zenon
by COPA-DATA

zenon driver manual
OMR_FINS

v.8.20

Contents

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 OMR_FINS

3 OMR_FINS - data sheet

General:	
Driver file name	OMR_FINS.exe
Driver name	Omron FINS Driver

General:	
PLC types	OMRON SYSMAC
PLC manufacturer	Omron

Driver supports:	
Protocol	Omron FINS
Addressing: Address-based	Address based
Addressing: Name-based	--
Spontaneous communication	--
Polling communication	X
Online browsing	--
Offline browsing	--
Real-time capable	--
Blockwrite	X
Modem capable	--
RDA numerical	--
RDA String	--
Hysteresis	--
extended API	--
Supports status bit **WR-SUC**	--
alternative IP address	--

Requirements:	
Hardware PC	RS 232 serial interface, cable type: SYSMAC WAY; standard network card
Software PC	--

Requirements:	
Hardware PLC	--
Software PLC	--
Requires v-dll	--

Platforms:	
Operating systems	Windows 10; Windows 7; Windows 8; Windows 8.1; Windows Server 2008 R2; Windows Server 2012; Windows Server 2012 R2; Windows Server 2016

4 Driver history

Date	Driver version	Change
07.07.08	1300	Created driver documentation

DRIVER VERSIONING

The versioning of the drivers was changed with zenon 7.10. There is a cross-version build number as of this version. This is the number in the 4th position of the file version,
For example: **7.10.0.4228** means: The driver is for version **7.10** service pack **0**, and has the build number **4228**.

Expansions or error rectifications will be incorporated into a build in the future and are then available from the next consecutive build number.

Example

A driver extension was implemented in build **4228**. The driver that you are using is build number **8322**. Because the build number of your driver is higher than the build number of the extension, the extension is included. The version number of the driver (the first three digits of the file version) do not have any significance in relation to this. The drivers are version-agnostic

5 Requirements

This chapter contains information on the requirements that are necessary for use of this driver.

5.1 PC

HARDWARE

Serial interface RS232 or standard network card (TCP/IP)

SOFTWARE

Copy the driver file OMR_FINS.exe into the current program directory (unless it is already there) and enter it into the file TREIBER_EN.XML with the tool driverinfo.exe.

> **Information**
>
> Windows CE is not supported at the moment.

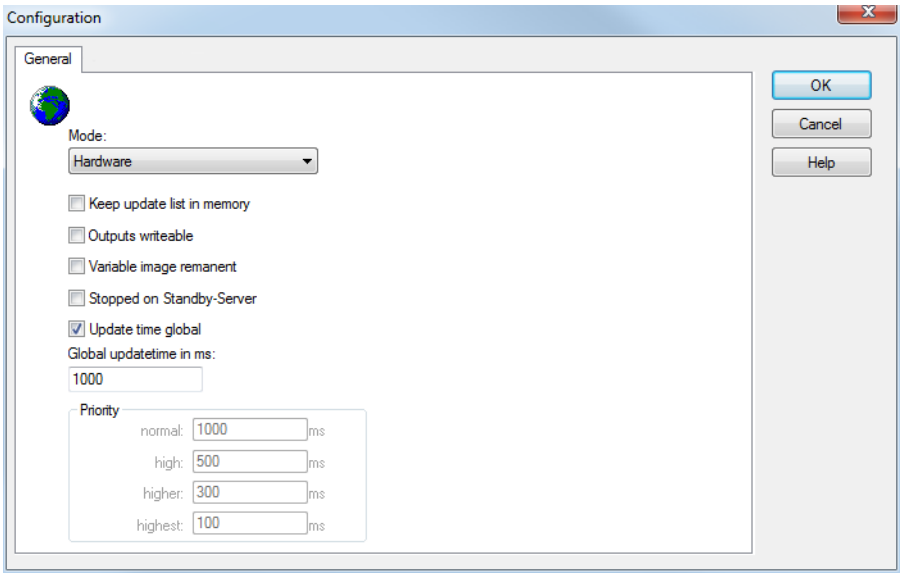
6 Configuration

In this chapter you will learn how to use the driver in a project and which settings you can change.

> **Information**
>
> Find out more about further settings for zenon variables in the chapter Variables of the online manual.

6.1 General

The configuration dialog is opened when a driver is created. In order to be able to open the dialog later for editing, double click on the driver in the list or click on the **Configuration** property.



Option	Description
Mode	Allows to switch between hardware mode and simulation mode ▸ *Hardware*: A connection to the control is established. ▸ Simulation - static: No communication between to the control is established, the values are simulated by the driver. In this modus the values remain constant or the variables keep the values which were set by zenon Logic. Each variable has its own memory area. E.g. two variables of the type marker with offset 79 can have different values in the Runtime and do not influence each other. Exception: The simulator driver. ▸ *Simulation - counting*: No communication between to the control is established, the values are simulated by the driver. In this modus the driver increments the values within a value range automatically. ▸ *Simulation - programmed*: No communication is established to the PLC. The

Option	Description
	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 a zenon Logic Runtime which is integrated in the driver. For details see chapter Driver simulation.
Keep update list in the memory	Variables which were requested once are still requested from the control even if they are currently not needed. This has the advantage that e.g. multiple screen switches after the screen was opened for the first time are executed faster because the variables need not be requested again. The disadvantage is a higher load for the communication to the control.
Output can be written	▸ *Active*: Outputs can be written. ▸ *Inactive*: Writing of outputs is prevented. **Note**: Not available for every driver.
Variable image remanent	This option saves and restores the current value, time stamp and the states of a data point. Fundamental requirement: The variable must have a valid value and time stamp. The variable image is saved in hardware mode if one of these statuses is active: ▸ User status *M1* (*0*) to *M8* (*7*) ▸ *REVISION(9)* ▸ *AUS(20)* ▸ *ERSATZWERT(27)* The variable image is always saved if: ▸ the variable is of the **Communication details** object type ▸ the driver runs in simulation mode. (not programmed simulation) The following states are not restored at the start of the Runtime:

Option	Description
	▸ *SELECT(8)* ▸ *WR-ACK(40)* ▸ *WR-SUC(41)* The mode **Simulation - programmed** at the driver start is not a criterion in order to restore the remanent variable image.
Stop on Standby Server	Setting for redundancy at drivers which allow only one communication connection. For this the driver is stopped at the Standby Server and only started at the upgrade. **Attention:** If this option is active, the gapless archiving is no longer guaranteed. ▸ *Active*: Sets the driver at the not-process-leading Server automatically in a stop-like state. In contrast to stopping via driver command, the variable does not receive status **switched off** but an empty value. This prevents that at the upgrade to the Server irrelevant values are created in the AML, CEL and Historian. Default: *inactive* **Note:** Not available if the CE terminal serves as a data server. You can find further information in the zenon Operator manual in the CE terminal as a data server chapter.
Global Update time	Setting for the global update times in milliseconds: ▸ *Active*: The set **Global update time** is used for all variables in the project. The priority set at the variables is not used. ▸ *Inactive*: The set priorities are used for the individual variables. **Exceptions:** Spontaneous drivers ignore this option. They generally use the shortest possible update time. For details, see the **Spontaneous driver update time** section.

Option	Description
Priority	The polling times for the individual priority classes are set here. All variables with the according priority are polled in the set time. The variables are allocated separately in the settings of the variable properties. The communication of the individual variables can be graded according to importance or required topicality using the priority classes. Thus the communication load is distributed better. **Attention:** Priority classes are not supported by each driver, e.g. spontaneously communicating zenon drivers.

CLOSE DIALOG

Option	Description
OK	Applies all changes in all tabs and closes the dialog.
Cancel	Discards all changes in all tabs and closes the dialog.
Help	Opens online help.

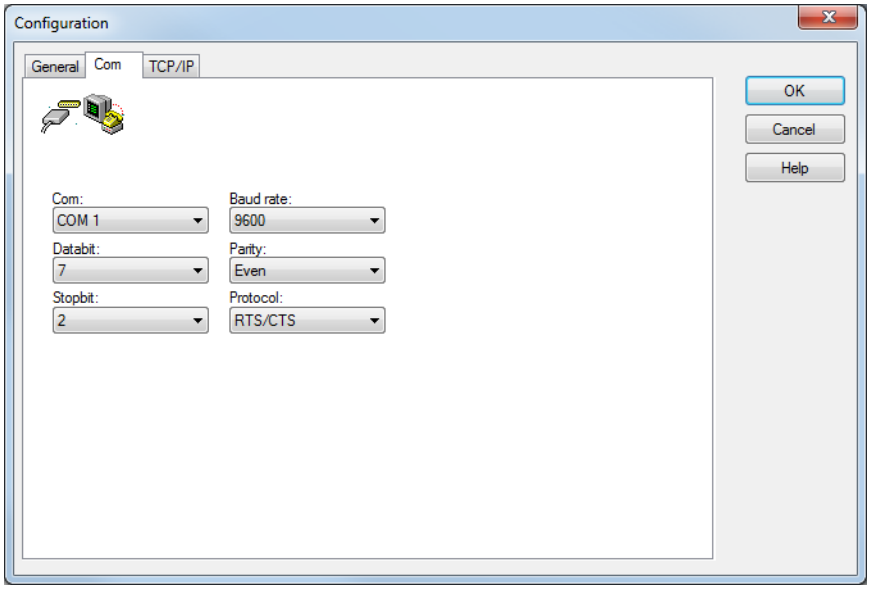
UPDATE TIME FOR SPONTANEOUS DRIVERS

With spontaneous drivers, for **Set value**, **advising** of variables and **Requests**, a read cycle is triggered immediately - regardless of the set update time. This ensures that the value is immediately available for visualization after writing. The update time is generally 100 ms.

Spontaneous drivers are **ArchDrv**, **BiffiDCM**, **BrTcp32**, **DNP3**, **Esser32**, **FipDrv32**, **FpcDrv32**, **IEC850**, **IEC870**, **IEC870_103**, **Otis**, **RTK9000**, **S7DCOS**, **SAIA_Slave**, **STRATON32** and **Trend32**.

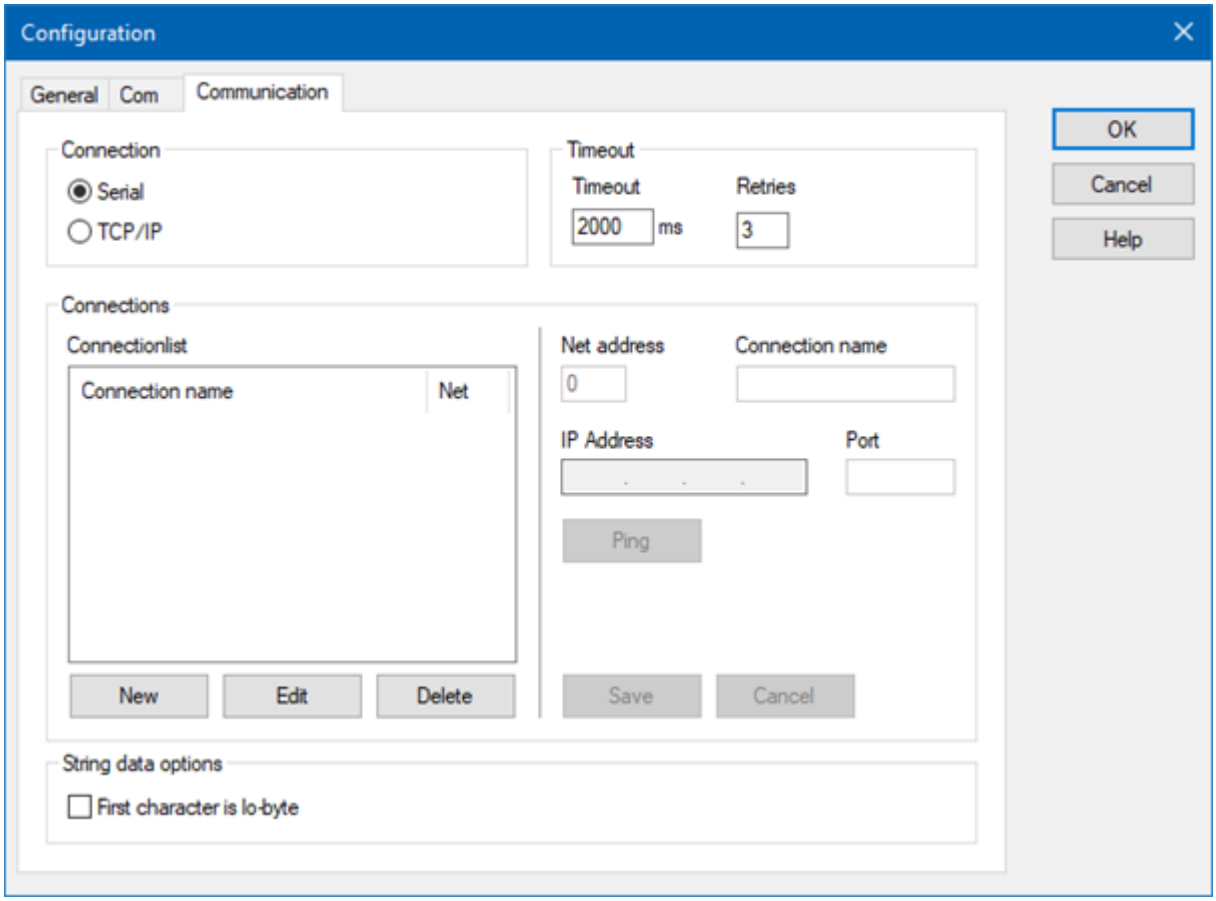
6.2 COM

Configuration of the communication parameters for the serial interface



Parameter	Description
Com	Selection of the serial interface, to which the PLC is connected.
Baud rate	Selection baud rate. Amend to controller. Select from drop-down list: Default: *9600* Input range: *110* to *256000*
Data bit	Data word size in Bit: Default: *7*
Parity	Settings for the parity of the connection Default: *Even*
Stop bit	Number of stopbits for the connection Default: *2*
Protocol	Protocol of the connection. Default: *RTS/CTS*

6.3 Communication



CONNECTION

Parameter	Description
Connection	Connection type. Selection from option field. ▸ *Serial*: Communication via serial connection. ▸ *TCP/IP*: communication via TCP/IP protocol.

TIMEOUT

Options for Error response time.

Parameter	Description
Timeout	Waiting time for establishing a connection in milliseconds.

Parameter	Description
Wiederholungen	Number of retries if establishing a connection is not successful.

CONNECTIONS

Settings of the connections.

Parameter	Description
Connection list	List of defined connections to PLCs.
Net address	Corresponds to the **Net address** for variable properties.
Connection name	Freely definable name.
IP address	IP address of PLC.
Port	Port address of PLC. You can find details in the manual of your PLC.
Ping	Sends a ping to the IP address that is configured for this connection. Allows the connection to the device to b tested. If the ping is concluded negatively, check the IP address and check to see if the device is online.
New	Establishes a new connection.
Edit	Opens highlighted connection for editing.
Delete	Deletes highlighted connection from the list.
Save	Accepts all changes for edited connection and closes editing option.
Cancel	Discards all changes for edited connection and closes editing option.
OK	Accept changes in the dialog and close dialog. Only available if no connection is in the "edit" state.
Cancel	Discards all changes and closes the dialog.
Help	Opens online help.

STRING DATA OPTIONS

Setting for STRINGS.

Parameter	Description
First character in the lo-byte	Set the byte order for STRINGS in the controller with this checkbox. ▸ *Active*: Byte order is reversed. Is required by the Omron NJ, for example. Default: *inactive*

Information

Maximum number of connections: 256 (0-255).

7 Addressing

Group/Property	Description
General	Property group for general settings.
Name	Freely definable name. **Attention:** For every zenon project the name must be unambiguous.
Identification	Freely definable identification. E.g. for Resources label, comments, ...
Addressing	
Net address	Network address of variables. This address refers to the bus address in the connection configuration of the driver. This defines the PLC, on which the variable resides.
Data block	For variables of object type *Extended data block*, enter the datablock number here. Adjustable from *0* to *4294967295*. You can take the exact maximum area for data blocks from the manual of the PLC.

Group/Property	Description
Offset	Offset of variables. Equal to the memory address of the variable in the PLC. Adjustable from *0* to *4294967295*.
Alignment	not used for this driver
Bit number	Offset of variables. Equal to the memory address of the variable in the PLC. Adjustable from *0* to *4294967295*.
String length	Only available for String variables. Maximum number of characters that the variable can take.
Driver connection/Data Type	Data type of the variable. Is selected during the creation of the variable; the type can be changed here. **Attention:** If you change the data type later, all other properties of the variable must be checked and adjusted, if necessary.
Driver connection/Driver Object Type	Object type of the variables. Depending on the driver used, is selected when the variable is created and can be changed here.
Driver connection/Priority	Setting the priority class. The variable of the priority class is thus assigned as it was configured in the driver dialog in the **General** tab. The priority classes are only used if the **global update time** is deactivated. If the **global update time** option is activated and the priority classes are used, there is an error entry in the log file of the system. The driver uses the highest possible priority.

VARIABLE ADDRESSING VIA

Address

The variables are allocated to the memory area of the PLC by their offset.

> **Information**
>
> The addressing of the PLC is carried out as WORD. For uneven string length always one additional character is read or written.

8 Driver objects and datatypes

Driver objects are areas available in the PLC, such as markers, data blocks etc. Here you can find out which driver objects are provided by the driver and which IEC data types can be assigned to the respective driver objects.

8.1 Driver objects

DRIVER OBJECT TYPES AND SUPPORTED IEC DATA TYPES FOR PROCESS VARIABLES IN THE CONTROL SYSTEM

Driver object types	Channel type	Supported data types (DataType)	Read	Write	Comment
IR Area	64	*BOOL, INT, UINT*	X	X	
HR Area	65	*BOOL, INT, DINT, REAL, UINT, UDINT, STRING*	X	X	
AR Area	66	*BOOL, INT, UINT*	X	X	
TIM/CNT	68	*BOOL*	X	X	
EM Area	73	*BOOL, INT, DINT, REAL, UINT, UDINT, STRING*	X	X	Define Area No 0...12 in the data block Offset 0...32767
DM Area	70	*BOOL, INT, DINT, REAL, UINT, UDINT, STRING*	X	X	
DM Area BCD	71	*INT, DINT, UINT, UDINT*	X	X	
CIO Area	71	*BOOL, INT, DINT, REAL UINT, UDINT, STRING*	X	X	
WORK Area	72	*BOOL, INT, DINT, REAL, UINT, UDINT, STRING*	X	X	

CHANNEL TYPE

The term **Kanaltyp** is the internal numerical name of the driver object type. It is also used for the extended DBF import/export of the variables.

"**Kanaltyp**" is used for advanced CSV import/export of variables in the "**HWObjectType**" column.

9 Mapping of the data types

All variables in zenon are derived from IEC data types. The following table compares the IEC datatypes with the datatypes of the PLC.

EXAMPLES FOR ALL POSSIBLE IEC DATA TYPES

SPS	zenon
I16	INT
I32	DINT
U16	UINT
U32	UDINT
REAL	REAL
Boolean	BOOL

DATA TYPE

The term **data type** is the internal numerical identification of the data type. It is also used for the extended DBF import/export of the variables.

10 Driver-specific functions

The driver supports the following functions:

Driver-specific function	Description
Blockwrite	**X** For further details see the **blockwrite** section.

Driver-specific function	Description
Browsing	--
Real-time stamping	--
Extended error file	--
Error file	The driver supports the common error files; with zenon 6.20 or higher, central logging is possible.
Error timeout	No settings possible.
RDA	--
Redundancy	**X**
Serial logging	**X** For details, see the **serial logging** section.
Access methods	▸ **Polling**: Cyclical query of the variable or the driver. ▸ **Spontaneous**: Not supported

Key:

- **X**: supported
- --: not supported

BLOCKWRITE

To activate **Blockwrite**, an entry must be set in **project.ini**:

- Section:
  [OMR_FINS]
- Entry:
  BLOCKWRITE=*1*

SERIAL LOGGING

To activate serial logging, an entry must be set in **project.ini**:

- Section:
  [RS232LOG]
- Entry:
  LOGCOMx=

  x: Number of the selected interface.

- *0*: no logging
- *1*: serial logging active
  A file called **LOG_COMxxx.TXT** is created in the folder of the driver.

Examples for the activation of logging for **COM 1**:

[RS232LOG]

LOGCOM1=*1*

Information

Activate the logging only in the event of problems and only for a short time. Logging needs considerable resources. Also, the LOG file occupies a lot of memory within a short time.

11 Driver command function

The zenon **Driver commands** function is to influence drivers using zenon.
You can do the following with a driver command:

- Start
- Stop
- Shift a certain driver mode
- Instigate certain actions

Note: This chapter describes standard functions that are valid for most zenon drivers.
Not all functions described here are available for every driver. For example, a driver that does not, according to the data sheet, support a modem connection also does not have any modem functions.

Attention

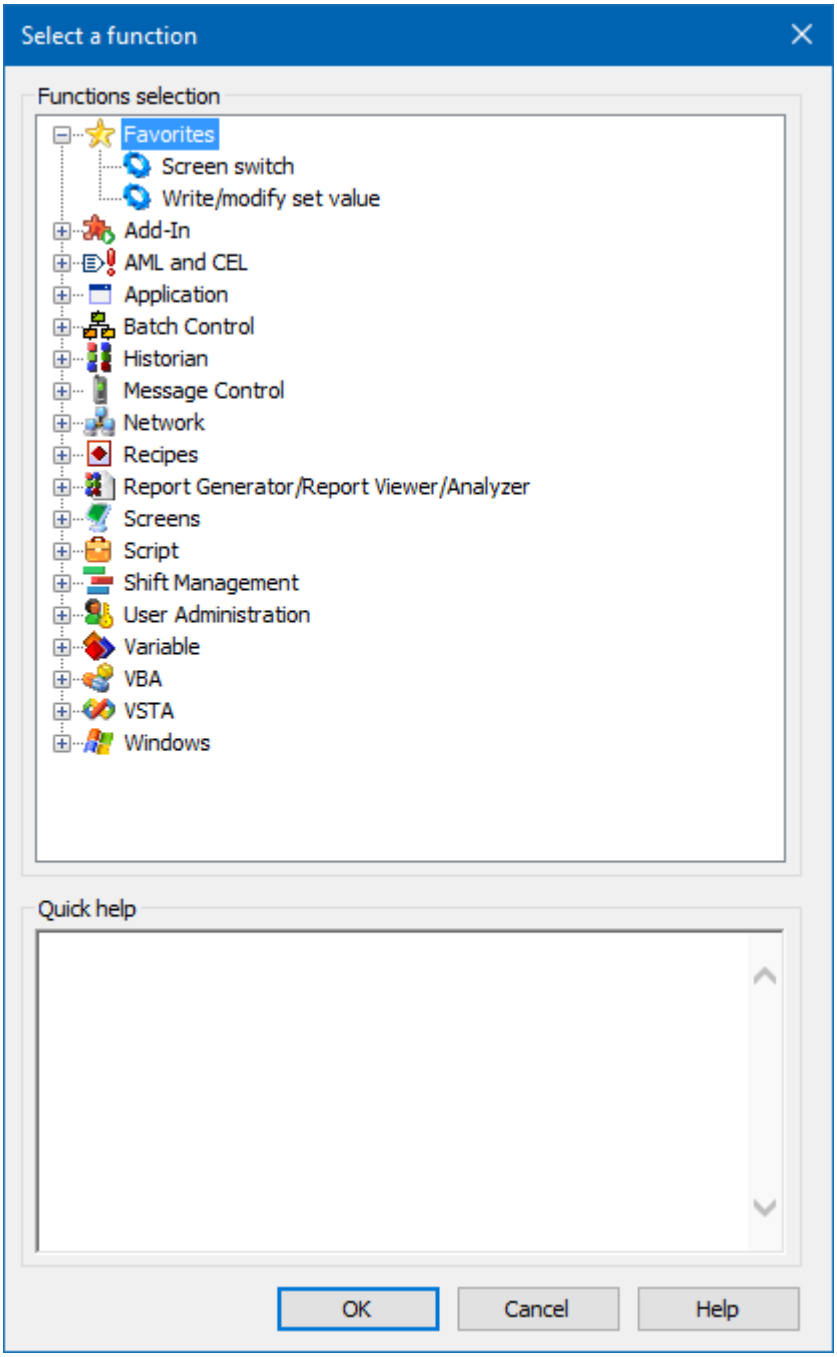
The zenon **Driver commands** function is not identical to driver commands that can be executed in the Runtime with Energy drivers!

CONFIGURATION OF THE FUNCTION

Configuration is carried out using the **Driver commands** function.
To configure the function:

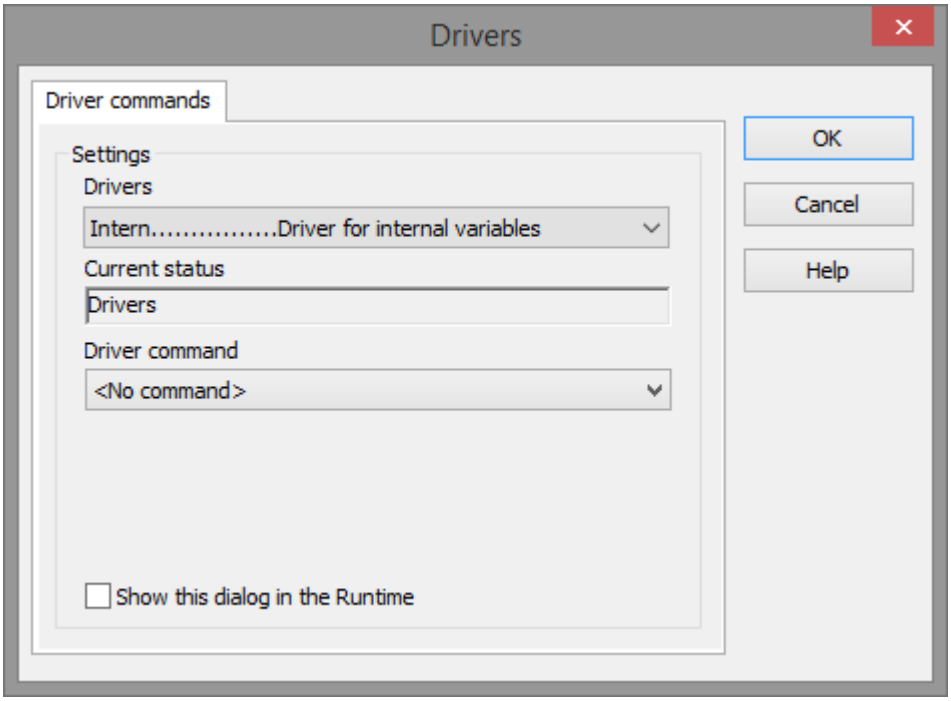
1. Create a new function in the zenon Editor.

The dialog for selecting a function is opened



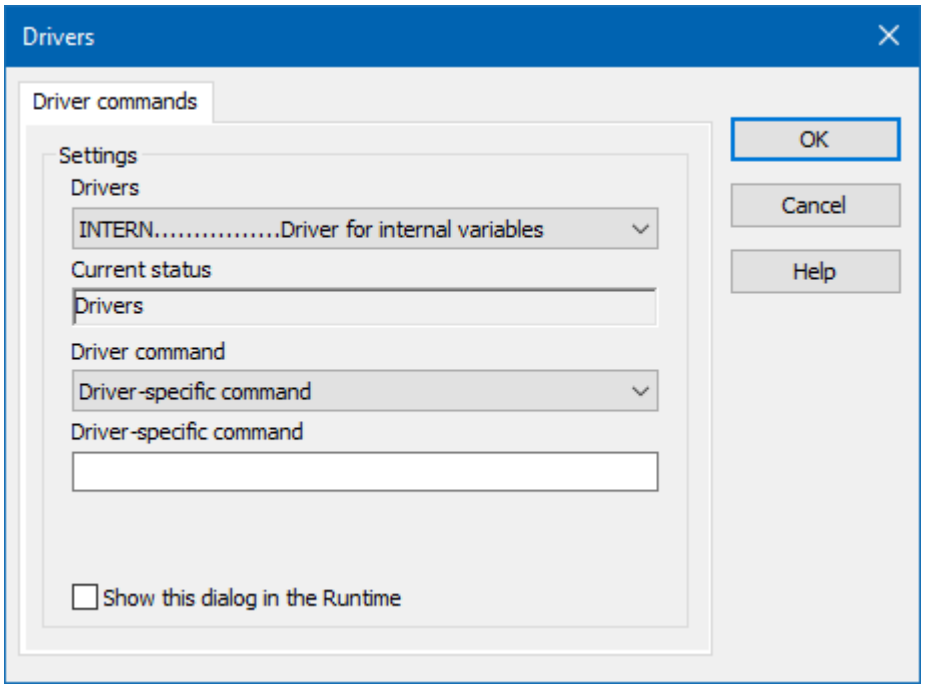
2. Navigate to the node **Variable.**
3. Select the **Driver commands** entry.

The dialog for configuration is opened



4. Select the desired driver and the required command.
5. Close the dialog by clicking on **OK** and ensure that the function is executed in the Runtime. Heed the notices in the **Driver command function in the network** section.

DRIVER COMMAND DIALOG



Option	Description
Driver	Selection of the driver from the drop-down list. It contains all drivers loaded in the project.
Current condition	Fixed entry that is set by the system. no function in the current version.
Driver command	no function in the current version. For details on the configurable driver commands, see the **available driver commands** section.
Driver-specific command	Entry of a command specific to the selected driver. **Note:** Only available if, for the **driver command** option, the *driver-specific command* has been selected.
Show this dialog in the Runtime	Configuration of whether the configuration can be changed in the Runtime: ▸ *Active*: This dialog is opened in the Runtime before executing the function. The configuration can thus still be changed in the Runtime before execution. ▸ *Inactive*: The Editor configuration is applied in the Runtime when executing the function. Default: *inactive*

CLOSE DIALOG

Options	Description
OK	Applies settings and closes the dialog.
Cancel	Discards all changes and closes the dialog.
Help	Opens online help.

AVAILABLE DRIVER COMMANDS

These driver commands are available - depending on the selected driver:

Driver command	Description
No command	No command is sent. A command that already exists can thus be removed from a configured function.

Driver command	Description
Start driver (online mode)	Driver is reinitialized and started. **Note:** If the driver has already been started, it must be stopped. Only then can the driver be re-initialized and started.
Stop driver (offline mode)	Driver is stopped. No new data is accepted. **Note:** If the driver is in offline mode, all variables that were created for this driver receive the status *switched off* (*OFF*; Bit *20*).
Driver in simulation mode	Driver is set into simulation mode. The values of all variables of the driver are simulated by the driver. No values from the connected hardware (e.g. PLC, bus system, ...) are displayed.
Driver in hardware mode	Driver is set into hardware mode. For the variables of the driver the values from the connected hardware (e.g. PLC, bus system, ...) are displayed.
Driver-specific command	Entry of a driver-specific command. Opens input field in order to enter a command.
Driver - activate set setpoint value	Write set value to a driver is possible.
Driver - deactivate set setpoint value	Write set value to a driver is prohibited.
Establish connecton with modem	Establish connection (for modem drivers) Opens the input fields for the hardware address and for the telephone number.
Disconnect from modem	Terminate connection (for modem drivers)
Driver in counting simulation mode	Driver is set into counting simulation mode. All values are initialized with *0* and incremented in the set update time by *1* each time up to the maximum value and then start at *0* again.
Driver in static simulation mode	No communication to the controller is established. All values are initialized with *0*.
Driver in programmed simulation mode	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.

DRIVER COMMAND FUNCTION IN THE NETWORK

If the computer on which the **Driver commands** function is executed is part of the zenon network, further actions are also carried out:

- A special network command is sent from the computer to the project server.
  It then executes the desired action on its driver.
- In addition, the Server sends the same driver command to the project standby.
  The standby also carries out the action on its driver.

This makes sure that Server and Standby are synchronized. This only works if the Server and the Standby both have a working and independent connection to the hardware.

12 Test

TESTED WITH THE FOLLOWING HARDWARE AND SOFTWARE

omron SYSMAC CJ1M /CPU11 (Programmable Controller)

TESTING ENVIRONMENT

Tested with serial interface (serial cable, no null modem cable, cable type: SYSMAC WAY), Parameter: 9600, 7, 2, even, no and TCP/IP connection with default port 9600. The IP-Address of the PLC can be set via Web-interface eg: . http://[IP-Address of the PLC\\/0 or via the DM memory area with serial connection (Offset: m = D30000 + (100 x unit number) + 98; in these 4 Bytes, the IP-Addresse is entered, which will be used by the PLC).

For details, please check the Omron Documentation 'W441E101_Ethernet+CPUs_Operation_Manual.pdf'.

LIMITATIONS

In our test, wrting on HR and AR was impossible. With the driver object type IR neither write nor read was possible.

13 Error analysis

Should there be communication problems, this chapter will assist you in finding out the error.

13.1 Analysis tool

All zenon modules such as Editor, Runtime, drivers, etc. write messages to a joint log file. To display them correctly and clearly, use the Diagnosis Viewer program that was also installed with zenon. You can find it under **Start/All programs/zenon/Tools 8.20 -> Diagviewer.**

zenon driver log all errors in the LOG files.LOG files are text files with a special structure. The default folder for the LOG files is subfolder **LOG** in the folder **ProgramData**. For example:

%ProgramData%\COPA-DATA\LOG.

Attention: With the default settings, a driver only logs error information. With the Diagnosis Viewer you can enhance the diagnosis level for most of the drivers to "Debug" and "Deep Debug". With this the driver also logs all other important tasks and events.

In the Diagnosis Viewer you can also:

- Follow newly-created entries in real time
- customize the logging settings
- change the folder in which the LOG files are saved

Note:

1. The Diagnosis Viewer displays all entries in UTC (coordinated world time) and not in local time.
2. The Diagnosis Viewer does not display all columns of a LOG file per default. To display more columns activate property **Add all columns with entry** in the context menu of the column header.
3. If you only use **Error-Logging**, the problem description is in the column **Error text**. For other diagnosis level the description is in the column **General text**.
4. For communication problems many drivers also log error numbers which the PLC assigns to them. They are displayed in **Error text** or **Error code** or **Driver error parameter** (**1** and **2)**. Hints on the meaning of error codes can be found in the driver documentation and the protocol/PLC description.
5. At the end of your test set back the diagnosis level from **Debug** or **Deep Debug**. At **Debug** and **Deep Debug** there are a great deal of data for logging which are saved to the hard drive and which can influence your system performance. They are still logged even after you close the Diagnosis Viewer.

> **Attention**
>
> In Windows CE errors are not logged per default due to performance reasons.

You can find further information on the Diagnosis Viewer in the Diagnose Viewer manual.

13.2 Driver monitoring

Runtime monitors the availability of the driver by means of a watchdog. If a driver is no longer available, the *INVALID* status bit is also set for all checked-in variables.

Possible causes for a triggering of the watchdog:

- The driver process is no longer running.

  Check whether the driver EXE file is still running in the Task Manager.

- Operating system is busy with processes that have a higher priority.

  Check the configuration of your system to see whether there is sufficient memory and CPU power. In this case, the driver only resets the *INVALID* status bit if there is a value change on the connected party. Static values retain the *INVALID* status bit until the next time the Runtime or the driver is started.

CONFIGURATION OF WATCHDOG

For the monitoring of communication in the Runtime, the connection to the driver is checked in a fixed, prescribed time period of 60 seconds. This process is repeated several times. If, within 5 attempts (= within 5 minutes), no valid connection to the driver is detected, the *INVALID* bit is set for the checked-in (*advised*) variables. In addition, the *INVALID* bit is also set when new variables are advised. The *INVALID* bit will no longer be reset.

Corresponding LOG entries are created for this.

LOG ENTRY

An error message is logged in the LOG when the watchdog is triggered:

Parameter	Description
Communication with driver:<drvExe>/<drvDesc>(id:<drvId>) timed out. No communication for <time> ms.	No communication with driver within the given time. - <*time*>: Time (in milliseconds) - <*drvDesc*>: Driver name - <*drvExe*>: Driver EXE name - <*drvId*>: Driver ID in the zenon project
Communication with %s timed out. Invalid-Bit will be set.	Communication to the *%s* driver could not be established after 5 attempts within 60 seconds. The *INVALID* bit is set for the variable.
Communication with %s timed out. Timeout happened %d times	Communication to the *%s* driver could not be established after *%d* times within 60 seconds.

13.3 Error numbers

In case of communication problems an entry in the error log file of the driver is generated; here the error cause is stated with a number.

Information

Refer to the FINS documentation section 5-1-3 „End Codes" for an explanation of the error numbers. Here all error numbers are documented.

13.4 Check list

- If the PLC is connected correctly (serial connection cable – no null modem cable with serial connection) and the connection has been configured correct 9600,7,2,even,no)?
- Is the IP address of the PLC configured correctly (with connection via TCP/IP)? Can the PLC be contacted via the Web interface or with e.g. ping in the driver configuration/Conn. TCP/IP?
- Did you analyze the error file (up to zenon version 6.01) or the central log file (version zenon 6.20 and higher)?