logic

It is very easy to further process variables from PLC systems such as e.g. Siemens S7 or Beckhoff TwinCAT in zenon Logic, variables which were read in by the according zenon drivers in zenon.

There are two possibilities available for this:

- Allocation:
  The variable from the PLC (e.g. S7) is linked to a zenon Logic variable with an allocation.
  In this case only a unidirectional communication is possible.
- **Logic to SCADA connection**:
  The PLC variables (e.g. from a S7) are exchanged between zenon and zenon Logic using a **Logic to SCADA connection**. In order to switch the PLC variables in zenon Logic Workbench to visible, you must activate the **Visible externally** property in the properties of the variables. The variable is thus included in the variable list of zenon Logic Workbench.

  The values can be updated to the Runtime via a Shared-Memoryconnection. Therefore, you have to create the **Logic-to-SCADA connection** driver in the zenon Logic field bus configuration. The value of the variable can be defined either in zenon by a PLC or in zenon Logic Runtime and then written to the PLC. However the values must be neither updated at he same time nor in every zenon Logic cycle.

  Advantage: This process works in two directions. It can also be configured very quickly and easily.
  Note: Such a connection only works locally on a PC or CE device. Here it is not possible to communicate with a zenon Logic Runtime on another device.

LOGIC TO SCADA CONNECTION

A **Logic to SCADA connection** variable is specially marked in zenon Logic, because it can no longer be changed in zenon Logic. Changing the variable here would lead to problems, as the source of the variable is the PLC programming environment in the external PLC (e.g. Step7).

In order for **Logic-to-SCADA connection** variables to be able to communicate in Runtime, an I/O driver - the **Logic-to-SCADA connection** driver - must also be created in zenon Logic.

Attention: If the **No automatic writing** property is activated for the **Logic-to-SCADA connection** in the zenon Logic Workbench, the driver does not return any changes from Runtime as set value operation..

Configuration:

1. Open the field bus configuration in the zenon Logic Workbench.
   To do this, double click on the **field bus configuration** module or click on the symbol in the toolbar.

2. Right-click in the I/O driver area.
3. Select **Add configuration** in the context menu.
4. In the **All** group, select the **Logic to SCADA connection** entry.
   This configuration is entered and displayed.

It is thus ensured that the externally-visible variables from zenon communicate with zenon Logic.

> ⚠**Attention**
>
> Note when choosing the variable name:
>
> The name of the variable from an external PLC must comply with IEC 61131-3 with regard to how it is written.
> Allowed are:
>
> - Numbers
> - Letters from A to Z (no umlauts)
> - Underscores
>
> In addition the name must not start with a number.
> (for details, see also variable names (on page 25).)
> **Tip:** To use a variable with non-permitted characters in zenon Logic, the name can be set in the code directly in curly brackets.
> Example: The variable **myArray[3]** is called up in the program with {**myArray[3]**}.

POSSIBLE INTERACTIONS

Note for **Logic to SCADA connections**, possible interactions with the **Hysteresis** property.
If a **Hysteresis** was configured for the zenon variable, the value in zenon Logic does not match the value in the PLC. It is also possible that it does not match the value in zenon.
The following rules apply:

- The value changes in the PLC and within the configured hysteresis bandwidth:
  The value is not communicated to zenon Runtime and therefore also not to zenon Logic.
- The value is changed by the user in zenon (by means of a function for example) and the value remains within the configured hysteresis bandwidth:
  The value remains the same in zenon and in zenon Logic but it changes in the PLC.
- The value is changed in zenon Logic and remains within the configured hysteresis:
  The value changes in the PLC and in zenon Logic, but remains unchanged in zenon.
  If in this situation the value changes in the PLC and stays within the hysteresis band, three different values exist:
  one in the PLC, one in zenon and one in zenon Logic.

⚠Attention

Events are only triggered in the event of value changes. In special groups of applications, there may be inconsistencies between values.

This can be avoided very easily in that different variables are used for reading and writing.

4.2.3.4Variable names

In zenon Logic variable names must be in accordance with the IEC 61131-3 regulations. The correctness of the entry is checked by zenon as well as by zenon Logic (communication take place using the STRATON driver).

In contrast, for variables of other drivers (e.g. S7 TCP/IP driver), the person configuring the project has to ensure that the variable name corresponds to the conventions of IEC 61131-3 (communication using **Logic to SCADA connection**).

EXAMPLES OF PERMITTED AND NON-PERMITTED IEC 61131-3 WAYS OF WRITING

Permitted name	Impermissible name	Invalid because:
_XYZ	*__XYZ*	Two underscores.
xy_z	*xy__z*	Two underscores.
*xyz_*	*xyz__*	Two underscores.
MotorOn	*Motor On*	Space.
Motor_On	*Motor/On*	Slash.
mw10	*1mw10*	Number at the beginning.
xyz	*#xyz*	Hash.
_Motor1	*_Motor 1*	Space.
Stoerung	*Störung*	Umlaut.
Ueberbruecken	*Überbrücken*	Umlaut.
Mo_tor	*Mo-tor*	Hyphen.

From version 6.22 SP1 on, it is possible to use variables that are not compliant with IEC 61131-3 (free variable names). Take care when using free variable names.

Follow these rules when using free variable names:

- The following characters are not allowed to be part of a variable name: '(*', '*)', '//', '{' and '}'
- Variable names are not allowed to start with '__' (two underscores)
- If free variable names are used when using programming language ST or IL, they must be between curly brackets {}.
- If free variable names are used with structure of array, they must be between curly brackets {}.
- Variables with free variable names with a basis data type can be used without curly braces {} in programming languages FBD and LD. They must be declared as **global** variables.
- Variables with the same name once with {} and once without {} are regarded as two different variables.

> **Information**
>
> Structure variables and arrays of structure variables with dot (".") in the name are displayed correctly. Standard arrays with slash ("/") or dot (".") in the name do not display values in the **spy list**.

4.2.3.5 Arrays

Arrays can be created in zenon and zenon Logic for both systems. The declaration as O&M variable takes place as described in chapters Creating O and M variables in zenon Logic (on page 18) and Creating O and M variables in zenon (on page 22). Take care when creating arrays in zenon Logic that you must activate the single elements in zenon.

When creating arrays in zenon, set the array start to 0. Thus the arrays have the same indices in both systems.

> **Attention**
>
> Arrays with initial values should not be declared as O&M! If it is inevitable, the string describing the initial value must not be longer than 255 characters.

4.2.3.6 Data structures

Structure datatypes can be created in zenon or in the zenon Logic Workbench.

Note: For the communication of B&B variables (on page 18) with zenon, the **Complex variables in separate segments** option must be activated in zenon Logic Workbench. You can find the option under **Project**, **Parameter...** in the **Runtime** group. The size of the structure definition per project may not exceed 128kb. The current size of the definition for complex data types is displayed in the compiler output.

Information

When creating structure data types in zenon the IEC 61131-3 naming convention has to be used otherwise the structure data type is not available in zenon Logic. Also see Variable names (on page 25). Here the spelling is not checked by zenon!

Information

If you create structure data types in zenon Logic, they are always declared as linked in zenon.

Therefore it is better to create a structure data type in zenon because then you can decide whether you want a linked or an embedded data type.

4.2.4 Distributed engineering (multiuser)

Several people configuring a project can work together on one zenon project. Thus for example one project engineer works on a zenon Logic project and another works on the zenon project.

BASIC REQUIREMENTS AND PROCEDURES

Based on the mechanisms of distributed engineering in zenon, it is possible to distinguish between the following basic commands for the processing of zenon Logic projects:

- Checkout (enable changes) for an individual zenon Logic project
- Checkout of the entire project list

Here, the following requirements must be taken into account:

- The exclusive processing of a zenon Logic project requires the checkout (“enable changes”) of the complete variable list, drivers and data types in the zenon Editor.
- In order to create a new zenon Logic project or delete a project, the entire zenon Logic module node must be checked out.

- In order to create or edit library projects and cross-project components (e.g. global spy list, global binding configuration etc.), the zenon Logic Module must be exclusively checked out.
- Components that may not be edited due to the given requirements are locked in the project list of the zenon Logic Workbench.

Note: The requirements for the editing of zenon Logic projects are checked in the zenon Editor. A user prompt (message box) may be performed to enable additionally required module checkouts, if needed.

VARIABLES

In distributed engineering O&m variables (operating and monitoring) are handled separately by the zenon Logic project.

You can create variables in zenon Logic as usual. If the variables should also be available in zenon, they have to be equipped with the corresponding property (element group, profile **STRATON** (*Operating and Monitoring for **zenon***)). After that the variable in created in zenon. You can check it in for other engineers (**apply changes**). You cannot edit a checked in variable in the zenon Logic Workbench. It is marked accordingly in the zenon Logic variable list.

DRIVERS

STRATON drivers are attended separately in distributed engineering by the zenon Logic project. That means if you want to make changes, you have to check out the STARTON driver (**enable changes**).

ZENON LOGIC PROJECTS

In order to edit a zenon Logic project in a multi-user zenon project, the zenon Logic project must be checked out (**Check out**). After the engineering, the changes can be accepted or undone. Library projects and cross-project definitions can only be created, deleted or edited in the zenon Logic Workbench if the entire zenon Logic module is exclusively checked out.

ZENON LOGIC DELETE PROJECT

In order to delete zenon Logic projects, it must be checked out (**check out**) or edited in offline mode.

4.2.5 Reusing projects

zenon Logic projects can easily be integrated into a zenon project:

1. Select, in the context menu of the detail view of the **zenon Logic (IEC 61131-3)** module, **External zenon Logic Import project...**.

2. Enter an existing zenon Logic project (highlight folder).
3. Click on **OK** to integrate the existing zenon Logic project into the zenon project.
4. Define a name for the imported project.
5. Select a driver for the integrated solution.

Another possibility is the XML import:

1. In the context menu of the detail view of the **zenon Logic (IEC 61131-3)** module, click on **Import XML**.
2. Select the desired XML file.
3. Click on **OK** to import the zenon Logic project.
4. Create a fixed name for the imported project.
5. Select a driver for the integrated solution.

Note: At the XML import you must deactivate **online change** so that the reload of the zenon Logic variable works.

4.3 zenon Logic to zenon connection

A direct, bi-directional variable exchange between zenon Runtime and zenon Logic Runtime is possible if both Runtimes are operating on the same computer.

The communication is then carried out via a shared memory interface. This type of connection supports the communication of variable values as well as time stamps and status bits. Therefore, the variables of any driver in zenon can also be processed directly in zenon Logic. This can be used, for example, for logical processing on the basis of variable values in zenon ("SCADA Logic") or for runtime monitoring of zenon variable values. On the other hand, variable values from the field level can also be forwarded directly to the zenon Runtime in this configuration.

In order to use the zenon Logic to zenon connection, this must be activated in the zenon Logic project for the zenon Logic Runtime (zenon Logic Field bus configuration **Logic-to-SCADA RT connection**). In the zenon project, variables for shared use (shared memory) are released per variable by using a respective configuration. Variables of any driver and driver type in zenon can basically be released for shared communication.

Information

You can find further information on this in the "Linking of zenon variables to zenon Logic" section in this chapter.

Notes:

- The variable communication via this type of connection can only be used during runtime if zenon Runtime and zenon Logic Runtime are running on the same computer.
- The use of the zenon Logic to zenon connection can basically be configured at any time during the course of engineering and then be modified later on. Both in the zenon project (release of variables) as well as in the zenon Logic project (add the **Logic-to-SCADA RT connection**).

CONFIGURATION OF THE ZENON LOGIC TO ZENON CONNECTION

Certain configurations are necessary in zenon Logic so that variables of the zenon project can be used in the zenon Logic Runtime.

In order to establish communication between zenon and the zenon Logic Runtime:

1. Create a new zenon Logic project or open an existing zenon Logic project.
2. Select the **Field Bus Configuration** node in this zenon Logic project.
3. Open the **I/O Driver** view by double-clicking on the node.
4. Click on the **Add configuration** symbol.
   This opens the **Add configuration** dialog.
5. Expand the view of **SCADA Runtime Connectivity** by clicking on the arrow symbol located in front of it.
   This displays the **Logic-to-SCADA RT connection** entry.
6. Double click on **Logic-to-SCADA RT connection**.
   This creates the connection.

CHECK THE COMPILER SETTINGS OF THE ZENON LOGIC PROJECT

Procedure:

1. Go to **Project** and **Parameter...** in the menu bar.
   This opens the **Options** dialog.
2. Click on **Compiler** to open the view of the advanced project settings.
3. Make sure that the following settings are set to **Yes**:
   - **Embed symbols of all variables**
   - **Retain capitalization of symbols**

Note: These compiler settings can be deactivated for certain embedded applications to save resources. These settings are absolutely required for the zenon Logic to zenon connection. Make sure that the project with the above-mentioned setting has been compiled and transferred to the target system.

LINKING OF ZENON VARIABLES WITH ZENON LOGIC

It is possible to link as many zenon variables as you want with zenon Logic. The variables can only be linked in zenon.

The respective variable then appears with an additional symbol in the variable list of the zenon Logic project.

In order to link variables:

1. Highlight the desired variables in the zenon Editor. Multiple selection is possible.
2. Go to**External settings** in the variable properties.
3. Activate the checkbox of the **Externally visible** property.
   The variables are now also available for zenon Logic projects.

Note: The **Externally visible** property allows you to define the zenon Logic projects for which the variable is to be visible. If you leave the input field empty, the variable is visible for all zenon Logic projects.

From zenon Version 6.50 and higher, it is possible to allow structure data types to communicate via this connection.

However, there are limitations to the communication of structure data types:

- The functions for requesting the status of the (*isvalid* or *isdirty*) variables and the *setdirty* function are not available for the use of variables with structure data types.
- The reading of status bit information and the date/time stamp does not function when using variables with structure data types.

DATA TYPES

All data types are permitted.

RELATED FUNCTIONS

When using the zenon Logic to zenon connection, it may be necessary in certain situations to check the connection status or the validity of the variable information for the zenon Logic Runtime. Functions for use in the 61131-3 application program are available for this purpose.

VARIABLE STATUS BITS

It is possible to read the variable status bits and the date/time stamp that come from zenon. For further information on this, see the Variable status bits chapter of the 61131-3 documentation.

4.3.1 zenon Logic to zenon functions

The following functions are available in zenon Logic Workbench for the use of the zenon to Runtime I/O connection:

Command	Description
zenrt_isValid	Checks whether the variable is *valid*.
zenrt_setDirty	Marks the variable as *dirty* in order to force a transfer without value change too.
zenrt_isDirty	Checks whether a variable has been marked as amended or *dirty*.
zenrt_red	Checks whether there is a connection to the zenon server.

ZENRT_ISVALID

Checks whether the variable is valid.

If the zenon Logic variable has not activated the status bits, *zenrt_isValid* can be used in order to establish whether a variable has been read/initialized by zenon. The information received is the same as for the time stamp.

Input	Data type	Description
VARIABLE	*ANY*	Variable name of the variable that is to be checked for validity.

Output	Data type	Description
Q	*BOOL*	Indicates whether the zenon variable is valid in zenon Logic. ▸ *TRUE*: zenon variable is valid ▸ *False*: zenon variable is not valid

ZENRT_SETDIRTY

Marks the variable as dirty in order to also force a transfer without a change.

> **Information**
>
> If the value is changed after a function call, only the new value is transferred. The current value is lost.

Input	Data type	Description
VARIABLE	*ANY*	Variable name of the variable that is to be highlighted as *dirty*.

Output	Data type	Description
Q	*BOOL*	*TRUE* If successfully marked as *dirty*.

ZENRT_ISDIRTY

Checks whether a variable has been marked as amended or *dirty*.

Input	Data type	Description
VARIABLE	*ANY*	Variable name of the variable that is to be checked for being marked as dirty.

Output	Data type	Description
Q	*BOOL*	*TRUE* If the variable is marked as dirty.

ZENRT_RED

Checks whether there is a connection to the zenon server. Uses the zenon error timeout that is set in the bus driver configuration.

Input	Data type	Description
MAX_SWITCH	*DINT*	Maximum number of redundancy switching attempts.

Output	Data type	Description
SWITCH	*BOOL*	If *TRUE*, the zenon Runtime can no longer be reached or the zenon Runtime is standby. This means that the number of **MAX_SWITCH** has not yet been exceeded.
ERROR	*BOOL*	If *TRUE*, the maximum number of redundancy switching has been reached.

5 Driver and connection

Runtime target systems for zenon Logic offer a range of options for networking with different systems. There are different variants available for this:

- **Field bus configuration**

  Different field buses and protocols are activated by means of a selection dialog. Configuration is carried out on the basis of a uniform basic structure. In doing so, specific parameters and elements of the respective prototype are offered at the corresponding levels of this structure. For details and an overview, see the field bus configuration chapter.

- **Networking of distributed Runtimes**

  zenon Logic Runtimes have, as standard, a protocol that enables direct, reciprocal exchange of variables. The setting of which variables are sent from a Runtime into the network and/or by which Runtimes these are received can be set directly in an integrated configuration using the variables declared in the projects. For details and an overview, see the Linking applications – connection chapter.

- **Communication profiles**

  For selected drivers, 'profiles' offer a flexible way to set the availability of variables using a protocol. In doing so, only general driver communication parameters are defined using the field bus configuration. The availability of variables, as well as protocol-specific types and addressing information, is sent to the variable directly as a property. The configuration of variables using profiles includes slave and server protocols for energy applications in particular. The variables that are to be replaced as part of the driver-bound connection between zenon Logic Runtime and zenon Runtime ("operation and monitoring") are also defined by means of a profile setting for the variable. You can find details in the description chapters of the respective drivers. You can find a detailed description on the setting of zenon Logic variables for "operation and monitoring" with zenon in the Creation of B&B variables in zenon Logic chapter.

- **Function blocks for communication**

  Application-related processes can be implemented through the use of function blocks in the 61131-3 programs. zenon Logic offers support for a range of protocol-specific functions using function blocks. You can find details in the Advanced Operations chapter, in the "Communication" section.

Note: zenon Logic Workbench contains a number of configuration tools and program resources that can be used on different target systems. Please note the communication options that are actually available for your Runtime target system.

6 Programmed driver simulation

Direct drivers in zenon have an option for operation in 'programmed simulation'. In the process, the variable values should not come from the corresponding driver counterpart (the PLC for example), but should be generated locally by a virtual simulation program. The programmed driver simulation can thus be used, for example, to simulate the behavior of devices for development time or during partial putting into operation. It is in precisely this way that process behavior can be recreated with digital or analog values.

The simulation programs are created in the zenon Logic Workbench by an integrated 61131-3 project. The variables of the respective driver are available directly as variables in the 61131-3 programming system. When starting the zenon Runtime, available simulation projects – depending on the setting of the operation type of individual drivers – are loaded and executed by an integrated zenon Logic Runtime. Corresponding driver simulation projects can be checked and/or edited by means of an online connection to the zenon Logic Workbench.

You can find details on the technical functionality and operation in the Programmed simulation for zenon direct drivers chapter.

7 zenon Logic Workbench

This help can be called up in the zenon Logic Workbench either via the **Help** menu option. In order to access a help chapter about a certain topic (e.g. function block), select it with a mouse click and then press **F1**.

8 zenon Logic Runtime for Windows

zenon Logic Runtime interprets the compiled PLC Code of the zenon Logic Workbench.
This manual gives you an overview of the use of zenon Logic Runtime.

SUPPORTED SYSTEMS

Overview of supported desktop operating systems for Windows 10

Windows version	zenon Supervisor / Operator	Everywhere by zenon	zenon Logic Runtime
Windows 10 Home	X	X (PC)	X
Windows 10 Mobile	--	X (Mobile)	--

Windows version	zenon Supervisor / Operator	Everywhere by zenon	zenon Logic Runtime
Windows 10 Pro	X	X (PC)	X
Windows 10 Enterprise	X	X (PC)	X
Windows 10 Education	X	X (PC)	X
Windows 10 Mobile Enterprise	--	X (Mobile)	--
Windows 10 IoT Core	--	--	--
Windows 10 IoT Enterprise (Windows 10 Enterprise LTSB)	X	X (PC)	X
Windows 10 Enterprise LTSC	X	X (PC)	X

Key:

- **X**: available
- **--**: Cannot run

You can find an overview of other operating system requirements here.

ZENON WEB CLIENT

zenon Logic is not supported by the zenon web client.

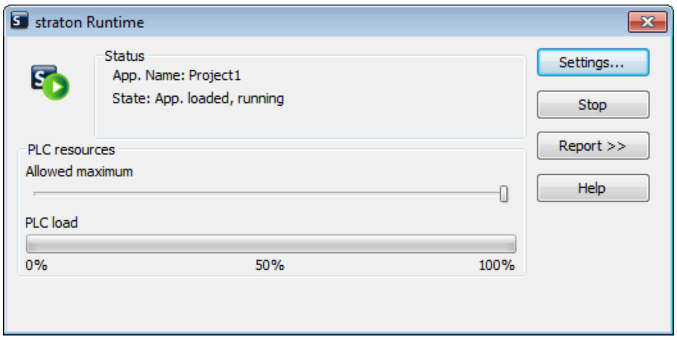
8.1 User interface

The zenon Logic Runtime is available as a symbol in the system tray. This shows the current status of Runtime. A double-click on the symbol or on the **Show** item in the context menu opens the zenon Logic Runtime user interface.

The user interface shows a different status depending on the current status:

- *runs*
- *Step mode*
- *Breakpoints set*

- *stopped*



Status
App. Name: Project1
State: App. loaded, running

Allowed maximum

A progress bar shows the PLC load.

In addition to information about the status of the zenon Logic project running, the following actions can be executed in the zenon Logic Runtime user interface:

- Slider **Allowed maximum** of the **PLC resources:** set allowed maximum.
- **Settings...** (on page 38) button **:** Makes configuration possible via three tabs
  - General settings (on page 38)
  - Redundancy (on page 41)
  - Advanced users (on page 42)
- **Start/Stop** button: opens the dialog to start Runtime or stops Runtime
- **Messages** (on page 45) button: opens and closes the message window

8.1.1 Slider Allowed maximum

The slider **Allowed maximum** is used to guarantee other applications which run on the same device - such as zenon - system resources.

With this slider the maximal allowed execution time related to the cycle time can be set. If with a set cycle time of 10 ms the slider is set to 50% the PLC cycle may take a maximum of 5ms. If the PLC cycle takes longer that 5 ms, the remaining calculation is done in the next cycle. So one cycle has to be omitted from the PLC. This results in a cycle overflow.

The slider can be set between 10% and 100% steplessly.

The bar graph (PLC load) shows, how high the PLC load is in relation to the setting of the slider. From this you can conclude how to set the slider for an optimal tuning.

PLC load high (Slider between 50 and 100%): Set slider towards 100% or increase cycle time.

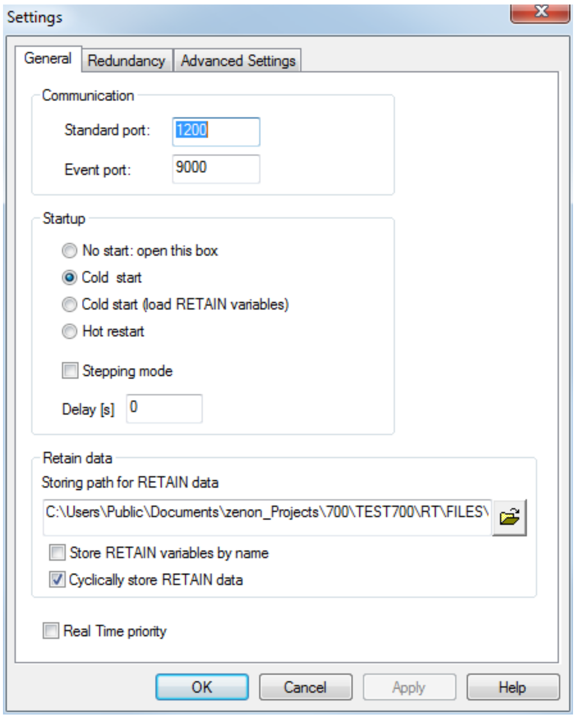
8.1.2 Settings

The **settings** make it possible to configure the zenon Logic Runtime with the help of three tabs:

- General settings
- Redundancy
- Advanced users

8.1.2.1 General

In the **General** tab, the different parameters of the zenon Logic Runtime can be changed:



Communication

Standard port: 1200

Event port: 9000

Startup

○ No start: open this box

● Cold start

○ Cold start (load RETAIN variables)

○ Hot restart

□ Stepping mode

Delay [s] 0

Retain data

Storing path for RETAIN data

C:\Users\Public\Documents\zenon_Projects\700\TEST700\RT\FILES\

□ Store RETAIN variables by name

☑ Cyclically store RETAIN data

□ Real Time priority

Parameter	Effect
Communication	
Port (Main):	IP port of the zenon Logic Runtime for the cyclic data transfer (e.g. zenon Logic Workbench).
Port (Binding):	IP port of zenon Logic Runtime for spontaneous data transfer (e.g. *straton32.exe* driver of zenon or binding).
Run-up	
No start. Open this box:	When the zenon Logic Runtime is started, first this box is opened.

Parameter	Effect
Cold start:	Initialized run-up. Also retain variables are started initialized.
Cold start (loading of RETAIN variables):	Initialized run-up with current values of the retain variables. They only contain values, no states (see Info box).
Warm start:	Restart with all variable values from the last stop.
Start in step mode	Starts the zenon Logic Runtime and immediately stops it in debug mode.
Delay [s]	Delays the start by the set number of seconds. Default: *0*
Retain Data	
Storage path Retain data	Here the file with the retain variables is stored. The default setting depends on how the zenon Logic Runtime is started.
Save retain variables with their names	Activate this checkbox if you want to save retain data in the format with the variable names. With this the values of the retain variables can also be saved if you add or delete a retain variable. **Note:** To save a variable, it must be highlighted as *embedded* in the zenon Logic Workbench. If you do not activate the checkbox, the retain data are saved the conventional way. After changing the retain variables, the Runtime starts with the initial values for all retain variables. Default: *deactivated*
Cyclic saving of retain variables	If you deactivate this checkbox, the retain data are no longer saved cyclically, but only when the Runtime is closed. They are still saved after a change, but not more frequently than about every 10 ms. Default: activated
Real-time priority	Sets the zenon Logic Runtime process to the windows priority *Real-time*. If the flag is not set, it means priority *Normal*.

Information

If you activated the **Save retain variables with names** option in zenon and also want to use it for a manual Runtime start, you must make sure that the RETAINBYNAME option is set in the K5DBXS.INI file in the [CMD] area. This ensures that all retain variables are highlighted as embedded when the zenon Logic Workbench is started.
Note: The INI file is only checked at the start of the zenon Logic Workbench. Changes during the Workbench runs have no effect.

Information

Retain variables

Retain data contain only the value of the zenon Logic variables not their status. This means for the start:

- Warm start: The status which was set for a variable is restored - regardless of whether it is a retain variable or not.
- Cold start with retain variables: Only the value of the retain variable in zenon Logic is restored, not the status.

Information

If the zenon Logic Runtime has been started with the zenon Runtime or with the zenon Logic Runtime Manager, changes done in this user interface are not effective after a restart of the zenon Logic Runtime.

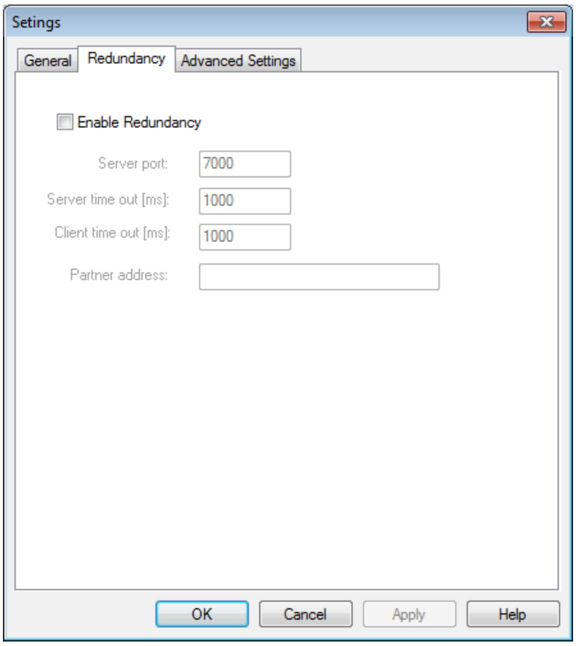
Please make the changes as usual in the zenon Logic project properties of the zenon Editors or in the zenon Logic Runtime Manager.

In such cases the dialog helps to determine which parameters were forwarded to the zenon Logic Runtime.

The changes to the settings which were done in this dialog are only effective if the zenon Logic Runtime is started directly (e.g. using the Windows Explorer).

8.1.2.2 Redundancy

In tab **Redundancy** you can see the redundancy settings for the zenon Logic Runtime.



□ Enable Redundancy

Server port: 7000

Server time out [ms]: 1000

Client time out [ms]: 1000

Partner address:

You can check or enter the following entries:

Entry	Meaning
Activate redundancy	Runtime runs in redundancy mode
Server Port:	Port for replication connection
Server Timeout [ms]	Timeout in order to determine if the passive client is available. After the timeout expires, the replication is stopped.
Client Timeout [ms]	Timeout in order to determine that the active server is no longer available. After the timeout expires, the passive client becomes the active server.
Partner address:	IP address of the redundancy partner.

Information

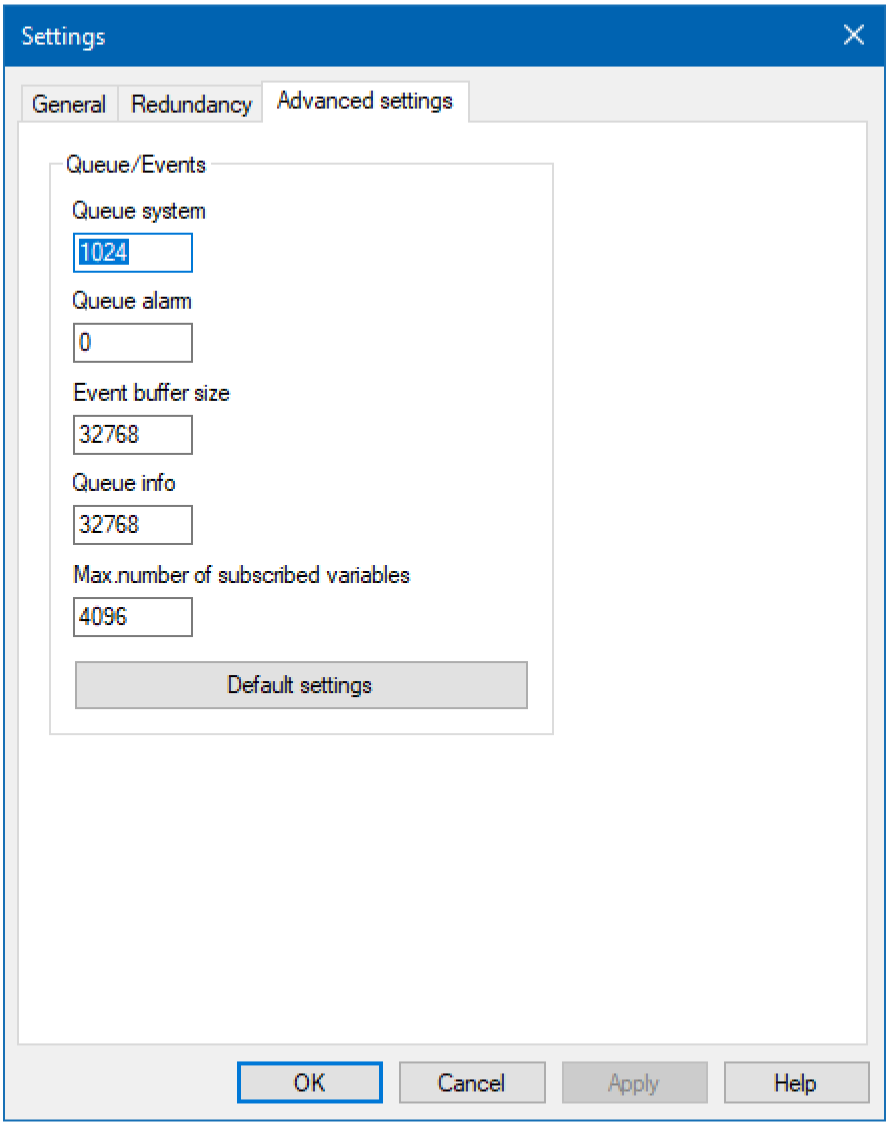
You can find details on configuring the redundancy of zenon Logic and zenon in the Redundancy zenon Logic and zenon (on page 73) chapter.

LIMITATIONS

Redundancy is not supported in zenon Logic RTK Runtime.

8.1.2.3 Advanced settings

Using this tab, settings are made for spontaneous communication with external applications (binding, zenon driver etc.).



Queue system: 1024

Queue alarm: 0

Event buffer size: 32768

Queue info: 32768

Max.number of subscribed variables: 4096

The following settings are available:

Parameter	Description	Range of values
System	Max. length of the system queue. In this queue, for example, overflows of the event queue are buffered.	*1024 ... 65520* Default:*1024*

Parameter	Description	Range of values
Alarms	Length of the alarm queue (not used at the moment)	*0 ... 65520* Default:*0*
Buffer size for events	Length of the event queue in bytes. The events of the zenon Logic Runtime event sever are written in the queue. From there they are sent via TCP. In this way, every value change in Runtime which is communicated via an appropriate interface (e.g. straton NG or binding for distributed applications), generates an event. If it is communicated via straton NG, the event queue is processed in the hardware cycle of the driver. If this queue overflows, the overflow messages are written in the system queue. This means the remaining value changes are discarded. **Attention:** A maximum of 255 events can be transferred per cycle when communicating via the **stratonNG driver**.	*0 ... 65535* Default:*32768*
Messages	Length of the info queue. In this queue, the messages to the zenon Logic Workbench are buffered.	*1024 ... 65520* Default:*32768*
Max. number of advised variables	This parameter defines how many variables can be processed by the event server per cycle. Together with the zenon Runtime, variables are advised for communication if they, for example, are displayed in a screen or used in an archive. If the number of advised variables exceeds the set number, these variables show INVALID Bit in the zenon Runtime.	*128 ... 65535* Default:*4096*
Reset to default settings	Restores the default settings.	

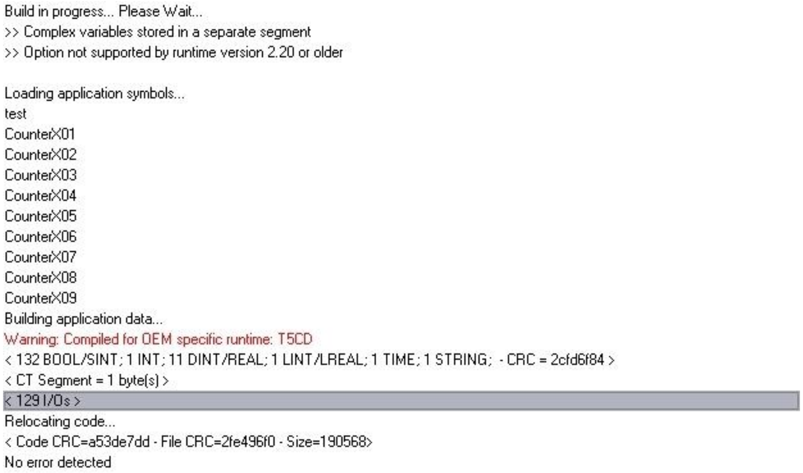
Parameter	Description	Range of values

Information

A value change in the event queue occupies 12 bytes (overhead) + the value (e.g. 4 bytes for a *DWORD*, 256 bytes for a string that is 255 characters long). Therefore, in order to be able to buffer 1000 value changes within a zenon driver hardware cycle for *DWORDs*, the event queue must be 1600 bytes large [=1000 * (12 + 4)].

Information

If in large projects more than 1024 variables are transferred between zenon Logic and zenon via the *straton32.exe* driver, the **Max. number of advised variables** parameter must be increased accordingly. An easy method to find out the number for the **Max. number of advised variables** is to look at the value for the compiled I/Os in the output window of the zenon Logic compilers. See illustration below.



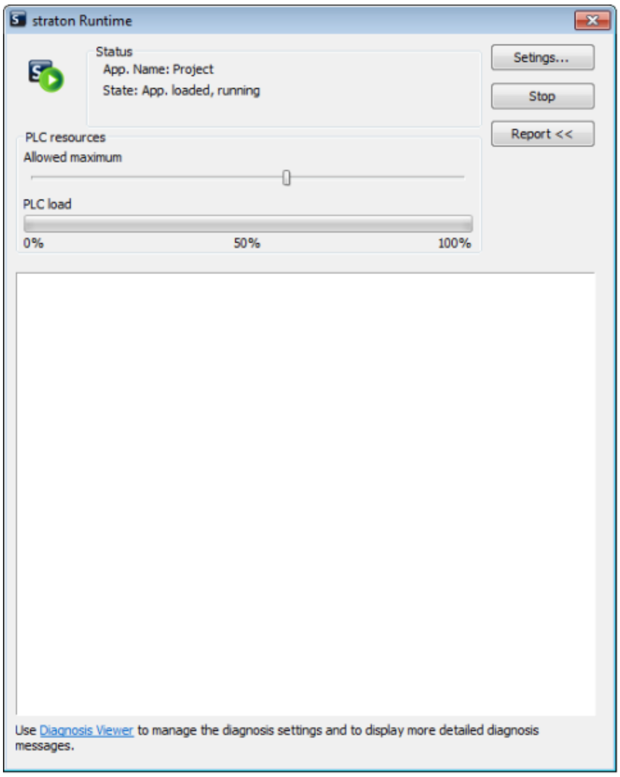
```
Build in progress... Please Wait...
>> Complex variables stored in a separate segment
>> Option not supported by runtime version 2.20 or older

Loading application symbols...
test
CounterX01
CounterX02
CounterX03
CounterX04
CounterX05
CounterX06
CounterX07
CounterX08
CounterX09
Building application data...
Warning: Compiled for OEM specific runtime: T5CD
< 132 BOOL/SINT; 1 INT; 11 DINT/REAL; 1 LINT/LREAL; 1 TIME; 1 STRING;  - CRC = 2cfd6f84 >
< CT Segment = 1 byte(s) >
< 129 I/Os >
Relocating code...
< Code CRC=a53de7dd - File CRC=2fe496f0 - Size=190568>
No error detected
```

Information

The communication interface provides support for up to 255 clients. Expanding or decreasing the queue parameter has a corresponding effect on the required memory.

8.1.3 Messages

Runtime messages can be displayed in the zenon Logic Runtime user interface. Click on the **Messages>>** button to activate the view. The message display opens. Click on the **Messages<<** button to close the message display.



App. Name: Project
State: App. loaded, running

Use Diagnosis Viewer to manage the diagnosis settings and to display more detailed diagnosis messages.

For actions in the message display, click in the message display with the left or right mouse button. You obtain a context menu:

Menu entry	Description
Display time stamp of messages	Adds a time stamp to each message.
Copy messages to the clipboard	Copies selected entries to the clipboard.
Delete all messages	Deletes all messages from the message window.

You administer the settings for zenon Logic logging and the drivers with the Diagnosis Viewer.

8.2 Licensing

zenon Logic Runtime needs a valid license to start. Otherwise the 30 minute demo mode (on page 47) is started. Licensing is carried out by means of **COPA-DATA license administration**.

SHOW INFORMATION ABOUT THE ZENON LOGIC RUNTIME LICENSE

To view the license information:

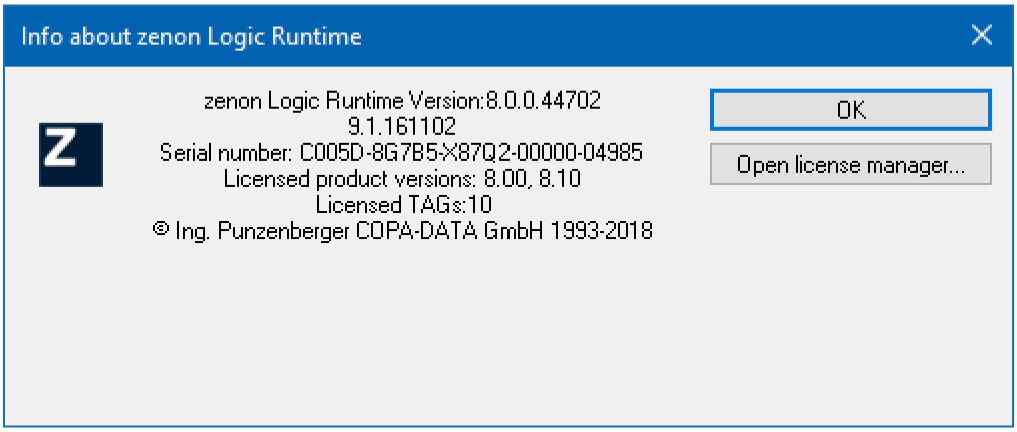
1. With Runtime running, double-click on the symbol for zenon Logic in the system tray.

   The dialog with status information is opened.

2. In the top left corner, click on the zenon Logic logo.
3. In the drop-down list, select **About zenon Logic Runtime...**.

   The dialog with the license information is opened.

LICENSE INFORMATION DIALOG



Info about zenon Logic Runtime

zenon Logic Runtime Version:8.0.0.44702
9.1.161102
Serial number: C005D-8G7B5-X87Q2-00000-04985
Licensed product versions: 8.00, 8.10
Licensed TAGs:10
© Ing. Punzenberger COPA-DATA GmbH 1993-2018

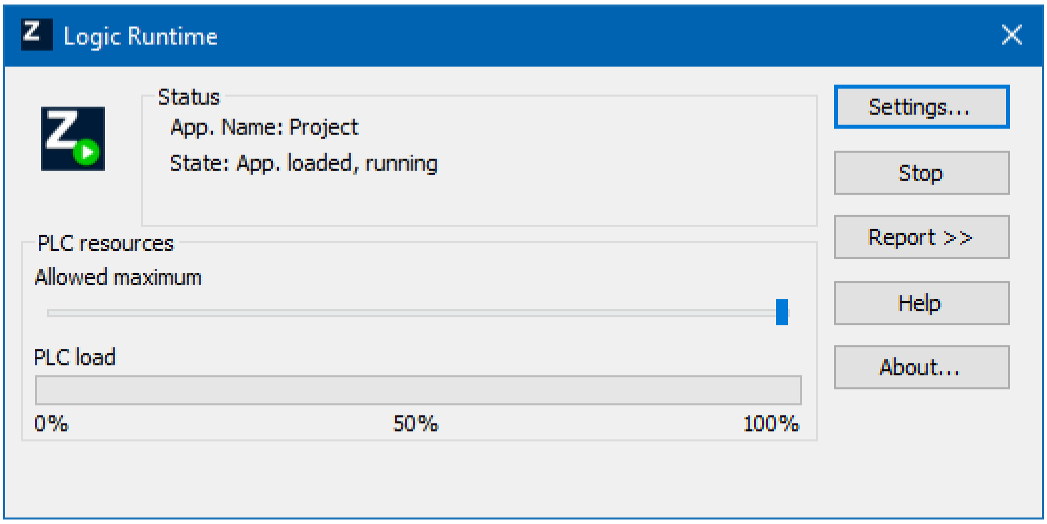
The dialog shows information about the license currently being used.

Option	Description
OK	Closes the dialog.
Open license administration...	Opens the **License Manager**. Licenses can be activated, returned and managed with this tool. The serial number is needed for this.

8.2.1 TAG handling zenon - zenon Logic

The number of the TAGs used in zenon Logic is the number of I/Os in zenon Logic minus the number of the used I/Os with a zenon or zenon Logic profile. These I/Os are already considered at the TAG calculation in zenon.

This makes sure that no I/Os are counted twice.



Status
App. Name: Project
State: App. loaded, running

The I/Os used are displayed in the <CA_PRODUCTNAME< Runtime user interface.

8.2.2 Demo mode

If, when starting zenon Logic Runtime, no valid serial number is found, zenon Logic Runtime starts in demo mode. The Demo Mode stops the zenon Logic Runtime after 10 minutes. With a special demo serial number, zenon Logic Runtime runs continuously for 30 days. After these 30 days then the maximal time of the zenon Logic Runtime is again 10 minutes.

ZENON LOGIC MINI

You did not buy the zenon Logic Runtime license but you own a valid zenon Runtime license. Then a mini version of the zenon Logic Runtime is licensed automatically.
With the mini version up to eight TAGs (=variables) can be used by zenon Logic via input/output mechanisms. TAGs are variables of field bus drivers and variables with profiles (e.g. O&M profile zenon Logic). Excluded from of the calculation are the zenon internal profiles **STRATON** and **zenOnRT** for zenon-internal communication.

Calculation example:

- Four variables for communication with zenon
  (are not calculated)
- four variables for the communication with Profibus
- four variables for the communication with the **IEC 61850** Client.

This equals the maximum number of eight TAGs (0 + 4 + 4 = 8).

8.3 Start the zenon Logic Runtime

The zenon Logic Runtime supports multi-instances and therefore needs parameters (transfer parameters), when it is started.

There are the following possibilities for starting zenon Logic Runtime:

- with the help of the zenon Runtime
- with the help of the zenon Logic Runtime Manager
- With a BAT file
- Using the command line

START WITH THE HELP OF THE ZENON RUNTIME

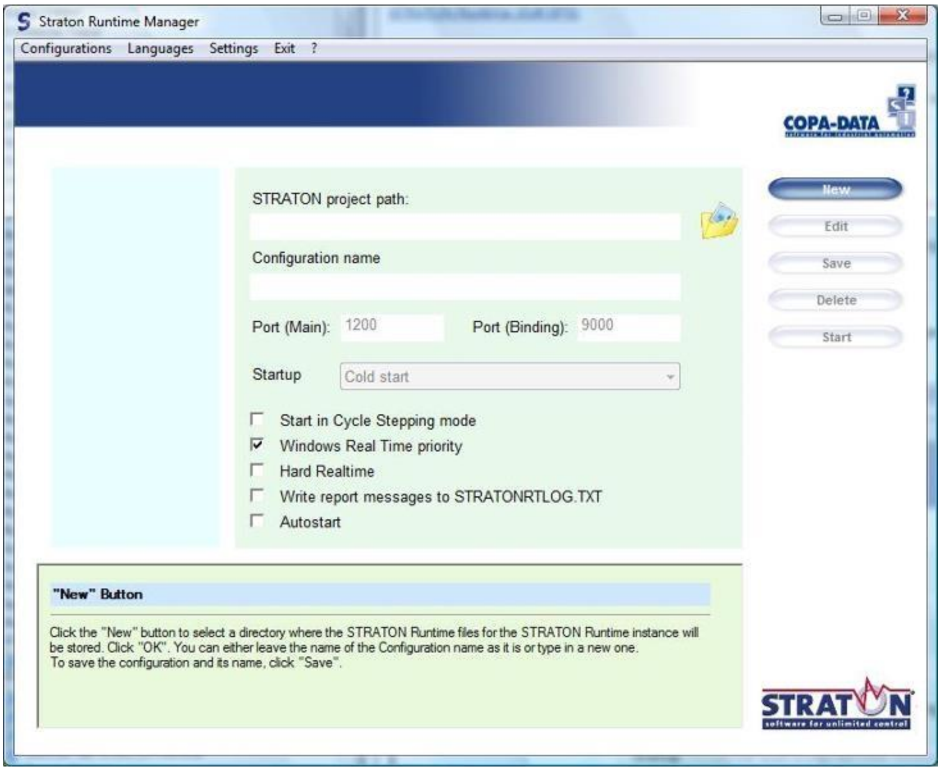
In order to start the zenon Logic Runtime with the zenon Runtime:

- Select the zenon Logic project in the zenon Editor.
- Change the *Start type* (**Runtime**) as you wish. The properties of **Start type** are described in the embedded help in zenon.

Note: If the **Start place** property is set to *Server 1 and Server 2*, then *localhost* has to be used in the **Host** as a value. Otherwise zenon Logic Runtime can no longer start.

START WITH THE HELP OF THE ZENON LOGIC RUNTIME MANAGER

You can use the zenon Logic Runtime Manger to start the zenon Logic Runtime manually. You can find the zenon Logic Runtime Manager in the COPA-DATA area in the Windows® start menu. The zenon Logic Runtime manager is used to start zenon Logic Runtime instances on a PC. You can for example chose *hard real time* here in order to start the zenon Logic real time version (zenon Logic RTK) or to start a zenon Logic Runtime instance during the booting of the system ("Auto start").



STRATON project path:

Configuration name

Port (Main): 1200 Port (Binding): 9000

Startup Cold start

□ Start in Cycle Stepping mode
☑ Windows Real Time priority
□ Hard Realtime
□ Write report messages to STRATONRTLOG.TXT
□ Autostart

"New" Button

Click the "New" button to select a directory where the STRATON Runtime files for the STRATON Runtime instance will be stored. Click "OK". You can either leave the name of the Configuration name as it is or type in a new one.
To save the configuration and its name, click "Save".

You receive accurate information about the individual elements of the zenon Logic Runtime manager by moving the mouse pointer over them. The corresponding help is displayed in the property help.

You receive the complete help about this application by clicking the **?** in the menu bar.

> **Attention**
>
> Runtime Manager must be started with administrator rights.

START IN A DOS SHELL

zenon Logic Runtime can be started using the command line interface or a BAT file. This start process is also valid for starting under Windows CE. If the path under CE contains a space, the path parameter must be defined between individual quote marks. Example: -PATH='\hard disk\project1\'

For the command the following is true:

- all commands must be entered in capitals

- the following signs are allowed:
  - -
  - / (must not be used as a last character for the value of **PATH**!)

The following handover parameters are available:

MAIN PARAMETER

Settings as under **Settings -> General** (on page 38) in the user interface of zenon Logic Runtime.

Parameter	Description
PORT=<Zahl>	Communication port. If no parameter is set: *1200* (Default)
PORTTRACE=<Zahl>	Event port. If no parameter is set: *PORT+7800*
PATH=<DB-Path>	Storage area of the zenon Logic application code. If no parameter is set: zenon Logic Runtime folder. **Note:** - Must be set. - Path must not end with a Backslash (\). - For error detection, the driver checks the project name at the end of the path. - zenon Runtime discovers, when starting, zenon Logic Runtimes that are already running by comparing the complete path.
ZENPATH=<Pfad>	zenon path
ZENPROJECT=<Name>	zenon project name. **Note:** Must be set.
DONTHIDE	Do not hide the Runtime after starting (it is the only parameter without the = character)
STEP=<0|1>	Activation/deactivation of the stepping mode: - *1*: Start in stepping mode - *0*: no stepping mode

Parameter	Description
	- Without parameters: no stepping mode
START=<0|1|2|3>	Type of Runtime start: - 0: Display setup dialog - 1: Coldstart - *2*: Warm start - *3*: Hot start - Without parameters: Warm start
LOWPRIORITY=<0|1>	Priority: - *0*: low priority - *1*: Windows real time priority - Without parameters: low priority
WRITELOG=<0|1>	Setting for logging: - *0*: Do not create LOG - *1*: Create LOG - Without parameters: Create LOG The log file that is stated in the **PATH** parameter is created. Name: **STRATONRTLOG.TXT**

ADDITIONAL PARAMETERS

Parameter	Description
REMPATH=<Pfad>	Path for remanent data. If no parameter is set: **PATH** folder
DELAY=<Zahl>	Start delay in [s]: Delayed start of the zenon Logic Runtime. If no parameter is set: *0*
MAXUTILPRCT=<Zahl>	Position of the slider for maximum PLC workload in %. If no parameter is set: *100*

EVENT PARAMETER

Settings as under **Settings -> Advanced settings** (on page 42) in the user interface of zenon Logic Runtime for the queue/events.

Parameter	Description
QSSYSTEM=<Zahl>	Size of system queue. If no parameter is set:*1024*
QSALARM=<Zahl>	Size of alarm queue. If no parameter is set: *0*
QSEVENT=<Zahl>	Size of event queue. If no parameter is set: *8192*
QSINFO=<Zahl>	Size of info queue. If no parameter is set: *32768*
NBEVENT=<Zahl>	Maximum number of recordable events. If no parameter is set: *1024*
EVENTCON=<Zahl>	Number of possible event connections. If no parameter is set: *256*

REDUNDANCY PARAMETER

Settings as under **Settings -> Redundancy** (on page 41) in the user interface of zenon Logic Runtime.

Parameter	Description
REDENABLE=<0,1>	Redundancy on/off. Without parameters: *0*
REDADDR=<IP>	Redundancy partner address.
REDPORT=<Zahl>	Redundancy port for replication.
REDCLITIMEOUT=<Zahl>	Redundancy client time out [ms]; time out which is used to determine when a passive client becomes an active server. If no parameter is set: *1000*
REDSRVTIMEOUT=<Zahl >	Server Timeout redundancy [ms]. Time-out for ending the replication. If no parameter is set: *1000*
REDZENON=<0,1>	If set to *1*, the partner address is read from the **PROJECT.INI**.

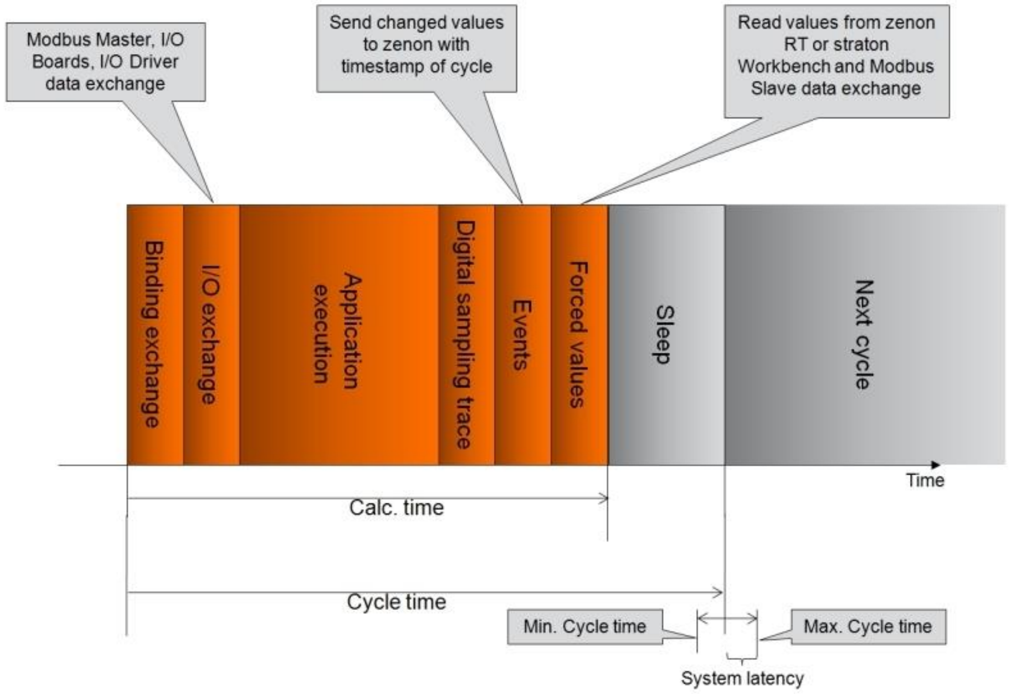
Parameter	Description
	If no parameter is set: *0*

8.4 zenon Logic Runtime cycle

The zenon Logic Runtime cycle consists of:

- **Calc. time:** Time for data exchange, calculation, save calculated values, read and write modified values, idle time.
- **Cycle time:** Complete cycle time determined by system latency.
- **System latency:** Response time of the system.

These values can be displayed in the zenon Logic user interface. Use the shortcut **Alt+S** or **Shift+Alt+A**.

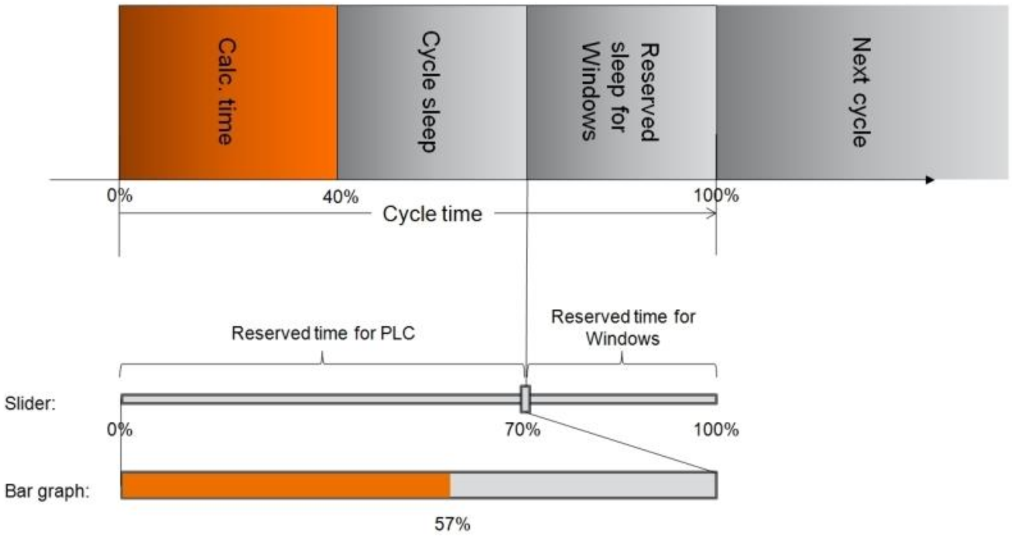

Modbus Master, I/O Boards, I/O Driver data exchange
Send changed values to zenon with timestamp of cycle
Read values from zenon RT or straton Workbench and Modbus Slave data exchange
Binding exchange
I/O exchange
Application execution
Digital sampling trace
Events
Forced values
Sleep
Next cycle
Time
Calc. time
Cycle time
Min. Cycle time
Max. Cycle time
System latency

Property	Description
Binding exchange	Reading of the Binding values which are published in the network and which were configured by the application (if they were configured).
I/O exchange	Data exchange between the I/O drivers (except Modbus slave/server) and the I/O boards (if they were configured).

Property	Description
Application execution	Execution of the programs.
Digital sampling trace	Execution of the digital sampling (if it was configured).
Events	Sending of the events (Binding or **straton32.exe** driver of zenon) (if they were configured).
Forced values	Reading of the forced values of zenon from the zenon Logic Workbench (online debugging) and data exchange of the Modbus slave/server (if they were configured).
Sleep	Wait the remaining time until the cycle time is over. When cycle setting is *As fast as possible (free run)*, the time is always 1 ms. The **Sleep** time is used in order to execute tasks of the operation system and of other windows applications.

TIME ALLOCATION PLC AND WINDOWS

With the help of the slider in the zenon Logic Runtime user interface, you can determine how much time is reserved for the PLC and for Windows. The value displayed in the bar display always refers to the ratio set in the slider.



BEHAVIOR WITH CONFIGURED FREE RUNNING MODE

The “As fast as possible (free run)” option is not available for the zenon Logic Runtime. If this option is activated during configuration in the zenon Logic Workbench, this is automatically changed into a *triggered cycle time* with a value of *1 ms* when executed in the zenon Logic Runtime.

Note: Free running (as fast as possible) is configured in zenon Logic in the **Options** dialog. To do this, call up the **Project** menu entry in the zenon Logic Workbench. Select the **Parameter...** entry there Select the the **cycle time** property in the **Options** dialog in the **Runtime** area. The **Cycle time** dialog with the corresponding options opens when double clicking on the property.

8.4.1 Error message cycle time

If the **cycle time** is exceeded, messages are displayed in the zenon Logic Runtime or in the Workbench and corresponding entries are written to the LOG file.

Error message user interface	Entry log file	Meaning
Timing configuration error. (x) cycles lost.	*Timing configuration error. (x1) cycles lost. Configured cycle time = (x2) us, maximum execution time = (x3) us, maximum PLC utilisation = (x4).*	Number (*x*) of not executed cycles due to cycle time out. The execution time is higher than the set cycle time or there are not enough CPU resources available.
Timing configuration error. PLC utilization peaked at (x)%.	*Timing configuration error. PLC utilization exceeds 80%. Configured cycle time = (x1) us, maximum execution time = (x2) us, maximum PLC utilisation =* *(x3)%."*	The execution time is higher than *80%* of the set cycle time; actually (*x*)%.

TROUBLESHOOTING

You can rectify the **cycle time** being exceeded by:

- **cycle time** increasing
- decreasing execution time, e.g.:
  - use a system with more CPU resources
  - Use code compiled with C-Compiler
  - reduce the program

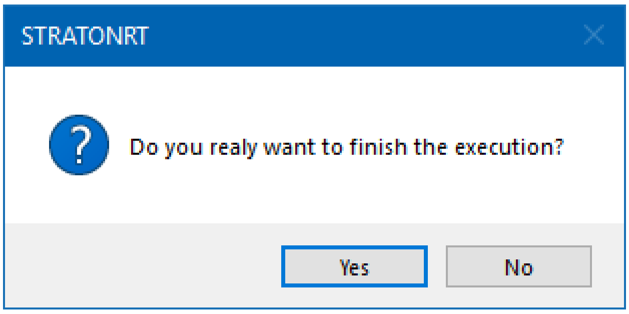
8.5 Close the zenon Logic Runtime

The zenon Logic Runtime can be closed manually either by using the context menu of the symbol in the system tray or by using the open window of the zenon Logic Runtime:

CONTEXT MENU SYMBOL

Procedure:

1. Right click on the icon in the system tray of the taskbar to open the context menu.
2. Click **Exit**.
   The following dialog will open.



STRATONRT

Do you realy want to finish the execution?

3. Click **Yes** to close the Runtime.

IN THE WINDOWS OF THE ZENON LOGIC RUNTIME

Another possibility to close the Runtime is in the open window of the zenon Logic Runtime.

1. Right click in the top left corner of the window to open the context menu.
2. Click **Close Alt+F4**.
   This will immediately close the zenon Logic Runtime.

> **Information**
>
> If the zenon Logic Runtime was started together with the zenon Runtime and for the zenon Logic project the start type 'autom. start/stop' was selected in the zenon Editor, the zenon Logic Runtime is automatically closed when the zenon Runtime is closed.

8.6 Creating the hot restart file

When you close the zenon Logic Runtime, a *Hot Restart* file is created automatically. You can trigger the writing of the *WriteHot* file via function *Hot Restart*.

Every time a *Hot Restart* file is created, a backup file is also created for which the last character of the file name is replaced with a '_'. In addition the files are created with a header which contains the *Checksum* of the data. When reading the data, header and *Checksum* are checked. If an error occurs, the backup file is used.

Function *WriteHot* has the following input parameters:

Parameter	Description
I	Trigger; if this parameter is *TRUE* the function is executed.
SYNC	If this parameter is *TRUE*, the cycle is hold for as long as the *Hot Restart* file is written. If the parameter is *FALSE* , the function is carried out asynchronous.
BUSY	This parameter is *TRUE* if the function is executed.

Information

If you call up the function several times in a cycle, only the last call is considered.

8.7 Remote systems

This chapter deals with target systems which are not on the local computer. The target system is accessed via Ethernet TCP/IP.

8.7.1 zenon Runtime and zenon Logic Runtime on one remote PC

INSTALLATION ON THE TARGET SYSTEM

First you install and license the zenon Runtime on the remote PC. The zenon Logic Runtime is automatically installed and licensed.

DOWNLOAD THE PROJECT DATA WITH ZENON LOGIC RUNTIME, START TYPE "AUTOM. START/STOP" OR "AUTOM. START"

In order to download the zenon and zenon Logic Runtime files, use the Remote Transport function of zenon. You will find more information in the chapter Remote Transport.

The zenon Logic Runtime is started and closed together with the zenon Runtime automatically.

DOWNLOAD OF THE APPLICATION DATA WITH THE ZENON LOGIC RUNTIME START TYPE "START MANUALLY"

If the zenon Logic Runtime is started manually on the target PC (e.g. during boot up of the operating system), the PLC code has to be loaded manually to the target platform with the zenon Logic Workbench. For that

- In the zenon Editor, enter the right parameters in the properties in the **Workbench** group. **Target host** and **Target port**
- in the zenon Logic Workbench click on the **Online** button.
- with this the zenon Logic Workbench establishes a connection to the target system.
- now you can transfer the PLC code to the target system via button **Download**; the progress of the download is displayed with the help of a progress bar
- after the download the PLC starts automatically with the new application.
- If the download was successful and the PLC program runs on the target system faultless, **RUN** is displayed in the toolbar

8.8 zenon Logic Runtime with Real Time Kernel (RTK)

The zenon Logic RTK (Realtime Kernel) is the zenon Logic Runtime system for hard realtime. In contrast to the "normal" zenon Logic Runtime, this runs at the same time and independently of the operating system. The highest availability and security against failure is thus guaranteed. zenon Logic is thus also still available if the operating system is blocked or no longer available.

Requirements:

- zenon Logic RTK is available for the following Windows operating systems:
  - Windows 7, 32-bit version
  - Windows 8 and 8.1, 32-bit version
  - Windows 10, 32-bit version

Attention: zenon Logic Runtime with RTK can only run on 32-bit operating systems.

- zenon Logic RTK can be started with one of the following possibilities:
  - zenon Logic Runtime Manager (**hard real time** (on page 93) option must be set)
  - By means of the command line
  - With a BAT file.

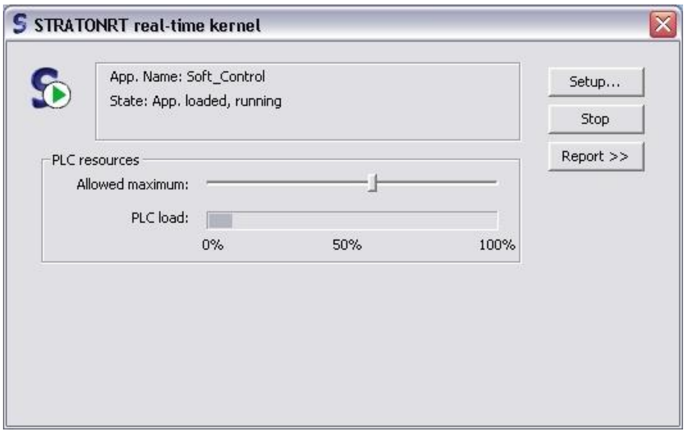
Attention

Do not operate two real-time extensions in parallel on one computer.
This means:

- If another real-time extension is already active on the computer, do not start the zenon Logic RTK.
- Only one instance of the zenon Logic RTK can be started at a time.

The Realtime Kernel is designed to trigger the zenon Logic Runtime cycle independent of the operation system. So the zenon Logic Runtime cycle is guaranteed to be executed.

The PC's part of the calculating capacity can be set with a slider.



STRATONRT real-time kernel

App. Name: Soft_Control
State: App. loaded, running

PLC resources
Allowed maximum:
PLC load:

Depending on the size of the PLC application and the set cycle time, the slider has to be adjusted to the PLC load. The slider can be set between 10% and 85%. The current load is visualized as a bar and indicates whether the permitted maximum for executing the PLC cycle is sufficient.

If the display reaches the 100% mark, there is a danger of cycle overflows! In this case set the slider higher (towards 85%). If this is not possible due to programs in Windows (e.g. long screen opening times of zenon), the cycle time of the zenon Logic application has to be increased.

Attention

If the bar reaches the 100% mark, there is a danger of cycle overflows!

Example

Slider at 10%, cycle time 10ms:

The zenon Logic RTK must process the PLC program in 1 ms which equals 10% of the set cycle time of 10 ms.

Slider at 50%, cycle time 5ms:

For executing the PLC program 2.5ms are available for the zenon Logic RTK.

Slider at 85%, cycle time 20ms:

For executing the PLC program 17ms are available for the zenon Logic RTK.

Formula

To ensure that the slider setting does not lead to cycle overflows in conjunction with the cycle time, the following formula can be applied:

```
cycle time > (100 / Slider position [%] ) * calculating time
```

In order to find out the calculating time of the PLC cycle, click **Shift+Alt+A** and note the *Max. Cal.* value. *time [µs]*.

Assumption:

Max. cal. Time: 274µs

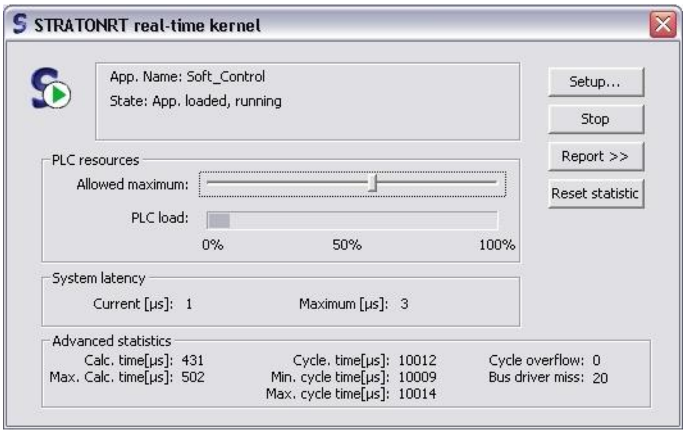
Slider position: 50%

Set cycle time: 1ms (1000µs)

Calculation:

```
(100/50)*274 = 548µs
```

This means that the defined cycle time of 1 ms is sufficient.



STRATONRT real-time kernel

App. Name: Soft_Control
State: App. loaded, running

PLC resources
Allowed maximum:
PLC load:

System latency
Current [µs]: 1
Maximum [µs]: 3

Advanced statistics
Calc. time[µs]: 431
Max. Calc. time[µs]: 502
Cycle. time[µs]: 10012
Min. cycle time[µs]: 10009
Max. cycle time[µs]: 10014
Cycle overflow: 0
Bus driver miss: 20

⚠Attention

Note: If you do not stick to the described formula, the determinism which is based in the cycle time is not given anymore!

CYCLE TIME

Cycle times from 100 µs or higher can be set.
Note: The actual use of the set cycle time depends on the system and the selected HAL Timer. Information on the current system load is provided by the system load time (keyboard shortcut: **ALT+S**) and **Advanced statistics** (keyboard shortcut: **SHIFT+ALT+A**).

Description of the measured values displayed:

System latency (Alt+S)	
Current [µs]	Current system latency (see also Runtime cycle (on page 53))
Maximum [µs]	Maximum system latency (since the start of the Runtime)

Advanced statistics (SHIFT+ALT+A)	
Cal. time [µs].	Current calculation time of the last cycle
Max. cal. time [µs].	Maximal calculation time for a PLC cycle (since the start of the Runtime)
Cycle time [µs]	Current cycle time
Min. cycle time [µs]	Minimal cycle time (since the start of the Runtime)
Max. cycle time [µs]	Maximal cycle time (since the start of the Runtime)
Cycle overflow [1]	Counter for cycle time overruns (since the start of the Runtime)
Driver overflow [1]	Counter for failed PLC cycles of Windows (non-realtime) drivers (since the start of the Runtime)

The value of the cycle time can always only be a multiple of the HAL Timerperiod (time basis). The time basis depends on the selected timer and its time basis (see also zenon Logic RTK configuration (on page 63)).

That means: A set cycle time of 10 ms of a hardware with a time basis of 122 µs is not executed with exactly 10,000 ms but with 10,004 ms.

8.8.1 Restrictions and requirements of the zenon Logic RTK

HARDWARE REQUIREMENTS FOR THE ZENON LOGIC RTK

ACPI:

The hardware (PC or IPC) and the operating system must support ACPI:

- ACPI-Multiprocessor-P
- ACPI-PC (Advanced Configuration and Power Interface)
- ACPI uniprocessor PC

You can check whether your PC supports ACPI in the system control (system/device manager/computer).

CPUS:

The zenon Logic RTK is supported from the following processor generations on:

- Pentium II or younger
- AMD Athlon XP or younger

LIMITATIONS

The zenon Logic Real Time Kernel (RTK) runs as a separate process at hardware level independently of the operating system. In contrast, zenon Logic Runtime runs on Windows®. The real-time-enabled zenon Logic RTK has a few restrictions compared to the zenon Logic Runtime.

- Virtual Machines are not supported.
- zenon Logic redundancy is not supported
- Not all function blocks are supported.
- Not all I/O drivers are supported.
- Some real-time drivers must be registered/deregistered manually.

FUNCTION BLOCKS

The following funciton blocks are not processed directly in the Kernel Mode, but only emulated:

- all file operations (category 'Files')
- all TCP/IP modules (category 'TCP-IP')

Following function blocks are not supported by the zenon Logic RTK:

- day_time_local (Category 'Clock')

> **Information**
>
> The respective configuration of Runtime can be loaded in the zenon Logic workbench. To do this, select the **Configuration** entry in the Workbench in the context menu of the loaded project.
>
> After you have selected the configuration, the function blocks which are not supported are marked red in the library.

I/O DRIVER

The following zenon Logic I/O drivers are executed by the zenon Logic RTK in Kernel Mode (real time):

- Profibus (Hilscher CIF Profibus)
- Hilscher SYCON universal (Hilscher SYCON configuration)

You can find more information about supported I/O drivers here: zenon Logic I/O driver support (on page 82)

REGISTER AND DE-REGISTER REAL TIME DRIVERS

There is a small chance that the real time driver of the zenon Logic RTK must be registered and de-registered manually. In order to do this start the command prompt via:
Start > execute > cmd.

Use the following syntax:

- Register:

  <Folder of the zenon Logic RTK>:\ stratonrtkvm -regsrv
- De-register:

  Folder of the zenon Logic RTK>:\ stratonrtkvm -unregsrv

Example of the <folder of the zenon Logic RTK>: *%Programme (x86)%\COPA-DATA\zenon 820*

8.8.2 zenon Logic RTK configuration

The zenon Logic Real Time Kernel (RTK) runs as an own instance on hardware level and is independent of the operating system. The Timerticks which are used for control are directly obtained from the motherboard. Which timer are available depends on the used hardware. The configuration is carried out by the zenon Logic RTK configurator.

8.8.2.1 Start RTK configuration

Double click the button for zenon Logic RTK Configurator in the windows control panel.

STRATON RTK

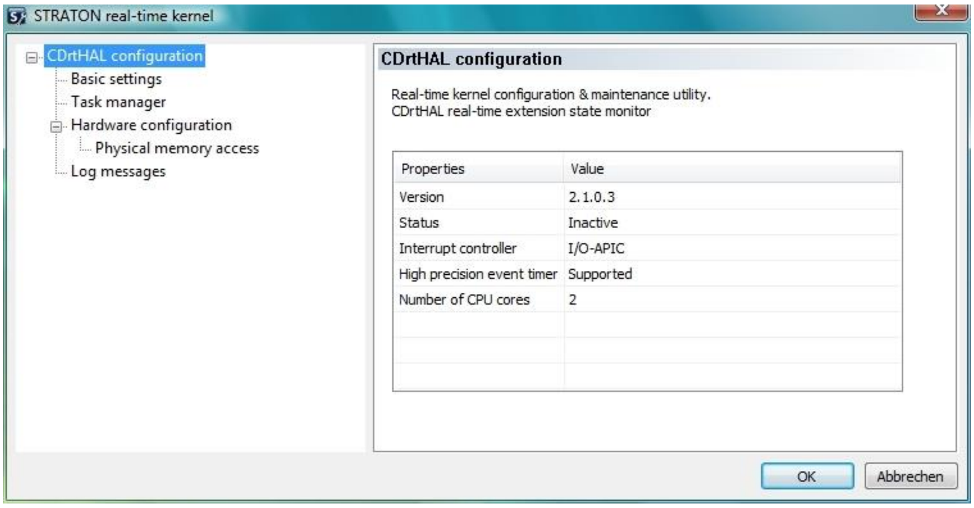
The window for editing the configuration opens. In the left window you see a menu tree with the following options:

- CDrtHAL configuration
- Default settings
- Task manager
- Hardware configuration
- Access to the physical memory.
- LOG messages

In the right-hand window you can see information about the menu items and configure the RTK.

8.8.2.2 RTK basis information

After the start menu item 'CDrtHal configuration' is active in the left-hand window of the configuration software. In the right-hand window displays basic information about the configuration and the system:



CDrtHAL configuration

Real-time kernel configuration & maintenance utility.
CDrtHAL real-time extension state monitor

Properties	Value
Version	2.1.0.3
Status	Inactive
Interrupt controller	I/O-APIC
High precision event timer	Supported
Number of CPU cores	2

Property	Information
Version	Version of the real time driver (*CDrtHAL*).

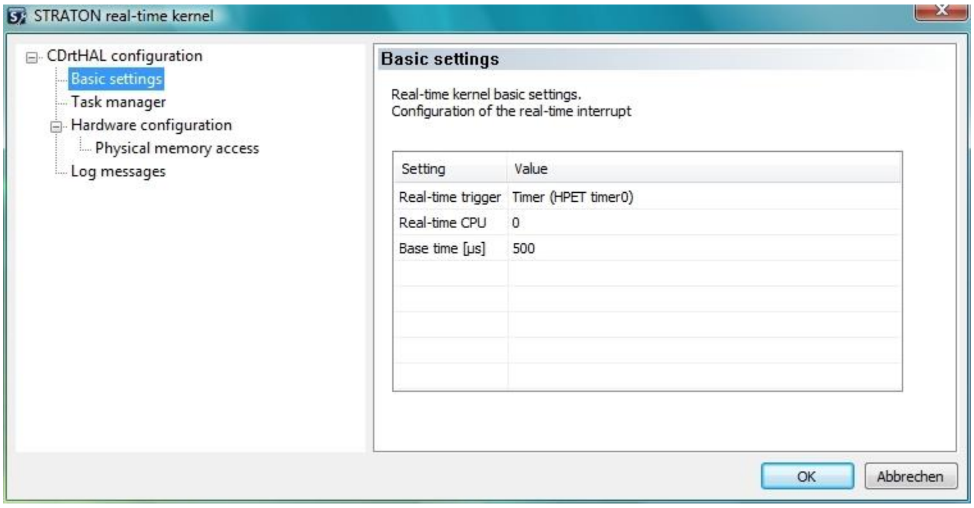
Property	Information
State	RTK is active or inactive
Interrupt controller	Type of interrupt controller.
High precision event timer	Support depends on hardware:
Process cores	Number of available process cores.

8.8.2.3 RTK basic settings

RTK BASIC SETTINGS

In the basic setting you define the following:

- Source for the Timertick
- CPU on which the RTK is carried out
- Basic time as basis for the multiplication factor for the calculation of the cycle time



Basic settings

Real-time kernel basic settings.
Configuration of the real-time interrupt

Setting	Value
Real-time trigger	Timer (HPET timer0)
Real-time CPU	0
Base time [µs]	500

TIMER FOR REAL-TIME SOURCE

Which timer is displayed on the selection list, depends on the used hardware. This information is also displayed on the start page of the RTK configuration. Available can be:

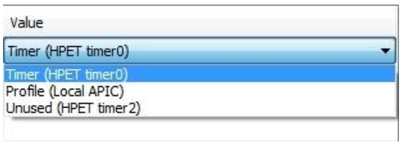
- Interrupt-Controller APIC
- Interrupt-Controller PIC
- HPET (High precision event timer)

The follwoing possibilities arisefrom this:

1. PIC without HPET
    - Timer (8254 Timer), basic times: 334 µs, 143 µs, 112 µs, 53 µs
    - Profiles (real-time clock), basic times: 488 µs, 244 µs, 122 µs
2. APIC without HPET
    - Profiles (real-time clock), basic times: 488 µs, 244 µs, 122 µs
    - Profiles (local APIC), basic times: choose anywhere between 50 - 500 µs
    - Unused (8254 Timer), basic times: choose anywhere between 50 - 500 µs
3. APIC with HPET
    - Timer (HPET timer o), basic times: 500 µs, 250 µs, 125 µs
    - Profiles (local APIC), basic times: choose anywhere between 50 - 500 µs
    - Unused (HPET timer 2), basic times: choose anywhere between 50 - 500 µs

Whether and to what extend basic times are available depends on the hardware (real-time clock) and the operating system. When using the 'timer' real-time source, the source for the Windows Timer Interrupt is used for the real time execution. In this case the set basic time must be an integral factor of the Windows Timer Interrupt basic time.

real-time source: depends on the operating system Select using list field (drop down).



We recommend the following basic setting: Unused (HPET Timer2).

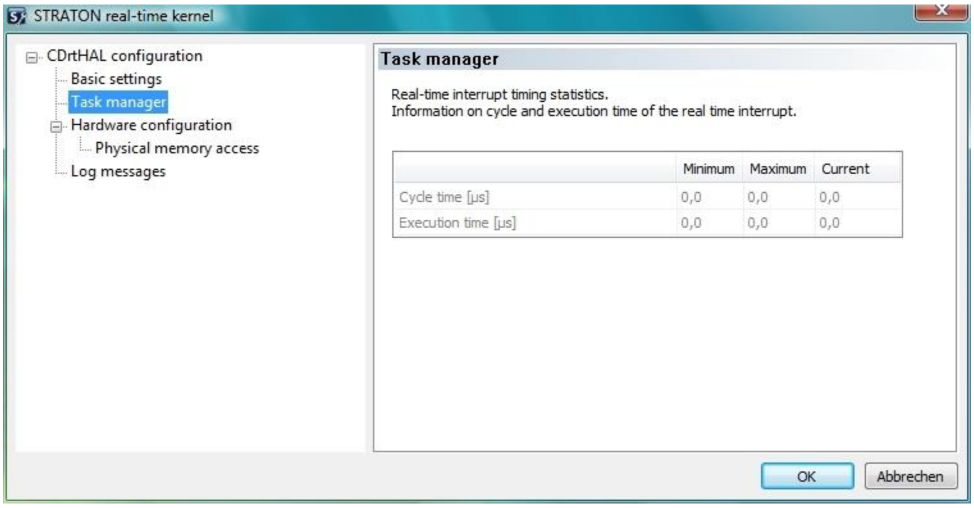
Real time CPU: Select the CPU on which the process of the RTK should be carried out.

Basic time: Basic time of the timerticks. Depending on the real time source, the basic time is selected from a list or entered freely. During operation the multiplication factor for the timer ticks refers to the entered basic time.

> **⚠Attention**
>
> When using *Profile: Real time source* the kernel profiling must not be activated under any circumstances as long as a zenon Logic RTK application is executed. For more information about kernel profiling see: http://www.microsoft.com/downloads/details.aspx?familyid=9d467a69-57ff-4ae7-96ee-b18c4790cffd&displaylang=en (http://www.microsoft.com/downloads/details.aspx?familyid=9d467a69-57ff-4ae7-96ee-b18c4790cffd&amp;displaylang=en)

8.8.2.4 RTK Taskmanager

The task manager displays cycle times and execution times.



Task manager

Real-time interrupt timing statistics.
Information on cycle and execution time of the real time interrupt.

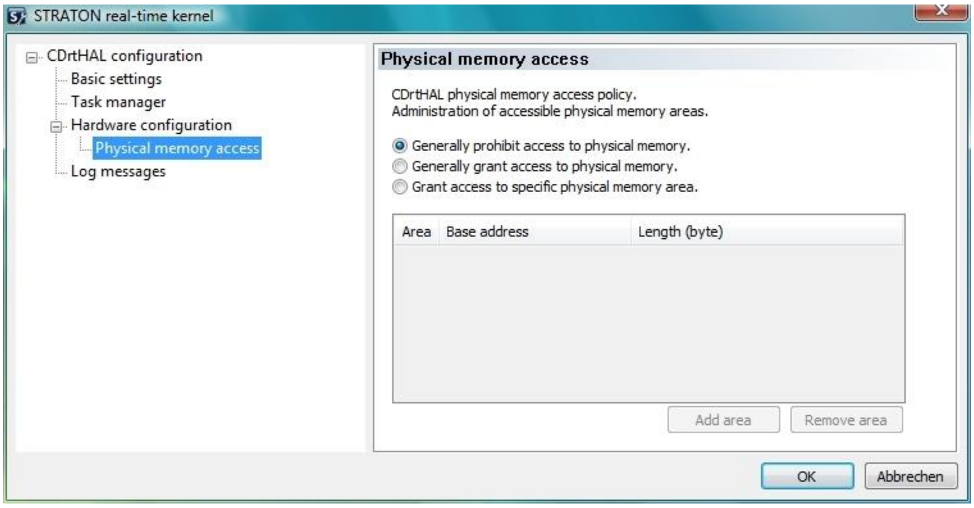
	Minimum	Maximum	Current
Cycle time [µs]	0,0	0,0	0,0
Execution time [µs]	0,0	0,0	0,0

Property	Information
Cycle time (in microseconds)	Minimal, maximal and current value. Cycle refers to the cycles of the Timerticks based on the basic time.
Execution time (in microseconds)	Minimal, maximal and current value.

8.8.2.5 RTK hardware configuration

A separate reserved storage area is needed for the communication with the I/O drivers (e.g. Profibus) which are executed in the kernel mode. zenon Logic RTK and the I/O driver system agree on an area on which both have read and write access.

The configuration offers three settings:



Physical memory access

CDrtHAL physical memory access policy.
Administration of accessible physical memory areas.

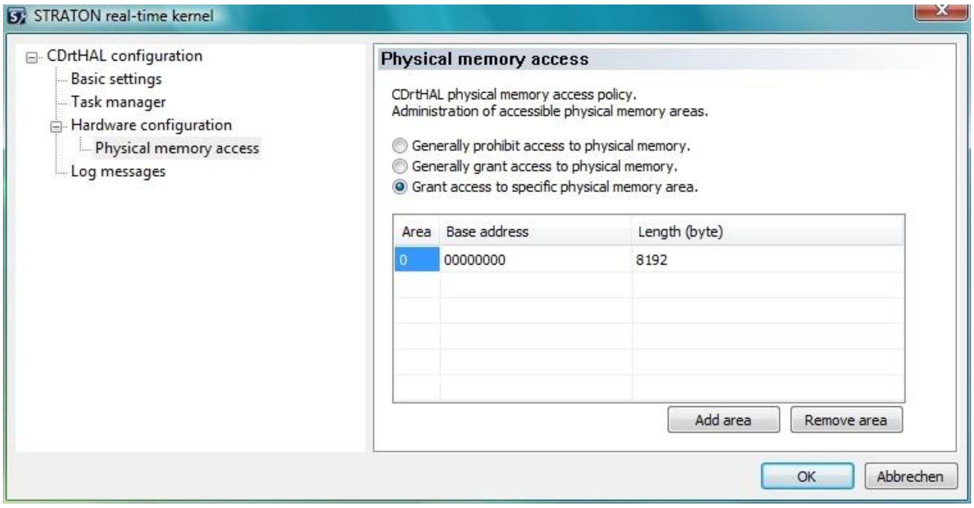
● Generally prohibit access to physical memory.
○ Generally grant access to physical memory.
○ Grant access to specific physical memory area.

Area	Base address	Length (byte)

Option	Effect
Generally prohibit access to the physical memory.	Free access memory areas are a safety risk with regards to malware. Thus access to the physical memory is denied in the shipping state. Default settings
Generally allow access to the physical memory.	Allows the communication to use the total available memory for communication, zenon Logic RTK and Profibus set their communication areas randomly. Unsafe setting which is not recommended.
Allow access to a defined physical memory area.	zenon Logic RTK and I/O driver have a strictly allocated memory area. **Recommended setting**

8.8.2.5.1Define memory area

In order to allocate a fixed memory area for zenon Logic and I/O driver, click on **Access to the physical memory** in the left window and then on **Allow access to defined physical memory area** in the right selection window.



CDrtHAL physical memory access policy.
Administration of accessible physical memory areas.

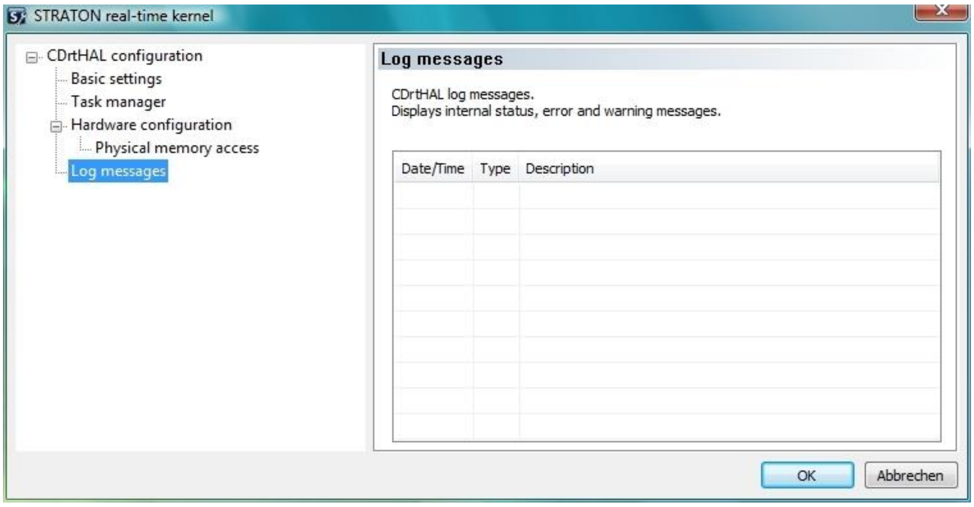
○ Generally prohibit access to physical memory.
○ Generally grant access to physical memory.
● Grant access to specific physical memory area.

Area	Base address	Length (byte)
0	00000000	8192

Property	Information
Area	The number is assigned by the configurator automatically.
Base address	Start address for the free memory. It can for example be determined with the help of the **Hilscher Tool** (included in the delivery of the card).
Length	Length of the free area. **Attention:** must be stated in Bytes.
Add area	Creates a new entry in the list.
Delete	Deletes an existing entry from the list.

8.8.2.6 RTK Logs

RTK messages are shown in the **Log messages**.



CDrtHAL log messages.
Displays internal status, error and warning messages.

Property	Information
Date/Time	When did the event happen
Type	What type of message (status, error, ...)
Description	What has happened?

8.8.2.7 Close RTK configuration

By clicking button **OK** all changes are saved and the configurator is closed. If the system needs to be rebooted, a corresponding message is displayed.

If you close the configurator by clicking the close symbol (**X**) or by clicking **Cancel**, a dialog appears asking whether you want to save the changes.

8.8.3 Blue Screen Handling

A possible Bluescreen of Death (BSOD) is recognized by the zenon Logic RTK immediately.

- If the zenon Logic RTK is configured for stopping in case of a BSOD, the shut down program - if defined - is executed immediately, i.e. before the next cycle.
- If the zenon Logic RTK is configured to continue, the next cycle is executed after a minimum delay of approximately 10 [ms] (switching of the graphics mode).
  To achieve this, in the zenon Logic application, the zenon Logic **RTK_OnBugCheck** function must be added to the zenon Logic standard library with *TRUE*.

⚠Attention

In case of a BSOD the execution of the zenon Logic application can be limited.

Following functions are no longer supported when this occurs:

- File Operations
- TCP functions
- Functions of the serial interface
- Non real-time field bus driver
- System clock functions
- RETAIN variables
- Writing of log messages (the right to make changes due to product version cycles is reserved)

In order to check in the zenon Logic application if a BSOD has occurred, use the zenon Logic function **RTK_IsBugCheck**.

OPTIONS FOR PROFIBUS MASTER (HILSCHER CIF PROFIBUS):

No restart after stop (keeping outputs): If this option is active, the outputs of the Profibus I/Os remain set after the shut down. Otherwise the outputs would be set to the defined alternate values after the shut down sequence.

8.8.4 Error codes at starting the zenon Logic RTK

The displayed error code refers to starting the *STRATONRTKVM.exe*.

Error code	Meaning
0x00000000	If an error occured in the zenon Logic RTK, usually the error is explained in a message: The log message with the error code 0x00000000 can be ignored.
0xFFFFFFFF	A timeout occured on starting the *STRATONRTKVM.exe*. Possible reasons: *STRATONRTKVM.exe* does not exist, too high load of the system. **Attention:** After such an error make sure that the *STRATONRTKVM.exe* as well as the *STRATONRTK.exe* no longer run (task manager) before restarting the zenon Logic RTK.
0x00000001 – 0x0000FFFF	Is an error code of the zenon Logic virtual machine (COPALP error message).
0xFFFF01,	Error loading the non-real-time bus drivers! probable reason:

Error code	Meaning
0xFFFF02	incompatible (outdated) version of one or more bus driver(s).
0xFFFFF000	rtHAL error code (0xFFFFF000 is added to the rtHAL code)
0xFFFFF001	Make sure, that the rtHAL driver has been installed correctly. If the problem still exists, contact your zenon support and tell them the Windows version (service pack, hot fix, ...) and the used Windows HAL. The name of the used HAL is shown in the device manager under computer. **System requirements:** ACPI, Pentium II or AMD Athlon or higher.

RTHAL ERROR CODES:

0x0	No error

POSSIBLE ERROR CODES ON STARTING

Error code	Meaning
0x1	Error while loading the driver. Reason: Driver is not registered correctly.
0x2	START could not be executed. Reason: Wrong version of the driver is installed.

ERROR CODES DURING RUNTIME

Error code	Meaning
0x6	SUSPEND could not be executed.
0x7	STOP could not be executed.

INTERNAL ZENON LOGIC VIRTUAL MACHINE ERROR CODES

```
T5RET_OK 0
T5RET_ERROR 1
T5RET_BADCODE 2 invalid app code - bad header
T5RET_BADCODETARGET 3 invalid code - bad target id
T5RET_BADCODEVERSION 4 invalid code - bad version
T5RET_EXECSIZE 5 exec buffer overflow
T5RET_BADREQUEST 6 unknown comm request
T5RET_SERVEREAD 7 cannot process read request
T5RET_UNKNOWNTIC 8 unknocn tic code
T5RET_DATA8OVERFLOW 9 too many DATA8 used
T5RET_DATA16OVERFLOW 10 too many DATA16 used
```

```
T5RET_DATA32OVERFLOW 11 too many DATA32 used
T5RET_DIVZERO 12 division by 0 in TIC (32bit)
T5RET_TIMEOVERFLOW 13 too many TIME variables
T5RET_CANTACTIME 14 active timers not supported
T5RET_SAFETICLOOP 15 awful loop in TIC (safe mode)
T5RET_OUTOFTIC 16 TIC code ointer out of range
T5RET_UNKNOWNSTF 17 not supported std function
T5RET_LDIVZERO 18 division by 0 in TIC (64bit)
T5RET_DATA64OVERFLOW 19 too many DATA64 used
T5RET_STRINGOVERFLOW 20 too many strings
T5RET_SDIVZERO 21 division by zero in TIC (8 bit)
T5RET_WDIVZERO 22 division by zero in TIC (16 bit)
T5RET_HOTRESOURCES 23 HOT restart: system resources used
T5RET_HOTMAP 24 invalid var map in hot restart
T5RET_HOTACTIME 25 change of active timer map
T5RET_HOTSTAMP 26 bad build stamp for hot restart
T5RET_HOTSFC 27 SFC changed - no hot restart!
T5RET_HOTRETAIN 28 retain def has changed - no hot
T5RET_HOTFORMAT 29 bad DB format - no hot restart!
T5RET_CYCLEOVERFLOW 30 cycle time overflow
T5RET_RETAINSTORING 31 cant store retain variables
T5RET_ARRAYBOUND 32 array index out of bounds
T5RET_RECURSE 33 recursive call of sub-program
T5RET_XVOVERFLOW 34 too many external variables used
T5RET_NOBSAMPLING 35 sampling trace not supported
T5RET_TASKOVERFLOW 36 too many tasks
T5RET_HOTCALLSTACK 37 call stack overflow (hot)
T5RET_STACKOVER 38 stack overlow
T5RET_HOTASI 39 cannot hot restart ASi
T5RET_TOXC 40 OXC limit reached
T5RET_WOXC 41 OXC info
```

8.9 Redundancy zenon Logic and zenon

zenon Logic can be used redundantly and offers the possibility to combine the zenon redundancy with the zenon Logic redundancy.

In this chapter the configuration of the zenon/zenon Logic redundancy is described. The zenon/zenon Logic redundancy is referred to as SCADA logic redundancy from here on.

Further terms:

- zenon server = dominant zenon server
- zenon SB = zenon Standby Server
- Active zenon Logic Runtime = zenon Logic Runtime instance being executed
- Passive zenon Logic Runtime = backup system (Standby) of zenon Logic Runtime

OPERATING SYSTEMS

You can find information on supported operating systems in the **zenon Logic Runtime** (on page 35) chapter, **Supported systems** section.

LIMITATIONS

Redundancy is not supported in zenon Logic RTK Runtime.

8.9.1 Requirements

Two PCs are necessary for SCADA logic redundancy. The both PCs are connected via Ethernet TCP/IP (100 MBit, recommended 1 GBit).

> ⚠**Attention**
>
> The SCADA logic redundancy requires both systems, zenon and zenon Logic, to be installed on one PC for each server.

We recommend to set the cycle time of the zenon Logic project to twice the cycle time of a single project, but at least to 50 ms.

Since SCADA logic assumes that zenon Logic and zenon communicate with each other using the **Logicto SCADA** driver, it must be ensured that the zenon server and the active zenon Logic Runtime run on the same PC.

In order to prevent a one-sided switching from zenon Logic or zenon, a synchronization between the zenon Logic Runtime and the zenon Runtime must be engineered. A watchdog is used for this.

WATCHDOG

ZENON LOGIC

The status of the zenon project (standalone/server/standby server/client) is transferred via the **Logic to SCADA** connection and is evaluated by the redundancy function block. It has a freely definable time out value as input. As output binary values for redundancy switch and active, passive and failed Runtimes etc. are output. They determine the reaction of the zenon Logic Runtime.

ZENON

A watchdog generated by the zenon Runtime driver is transferred using the **Logic to SCADA** connection and evaluated in zenon. This evaluation takes place in accordance with the settings of properties **Failure recognition after [s]** and **Function for failure recognition**. They define a time out time and a function (normally **Exit Runtime**).

In order for the correct watchdog to be evaluated, it must be clear for several active zenon Logic projects from which zenon Logic project the watchdog originated. If zenon Logic is started manually, by means of a zenon function, batch file or command line, the zenon project name (**ZENPROJECT=<Name>**) must also be given as a transfer parameter (on page 48).

⚠Attention

The start of the zenon Runtime must not be performed from the zenon Editor but with a direct call from the zenon Runtime!

The zenon redundancy must be set up. Details: See help chapter Network.

8.9.2 Configure redundancy

zenon Logic and zenon can control each other in redundant operation by means of a watchdog. If the zenon Logic Runtime fails:

- the local zenon Runtime is also closed
- both systems are switched to redundancy mode

CONFIGURATION IN ZENON

To activate redundancy:

- activate the **Redundant operation** property in the **Runtime** group of the properties of the zenon Logic project in zenon
- define:
  - **Failure recognition after [s]**
  - **Function for failure recognition**

CONFIGURATION IN ZENON LOGIC

The settings for redundancy are made in tab Redundancy (on page 41) of the zenon Logic Runtime.

WATCHDOG

With the watchdog each side checks the other for changes to this counter. If the counter for a projected time is not changed, this is reported to the program:

- zenon: a function configured in an integrated project is carried out (**Function for failure recognition**)
- zenon Logic: an **Output** function block becomes active

The watchdog examination starts as soon as the other side has changed its watchdog for the first time. This is necessary because the time is needed to start or the zenon Logic Runtime does not necessarily need to start with the zenon Runtime.

Information

If zenon Logic Runtime is a started manually, the following command line settings (on page 48) must be made:

- zenon path:

  ZENPATH=<Pfad>

- Name zenon project:

  ZENPROJECT=<Name>

- Information on zenon network project:

  REDENABLE=<0,1>

  REDZENON=<0,1>

WATCHDOG IN ZENON

zenon checks the zenon Logic Runtime Watchdog if:

- zenon Logic Runtime executes code, timeout is <> 0 and Runtime can be assigned to an integrated zenon Logic project
- zenon Logic Runtime is stopped
- zenon Logic Runtime is passive or started passively
- zenon Logic Runtime is ended and was previously passive or active

FUNCTION FOR RECOGNIZING FAILURES

With the **Function for failure recognition** property, a function is selected that is executed as soon as the timeout defined in **Failure recognition after [s]** for the watchdog has expired. The watchdog is only rechecked if Runtime is registered again.

If a function from another project is used, the project must already have been loaded when the project with the integrated zenon Logic project is loaded.

LOGGING

The zenon Logic zenRt driver connection can be logged with the Diagnosis Viewer. The module name is zenon Logic. Each message contains the name of the zenon project and the name of the zenon Logic project. The logging includes:

- Error
  - *zprj:<z-projekt> sprj:<s-projekt> Runtime <Runtime process ID> watchdog expired.* Runtime watchdog was not operated by Runtime for too long.

  - *zprj:<z-projekt> sprj: <s-projekt> Runtime: <Runtime process ID> Unknown zenon Logic project.*
    zenon Logic project cannot be assigned to an integrated zenon project.
- Warnings
  - *vm: <Vm-Prozess-ID> zenon project '<Name>' unknown*
    Name of the zenon Logic project not known in zenon.

WATCHDOG IN ZENON LOGIC RUNTIME

The following applies for zenon Logic watchdog:

- The zenon watchdog is checked regularly as long as the timeout of the >CD_PRODUCTNAME< failure recognition is <>*0*.
  This check takes place if:
  - Code is executed
  - Runtime is active or passive.

FUNCTION BLOCK ZENRT_RED AND REDSWITCH

In zenon Logic it is possible, with the help of function blocks, to check whether there is a connection to a zenon server and if it is possible to switch the zenon Logic state if necessary.

The function block **zenrt_Red** checks via the timeout information from driver "**straton_to_zenon_Runtime_Connection**" whether zenon Logic is connected to a zenon Server. If this is not the case, you can switch the affected zenon Logic to *inactive* with the help of function block **RedSwitch** and switch the Standby to *active*.

For more information about function blocks see the zenon Logic online help.

TIME HANDLING IN FUNCTION BLOCK ZENRT_RED

The number of switch attempts increases at the cycle rate of the timeout. If the zenon Logic Runtime project cannot be assigned to a zenon network project, the network timeout time (**Timeout [s]** property in **Network** group) is added to the failure recognition time (**Failure recognition after [s]** property). The time that the **ERROR** output is active is not **MAX_SWITCH * cycle** but **MAX_SWITCH * timeout [+ network timeout]**.

The non-process-handling, non-dominant zenon computer is considered to be the Standby (Server 2).
The non-process handling, dominant computer (server starts as standby) is not considered a standby.

8.10 OEM specification

In this OEM specification the possibilities of the COPA-DATA zenon Logic Runtime are described which are referred to in the zenon Logic Workbench help with "**see OEM specification**".

CONFIGURATION (PROJECT PROPERTIES)

As a standard the zenon Logic Workbench contains several features and an extensive library of functions and function blocks.. Depending on the Runtime configuration, some features or blocks may be not available. Therefore it is possible to select configurations for the Workbench in order to highlight the features which are not supported by the Runtime.

The respective configuration of Runtime can be loaded in the zenon Logic workbench. To do this, select the **Configuration** entry in the Workbench in the context menu of the loaded project.

FILE ADMINISTRATION FUNCTION (FUNCTION BLOCKS)

This function makes it possible to read and write files sequentially.
Possible paths depend on the operating system used. Some folders are write protected if UAC (User Account Control) is turned on.

The following functionality is supported by zenon Logic Runtime versions:

Function	CE	RT (PC)	RTK (PC)	Implementation
F_ROPEN	X	X	X	not blocking
F_WOPEN	X	X	X	not blocking
F_AOPEN	X	X	X	not blocking
F_CLOSE	X	X	X	not blocking
F_EOF	X	X	X	not blocking
FA_READ	X	X	X	not blocking
FA_WRITE	X	X	X	not blocking
FM_READ	X	X	X	not blocking
FM_WRITE	X	X	X	not blocking
FB_READ	X	X	X	not blocking
FB_WRITE	X	X	X	not blocking
LogFileCSV	X	X	X	not blocking
F_EXIST	X	X	--	blocking

Function	CE	RT (PC)	RTK (PC)	Implementation
F_GETSIZE	X	X	--	blocking
F_COPY	X	X	--	blocking
F_DELETE	X	X	--	blocking
F_RENAME	X	X	--	blocking

Key:

X: supported

`--: not supported`

SFC EXECUTION

The execution of the SFC program equals that of a T5 standard target system und follows the information of the zenon Logic Workbench help.

The exclusiveness of transitions within a divergence is supported.

INTERLINK APPLICATIONS - BINDING

A maximum of 65535 Binding variables can be published in the network. However this value is reduced by every variable which is sent to zenon using the event service.

The standard IP port number of a zenon Logic Runtime is 9000.

PROJECT PROPERTIES - "C" COMPILER

This feature is supported by all zenon Logic target systems (RT, CE, RTK). All C compilers which are normally proposed by the zenon Logic Workbench can be used. **Attention:** Only use the default name of the created Runtime-DLL.

LOGFILECSV (FUNCTION BLOCK)

See first point: File administration function (function blocks).

ADDITIONAL OEM RELEVANT FEATURES AND FUNCTIONS:

Feature	RT (PC)	RTK (PC)	CE
DTAT (function block)	x[3]	x[3]	x[3]
TCP/IP management functions	x	x	x

Feature	RT (PC)	RTK (PC)	CE
Dynamic memory allocation function	x	x	x
PrintF (Function)	x	x	x
Step by Step Debugging	x	x	x
Functions for real time clock administration	x	x	x
Digital sampling recording	x	x	x
UDP management functions	x	x	x
T5 registry management functions	-	-	-
DTFORMAT (Function)	x	x	x
GETSYSINFO (function) - _SYSINFO_CYCLESTAMP_MS	x	x	x
Variable status bits	x	-[1]	x
DAY_TIME (Function)	x	x	x
SET_DAY_TIME (Function)[2]	x	-	x

Key:

- x: Supported
  - [1]: Driver which support the **variable status bits** are only emulated or not supported by the zenon Logic RTK. Therefore no access is granted to the status bits or the time stamps.
  - [2]: zenon works internally with UNIX time. Therefore only system times within their validity are allowed (0-MAXDINT starting with 1.1.1970 0:00 GMT). Data which refer to an earlier point in time affect the functionality of the zenon Runtime.
  - [3]: zenon works internally with UNIX time. Change the settings for this function block in UTC.
- -: Not supported

SUPPORT FOR PRINTF FUNCTION

zenon Logic Runtime supports the "**printf**" function to track messages whilst the program is being executed. These messages are displayed in the output window of the Workbendh. To do this, the Workbench be online in zenon Logic Runtime.

However this only works if, in the Diagnosis Viewer, the logging level "**Messages**" is active for all modules on the Diagnosis Client. You can find details on these logging levels in the Tools manual - Diagnosis Viewer, in the Diagnosis Client chapter.

The **printf** command works with the simulator of the zenon Logic Workbench without additional configuration.

RETAIN VARIABLES (REMANENT FLAGS)

Retain variables saved at the defined saving location - normally in the instance path (hand over parameter *PATH*) - when the Runtime is stopped or closed in a normal way. Depending on the system, it can be the hard disk or a flash card.

With function "*WriteHot*" retain data and hot restart data can be saved explicitly, e.g. after every change or cyclically.

Some zenon Logic I/O drivers support saving the RETAIN values to the SRAM (battery buffered RAM).

In order for the RETAIN values to be loaded when the PLC application is restarted, the zenon Logic Runtime must be started with the corresponding handover parameters/settings (cold start - load RETAIN variables).

You can find information regarding the configuration of Retain data with the zenon Logic Runtime in the chapter settings (on page 38)/general (on page 38).

I/O DRIVER

You can find more information about supported I/O drivers here: zenon Logic I/O driver support (on page 82)

You can find information about the support of Online Change or Hot Restart here: zenon Logic I/O driver: Hot-Restart/Online Change (on page 84)

8.10.1 Support for printf function

zenon Logic Runtime supports the "**printf**" function to track messages whilst the program is being executed. These messages are displayed in the output window of the Workbendh. To do this, the Workbench be online in zenon Logic Runtime.

However this only works if, in the Diagnosis Viewer, the logging level "**Messages**" is active for all modules on the Diagnosis Client. You can find details on these logging levels in the Tools manual - Diagnosis Viewer, in the Diagnosis Client chapter.

The **printf** command works with the simulator of the zenon Logic Workbench without additional configuration.

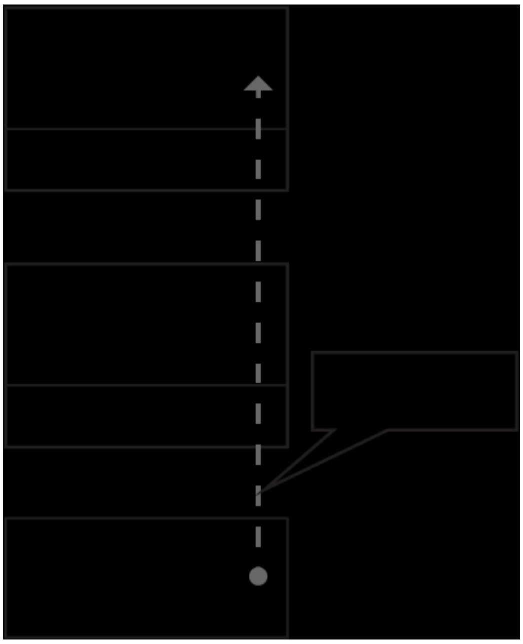
8.11 Forwarding time stamps and the variable status information

As of version 6.22 SP1 Build 3 the zenon Logic Runtime automatically forwards time stamp and variable status information from the zenon Logic I/O driver to the event handler of the zenon Logic Runtime and thus via **straton32** to zenon. Therefore in zenon (e.g. CEL) the time stamp of the source (I/O driver slave or server) are displayed and not the time stamp of the zenon Logic Runtime.

The following prerequisites are necessary:

- The variable must have a time stamp in the zenon Logic I/O driver. It depends on the protocol. Some drivers do not provide time stamps.
- The zenon Logic project must have been compiled and loaded with compiler flag *Create status bit for variable with profile* turned on.

Overview forwarding the time stamp:



8.12 zenon Logic I/O driver support

I/O drivers which are supported by different zenon Logic Runtime types:

Driver	RT (PC)	RTK (PC)	CE
Advantech ADAM 5550	--	--	--

Driver	RT (PC)	RTK (PC)	CE
Anybus Konfiguration	X	E	--
ApplicomIO Konfiguration	X	E	--
AS-interface	X	--	--
BECKHOFF CX1000	--	--	x
Brodersen IOTOOLS	X	E	--
CAN-bus	--	--	--
Hilscher CIF Profibus	X	X[1]	X
Hilscher SYCON Konfiguration	X	E	X[2]
Hilscher SYCON Konfiguration [universal]	X	X	X
Hilscher SYCON.net Konfiguration	X	E	--
IEC 60870 Slave	X	--	X
IEC 61850 Client	X	E[1]	X
IEC 61850 Server	X	--	X
Interbus-S	X	E	--
MODBUS Master Protokoll	X	E	X
MODBUS Slave Protokoll	X	E	X
PROFINET IO	X	E	X
SoftNet ProfibusDP	X	E	--
Logic zu SCADA Verbindung	X	--	X
ThinkIO/IO System 758	--	--	X
Wago 750-860	--	--	--
Weihenstephan Standard - Client	X	E	X
Win32 Shared memory	X	--	--
XFlow	--	--	--

KEY

- X: Fully supported
- --: Not supported
- E: Carried out as emulation (performance loss possible, availability is limited in the PLC cycle, driver selection at a BSOD)
  - 1: The driver-specific function modules do not work with the zenon Logic RTK.
  - 2: Hilscher DLLs must be available on the target system.

8.13 zenon Logic I/O driver: Hot-Restart/Online Change

The zenon Logic I/O drivers support the Hot-Restart or Online Change in different ways:

Drivers	Hot-Restart	Online Change	Online Change including change of the bus driver configuration
Anybus Konfiguration	X	X	X[1]
ApplicomIO Konfiguration	X	X	X[1]
AS-interface	X	X	X[1]
BECKHOFF CX1000	X	X	--
Brodersen IOTOOLS	X	X	X[1]
Hilscher CIF Profibus	X	X	--
Hilscher SYCON Konfiguration	X	X	X[1]
Hischer SYCON Konfiguration [universal]	X	X	X[1]
Hilscher SYCON.net Konfiguration	X	X	X[1]
IEC 61850 Client	X	X	--
Interbus-S	X	X	--
MODBUS Master Protokoll	X	X	X[1]
MODBUS Slave Protokoll	X	X	X[1]
PROFINET IO	X	X	--
SoftNet ProfibusDP	X	X	X[1]
straton zu zenon Runtime Verbindung	X	X	--[2]
ThinkIO/IO System 758	X	X	--
Weihenstephan Standard - Client	X	X	X[1]
Win32 Shared memory	X	X	X[1]

LEGEND

- X: Functionality available
- --: Functionality not available
  - 1: Online Change including the change of the bus driver configuration is always followed by the reinitialization of the I/O driver. Because of the reinitialization I/Os can fail for a short period!

- 2: No need for change in the bus driver configuration.

8.14 Error message

Errors during the execution are logged in the log file.

Error message	Description
General error.	Error which could not be specified any further.
Invalid application code - bad header.	STRATON.cod file is invalid. Header invalid.
Invalid application code - bad target ID.	STRATON.cod file is invalid. Target ID invalid.
Invalid application code - bad version.	STRATON.cod file is invalid. Version invalid.
Unknown communication request.	Host and Runtime not compatible (T5 protocol).
Cannot process read request.	Incoming communication request cannot be processed.
Unknown TIC code.	Invalid instruction in file **straton.cod**.
Too many DATA8 used.	Too many SINT/USINT variables used.
Too many DATA16 used.	Too many INT/UINT variables used.
Too many DATA32 used.	Too many DINT/UDINT/REAL variables used.
Division by 0 in TIC (32bit).	Division by 0 in the PLC logic of the user.
Too many TIME variables.	Memory overflow for online change.
Infinite loop in TIC (safe mode).	Not closed loop in the application code.
Standard function used, which is not supported by this runtime.	PLC logic of the user refers to a function which is not available in this version of the Runtime.
Division by 0 in TIC (64bit).	Division by 0 in the PLC logic of the user.
Too many DATA64 used.	Too many LINT/LREAL variables used.
Too many strings.	Memory overflow for online change.
Division by zero in TIC (8 bit).	Division by 0 in the PLC logic of the user.
Division by zero in TIC (16 bit).	Division by 0 in the PLC logic of the user.
HOT restart: System resources used.	Hot restart not possible.
Invalid variable map in hot restart.	Hot restart not possible.

Error message	Description
Bad build stamp for hot restart.	Hot restart not possible. Invalid **straton.cod** version.
SFC changed - no hot restart!	Hot restart not possible.
Retain definition has changed - no hot restart.	Hot restart not possible.
Bad DB format - no hot restart!	Hot restart not possible.
Cycle time overflow.	Cycle time overflow. For details, see the **Cycle time error messages** section.
Can not store retain variables.	Not enough memory in order to save retain variable.
Array index out of bounds.	Access to invalid array index.
Recursive call of sub-program.	A sub-program calls up itself.
Too many external variables used.	Memory overflow for online change.
Too many tasks.	Memory overflow for online change.
Call stack overflow (hot).	Memory overflow for online change.
Stack overflow.	Too many call ups of intricate sub programs.
Can not hot restart (ASi).	Hot restart not possible (Asi bus driver).
Online change not supported.	Online change not possible
Online change failed - Bad code.	Online change not possible. Invalid **straton.cod** version.
Online change - Not the same application.	Online change not possible
Online change failed - IO drivers.	Online change not possible
Online change failed - CT segment changed.	Online change not possible
Online change failed - bus drivers.	Online change not possible. Bus driver does not permit online change.
Online change failed - variable map / profiles.	Online change not possible
Online change failed - tasks.	Online change not possible
Online change failed - MODBUS.	Online change not possible

Error message	Description
Online change failed - CAN bus.	Online change not possible
Online change failed - event production.	Online change not possible
Online change failed - change D8 alloc.	Online change not possible
Online change failed - change D16 alloc.	Online change not possible
Online change failed - change D32 alloc.	Online change not possible
Online change failed - change D64 alloc.	Online change not possible
Online change failed - change TMR alloc.	Online change not possible
Online change failed - change STR alloc.	Online change not possible
Online change failed - change FBI alloc.	Online change not possible
Online change failed - change CFC alloc.	Online change not possible
Online change failed - change XV alloc.	Online change not possible
Online change failed - STR buff size.	Online change not possible
Online change failed - STR length.	Online change not possible
Online change failed - FBI buffer.	Online change not possible
Online change failed - FBI changed.	Online change not possible
Online change failed - Program style.	Online change not possible
Online change failed - Number of programs.	Online change not possible

ERROR MESSAGE CYCLE TIME

If the **cycle time** is exceeded, messages are displayed in the zenon Logic Runtime or in the Workbench and corresponding entries are written to the LOG file.

Error message user interface	Entry log file	Meaning
Timing configuration error. (x) cycles lost.	*Timing configuration error. (x1) cycles lost. Configured cycle time = (x2) us, maximum execution time = (x3) us, maximum PLC utilisation = (x4).*	Number (*x*) of not executed cycles due to cycle time out. The execution time is higher than the set cycle time or there are not enough CPU resources available.

Error message user interface	Entry log file	Meaning
Timing configuration error. PLC utilization peaked at (x)%.	*Timing configuration error. PLC utilization exceeds 80%. Configured cycle time = (x1) us, maximum execution time = (x2) us, maximum PLC utilisation =* *(x3)%."*	The execution time is higher than *80%* of the set cycle time; actually (*x*)%.

TROUBLESHOOTING

You can rectify the **cycle time** being exceeded by:

- **cycle time** increasing
- decreasing execution time, e.g.:
  - use a system with more CPU resources
  - Use code compiled with C-Compiler

reduce the program

9 zenon Logic Runtime Manager

The zenon Logic Runtime Manager administrates all zenon Logic Runtime projects, which are stand alone or must be started manually, on your computer. It does not matter whether there is a zenon Runtime installed on the computer.

9.1 Why should you use the zenon Logic Runtime Manager?

- If you created your zenon Logic project in zenon and you set the zenon property of the zenon Logic project RT start type to *start manually*, the zenon Logic Runtime Manager is the ideal tool for organizing the parameters for the manual start.
- If you want to start your zenon Logic Runtime project during the booting of the Windows operating system, you can set this with one mouse click in the zenon Logic Runtime Manager.
- If you want to reorganize your zenon Logic project, you can do this with the zenon Logic Runtime Manager.
- If you want to run the zenon Logic Runtime autarchically (without the zenon Runtime), it is best to call up the zenon Logic Runtime instance using the zenon Logic Runtime Manager.

9.2 Main menu

- Configurations: There you can administrate the zenon Logic Runtime projects
- Languages: There you can switch between languages online
- Settings: Enter the path of the zenon Logic Runtime
- %: Open the help
- Exit: Exits the program - **Hint:** Do not forget to save the settings before you exit the zenon Logic Runtime Manager!

9.3 Properties help

Place the mouse cursor over an element in the zenon Logic Runtime Manager in order to see the corresponding help text in the property help.

In addition all elements of the zenon Logic Runtime Managers are described here:

9.3.1 Configuration list

This list contains all available zenon Logic Runtime configurations.

- Click on an entry in the list in order to display its parameter settings.
- After that click on **Edit** in order to change the parameters.
- Click on **Save** in order to save the changes.

9.3.2 zenon Logic project folder

The selected path of the zenon Logic Runtime project (storage location of the PLC code) is entered there automatically. You can change this entry manually or by left-clicking the folder symbol.

> **Information**
>
> Before you are able to edit any setting which you have already saved, you must first select the concerning configuration in the list. After that click on button **Edit**. In order to save the changes, click on button **Save**.

9.3.3 Name of the configuration

The name of the last path of the zenon Logic Runtime files is entered automatically if a new configuration is created. You can changes this name freely.

Information

Before you are able to edit any setting which you have already saved, you must first select the concerning configuration in the list. After that click on button **Edit**. In order to save the changes, click on button **Save**.

9.3.4 Port (Main)

Enter the main port for the zenon Logic Runtime instance. This port defines the connection of the zenon Logic Workbench and a zenon drivers. Per default this port is set to 1200.

Information

Before you are able to edit any setting which you have already saved, you must first select the concerning configuration in the list. After that click on button **Edit**. In order to save the changes, click on button **Save**.

9.3.5 Port (Binding)

Enter the binding port for the zenon Logic Runtime instance. The binding port defines the connection for spontaneous TCP/IP communication to other zenon Logic Runtimes. Per default this port is set to 7800 higher than the main port.

Information

Before you are able to edit any setting which you have already saved, you must first select the concerning configuration in the list. After that click on button **Edit**. In order to save the changes, click on button **Save**.

9.3.6 Startup

Define the startup behavior of the zenon Logic Runtime instance.

- *Cold start*
  zenon Logic Runtime starts with initialized variables.
- *Cold start (load RETAIN variables)*
  Die zenon Logic Runtime starts initialized with remanent markers.

- *Hot restart*
  zenon Logic Runtime starts with the variable values from the time Runtime was stopped.

> **Information**
>
> Before you are able to edit any setting which you have already saved, you must first select the concerning configuration in the list. After that click on button **Edit**. In order to save the changes, click on button **Save**.

9.3.7 Start in step mode

In order to start the zenon Logic Runtime in step mode, activate this checkbox.

> **Information**
>
> Before you are able to edit any setting which you have already saved, you must first select the concerning configuration in the list. After that click on button **Edit**. In order to save the changes, click on button **Save**.

9.3.8 Windows real time priority

In order for the zenon Logic Runtime instance to run in the Windows real time priority, activate this check box.

> **Information**
>
> We recommend to always have the real time priority activated. First and foremost deactivating the real time priority is used for testing purposes (e.g. if there may be infinite loops).

> **Information**
>
> Before you are able to edit any setting which you have already saved, you must first select the concerning configuration in the list. After that click on button **Edit**. In order to save the changes, click on button **Save**.

9.3.9 Hard real time

If you activate this checkbox, the zenon Logic real time PLC (zenon Logic RTK) is called up when the configuration is started. As the zenon Logic real time PLC may only run once on the computer, it is validated that this checkbox is only activated for one configuration.

> **Information**
>
> Before you are able to edit any setting which you have already saved, you must first select the concerning configuration in the list. After that click on button **Edit**. In order to save the changes, click on button **Save**.

9.3.10 Report output

Activate this checkbox in order to activate the writing of messages of the Runtime in a LOG file.

> **Information**
>
> Writing a LOG file will influence the performance.

> **Information**
>
> The LOG file is named *zenon Logic RTLOG.txt* and is saved in the zenon Logic Runtime project folder.

> **Information**
>
> Before you are able to edit any setting which you have already saved, you must first select the concerning configuration in the list. After that click on button **Edit**. In order to save the changes, click on button **Save**.

9.3.11 Auto start

In order to start a zenon Logic Runtime instance directly with the booting of the Windows operating system, activate this checkbox.

Information

Before you are able to edit any setting which you have already saved, you must first select the concerning configuration in the list. After that click on button **Edit**. In order to save the changes, click on button **Save**.

9.3.12 Button "New"

If you want to create a configuration, click on button **New**. After that a dialog opens in which you can select the folder in which you want to save the zenon Logic Runtime files (the PLC code). Confirm your selection with **OK**. You can change the name of the configuration. In order to save the configuration and the name of the configuration, click on **Save**.

9.3.13 Button "Edit"

Before you can change the parameters of a configuration, you must click on button **Edit** after you have selected an entry from the configuration list. In order to save the changes, you must click on **Save**.

9.3.14 Button "Save"

Click on button **Save** in order to save all changes.

9.3.15 "Delete" button

To delete a configuration, select the entry from the configuration list and click on the **Delete** button.

9.3.16 Button "Start"

In order to start the zenon Logic Runtime with the respective configuration, select a configuration from the list and click on button **Start**. The zenon Logic Runtime starts immediately.

Information

If the folder of the configuration does not contain any zenon Logic Runtime files (PLC code), the <CA_PRODUCTNAME >Runtime starts in mode *No program*.

9.3.17 zenon Logic Runtime (settings)

Enter the folder of the zenon Logic Runtime (folder of *stratonRT.exe*). In order to browse through your hard disk, left-click the folder symbol.

9.3.18 Language selection (languages)

Select the desired language by clicking on the respective flag.

9.3.19 Show splash screen (setting)

If you activate this check box, the splash screen of the zenon Logic Runtime Manager is displayed at the start.

9.3.20 Size of event queue

Length of the event queue in bytes.

> **Information**
>
> Before you are able to edit any setting which you have already saved, you must first select the concerning configuration in the list. After that click on button **Edit**. In order to save the changes, click on button **Save**.

9.3.21 Maximum number of recordable events

This parameter defines how many variables can be processed by the event server per cycle.

> **Information**
>
> Before you are able to edit any setting which you have already saved, you must first select the concerning configuration in the list. After that click on button **Edit**. In order to save the changes, click on button **Save**.