



COPADATA
do it your way

zenon guida

Driver simulation

v.7.00





© 2012 Ing. Punzenberger COPA-DATA GmbH

Tutti i diritti riservati.

Tutti i diritti riservati la distribuzione e la copia - indifferentemente dal metodo - può essere consentita esclusivamente dalla dittaCOPA-DATA. I dati tecnici servono solo per la descrizione del prodotto e non rappresentano in alcun modo parti legali. Modifiche - anche sotto aspetti tecnici sono a noi riservate

Contenuto

1. Benvenuti nell'help COPA-DATA.....	4
2. Driver simulation.....	4
3. Simulation static.....	5
4. Simulation - counting.....	6
5. Simulation - programmed	6
5.1 Editor.....	7
5.1.1 Create project.....	7
5.1.2 Delete project.....	10
5.1.3 Distributed engineering.....	11
5.1.4 Change driver	12
5.1.5 XML export/import.....	13
5.2 Driver configuration	13
5.2.1 Driver variable status	14
5.2.2 Cyclic comparison.....	16
5.3 zenon Logic Workbench.....	18
5.4 Runtime.....	20
5.4.1 Functionality.....	20
5.4.2 Runtime files.....	22
5.4.3 Data exchange	22
5.4.4 Data storage	24
5.4.5 Redundancy.....	25
5.4.6 Status.....	25
5.4.7 Start / stop.....	28
5.4.8 Driver commands	30
5.4.9 Linking variables	30
5.4.10 time stamp	31
5.5 Please not for variables in simulation projects	32
5.6 Error messages.....	33

1. Benvenuti nell'help COPA-DATA

GUIDA GENERALE

Nel caso in cui non abbiate trovato delle informazioni che cercavate o se avete dei consigli relativi al completamento di questo capitolo dell'help, mandate una Mail a documentation@copadata.com (<mailto:documentation@copadata.com>).

SUPPORTO ALLA PROGETTAZIONE

Se avete delle domande concernenti progetti concreti, potete rivolgervi per E-Mail al support@copadata.com (<mailto:support@copadata.com>).

LICENZE E MODULI

Nel caso in cui doveste constatare che avete bisogno di altri moduli o licenze, rivolgetevi ai nostri dipendenti all'indirizzo sales@copadata.com (<mailto:sales@copadata.com>).

2. Driver simulation

If the underlying process is not available when configuring, this can be simulated and tested in advance. Three modes are available for this:

- ▶ simulation - static (auf Seite 5): constant values simulated by the driver
- ▶ simulation - counting (auf Seite 6): values simulated by the driver are counted up

- ▶ simulation - programmed (auf Seite 6): Values are calculated via a simulation project with zenon Logic

 **Attenzione**

*If the driver is stopped in mode **Simulation - counting**, only the counting is stopped. The variable is not switched to **faulty**. In all other modes the driver is really stopped.*

Hint: **Simulation - programmed** is not supported by drivers for:

- ▶ Internal variables
- ▶ Mathematical variables
- ▶ Simulator variables
- ▶ System variables

 **Informazioni**

*zenon variables which represent the zenon Logic IO variables are available in the project in state **Driver simulation programmed**.*

 **Informazioni sulla licenza**

Contenuta nella licenza standard per Editor e Runtime

3. Simulation static

For a static simulation, no communication to the control is established; the values are simulated by the driver. In this mode the value remains constant. Values can be changed by the Runtime or the user. At a restart of the Runtime with **simulation - static** these values will however not be saved and are lost.

4. Simulation - counting

For a counting simulation, no communication to the control is established; instead the values are simulated by the driver. In this mode, the driver increments the values within a value range automatically, starting with 0. If the maximum value has been reached, the counting process starts at 0 again.



Informazioni

For negative starting values, the counting process only starts at 0.

INT uses the maximum value from USINT to count.



Attenzione

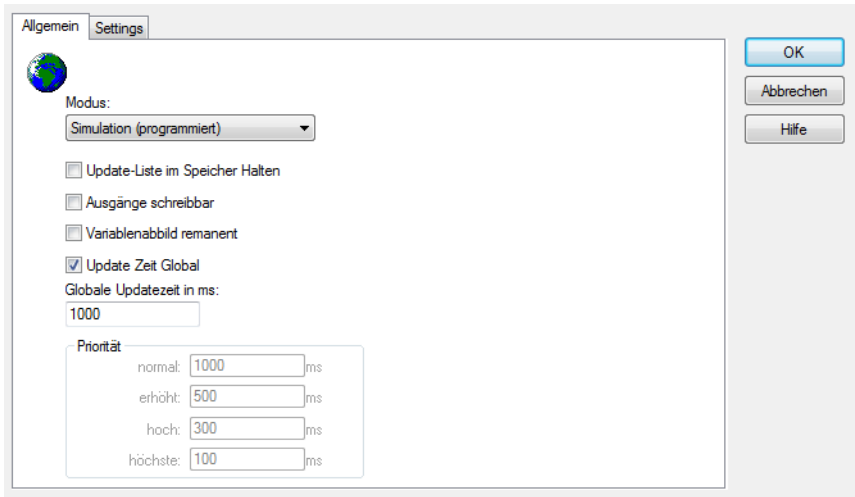
*If the driver is stopped in mode **Simulation - counting**, only the counting is stopped.*

The variable is not switched to `faulty`. In all other modes the driver is really stopped.

5. Simulation - programmed

For Simulation - programmed, no communication is established to the PLC; instead, the values are calculated by a freely programmable simulation project. The simulation project is created with the help of the zenon Logic Workbench and runs in the zenon Logic Runtime. It enables the status and time stamp of variables to be modified on the driver. Write commands to variables by zenon are forwarded to the simulation, redundant servers and standby servers are synchronized. This is how you also simulate complex processes.

To start the simulation program, select the Simulation - programmed mode in driver configuration (auf Seite 13).




Informazioni

Windows CE

Simulation - programmed is not available for Windows CE.

5.1 Editor

5.1.1 Create project

To create a program for Simulation - programmed, you must:

- ▶ select a single process driver
- ▶ make sure that the project name is valid.



Attenzione

Simulation - programmed is not supported by drivers for:

- ▶ Internal variables
- ▶ Mathematical variables
- ▶ Simulator variables
- ▶ System variables

To create a simulation project:

- ▶ click in the group `Progetto di simulazione del driver` in the property `Modifica` on `Click here ->`
- ▶ A new zenon Logic project is created.
- ▶ `Nome` and port numbers for `Porta eventi` and `Porta standard` are automatically issued; change these as you wish
- ▶ the zenon Logic Workbench is opened

The simulation project's workshop is automatically closed if:

- ▶ the zenon editor is closed
- ▶ a driver is deleted and its simulation project is currently being edited in Workbench
- ▶ the simulation project for the driver is deleted via the `Elimina` property
- ▶ the simulation project is renamed

PROJECT ARCHIVING

A zenon Logic project contains many files and folders. To manage them, in particular to simplify distributed engineering, all files are archived in compressed form in the `simul_<Treiber-ID>.zip` file in the `<Sql Projekt Pfad>\FILES\zenon\custom\drivers`. This file is in the driver files in the editor. When starting zenon Logic Workbench, the files are automatically decompressed and compressed again when it is ended.

CREATING VARIABLES

When creating variables in a simulation project, the succession is important:

1. create the variable
2. select the valid data type in the zenon Logic Workbench

3. after that activate property `embed symbol`

Background: Per default variables are created with the driver object type `PLC marker`. This object type does not support all data types for all drivers. If a variable is copied in the zenon Logic Workbench with active property `embed symbol`, you must deactivate this property in order to change the data type. With this the variable is then deleted in zenon.



Informazioni

Error message "Can not create variable"

A variable cannot be created in the simulation object if the driver does not have driver object types - exception

`driver variable` - which support this data type.

On activation of `embed symbol`, the error message "can not create variable" is displayed.

STRUCTURE DATA TYPES

At creating variable with a structure data type:

- ▶ they are created with the driver object type `PLC marker` if possible
- ▶ all driver object types with the exception of `driver variable` are tried until one can be used for all elements
- ▶ if no fitting driver object type is available,
 - no variable is created for non-structure variables
 - driver object type `PLC marker` is used for structure variables

EXAMPLE:

A structure data type contains elements of type `UINT` and `STRING`. A variable is created for `S7TCP` driver and `embed symbol` is activated.

- ▶ The variable is not created in the `PLC marker` but as `Ext. Datablock` in which all structure elements exist.
- ▶ A new type `LINT` is added; it is not supported by the `Ext. Datablock`.

- ▶ When a new variable is created, it is created as type `PLC marker`. Only the first complex variable can be activated (`UINT`). For the existing structure variable the object type `Ext . Datablock` remains and the last structure element (`LINT`) cannot be activated.

COMPILING THE FILES

If a simulation project is compiled in zenon Logic Workbench, this causes a variable to be created in zenon and the Runtime program file.

When compiling, a subfolder is created in `Runtime-Folder\RT\FILES\zenon\custom\drivers` with the name of the simulation project. The file with the code of the simulation is `SIMULRT.COD` is archived in this folder.

The following message is displayed in the output window:

`SIMULRT.COD`

And if compiled with the C compiler, then also:

`T5APP.DLL`

Attenzione

*For **multi-user projects** (auf Seite 11) you must not create simulation projects offline. They cannot be deleted anymore.*

5.1.2 Delete project

To delete a simulation project:

1. click on the `click Here->` button in the `Elimina` property in the `Progetto di simulazione del driver group`
2. confirm this when requested to do so
3. the ZIP file with the project files is deleted
4. the zenon Logic Workbench is ended

Note:

- ▶ the Runtime folder for the simulation project remains
- ▶ function `undo` is not available for this action,
- ▶ click on `Modifica` to create a new simulation project
- ▶ for `multi-user projects` (auf Seite 11):
 - the driver must be configured to `Make changes possible` so that the simulation program can be deleted
 - the project is not displayed as deleted on other Clients as long as it has not been synchronized;

if you try to open a deleted project before it has been synchronized (click on `Modifica`), a new simulation project will be created

- you cannot reverse the deletion of single-user projects via `Cancel changes`

Attenzione

The Simulation project is immediately removed from the local database and the server database when deleted. This action cannot be undone!

Note: Manual deletion of the ZIP file of the driver also leads to the simulation project being deleted.

Requirement: The zenon Logic Workbench for this driver is not opened. We do not recommend doing it this way!

5.1.3 Distributed engineering

For `multi-user projects`, all running zenon Logic Workbenches that are part of the project are closed when:

- ▶ `Accept changes` for modules that are completely locked for other users in `Enable changes`, such as variables, drivers and data types
- ▶ `Discard changes`:
 - new projects are not created on the server
 - Changes in the simulation program are lost
 - Driver files not present in the server database are also lost in the local database
- ▶ `Synchronize`:
 - changes made locally to the simulation program are lost

- ▶ Update local version:
changes made locally to the simulation program in the simulation program are lost

 **Attenzione**

The status of the simulation project's ZIP file may not be additionally modified (Accept Changes, Enable Changes, Discard Changes multi-user status), to ensure correct Accept Changes and Enables Changes for the driver!

DELETE PROJECT

For **multi-user projects**

- ▶ the driver must be configured to **Make changes possible** so that the simulation program can be deleted
- ▶ the project is not displayed as deleted on other Clients as long as it has not been synchronized;
 - if you try to open a deleted project before it has been synchronized (click on **Modifica**), a new simulation project will be created
- ▶ you cannot reverse the deletion of single-user projects via **Cancel changes**

 **Attenzione**

A simulation project with the status of **Enable Changes** cannot be deleted (auf Seite 10) in **Offline mode**.

A simulation project created **offline in a multi-user project** can therefore no longer be deleted.

5.1.4 Change driver

For a driver change in zenon the following is true:

- ▶ The simulation project is maintained.

- ▶ All ports are however set to 0. To receive working port numbers, open the project in the Editor. At this new port numbers are entered automatically. Port numbers can also be assigned manually.

 **Attenzione**

I driver che sono collegati con un progetto zenon Logic tramite la soluzione integrata, non possono essere cambiati.

5.1.5 XML export/import

*A project in mode **Simulation** - **programmed** can be exported via XML. It can however not be imported as new project for another driver.*

Workaround:

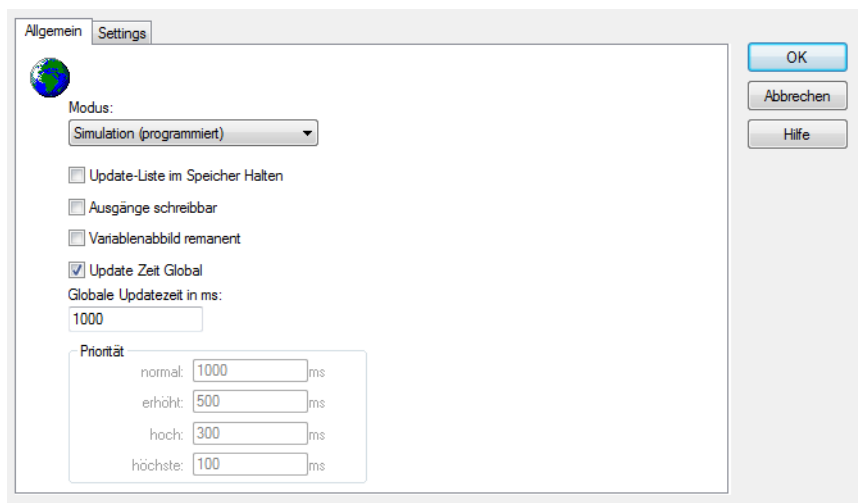
1. Export the zenon Logic programs.
2. Import them to the new project for the new driver.
3. Replace the variables via search and replace.

5.2 Driver configuration

There are four modes available when configuring the driver:

- ▶ **Hardware**
- ▶ **Simulation - counting**
- ▶ **Simulation static**
- ▶ **Simulation - programmed**

Select the **simulation - programmed** mode.



5.2.1 Driver variable status

The status of zenon Logic Runtime is displayed in the driver variable SimulRTState on Offset 60. The variable is numerical and cannot be written. The value informs you of the status of Runtime:

Bit	Meaning	Description
31	without application	Runtime has not loaded a program, the program was stopped or not loaded in Runtime.
30	not instanced	Runtime was not instanced, for example because the DLL with Runtime could not be loaded, there was too little memory available, there is no DDL or it is the wrong version.
18	p-code available	p-code is available together with compiled code; switching is possible.
17	compiled code active	Runtime runs with compiled code (otherwise: interpreted p-code).
16	compiled code available	Compiled code from the C compiler is available
15	application is loaded	an application is present, Runtime is running
14	cant start - missing handlers	Some "C" functions are missing.
13	active breakpoints installed	At least one breakpoint was set by the debugger.
12	CT segment exists	Application was compiled separately with the "complex variables in separate segment" option.
11	Reserved	(Internal, for Runtime only.)
10	sysinfo request is available	(Internal, for Runtime only.)
9	freeze event production	Binding does not transmit events.
8	single cycle mode	(Internal, for Runtime and debugger only.)
7	Reserved	(Internal, for Runtime only.)
6	locked variables	at least one variable is locked.
5	trigo functions are in degrees	Trigonometric functions are given in degrees.
4	log message(s) in stack	(Internal, for Runtime only.)
3	application stopped between 2 progs	Runtime stopped between two steps whilst debugging.
2	application stopped on SFC breakpt	(Internal, for Runtime only.)
1	application stopped on error	Runtime stopped with serious error, restart required.
0	application is running	TRUE = application is in "run" mode. FALSE = application paused in "cycle to cycle" mode.



Informazioni

The driver variable `SimulRTState` with offset 60 shows the status of the Runtime on the Server. This is also true when the variable is displayed on the Standby.

5.2.2 Cyclic comparison

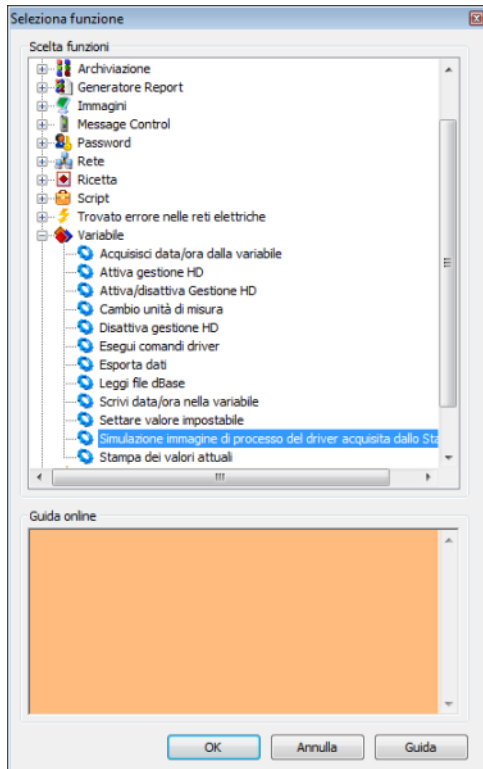
Simulations run in network operation on the server and standby server independently of one another. When the standby server is started, a comparison is made with the server. Nevertheless:

- ▶ This first comparison is delayed by the packet transfer time
- ▶ small differences in execution lead to increasing divergence

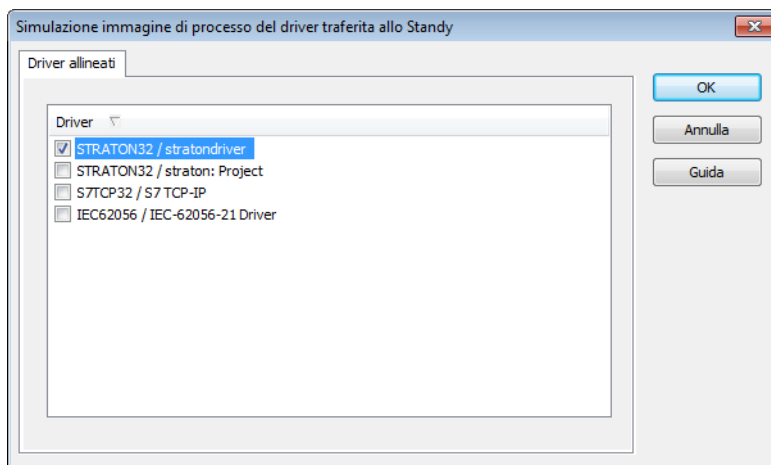
Thus the status of simulated variables is also different. To keep these differences small, use the `transfer driver simulation to stand by` function to compare the simulation image.

- ▶ create a new function

- ▶ select the **transfer driver simulation image to standby** function



- ▶ the dialog for the selection of drivers opens



- ▶ all process related drivers are available in the list (with the exception of the simulator) (clicking on the column header changes the sorting)
- ▶ select the driver to be compared
- ▶ An image is requested for each function request to the standby server for the selected driver

The differences in the current simulation status can be minimized through cyclical activation. Adjust the execution grid to the scope of data to be compared, because a load for the network and computer is generated as a result of this.

The `Esecuzione`: of the function is set `Standby` at and cannot be changed. This only happens if:

- ▶ the computer is a standby server
- ▶ the server is online
- ▶ the project is a network project
- ▶ Drivers were selected for comparison

TO CHANGE THE SELECTION OF DRIVERS

- ▶ click on driver properties in the `Generale` group in the `Parametri` property
- ▶ the dialog to select the driver opens

5.3 zenon Logic Workbench

The `Simulation - programmed` is programmed in the in the zenon Logic Workbench, which works in close conjunction with the zenon Editor. For the simulation, (in contrast to the integrated solution) only the variables of a driver are exchanged between the editors.

DEFINITION OF THE VARIABLES TO BE EXCHANGED:

When starting zenon Logic Workbench, all driver variables that are not yet present in zenon Logic are created as global variables. The `Embed symbol` is automatically activated for these.

VISIBILITY

If a new variable is defined in Workbench in the `global` or `retain` area, this is only visible in zenon Logic. To make it visible in zenon too, the `Embed symbols` option must be activated. From this point in time, all variables are updated by both editors.

If the `Embed symbol` option is deactivated, the variable is deleted in the zenon editor. This action enables `global` or `retain` variables to be defined in zenon Logic, which are not visible for zenon.

For a newly-created variable created by the zenon Logic Workbench in zenon, an attempt is made to assign this to the **SPSMERKER** area. In the event that the zenon driver (auf Seite 7) does not support the configured data type, a search is made for the first data area that supports the data type. If no area is found, the variable cannot be created.

A variable is only locked if in the case of a **multi-user project** and the `enable_changes` status has not been set.

PROJECT SETTINGS

Some settings under *Project -> Project parameters... -> Further options-> Extended* in zenon Logic influence the functionality of the simulation.

Extended settings	Description
Runtime: execute mode	Recommendation: Mode: triggered cycle time: 1/4 to 1/3 of the driver update time for frequent value changes to simulated variables lead to increased load placed on the computer.
compiler/options: Embed symbols of all variables	Recommendation: Deactivate. Active: Data for a variable are exchanged, because the symbol names are assigned in Runtime. But: Changes to the configuration of the variables are not updated between zenon and zenon Logic. It is no longer possible to make changes online.
compiler/options: Retain capitalization of symbols	without effect: Variable names are always converted to capitals in Runtime.
compiler/options: Create status bits for variables with a profile	Active: The status of variables is simulated or reacted to in the simulation program.
"C" compiler	Native code is supported. Name of the DLL that the compiled code must contain: t5app.dll.

5.4 Runtime

5.4.1 Functionality

zenon Logic Runtime offers the following functions:

- ▶ File transfer
- ▶ Data types:
 - BOOL
 - BYTE
 - DINT
 - DWORD
 - INT
 - LINT
 - LREAL
 - REAL
 - SINT
 - STRING
 - TIME
 - UDINT
 - UINT
 - USINT
 - WORD
- ▶ zenon arrays (see also Arrays and simulation (auf Seite 21))
- ▶ Date stamp/time stamp
- ▶ Digital recording of values
- ▶ Dynamically linked function blocks, stored in `t5block*.dll`
- ▶ Hot Restart
- ▶ Logging:
 - Trace messages from the program (`printf`)

- Error messages in Runtime
- all sent to Workbench and to the diagnosis server
- ▶ Machine code: Native compiled code with C post compiler (not available under Windows CE)
- ▶ Online Change
- ▶ Programming languages: IL, AS, FDB, LD, ST
- ▶ Retain Data (remanent Data)
- ▶ Special function blocks and features:
 - Data serialization
 - dynamic memory allocation
 - embedded recipes
 - embedded variable lists
 - File Management
 - Files
 - Math
 - Random
 - Signals
 - String tables
 - Trigonometry
 - Time
- ▶ Spontaneous communication
 - Value change as an event
 - 16 connections
 - Hysteresis
- ▶ UDP & TCP/IP for IEC languages
- ▶ Variable interlocking

Arrays and simulation

To simulate multidimensional arrays at driver simulation, you must activate property 'Save complex variables in own segment' in the zenon Logic Workbench.

ARRAYS WITH START INDEX 1

If at a simulation arrays with start index 1 exist, these arrays are created with one more index in the zenon Logic Workbench.

Esempio

In zenon an array has indices 1 to 4. This array has the indices 0 to 4 in the zenon Logic Workbench.

This guarantees that the same indices are used in zenon and in zenon Logic. Index 0 is not transferred to the driver.

If you change array in the zenon Logic Workbench, you must consider this additional index.

5.4.2 Runtime files

When creating Runtime files, a subfolder with the name of the simulation project is created in the `Cartella Runtime` of the project in the `\RT\FILES\zenon\custom\drivers` path. The `SIMULRT.COD` file with the code of the simulation is archived in this folder.

The following message is displayed after successful compilation in Workbench and transfer to the Runtime folder in the editor's output window:

```
SIMULRT.COD
```

And if compiled with the C compiler, then also:

```
T5APP.DLL
```

REMOTE TRANSPORT

The Runtime files for the simulation are generated as subfolders of the driver files. For Remote Transport, the subfolder with all files contained therein is transferred to the target computer

5.4.3 Data exchange

the exchange of data from zenon Logic Runtime to the driver starts with the 2nd cycle of zenon Logic Runtime. Therefore the first run is available to initialize all variables in Runtime.

Data is only exchanged if:

- ▶ The variable is requested and:
 - did not yet receive a value
 - does not have status `Switched off (OFF/_VSB_N_UPD)` or `Alternate value (ALT_VAL/_VSB_AVALUE)` in the driver
 - is still required by the driver.
 - has changed value or status

The exchange of data to the driver is carried out asynchronously to the Runtime cycle. The driver runs through the value changes once per update cycle. All values changed - with the exception of the `Switched off (OFF/_VSB_N_UPD)` or the `Alternate value (ALT_VAL/_VSB_AVALUE)` status - are entered into the list of variables to be updated in the driver and sent to Runtime.

SWITCH DRIVER SIMULATION SCREEN TO STANDBY **FUNCTION**

This function is only carried out on the standby server (for configuration, see Cyclical Comparison (auf Seite 16) chapter). The data is synchronously obtained from the server that composes the screen. The driver has 5 seconds of time for this. Because the data for the image is created in the same thread as that in which the simulation runs, it is necessary to ensure that the simulation is processed within 5 seconds. If this image is exceeded, no image is carried over for this driver.

VALUE SIMULATION

Value changes are not transferred immediately, but stored in a buffer. The size of the buffer amounts to 8192 value changes or the five time the number of variables, according to whatever value is greater. The delay in transferring resulting from the driver cycle can be up to 100 ms. If the value changes again whilst it is waiting to be transferred, all values that have not yet been transferred are sent to Runtime and the new value is then noted. This can lead to an increased load for the computer and network, but ensures that all values are transferred in Runtime.

DRIVER WRITE COMMAND

If `simulation - programmed` is configured and active, the write commands at standby are passed through in Runtime.

Write operations are carried out by the driver asynchronously to Runtime. Data is exchanged via a buffer. The size of the buffer amounts to 8192 value changes or the five time the number of variables, according to whatever value is greater.

Write commands are ran through after their values are changed in Runtime. If a write command cannot be executed correctly, the error message `Write queue full! Write command for <DP-Name> lost!` is displayed.

If the status bits are activated, setting `Writing successful` will display if the value was written from the driver to the value of the variable. The Runtime program should delete this status again after processing the write command. A status bit that is already active does not cause any delay in the driver writing again. Write commands are also executed if the Runtime program was stopped by a breakpoint.

Note: Not all write commands from the driver must be visible in Runtime. If several write commands are in the buffer, only the last one is visible in Runtime.

SWITCH OFF VARIABLES, SWITCH TO SUBSTITUTE VALUE

If the variable is switched off or switched to a substitute value, this is not recognizable for Runtime. Changes in values that are also carried out by Runtime are not assigned to the variables in the driver. If the variables are again switched to spontaneous value or activated, the current value is again carried over from Runtime to the driver. When turning off the spontaneous value status bit OFF is set. When you switch back to the spontaneous value, bit OFF is reset.

5.4.4 Data storage

RETAIN DATA

Retain data is stored in the `SimulRt<TreiberID>.ret` file. One file such as this is created per simulation, provided retain variables are present. The file is taken into account when a comparison is carried out in the network.

DRIVER REMANENT

The `simulation - programmed` setting at the driver start is not a criterion in order to restore the remanant variable image.

5.4.5 Redundancy

START ON STANDBY SERVER

If the driver is already running on a standby basis, then the image for the simulation should already be available. In this case, the simulation is started "hot" with this image. If there is no image, a warm start is carried out.

SWITCH DRIVER SIMULATION SCREEN TO STANDBY FUNCTION

This function is only carried out on the standby server (for configuration, see Cyclical Comparison (auf Seite 16) chapter). The data is synchronously obtained from the server that composes the screen. The driver has 5 seconds of time for this. Because the data for the image is created in the same thread as that in which the simulation runs, it is necessary to ensure that the simulation is processed within 5 seconds. If this image is exceeded, no image is carried over for this driver.

COMPARISON

In addition to the program files of the simulation, the following are also compared:

- ▶ Retain data (*.ret)
- ▶ all *.simul files

This file extension is created by the simulation program for simulation specific files and archived in the folder with the computer description below the `Cartella Runtime`.

5.4.6 Status

STATUS SIMULATION

For the status simulation, in zenon Logic under *Project -> Project parameters -> Extended -> Compiler -> Options -> Create status bits for variables with a profile* must be active. If the status of a variable is modified, this triggers the transmission of a variable to the driver.

Changes to the following states does not trigger a value change at the driver and is also not taken over from the simulation:

- ▶ Real time external (T_EXTERN/_VSB_RT_E)

- ▶ Real time internal (T_INTERN/_VSB_RT_I)
- ▶ Standard time (T_STD/_VSB_WINTER)
- ▶ Writing acknowledged (WR_ACK/_VSB_WR_ACK)
- ▶ Writing successful (WR_SUC/_VSB_WR_SUC)
- ▶ Normal status (NORM/_VSB_NORM)
- ▶ Deviation from normal status (N_NORM/_VSB_ABNORM)
- ▶ Select in the network (NET_SEL/_VSB_SELEC)
- ▶ Runtime exceeded (TIMEOUT/_VSB_RTE)
- ▶ In process (PROGRESS/_VSB_DIREC)
- ▶ Switched off (OFF/_VSB_N_UPD)
- ▶ Substitute value (ALT_VAL/_VSB_AVALUE)



Informazioni

State bit `Writing successful` (WR_SUC/_VSB_WR_SUC) is available in the Runtime program and can be set back in order to display successful writing.

However only setting back the state does not trigger the transfer to the <CD_PRODUCUTNAME> Runtime. Only value changes are transferred from the driver to the Runtime.

STATUS BITS HANDLED DIFFERENTLY

The following status bits cancel each other out:

Bit	Description
17: Spontaneous (SPONT/_VSB_SPONT)	Status SPONTAN is set. This is the status set if status handling is turned off or no status was defined. Status GI + INAVLID is deleted.
18: Invalid (INVALID/_VSB_I_BIT)	Status INVALID is set. Status SPONTAN + GI is deleted.
16: General interrogation (GI/_VSB_GR)	Status General Interrogation is set. Status SPONTAN + INAVLID is deleted.
8: Select in the network (NET_SEL/_VSB_SELEC)	<p>If this status bit is active (BSO activation/deactivation) the following status bits are accepted with 0 and also transmitted as 0 to the driver to identify a change:</p> <ul style="list-style-type: none"> ▶ Select in the network (NET_SEL) ▶ Cause of transmission (COTx) ▶ Select (SE_870) ▶ N_CONF (P/N-BIT) ▶ Test bit (TEST) ▶ Writing successful (WR_SUC)

SELECT BEFORE OPERATE (SBO)

Status simulation must be available in the Runtime program in order for *Seleziona prima di operare* to be reacted to. The procedure for *Seleziona prima di operare* is defined in Runtime.

ACTIVATION

If an `SBO select` is sent to the driver, it triggers a value being written with state *Select in the network* (NET_SEL) + *cause of transmission* (COT_act).

A corresponding SBO procedure must be implemented and must start in Runtime.

The state bit *Select in the network* (NET_SEL) must be set back in the Runtime program.

DEACTIVATION

The deactivation triggers the writing of a values with status `Select` in the network (`NET_SEL`) + Cause of transmission (`COT_deact`).

The SBO procedure must be ended in Runtime accordingly.

The state bit `Select` in the network (`NET_SEL`) must be set back in the Runtime program.

5.4.7 Start / stop

DRIVER START

The value updating only starts at the standby server if the driver has received the process image from its server. The image must be received within the time out module, otherwise the driver will start without an image. If the standby server is upgraded to the server within the waiting waiting time, waiting is also ended. Drivers that start in a network project or without a network have no waiting time. This behavior applies for both the simulation as well as for the hardware mode.

RUNTIME START

If the driver starts the `simulation - programmed`, the first stage is loading the DLL with the simulation Runtime. After this, Runtime is parametered and the Runtime program starts, provided it can be successfully loaded.

If an image of Runtime is present, Runtime is started `hot` with this. Otherwise, a `warm start` is carried out. If this is also not possible, an attempt is made to start Runtime `cold`. If there is no valid program, Runtime boots up stopped. The status of all variables is set to `INVALID`.

The simulation program runs in it own thread and is therefore completely independent from the driver cycle.



Informazioni

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.

STOP RUNTIME

If Runtime is stopped, the status `INVALID` is set for all variables. Variables that are requested when Runtime is stopped have the status `INVALID` as initialization. Runtime secures the data for the warm start and online change:

- ▶ `SIMULRT.HOT`: Contains the data for the hot restart.
- ▶ `SIMULRT.UPD`: Contains the data for the online change.

Both these files are created in the Runtime program folder. Runtime must have write authorization for this folder.

RELOAD

- ▶ Recompiling the simulation project:
causes the corresponding driver to be reloaded.
- ▶ Modification of the Time Out module:
is not recognized as a change by Runtime and does not cause the driver to be reloaded.

Standard implementation in the driver kit triggers a hot restart when reloading the simulation project. If this is not possible (for example because the program is different), a cold start is carried out with `retain`.

REMOVING RUNTIME

If Runtime is running, this is stopped. Before Runtime is removed, all outstanding value updates are sent to the driver. Only then is Runtime released and `simulRuntime.dll` removed from the PC.

5.4.8 Driver commands

Driver commands are used to influence drivers via zenon, e.g. start and stop. The engineering is implemented with the help of function `Driver commands`.

STOP/START DRIVER

If the driver is stopped in `simulation - programmed` mode, this will result in Runtime being removed from the memory. All variables obtain status `INVALID`. When the driver is restarted, the simulation is reloaded and Runtime is started.

SWITCH HARDWARE/SIMULATION DRIVER

When switching between modes `Hardware` and `Simulation`, the driver behaves according to the following pattern:

1. If the driver was configured for hardware mode in the Editor:
 - the driver is set to mode `simulation static` after function `Driver command` is executed with parameter `Driver in simulation mode`.
2. If the driver was configured in the Editor for one of the simulation modes (`static`, `counting`, `programmed`):
 - the driver is set to hardware mode and communicated with the control after function `Driver command` is executed with parameter `Driver in hardware mode`.
 - the driver then changes back to the configured simulation mode after function `Driver command` was executed with parameter `Driver in simulation mode`

5.4.9 Linking variables

The zenon driver is allocated to the zenon Logic variables via the names converted into capital letters. If the zenon Runtime requests a variable from the driver, it forwards this to the simulation. If no simulation is loaded, the variable receives status `Switched off (OFF/VSB_N_UPD)` when the driver is stopped. Otherwise it receives status `Invalid (INVALID/_VSB_I_BIT)`. Variables of `driver variable` object type are never requested by the simulation.

5.4.10 time stamp

o use time stamping, in zenon Logic under *Project -> Project parameters -> Extended -> Compiler -> Options -> Create status bit for variables with a profile* must be active.

If the the date is set for a variable in Runtime, the value of date and time is used as a time stamp. If the date or time of a variable changes during the last cycle, this always triggers transmission of the values to the driver. This also applies if neither the value nor the status have changed.

Most of the drivers only transfer a new time stamp to the Runtime when value and/or status have also changed. An exception is for example the IEC870 driver which also transfers new time stamps from the hardware (or *simulation - programmed*) without the value or status having changed to the Runtime.



Informazioni

It must be ensured that the date and time always increase. If this is not the case, this can lead to problems when archiving. The date must always have a value higher than or less than 0.

Variables stamped by Runtime are given the status of `Real time external (T_EXTERN/_VSB_RT_E)`.

All variables to be transmitted to the driver that have not been stamped by Runtime receive a joint time stamping in the cycle. As a result, it is ensured that the time corresponds to that of the change. These drivers receive the status `Real time internal (T_INTERN/_VSB_RT_I)`.

DATE & TIME

The following applies for all date and time indications:

- ▶ Times correspond to the UTC.
- ▶ All times must be between Jan 2 1970 und 2038.
- ▶ The dates are converted into a string in the format of YYYY/MM/DD.
- ▶ The time is converted into the format HH:MM:SS .

5.5 Please not for variables in simulation projects

The following is true for the creation and modification of variables in simulation projects:

1. zenon Logic

If variables are created for a simulation project in the zenon Logic Workbench, they do not have a valid addressing for the communication. If the communication is switched to hardware, the communication can be interrupted when the new variables are used in screens without adjusting the addressing correctly.

2. Rename

If a variable is renamed in a simulation project, the name of the variable is also changed in zenon accordingly. So for example you must adjust the names of variables which are used in a VBA project also in the VBA code.

3. Integrated solutions

Variables of integrated solutions such as zenon Logic, IEC870 or IEC850 must not be renamed in the simulation project. In this case the variables lose their project information. For example: If you rename `Project0/Global/NewVar` to `myNewVariableNewVar` in the simulation project, it becomes `myNewVar` in zenon. After the renaming a communication with the zenon Logic Runtime is impossible.

4. Communication based on variable names

Variables of a driver which communicates based on the variable names must not be renamed in the simulation project.

5. Deleting or importing variables

If a variable is deleted or imported while `online change` is activated in zenon Logic, all variables are removed and inserted again in zenon Logic.

6. Variables with active "embed" symbol

Variables which exist in a program and whose "embed" symbol is active are not available via the Shared Memory interface in zenon. If a variable with an active "embed" symbol is moved from the global area to a program, the variable is deleted in zenon.

5.6 Error messages

Error messages in zenon editor:

Error message	cause and solution
The chosen name was already used for another driver or contains invalid characters! Please select another name.	<p>The new name of the simulation project is already used in the project or is not valid.</p> <ul style="list-style-type: none"> ▶ Give it a valid name
The simulation project from the <File name>/<Description> driver has not been compiled and cannot be supported! Please compile the project in Workbench.	<p>A simulation project is available for the driver in Workbench. This was not compiled however.</p> <ul style="list-style-type: none"> ▶ Start the zenon Logic Workbench with this driver's simulation project ▶ compile the project <p>Tip: It does not make a difference if the driver settings are set to simulation - programmed in the driver settings</p>
Write queue full! Write command for <Name Variable> lost!	Displays the loss of a write command.