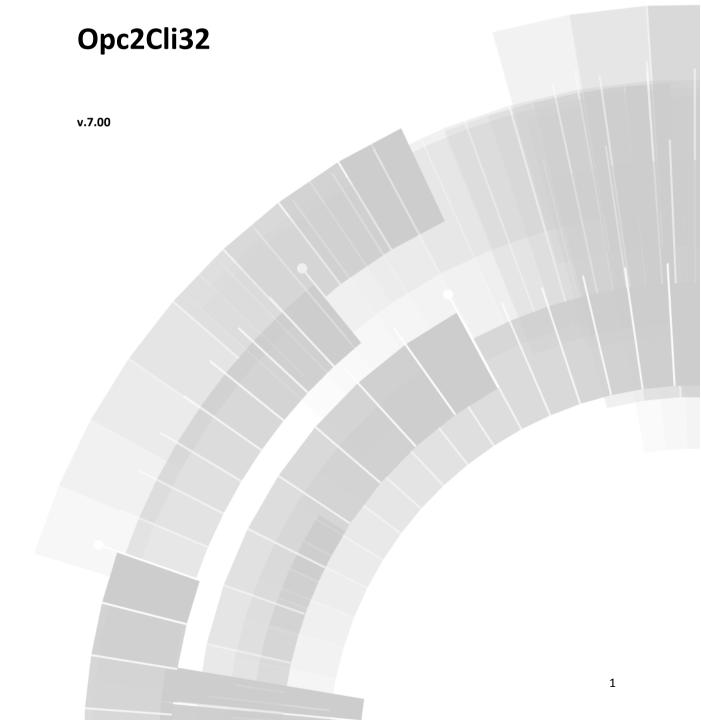


zenon driver manual





© 2012 Ing. Punzenberger COPA-DATA GmbH

All rights reserved.

Distribution and/or reproduction of this document or parts thereof in any form are permitted solely with the written permission of the company COPA-DATA. The technical data contained herein has been provided solely for informational purposes and is not legally binding. Subject to change, technical or otherwise.



Contents

1.	Welcome to COPA-DATA help			
2.	Opc2Cli32			5
3.	OPC2CLI32 - Data sheet5			
4.	Drive	r histor	ту	7
5. Requirements			7	
	5.1	PC		8
6.	Confi	guratio	on	8
	6.1	Creatin	ng a driver	8
	6.2	Setting	gs in the driver dialog	10
		6.2.1	General	10
		6.2.2	OPC	13
		6.2.3	Browser	15
7.	Creat	ing vari	iables	16
	7.1	Creatin	ng variables in the Editor	16
	7.2	Addres	ssing	20
	7.3	Driver o	objects and datatypes	20
		7.3.1	Driver objects	21
		7.3.2	Mapping of the data types	21
	7.4	Creatin	ng variables by importing	22
		7.4.1	XML import of variables from another zenon project	22
		7.4.2	DBF Import/Export	23
		7.4.3	Online import	29
	7.5	Driver \	variables	29
8.	Drive	er-specif	fic functions	35
9.	Drive	er comm	nands	36
10	Error	analysi	ic	27



10.1	Analysis tool	. 38
10.2	Logging	. 39
10.3	Check list	. 41



1. Welcome to COPA-DATA help

GENERAL HELP

If you miss any information in this help chapter or have any suggestions for additions, please feel free to contact us via e-mail: documentation@copadata.com (mailto:documentation@copadata.com).

PROJECT SUPPORT

If you have concrete questions relating to your project, please feel free to contact the support team via e-mail: support@copadata.com (mailto:support@copadata.com)

LICENSES AND MODULES

If you realize that you need additional licenses or modules, please feel free to contact the sales team via e-mail: sales@copadata.com (mailto:sales@copadata.com)

2. Opc2Cli32

3. OPC2CLI32 - Data sheet



General:	
Driver file name	OPC2CLI32.exe
Driver description	OPC Client V2-0
PLC types	The OPC Server must have implemented the interface in the version 1.0A or 2.0xBrowsing is supported.
PLC manufacturer	OPC;

Driver supports:	
Protocol	OPC;
Addressing: address based	x
Addressing: name based	x
Spontaneous communication	х
Polling communication	-
Online browsing	x
Offline browsing	-
Real-time capable	-
Blockwrite	-
Modem capable	-
Serial logging	-
RDA numerical	-
RDA String	-



Prerequisites:	
Hardware PC	-
Software PC	-
Hardware PLC	-
Software PLC	-
Requires v-dll	-

Platforms:	
Operating systems	Windows CE 5.0, CE 6.0; Windows XP, Vista, 7, Server 2003, Server 2008/R2;
CE platforms	x86; ARMV4;

4. Driver history

Date	•	Driver version	Change
07.07	7.08	4700	Created driver documentation

5. Requirements

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



5.1 PC

SOFTWARE

The OPC server must have implemented the interface version 1.0A or 2.0x. If both are implemented, the interface version 2 is used.

Copy the driver file OpcCli32.EXE to the current zenon directory (unless it is already there).

FUNCTIONALITY

Reading and writing of data fields, that are defined on the OPC server. The connection can be established via the variable name or the variable offset in the control system.

6. Configuration

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



💡 Info

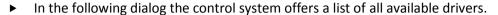
Find out more about further settings for zenon variables in the chapter Variables (main.chm::/15247.htm) of the online manual.

Creating a driver 6.1

In order to create a new driver:

▶ Right-click on priver in the Project Manage and select priver new in the context menu.

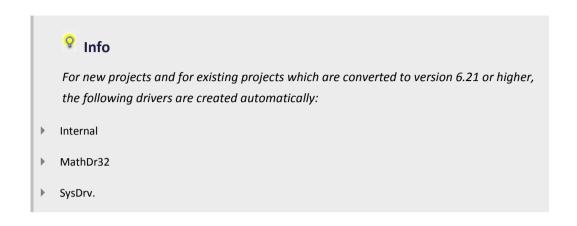






- Select the desired driver and give it a name:
 - The driver name has to be unique, i.e. if one and the same driver is to be used several times in one project, every time a new name has to be given each time.
 - The driver name is part of the file name. Therefore it may only contain characters which are supported by the operating system. Invalid characters are replaced by an underscore (_).
 - Attention: This name cannot be changed later on.
- ► Confirm the dialog with ox. In the following dialog the single configurations of the drivers are defined.
- Only the respective required drivers need to be loaded for a project. Later loading of an additional driver is possible without problems.

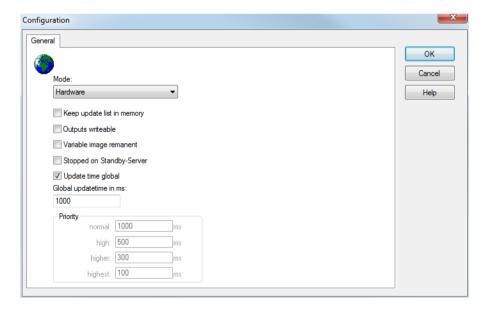




6.2 Settings in the driver dialog

You can change the following settings of the driver:

6.2.1 General





Parameters	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 N communication is established to the PLC. 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 a zenon Logic Runtime which is integrated in the driver. For details see chapter Driver
Keep update list in the memory	simulation (main.chm::/25206.htm). 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	This option saves and restores the current value, time stamp and the states
remanent	of a data point.
	Fundamental requirement: The variable must have a valid value and time stamp.
	The variable image is saved in mode hardware if:
	one of the states S_MERKER_1(0) up to S_MERKER8(7), REVISION(9), AUS(20) or ERSATZWERT(27) is active
	The variable image is always saved if:
	▶ the variable is of the object type Driver variable
	the driver runs in simulation mode. (not programmed simulation)
	The following states are not restored at the start of the Runtime:
	▶ 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 at the Standby Server	Setting for redundancy at drivers which allow only on 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 (statusverarbeitung.chm::/24150.htm) but an empty value. This prevents that at the upgrade to the Server irrelevant values are created in the AML, CEL and Historian.
Global Update time	Active: The set Global update time in ms 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.
Priority	Here you set the polling times for the individual priorities. All variables with the according priority are polled in the set time. The allocation is taken



place for each variable separately in the settings of the variable properties.

The communication of the individual variables are graduated in respect of importance or necessary topicality using the priorities. Thus the communication load is distributed better.

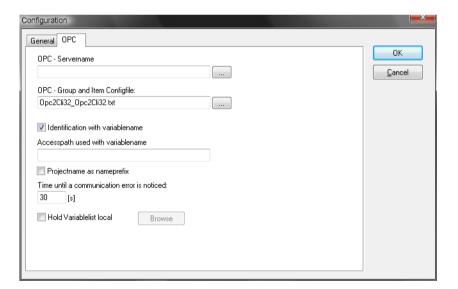
UPDATE TIME FOR CYCLICAL DRIVER

The following applies for cyclical drivers:

For Set value, Advising of variables and Requests, a read cycle is immediately triggered for all drivers regardless of the set update time. This ensures that the value is immediately available for visualization after writing. Update times can therefore be shorter than pre-set for cyclical drivers.

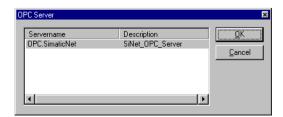
6.2.2 OPC

Select the OPC server and the configuration file.



OPC - SERVERNAME

Enter the OPC server to be used here. The button opens a list of available servers.





An installed OPC server can be selected.

OPC GROUP AND ITEM CONFIG FILE

Name of the configuration file: *.OPC file

Storage directory: Project directory.

Important when multiple OPC drivers were created.

IDENTIFICATION VIA VARIABLE NAME

If this checkbox is active, the zenon variable name will be used as the item name of the OPC server. That means: The variable name must be a valid Item name for the server. No settings are necessary in the browser. This browser tab will be shown automatically if the checkbox is activated. The current variable offset does not matter here.

ACCESS PATH USED WITH VARIABLE NAME

If the connection to the OPC server is realized via the variable name, the access path can be entered here. Is preset by the OPC server, and is only used by some OPC servers.

PROJECT NAME AS NAME PREFIX

If you check this box, the project name followed by a dot will be put in front of the variable for plaint text names to the OPC server.

e.g. project=test_projectvariable=var1OPC server item=test_projekt.var1

TIME UNTIL A COMMUNICATION ERROR IS RECOGNIZED

After this time has passed, a variable must have been initialized; otherwise it will be set to invalid (INVALID). If the value is 0, this check will be deactivated.

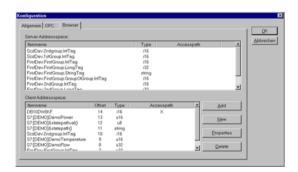


STORE VARIABLE LIST LOCALLY

If you check this box, the variable list read from the server will be stored locally and therefore it will be available later on without the need for another server request. If there were changes to the server, you can initiate a reload of the variable list with the Browse button.

6.2.3 Browser

This settings page is only available if the connection was made via the variable offset.



Parameters	Description
Server address space	Items automatically offered by the server.
Client address space	items available for zenon. These are available in the variable definition with the button "Select". The linking is done via the variable offset here.
Add	With this button, the selected items of the server address space are transferred to the client address space.
New	With this button, new items can be created.





Parameters	Description
Item name	Device: Device name
	Variable name.
	The format must conform to the convention of the OPC server
Access path	Optional. Is used by some OPC servers, preset by these.
Data type	For allowed data types see "4.1 objects for process variables in zenon"

Properties:

Enables later editing of items



Delete:

Enables deleting of Items.

7. Creating variables

This is how you can create variables in the zenon Editor:

7.1 Creating variables in the Editor

Variables can be created:

- as simple variables
- ▶ in arrays main.chm::/15262.htm

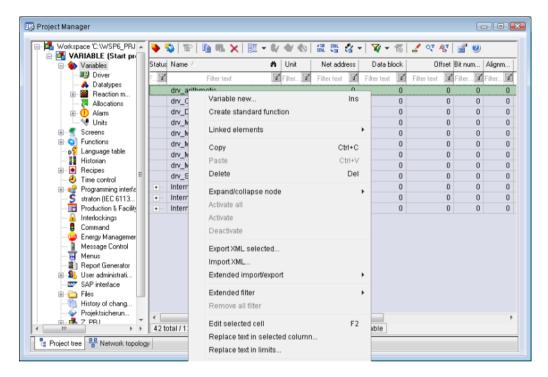


▶ as structure variables main.chm::/15278.htm

VARIABLE DIALOG

To create a new variable, regardless of which type:

Select the New variable command in the variables node in the context menu



- 2. The dialog for configuring variables is opened
- 3. configure the variable



4. The settings that are possible depends on the type of variables



Property	Description
Name	Distinct name of the variable. If a variable with the same name already exists in the project, no additional variable can be created with this name. Attention: The # character is not permitted in variable names. If non-permitted characters are used, creation of variables cannot be completed and the Finish button remains inactive.
Drivers	Select the desired driver from the drop-down list. Note: If no driver has been opened in the project, the driver for internal variables (Intern.exe (Main.chm::/Intern.chm::/Intern.htm)) is automatically loaded.
Driver object type (cti.chm::/28685.h tm)	Select the appropriate driver object type from the drop-down list.



Data type	Select the desired data type. Click on the button to open the selection dialog.
Array settings	Expanded settings for array variables. You can find details in the Arrays chapter.
Addressing options	Expanded settings for arrays and structure variables. You can find details in the respective section.
Automatic element activation	Expanded settings for arrays and structure variables. You can find details in the respective section.

INHERITANCE FROM DATA TYPE

Measuring range, Signal range and Set value are always:

- ▶ derived from the datatype
- ▶ Automatically adapted if the data type is changed

Note for signal range: If a change is made to a data type that does not support the set signal range, the signal range is amended automatically. For example, for a change from INT to SINT, the signal range is changed to 127. The amendment is also carried out if the signal range was not inherited from the data type. In this case, the measuring range must be adapted manually.



7.2 Addressing

Property	Description			
Name	Freely definable name.			
	ATTENTION: the name must be unique within each control system project.			
Identification	Any text can be entered here, e.g. for resource labels, comments			
Net address	not used for this driver			
Data block	For variables of object type Extended data block, enter the datablock number here.			
	Configurable [0 4294967295] . Please look up the exact maximum range for data blocks in the manual of the PLC.			
Offset	Offset of the variable; the memory address of the variable in the PLC. Configurable [0 4294967295]			
Alignment	not used for this driver			
Bit number	Number of the bit within the configured offset.			
	Valid input [0 65535]			
String length	Only available for String variables: Maximum number of characters that the variable can take.			
Driver object type	Depending on the employed driver, an object type is selected during the creation of the variable; the type can be changed here later.			
Data type	Data type of the variable, which is selected during the creation of the variable; the type can be changed here later.			
	ATTENTION: If you change the data type later, all other properties of the variable must be checked and adjusted, if necessary.			

7.3 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.



7.3.1 Driver objects

The following object types are available in this driver:

Driver object type	Channel type	Read / Write	Supported data types	Comment
PLC marker	8	R/W	REAL, BOOL, DINT, UDINT, USINT, INT, UINT, SINT, STRING	
Driver variable	35	R/W	BOOL, SINT, USINT, INT, UINT, DINT, UDINT, REAL, STRING	Variables for the statistical analysis of communication. Find out more in the chapter about the Driver variables (on page 29)

KANALNAME	DATENART	KANALTYP	ОВЈЕКТ
OPCSTRING	12	8	7
OPCFLOAT	5	8	6
OPCDWORD	4	8	5
OPCWORD	2	8	4
OPCBYTE	9	8	3
OPCBIT	8	8	2

7.3.2 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.



PLC	zenon
18	i/u8Bit (signed)
I16	i/u16Bit (signed)
I32	i/u32Bit (signed)
U8	i/u8Bit
U16	i/u16Bit
U32	i/u32Bit
F32	float32
Boolean	Boolean
String	String

Data type: The property Data type is the internal numerical name of the data type. It is also used for the extended DBF import/export of the variables.

7.4 Creating variables by importing

Variables can also be imported by importing them. The XML and DBF import is available for every driver.

7.4.1 XML import of variables from another zenon project

For the import/export of variables the following is true:

- ► The import/export must not be started from the global project.
- ► The start takes place via:
 - Context menu of variables or data typ in the project tree
 - or context menu of a variable or a data type
 - or symbol in the symbol bar variables



Attention

When importing/overwriting an existing data type, all variables based on the existing data type are changed.

Example:

There is a data type XYZ derived from the type INT with variables based on this data type. The XML file to be imported also contains a data type with the name XYZ but derived from type STRING. If this data type is imported, the existing data type is overwritten and the type of all variables based on it is adjusted. I.e. the variables are now no longer INT variables, but STRING variables.

7.4.2 **DBF Import/Export**

Data can be exported to and imported from dBase.

IMPORT DBF FILE

To start the import:

- 1. right-click on the variable list
- 2. in the drop-down menu of Extended export/import... Select the Import dBase Command
- 3. follow the import assistant

The format of the file is described in the chapter File structure.



💡 Info

Note:

- Driver object type and data type must be amended to the target driver in the DBF file in order for variables to be imported.
- dBase does not support structures or arrays (complex variables) at import.



EXPORT DBF FILE

To start the export:

- 1. right-click on the variable list
- 2. in the drop-down menu of Extended export/import... Select the Export dBase Command
- 3. follow the export assistant



Attention

DBF files:

- must correspond to the 8.3 DOS format for filenames (8 alphanumeric characters for name, 3 character suffix, no spaces)
- must not have dots (.) in the path name. e.g. the path C:\users\John.Smith\test.dbf is invalid. Valid: C: \users\JohnSmith\test.dbf
- must be stored close to the root directory in order to fulfill the limit for file name length including path: maximum 255 characters

The format of the file is described in the chapter File structure.



Info

dBase does not support structures or arrays (complex variables) at export.

File structure of the dBase export file

The dBaseIV file must have the following structure and contents for variable import and export:



Attention

dBase does not support structures or arrays (complex variables) when exporting.

DBF files must:

- ▶ correspond to the 8.3 DOS format for filenames (8 alphanumeric characters for name, 3 character suffix, no spaces)
- ▶ Be stored close to the root directory (Root)

DESIGN

Description	Туре	Field size	Comment
KANALNAME	Char	128	Variable name.
			The length can be limited using the MAX_LAENGE entry in project.ini .
KANAL_R	С	128	The original name of a variable that is to be replaced by the new name entered under "KANALNAME" (field/column must be entered manually).
			The length can be limited using the MAX_LAENGE entry in project.ini .
KANAL_D	Log	1	The variable is deleted with the $1\ \rm entry$ (field/column has to be created by hand).
TAGNR	С	128	Identification.
			The length can be limited using the MAX_LAENGE entry in project.ini .
EINHEIT	С	11	Technical unit
DATENART	С	3	Data type (e.g. bit, byte, word,) corresponds to the data type.
KANALTYP	С	3	Memory area in the PLC (e.g. marker area, data area,) corresponds to the driver object type.
HWKANAL	Num	3	Bus address
BAUSTEIN	N	3	Datablock address (only for variables from the data area of the PLC)
ADRESSE	N	5	Offset



BITADR	N	2	For bit variables: bit address
			For byte variables: 0=lower, 8=higher byte
			For string variables: Length of string (max. 63 characters)
ARRAYSIZE	N	16	Number of variables in the array for index variables
			ATTENTION: Only the first variable is fully available. All others
			are only available for VBA or the Recipe Group Manager
			are only available for VBA of the needbe Group Wallager
LES_SCHR	R	1	Write-Read-Authorization
			0: Not allowed to set value.
			1: Allowed to set value.
MIT_ZEIT	R	1	time stamp in zenon (only if supported by the driver)
	N.	2	Duivou anacific ID avantou of the autinoitive chicat
OBJEKT	N	2	Driver-specific ID number of the primitive object
			comprises KANALTYP and DATENART
SIGMIN	Float	16	Non-linearized signal - minimum (signal resolution)
	1.00.0		(-6
SIGMAX	F	16	Non-linearized signal - maximum (signal resolution)
7.7.7.7.7.7	 -	1.5	
ANZMIN	F	16	Technical value - minimum (measuring range)
ANZMAX	F	16	Technical value - maximum (measuring range)
			, , , , , , , , , , , , , , , , , , , ,
ANZKOMMA	N	1	Number of decimal places for the display of the values
			(measuring range)
	 -	10	
UPDATERATE	F	19	Update rate for mathematics variables (in sec, one decimal
			possible)
			not used for all other variables
MEMTIEFE	N	7	Only for compatibility reasons
	.,	•	only to companionly reasons
HDRATE	F	19	HD update rate for historical values (in sec, one decimal possible)
	l		
HDTIEFE	N	/	HD entry depth for historical values (number)
NACHSORT	R	1	HD data as postsorted values
		_	
DRRATE	F	19	Updating to the output (for zenon DDE server, in [s], one decimal
			possible)
IIVOE DI II	-	1.5	Desition house and for
HYST_PLUS	F	16	Positive hysteresis, from measuring range
HYST MINUS	F	16	Negative hysteresis, from measuring range
	<u> </u>		
PRIOR	N	16	Priority of the variable
	_		<u> </u>
REAMATRIZE	С	32	Allocated reaction matrix



F	16	Substitute value, from measuring range
F	16	Minimum for set value actions, from measuring range
F	16	Maximum for set value actions, from measuring range
R	1	Get value from standby server; the value of the variable is not requested from the server but from the standby-server in redundant networks
С	128	Resource label. Free string for export and display in lists. The length can be limited using the MAX_LAENGE entry in project.ini .
R	1	Non-linear value adaption: 0: Non-linear value adaption is used 1: non linear value adaption is not used
С	128	Linked VBA macro for reading the variable value for non-linear value adjustment.
С	128	Linked VBA macro for writing the variable value for non-linear value adjustment.
N	16	Linked counter REMA.
N	16	Gradient overflow for counter REMA.
F (= = = R	16

△ Attention.

When importing, the driver object type and data type must be amended to the target ${\it driver in the DBF file in order for variables to be imported.}$

LIMIT DEFINITION

Limit definition for limit values 1 to 4, and status 1 to 4:



Description	Туре	Field size	Comment	
AKTIV1	R	1	Limit value active (per limit value available)	
GRENZWERT1	F	20	Technical value or ID number of a linked variable for a dynamic limit (see VARIABLEx) (if VARIABLEx is 1 and here it is -1 , the existing variable linkage is not overwritten)	
SCHWWERT1	F	16	Threshold value for limit	
HYSTERESE1	F	14	Hysteresis in %	
BLINKEN1	R	1	Set blink attribute	
BTB1	R	1	Logging in CEL	
ALARM1	R	1	Alarm	
DRUCKEN1	R	1	Printer output (for CEL or Alarm)	
QUITTIER1	R	1	Must be acknowledged	
LOESCHE1	R	1	Must be deleted	
VARIABLE1	R	1	Dyn. limit value linking the limit is defined by an absolute value (see field GRENZWERTx).	
FUNC1	R	1	Function linking	
ASK_FUNC1	R	1	With interrogation before execution	
FUNC_NR1	N	10	ID number of the linked function (if "-1" is entered here, the existing function is not overwritten during import)	
A_GRUPPE1	N	10	Alarm/event group	
A_KLASSE1	N	10	Alarm/event class	
MIN_MAX1	С	3	Minimum, Maximum	
FARBE1	N	10	Color as Windows coding	
GRENZTXT1	С	66	Limit value text	
A_DELAY1	N	10	Time delay	
INVISIBLE1	R	1	Invisible	

Expressions in the column "Comment" refer to the expressions used in the dialog boxes for the definition of variables. For more information, see chapter Variable definition.

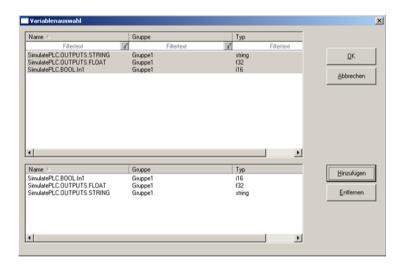


7.4.3 Online import

CREATING VARIABLES VIA THE ONLINE IMPORT FUNCTION



Via the context menu of the driver (right-click on the driver), you can open the online import dialog.



The list on top shows all variables that were assigned to the driver before and variables that come directly from the OPC server. Below that, you can see the variables currently being imported, which you can add to the variable list of the Editor by clicking OK.

7.5 Driver variables

The driver kit implements a number of driver variables. These are divided into:

- ▶ Information
- **▶** Configuration



- Statistics and
- **Error** messages

The definitions of the variables defined in the driver kit are available in the import file drvvar.dbf (on the CD in the directory: CD_Drive:/Predefined/Variables) and can be imported from there.

Hint: Variable names must be unique in zenon. If driver variables are to be imported from drvvar.dbf again, the variables that were imported beforehand must be renamed.



Info

Not every driver supports all driver variants.

For example:

- Variables for modem information are only supported by modem-compatible drivers
- Driver variables for the polling cycle only for pure polling drivers
- Connection-related information such as ErrorMSG only for drivers that only edit one connection at a a time



INFORMATION

Name from import	Туре	Offset	Description
MainVersion	UINT	0	Main version number of the driver.
SubVersion	UINT	1	Sub version number of the driver.
BuildVersion	UINT	29	Build version number of the driver.
RTMajor	UINT	49	zenon main version number
RTMinor	UINT	50	zenon sub version number
RTSp	UINT	51	zenon service pack number
RTBuild	UINT	52	zenon build number
LineStateIdle	BOOL	24.0	TRUE, if the modem connection is idle
LineStateOffering	BOOL	24.1	TRUE, if a call is received
LineStateAccepted	BOOL	24.2	The call is accepted
LineStateDialtone	BOOL	24.3	Dialtone recognized
LineStateDialing	BOOL	24.4	Dialing active
LineStateRingBack	BOOL	24.5	While establishing the connection
LineStateBusy	BOOL	24.6	Target station is busy
LineStateSpecialInfo	BOOL	24.7	Special status information received
LineStateConnected	BOOL	24.8	Connection established
LineStateProceeding	BOOL	24.9	Dialing completed
LineStateOnHold	BOOL	24.10	Connection in hold
LineStateConferenced	BOOL	24.11	Connection in conference mode.
LineStateOnHoldPendConf	BOOL	24.12	Connection in hold for conference
LineStateOnHoldPendTransfer	BOOL	24.13	Connection in hold for transfer
LineStateDisconnected	BOOL	24.14	Connection stopped
LineStateUnknow	BOOL	24.15	Connection status unknown
ModemStatus	UDINT	24	Current modem status
TreiberStop	BOOL	28	Driver stopped



			For driver stop, the variable has the value TRUE and an OFF bit. After the driver has started, the variable has the value FALSE and no OFF bit.
SimulRTState	UDINT	60	Informs the status of Runtime for driver simulation.

CONFIGURATION

Name from import	Туре	Offset	Description
ReconnectInRead	BOOL	27	If TRUE, the modem is automatically reconnected for reading
ApplyCom	BOOL	36	Apply changes in the settings of the serial interface. Writing to this variable immediately results in the method SrvDrvVarApplyCom being called (which currently has no further function).
ApplyModem	BOOL	37	Apply changes in the settings of the modem. Writing this variable immediately calls the method SrvDrvVarApplyModem. This closes the current connection and opens a new one according to the settings PhoneNumberSet and ModemHwAdrSet.
PhoneNumberSet	STRING	38	Telephone number, that should be used
ModemHwAdrSet	DINT	39	Hardware address for the telephone number
GlobalUpdate	UDINT	3	Update time in milliseconds (ms).
BGlobalUpdaten	BOOL	4	TRUE, if update time is global
TreiberSimul	BOOL	5	TRUE, if driver in sin simulation mode
TreiberProzab	BOOL	6	TRUE, if the variables update list should be kept in the memory
ModemActive	BOOL	7	TRUE, if the modem is active for the driver



Device	STRING	8	Name of the serial interface or name of the modem
ComPort	UINT	9	Number of the serial interface.
Baud rate	UDINT	10	Baud rate of the serial interface.
Parity	SINT	11	Parity of the serial interface
ByteSize	SINT	14	Number of bits per character of the serial interface
			Value = 0 if the driver cannot establish any serial connection.
StopBit	SINT	13	Number of stop bits of the serial interface.
Autoconnect	BOOL	16	TRUE, if the modem connection should be established automatically for reading/writing
PhoneNumber	STRING	17	Current telephone number
ModemHwAdr	DINT	21	Hardware address of current telephone number
RxIdleTime	UINT	18	Modem is disconnected, if no data transfer occurs for this time in seconds (s)
WriteTimeout	UDINT	19	Maximum write duration for a modem connection in milliseconds (ms).
RingCountSet	UDINT	20	Number of ringing tones before a call is accepted
ReCallIdleTime	UINT	53	Waiting time between calls in seconds (s).
ConnectTimeout	UDINT	54	Time in seconds (s) to establish a connection.



STATISTICS

Name from import	Туре	Offset	Description
MaxWriteTime	UDINT	31	The longest time in milliseconds (ms) that is required for writing.
MinWriteTime	UDINT	32	The shortest time in milliseconds (ms) that is required for writing.
MaxBlkReadTime	UDINT	40	Longest time in milliseconds (ms) that is required to read a data block.
MinBlkReadTime	UDINT	41	Shortest time in milliseconds (ms) that is required to read a data block.
WriteErrorCount	UDINT	33	Number of writing errors
ReadSucceedCount	UDINT	35	Number of successful reading attempts
MaxCycleTime	UDINT	22	Longest time in milliseconds (ms) required to read all requested data.
MinCycleTime	UDINT	23	Shortest time in milliseconds (ms) required to read all requested data.
WriteCount	UDINT	26	Number of writing attempts
ReadErrorCount	UDINT	34	Number of reading errors
MaxUpdateTimeNormal	UDINT	56	Time since the last update of the priority group Normal in milliseconds (ms).
MaxUpdateTimeHigher	UDINT	57	Time since the last update of the priority group Higher in milliseconds (ms).
MaxUpdateTimeHigh	UDINT	58	Time since the last update of the priority group нідь in milliseconds (ms).
MaxUpdateTimeHighest	UDINT	59	Time since the last update of the priority group Highest in milliseconds (ms).



PokeFinish BOOL	55	Goes to 1 for a query, if all current pokes were executed
-----------------	----	---

ERROR MESSAGES

Name from import	Туре	Offset	Description
ErrorTimeDW	UDINT	2	Time (in seconds since 1.1.1970), when the last error occurred.
ErrorTimeS	STRING	2	Time (in seconds since 1.1.1970), when the last error occurred.
RdErrPrimObj	UDINT	42	Number of the PrimObject, when the last reading error occurred.
RdErrStationsName	STRING	43	Name of the station, when the last reading error occurred.
RdErrBlockCount	UINT	44	Number of blocks to read when the last reading error occurred.
RdErrHwAdresse	UDINT	45	Hardware address when the last reading error occurred.
RdErrDatablockNo	UDINT	46	Block number when the last reading error occurred.
RdErrMarkerNo	UDINT	47	Marker number when the last reading error occurred.
RdErrSize	UDINT	48	Block size when the last reading error occurred.
DrvError	SINT	25	Error message as number
DrvErrorMsg	STRING	30	Error message as text
ErrorFile	STRING	15	Name of error log file

8. Driver-specific functions

This driver supports the following functions:

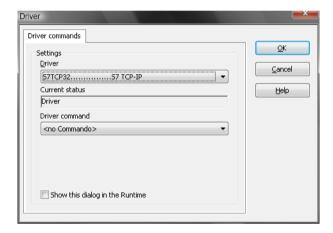


9. Driver commands

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.

Driver commands are used to influence drivers using zenon; start and stop for example. The engineering is implemented with the help of function <code>Driver</code> commands. To do this:

- create a new function
- ▶ select Variables -> Driver commands
- ▶ The dialog for configuration is opened



Parameters	Description
Drivers	Drop-down list with all drivers which are loaded in the project.
Current state	Fixed entry which has no function in the current version.
Driver commands	Drop-down list for the selection of the command.
<pre>Start driver (online mode)</pre>	Driver is reinitialized and started.
<pre>> Stop driver (offline mode)</pre>	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	Driver is set into simulation mode. The values of all variables of the driver are simulated by the



	mode	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	Enter driver-specific commands. Opens input field in order to enter a command.
•	Activate driver write set value	Write set value to a driver is allowed.
•	Deactivate driver write set value	Write set value to a driver is prohibited.
•	Establish connection 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)
	ow this dialog in the	The dialog is shown in Runtime so that changes can be made.

DRIVER COMMANDS IN THE NETWORK

If the computer, on which the driver command function is executed, is part of the zenon network, additional actions are carried out. A special network command is sent from the computer to the project server, which 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.

10. Error analysis

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



10.1 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 (main.chm::/12464.htm) program that was also installed with zenon. You can find it under Start/All programs/zenon/Tools 7.00 -> Diagviewer.

zenon driver log all errors in the log files. The default folder for the log files is subfolder <u>log</u> in directory ProgramData, example: C:\ProgramData\zenon \zenon700\LOG for zenon version 7.00 SPO. Log files are text files with a special structure.

Attention: With the default settings, a driver only logs error information. With the <code>piagnosis viewer</code> 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 currently created entries live
- customize the logging settings
- change the folder in which the log files are saved

Hints:

- 1. In Windows CE even errors are not logged per default due to performance reasons.
- 2. The Diagnosis Viewer displays all entries in UTC (coordinated world time) and not in local time.
- 3. 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.
- 4. If you only use Error logging, the problem description is in column Error text. For other diagnosis level the description is in column General text.
- 5. For communication problems many drivers also log error numbers which the PLC assigns to them. They are displayed in Error text and/or Error code and/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.
- 6. 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.



Info

You can find further information on the Diagnosis Viewer in the Diagnose Viewer (main.chm::/12464.htm) chapter.

Logging 10.2

The driver allows activating a configurable logging of the communication. The logged data is written to the file <name>_LOG.TXT in the project directory and can be read with any text editor.

Logging is activated by creating a section in the project.ini that has the driver name as a name and contains the following entries:

LOG_FLAGS0

Allows to configure the depth of the logging.



Bit/Numerical	Meaning
1/1	Log errors. Always active
2/2	Warnings.
3 /4	Add item will be logged
4/8	Add group will be logged.
5/16	Server starts will be logged.
6/32	Value changes will be logged.
7/64	Item activation will be logged.
8/128	Item deactivation will be logged.
9/256	Event DataChange will be logged.
10/512	Write will be logged.

The sum of the numerical values is the value for 'LOG_FLAGS0='; e.g.:

LOG_FLAGS0=65535

Activates logging for all.

LOGON

Allows to globally switch logging on(1) or off(0); e.g.:

LOGON=1

COMPLETE ENTRY IN THE PROJECT.INI:

[OPC Client V2.0]

LOG_FLAGS0=65535

LOGON=1





Info

For the entry between the brackets [] you have to use the driver name. You can find it in the driver properties. Select the driver in the Editor, under Properties – General – Description you will find the driver name.

Check list 10.3

- ▶ Is the variable definition correct ?
- ► Is the OPC server running?
- Have you analyzed the diagnosis viewer entries (which errors have occured)?