

zenon driver manual Idec32

v.8.10



© 2019 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. Technical data is only used for product description and are not guaranteed qualities in the legal sense. Subject to change, technical or otherwise.



Contents

1	Welcome to COPA-DATA help	5
2	Idec32	5
3	Idec32 - data sheet	6
4	Driver history	7
5	Requirements	8
	5.1 PC	8
	5.2 Control	8
6	Configuration	9
	6.1 Creating a driver	9
	6.2 Settings in the driver dialog	12
	6.2.1 General	13
	6.2.2 Driver dialog Com	17
	6.2.3 Driver dialog Idec settings	18
7	Creating variables	18
	7.1 Creating variables in the Editor	18
	7.2 Addressing	22
	7.3 Driver objects and datatypes	23
	7.3.1 Driver objects	
	7.3.2 Mapping of the data types	27
	7.4 Creating variables by importing	27
	7.4.1 XML import	28
	7.4.2 DBF Import/Export	29
	7.5 Communication details (Driver variables)	34
8	Driver-specific functions	40
9	Driver command function	41
10) Error analysis	46
	10.1 Analysis tool	46
	10.2 Error numbers	47
	10.3 Check list	48





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 Support Team, who 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 Idec32

Variable addressing via address

You can find the general driver configuration in the chapter "Process data model and variable definition".



3 Idec32 - data sheet

General:	
Driver file name	ldec32.exe
Driver name	ldec
PLC types	Idec MicroSmart, Idec OpenNet Controller, Idec FC4A -HPC1
PLC manufacturer	ldec

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



Requirements:	
Hardware PC	RS 232 interface, FC2A -KC4C cable
Software PC	
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	700	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

Protocol: Idec communication protocol

SOFTWARE

Copy the driver file Idec32.EXE into the current zenon directory (unless it is already there) and enter it into the TREIBER_EN.XML file with the tool DriverInfo.exe.

CONNECTION

Establish a serial connection with a FC2A-KC4C cable (Computerlink cabel to loader port RS2).

5.2 Control

- Idec MicroSmart
- ▶ Idec OpenNet Controller
- ▶ Idec FC4A-HPC1 (RS232 interface)



Protocol: Idec communication protocol

CONNECTION

Establish a serial connection with a FC2A-KC4C cable (Computerlink cabel to loader port RS2).

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 (main.chm::/15247.htm) of the online manual.

6.1 Creating a driver

In the Create driver dialog, you create a list of the new drivers that you want to create.

Create driver			
Available drivers			
Available drivers	Cleate driver Driver information Describion: Describion: Describion: Describion: Describion: Describion: Supported file by zenon. Supported file by zenon. Supported commention types: Local Supported commention types: Local Supported commention Supported commentin Supported commention Supporte		
B illes B illes B illes			
GE Automation&Controls			
Driver name			
File driver	· · · · · · · · · · · · · · · · · · ·		
	OK Cancel Help		
	on Carice hep		



Parameter	Description
Available drivers	List of all available drivers.
	The display is in a tree structure: [+] expands the folder structure and shows the drivers contained therein. [-] reduces the folder structure
	Default: no selection
Driver name	Unique Identification of the driver.
	Default: <i>Empty</i> The input field is pre-filled with the pre-defined Identification after selecting a driver from the list of available drivers.
Driver information	Further information on the selected driver. Default: <i>Empty</i> The information on the selected driver is shown in this area after selecting a driver.

CLOSE DIALOG

Option	Description
ОК	Accepts all settings and opens the driver configuration dialog of the selected driver.
Cancel	Discards all changes and closes the dialog.
Help	Opens online help.

Information

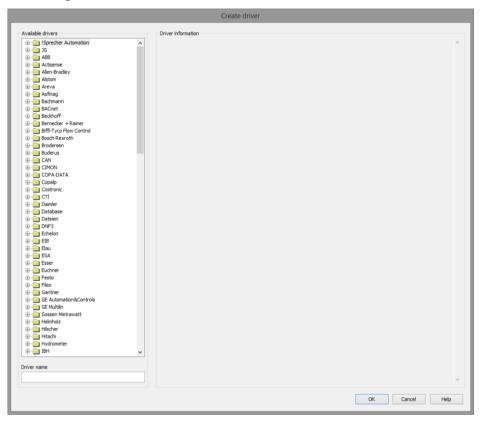
The content of this dialog is saved in the file called Treiber_[Language].xml. You can find this file in the following folder: C:\ProgramData\COPA-DATA\zenon[version number].

CREATE NEW DRIVER

In order to create a new driver:



- Right-click on Driver in the Project Manager and select New driver in the context menu. Optional: Select the New driver button from the toolbar of the detail view of the Variables. The Create driver dialog is opened.
- 2. The dialog offers a list of all available drivers.



- Select the desired driver and name it in the Driver name input field. This input field corresponds to the Identification property. The name of the selected driver is automatically inserted into this input field by default. The following is applicable for the Driver name:
 - The Driver name must be unique.
 If a driver is used more than once in a project, a new name has to be given each time.
 This is evaluated by clicking on the OK button. If the driver is already present in the project, this is shown with a warning dialog.
 - 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 by clicking on the OK button. The configuration dialog for the selected driver is opened.



Note: The language of driver names cannot be switched. They are always shown in the language in which they have been created, regardless of the language of the Editor. This also applies to driver object types.

DRIVER NAME DIALOG ALREADY EXISTS

If there is already a driver in the project, this is shown in a dialog. The warning dialog is closed by clicking on the **OK** button. The driver can be named correctly.

zenon Energy Edition	×
This driver name already exists! Please enter an unique driver name!	
ОК	

ZENON PROJECT

The following drivers are created automatically for newly-created projects:

- Intern
- MathDr32
- SysDrv

Information

Only the required drivers need to be present in a zenon project. Drivers can be added at a later time if required.

6.2 Settings in the driver dialog

You can change the following settings of the driver:



6.2.1 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.

Configuration			Х
General			_
		ОК	
Mode:		Cancel	
Hardware	•	Help	
🔲 Keep update list i	memory		
Outputs writeable			
Variable image re	nanent		
Stopped on Stand	by-Server		
Update time globa			
Global updatetime in 1000	ns:		
Priority			
	1000 ms		
high:	500 ms		
higher:	300 ms		
highest:	100 ms		

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



Option	Description
	within a value range automatically.
	 Simulation - programmed: No 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 simulation (main.chm::/25206.htm).
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 <i>M1</i> (0) to <i>M8</i> (7)
	► REVISION(9)
	► AUS(20)
	► ERSATZWERT(27)
	The variable image is always saved if:
	• the variable is of the object type Driver variable
	• the driver runs in simulation mode. (not



Option	Description
	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 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 (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.
	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.



Option	Description
	Exceptions: Spontaneous drivers ignore this option. They generally use the shortest possible update time. For details, see the Spontaneous driver update time section.
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
ОК	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.



6.2.2 Driver dialog Com

Here all general driver specific settings are done.

Configuration		
General Com Idec32-5	Setting	OK Cancel
Com: COM 1 Databit: 7 Stopbit: 1	Baudrate: 9600 ▼ Parity: Even ▼ Protocol: RTS/CTS ▼	

Parameters	Description
Com	Select serial interface
	COM1 - COM64
Data bit	Data word length
	5; 6; 7; 8
Stop bit	1; 1,5; 2
Baud rate	Data transfer rate
	110; 300; 600; 1200; 2400; 4800; 9600; 19200; 38400; 56000; 57600; 115200; 128000; 256000
Parity	No; Even; Odd;
Protocol	None; Xon/Xoff; RTS/CTS; DTR/DSR



6.2.3 Driver dialog Idec settings

Parameters	Description
Repetitions	If the transmission was not successful, it is retried this number of times.
Response time (ms)	Within this time, there must a response from the PLC.
MicroSmart	Activate this if the target PLC is of the MicroSmart series.

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:



Workspace 'C:\WSP6_PRJ VARIABLE (Start pr	🔶 🤹 🖶	🛅 🐘 🗙 🔠 🕶 🐓 🚸 🚳	mi in in	🏹 🗕 🛣	🖌 ्र 🐴	1	
😑 🔷 Variables	Status Name 🛆	n Unit	Net address	Data block	Offset B	Bit num	Alignm.
Driver		Filter text 🛛 🗹 Filter 🗹	Filter text 🗹 F	ilter text 📝	Filter text 📝 F	iter 🔽	Filter
A Datatypes Reaction m	drv_a*it		0	0	0	0	(
Allocations	drv_C	Variable new	Ins	0	0	0	(
	drv_E	Create standard function		0	0	0	(
👽 Units	drv_N			0	0	0	(
🕀 🛒 Screens	drv_N	Linked elements	+	0	0	0	(
Functions	drv_N	Сору	Ctrl+C	0	0	0	(
	drv_N	Paste	Ctrl+V	0	0	0	(
Historian	drv_N			0	0	0	(
Time control	drv_S	Delete	Del	0	0	0	(
- Programming interfa	+- Interr	Expand/collapse node		0	0	0	(
- S straton (IEC 6113	+ Interr		,	0	0	0	(
Production & Facilit	+ Interr	Activate all		0	0	0	(
Interlockings		Activate					
Command		Deactivate					
- 👜 Energy Managemer							
Message Control		Export XML selected					
- 2 Report Generator		Import XML					
S. User administrati		Extended import/export	•				
SAP interface							
😟 💼 Files		Extended filter	•				
		Remove all filter					
Projektsicherun	•						
1 1 7 PRI T	42 total / 11	Edit selected cell	F2	able			

1. Select the **New variable** command in the **Variables** node in the context menu

The dialog for configuring variables is opened

- 2. Configure the variable
- 3. The settings that are possible depends on the type of variables



CREATE VARIABLE DIALOG

ate variable	
Settings	
Name: S_Variable	
S_Variable	
Driver	
SIMUL32 - SIMUL_WIZ	
PLC marker	V
Datatype	
REAL	
Array settings	
Start index is 0 Start index is 1	
Dim 1 Dim 2	Dim 3
Addressing options	
Automatic addressing	
Addressing according to data type offset a	and start offset
Manual addressing	
Each datatype starts with new offset	
- courrouterype starts mutitien unset	
Automatic addressing	
Activate all elements	
 Activate element manually 	
roperty	Description
lame	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.
	Maximum length: 128 characters
	Attention: the characters # and @ are not permitted in variable names. If non-permitted characters are used, creation of variables cannot be completed and the Finish button remains inactive. Note: For some drivers, the addressing is possible over the property Symbolic address , as well.
Privers	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.
river Object Type	Select the appropriate driver object type from the drop-down list.
cti.chm::/28685.htm)	
cti.chm::/28685.htm) Pata Type	Select the desired data type. Click on the button to open the



Property	Description
	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 addressing	Expanded settings for arrays and structure variables. You can find details in the respective section.

SYMBOLIC ADDRESS

The **Symbolic address** property can be used for addressing as an alternative to the **Name** or **Identification** of the variables. Selection is made in the driver dialog; configuration is carried out in the variable property. When importing variables of supported drivers, the property is entered automatically.

Maximum length: 1024 characters.

The following drivers support the Symbolic address:

- ► 3S_V3
- AzureDrv
- BACnetNG
- ► IEC850
- KabaDPServer
- OPCUA32
- Phoenix32
- POZYTON
- RemoteRT
- S7TIA
- SEL
- SnmpNg32
- PA_Drv

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 Addressi	
Property	Description
Name	Freely definable name.
	Attention: For every zenon project the name must be unambiguous.
Identification	Freely definable identification. E.g. for Resources label, comments,
Net address	Bus address or net address of the variable.
	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 <i>Extended data block</i> , enter the datablock number here.
	Adjustable from <i>0</i> to <i>4294967295</i> .
	You can take the exact maximum area for data blocks from the manual of the PLC.
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	Number of the bit within the configured offset.
	Possible entries: <i>0</i> to <i>65535</i> . Working range [07]
String length	Only available for String variables. Maximum number of characters that the variable can take.
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/Data	Data type of the variable. Is selected during the creation of the variable; the type can be changed here.
Туре	Attention: If you change the data type later, all other properties of the

7.2 Addressing



Property	Description
	variable must be checked and adjusted, if necessary.
Driver connection/Priorit y	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.

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

Driver Object Type	Channel type	Read	Write	Supported data types	Comment
Output	11	Х	Х	BOOL	
Data Register	65	Х	Х	BOOL, UDINT, DINT, USINT, UINT, INT, SINT	
Input	10	Х		BOOL	
Link Register	67	Х	Х	BOOL, UDINT, DINT, USINT, UINT, INT, SINT	
Link Relay	64	Х	Х	BOOL, UDINT, DINT, USINT, UINT, INT, SINT	
Shift Register	68	Х	Х	BOOL	

The following object types are available in this driver:



Driver Object Type	Channel type	Read	Write	Supported data types	Comment
PLC marker	8	Х	Х	BOOL	
Counter	23	Х	Х	UDINT, UINT	
Counter current	70	Х	Х	UDINT, UINT	
Timer	22	X	X	BOOL, SINT, USINT, UINT, DINT, INT, UDINT	
Times current	69	Х	Х	UDINT, UINT	
Communication details	35	X	X	BOOL, SINT, USINT, INT, UINT, DINT, UDINT, REAL, STRING	Variables for the static analysis of the communication; is transferred between driver and Runtime (not to the PLC).
					Note : The addressing and the behavior is the same for most zenon drivers.
					You can find detailed information on this in the Communication details (Driver variables) (on page 34) chapter.

OBJECTS FOR PROCESS VARIABLES IN ZENON

KANALNAME	DATENART	KANALTYP	OBJEKT	LESEN	SCHREIBEN
X Bool	8	10	8	Х	
Y Bool	8	11	14	Х	Х
M Bool	10	8	3	Х	Х
Time target WORD	2	22	22	Х	Х



KANALNAME	DATENART	KANALTYP	OBJEKT	LESEN	SCHREIBEN
Time target INT	1	22	22	Х	Х
Time target DWORD	4	22	23	Х	Х
Time target DINT	3	22	23	Х	Х
Counter target WORD	2	23	28	Х	Х
Counter target INT	1	23	28	Х	Х
Counter target DWORD	4	23	29	Х	Х
Counter target DINT	3	23	29	Х	Х
Link relay BOOL	8	64	32	Х	Х
Link relay BYTE	9	64	33	Х	Х
Link relay CHAR	10	64	33	Х	Х
Link relay WORD	2	64	34	Х	Х
Link relay INT	1	64	34	Х	Х
Link relay DWORD	4	64	35	Х	X
Link relay DINT	3	64	35	Х	Х
Shift Register BOOL	8	68	38	Х	X
Data Register BOOL	8	65	44	Х	Х
Data Register BYTE	9	65	45	Х	Х
Data Register CHAR	10	65	45	Х	Х
Data Register WORD	2	65	46	Х	Х



KANALNAME	DATENART	KANALTYP	OBJEKT	LESEN	SCHREIBEN
Data Register INT	1	65	46	Х	Х
Data Register DWORD	4	65	47	Х	Х
Data Register DINT	3	65	47	Х	Х
Link Register BOOL	8	67	56	Х	Х
Link Register BYTE	8	67	57	Х	Х
Link Register CHAR	10	67	57	Х	Х
Link Register WORD	2	67	58	Х	Х
Link Register INT	1	67	58	Х	Х
Link Register DWORD	4	67	59	Х	Х
Link Register DINT	3	67	59	Х	Х
Time current WORD	2	69	64	Х	Х
Time current INT	1	69	64	Х	Х
Time current DWORD	4	69	65	Х	Х
Time current DINT	3	69	65	Х	Х
Counter current WORD	2	70	70	Х	Х
Counter current INT	1	70	70	Х	Х
Counter current DWORD	4	70	71	Х	Х



KANALNAME	DATENART	KANALTYP	OBJEKT	LESEN	SCHREIBEN
Counter current DINT	3	70	71	Х	Х

Key:

X: supported

--: not supported

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.

EXAMPLES FOR ALL POSSIBLE ZENON DATA TYPES

PLC	zenon
18	i/u8Bit (signed)
116	i/u16Bit (signed)
132	i/u32Bit (signed)
U8	i/u8Bit
U16	i/u16Bit
U32	i/u32Bit
Boolean	Boolean

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.

7.4 Creating variables by importing

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



Information

You can find details on the import and export of variables in the Import-Export (main.chm::/13028.htm) manual in the Variables (main.chm::/13045.htm) section.

7.4.1 XML import

During XML import of variables or data types, these are first assigned to a driver and then analyzed. Before import, the user decides whether and how the respective element (variable or data type) is to be imported:

- Import: The element is imported as a new element.
- Overwrite: The element is imported and overwrites a pre-existing element.
- Do not import: The element is not imported.

Note: The actions and their durations are shown in a progress bar during import. The import of variables is described in the following documentation. Data types are imported along the same lines.

REQUIREMENTS

The following conditions are applicable during import:

Backward compatibility

At the XML import/export there is no backward compatibility. Data from older zenon versions can be taken over. The handover of data from newer to older versions is not supported.

Consistency

The XML file to be imported has to be consistent. There is no plausibility check on importing the file. If there are errors in the import file, this can lead to undesirable effects in the project.

Particular attention must be paid to this, primarily if not all properties exist in the XML file and these are then filled with default values. E.g.: A binary variable has a limit value of *300*.

Structure data types

Structure data types must have the same number of structure elements. Example: A structure data type in the project has 3 structure elements. A data type with the same name in the XML file has 4 structure elements. Then none of the variables based on this data type in the file are imported into the project.



🗉 🖌 Hint

You can find further information on XML import in the **Import - Export** manual, in the **XML import (main.chm::/13046.htm)** chapter.

7.4.2 DBF Import/Export

Data can be exported to and imported from dBase.

Information

Import and Export via CSV or dBase supported; no driver specific variable settings, such as formulas. Use export/import via XML for this.

IMPORT DBF FILE

To start the import:

- 1. right-click on the variable list
- 2. in the drop-down list 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.

Information

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 list 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.

Information

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) at export.

DBF files must:

- conform with their name to the 8.3 DOS format (8 alphanumeric characters for name, 3 characters for extension, no space)
- Be stored close to the root directory (Root)

STRUCTURE

Identification	Тур е	Field size	Comment
KANALNAME	Cha r	128	Variable name. The length can be limited using the MAX_LAENGE entry in the project.ini file.
KANAL_R	С	128	The original name of a variable that is to be replaced by the new name entered under "VARIABLENNAME" (variable name) (field/column must be entered



Identification	Typ e	Field size	Comment
			manually).
			The length can be limited using the MAX_LAENGE entry in the project.ini file.
KANAL_D	Log	1	The variable is deleted with the 1 entry (field/column has to be created by hand).
TAGNR	С	128	Identification.
			The length can be limited using the MAX_LAENGE entry in the project.ini file.
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	Nu m	3	Net address
BAUSTEIN	Ν	3	Datablock address (only for variables from the data area of the PLC)
ADRESSE	Ν	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 Recipegroup Manager
LES_SCHR	L	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)
OBJEKT	Ν	2	Driver-specific ID number of the primitive object comprises TREIBER-OBJEKTTYP and DATENTYP



Identification	Тур e	Field size	Comment
SIGMIN	Floa t	16	Non-linearized signal - minimum (signal resolution)
SIGMAX	F	16	Non-linearized signal - maximum (signal resolution)
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)
UPDATERATE	F	19	Update rate for mathematics variables (in sec, one decimal possible) not used for all other variables
MEMTIEFE	Ν	7	Only for compatibility reasons
HDRATE	F	19	HD update rate for historical values (in sec, one decimal possible)
HDTIEFE	Ν	7	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)
HYST_PLUS	F	16	Positive hysteresis, from measuring range
HYST_MINUS	F	16	Negative hysteresis, from measuring range
PRIOR	Ν	16	Priority of the variable
REAMATRIZE	С	32	Allocated reaction matrix
ERSATZWERT	F	16	Substitute value, from measuring range
SOLLMIN	F	16	Minimum for set value actions, from measuring range
SOLLMAX	F	16	Maximum for set value actions, from measuring range
VOMSTANDBY	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
RESOURCE	С	128	Resources label. Free string for export and display in lists.



Identification	Тур е	Field size	Comment
			The length can be limited using the MAX_LAENGE entry in project.ini .
ADJWVBA	R	1	Non-linear value adaption: <i>0</i> : Non-linear value adaption is used <i>1</i> : Non-linear value adaption is not used
ADJZENON	С	128	Linked VBA macro for reading the variable value for non-linear value adjustment.
ADJWVBA	С	128	ed VBA macro for writing the variable value for non-linear value adjustment.
ZWREMA	Ν	16	Linked counter REMA.
MAXGRAD	Ν	16	Gradient overflow for counter REMA.

Attention

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

LIMIT VALUE DEFINITION

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

Identification	Туре	Field size	Comment
ΑΚΤΙV1	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 value (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 value
HYSTERESE1	F	14	Is not used
BLINKEN1	R	1	Set blink attribute
BTB1	R	1	Logging in CEL
ALARM1	R	1	Alarm



Identification	Туре	Field size	Comment	
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	Functions linking	
ASK_FUNC1	R	1	Execution via Alarm Message List	
FUNC_NR1	Ν	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.

7.5 Communication details (Driver variables)

The driver kit implements a number of driver variables. This variables are part of the driver object type *Communication details*. These are divided into:

- Information
- Configuration
- Statistics and
- Error message



The definitions of the variables implemented in the driver kit are available in the import file **DRVVAR.DBF** and can be imported from there.

Path to file: %ProgramData%\COPA-DATA\zenon<Versionsnummer>\PredefinedVariables

Note: Variable names must be unique in zenon. If driver variables of the driver object type *Communication details* are to be imported from **DRVVAR.DBF** again, the variables that were imported beforehand must be renamed.

Information

Not every driver supports all driver variables of the driver object type *Communication details.*

For example:

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

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

INFORMATION



Name from import	Туре	Offset	Description
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
LineStateOnHoldPendTransfe r	BOOL	24.13	Connection in hold for transfer
LineStateDisconnected	BOOL	24.14	Connection terminated.
LineStateUnknow	BOOL	24.15	Connection status unknown
ModemStatus	UDINT	24	Current modem status
TreiberStop	BOOL	28	Driver stopped
			For <i>driver stop</i> , the variable has the value <i>TRUE</i> and an OFF bit. After the driver has started, the variable has the value <i>FALSE</i> and no OFF bit.
SimulRTState	UDINT	60	Informs the status of Runtime for driver simulation.
ConnectionStates	STRING	61	Internal connection status of the driver to the PLC.
			Connection statuses:
			<i>0:</i> Connection OK
			1: Connection failure
			2: Connection simulated
			Formating:
			<netzadresse>:<verbindungszustand>;;;</verbindungszustand></netzadresse>



Name from import	Туре	Offset	Description
			A connection is only known after a variable has first signed in. In order for a connection to be contained in a string, a variable of this connection must be signed in once.
			The status of a connection is only updated if a variable of the connection is signed in. Otherwise there is no communication with the corresponding controller.

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



Name from import	Туре	Offset	Description
Device	STRING	8	Name of the serial interface or name of the modem
ComPort	UINT	9	Number of the serial interface.
Baudrate	UDINT	10	Baud rate of the serial interface.
Parity	SINT	11	Parity of the serial interface
ByteSize	USINT	14	Number of bits per character of the serial interface
			Value = 0 if the driver cannot establish any serial connection.
StopBit	USINT	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	UINT	54	Time in seconds (s) to establish a connection.

STATISTICS

Name from import	Туре	Offse t	Description
MaxWriteTime	UDINT	31	The longest time in milliseconds (ms) that is required for writing.



Name from import	Туре	Offse t	Description
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 High 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 MESSAGE

Name from import	Туре	Offse t	Description
ErrorTimeDW	UDINT	2	Time (in seconds since 1.1.1970), when the last error occurred.



Name from import	Туре	Offse t	Description
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	DINT	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	USINT	25	Error message as number
DrvErrorMsg	STRING	30	Error message as text
ErrorFile	STRING	15	Name of error log file

8 Driver-specific functions

The driver supports the following functions:

CREATING A VARIABLE

The neta address matches the device number. 255 is for all PLCs.

Independent of the addressed range, the PLC element number matches the offset of the variable.

e.g. in the program WINDLDR, an output is named "Q0037", which means an offset of 37.

This offset consists of module 3 and output 7.

If there are more than 8 outputs - inputs on one module, the next highest module number will be used.



Example

Output 10 at module 3 in this case has an offset of 40.

The following PLC areas are supported:

Parameters	Description
Input	Bit
Output	Bit
Internal relay	Bit
Link relay	Bit, not available for MicroSmart
Shift relay	Bit
Link relay	Bit
Timer preset und Istwert	Word
Counter preset und Istwert Word	
Data register	Word
Link register	Bit, not available for MicroSmart

9 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

Select a function	×
Functions selection	
Favorites	
Screen switch	
S Write/modify set value	
ia- instantia and CEL	
E - B Batch Control	
E. Message Control	
🗄 🖓 Network	
• Recipes	
E Report Generator/Report Viewer/Analyzer	
🗄 📲 🖅 Screens	
🗄 💼 Script	
🗄 🔤 Shift Management	
ianse Administration ianse Variable	
terret vanable terret vanable	
₩··· 🐼 VSTA	
How Windows	
	_
Quick help	
	-
· · · · · · · · · · · · · · · · · · ·	4
1	
OK Cancel Help	
Current Prep	

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



The dialog for configuration is opened

Drivers	
ver commands	
Settings	ОК
Drivers	Cancel
InternDriver for internal variables $\qquad \qquad \lor$	Concer
Current status	Help
Drivers	
Driver command	
<no command=""> V</no>	
Show this dialog in the Runtime	

- 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

rivers Driver commands	>
Settings Drivers INTERNDriver for internal variables Current status Drivers Driver command Driver-specific command Currentsettic command	OK Cancel Help
Show this dialog in the Runtime	

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. Has no function in the current version.
Driver command	Selection of the desired driver command from a drop-down list.
	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.



Option	Description
	Note: Only available if, for the driver command option, the <i>driver-specific command</i> 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.
	• <i>Inactive</i> : The Editor configuration is applied in the Runtime when executing the function.
	Default: inactive

CLOSE DIALOG

Options	Description
ОК	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>	No command is sent. A command that already exists can thus be removed from a configured function.
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 <i>switched</i> <i>off</i> (<i>OFF</i> ; Bit <i>20</i>).



Driver command	Description
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.
Activate driver write set value	Write set value to a driver is possible.
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)
Driver in counting simulation mode	Driver is set into counting simulation mode. All values are initialized with <i>0</i> and incremented in the set update time by <i>1</i> each time up to the maximum value and then start at <i>0</i> again.
Driver in static simulation mode	No communication to the controller is established. All values are initialized with <i>0</i> .
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.

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 8.10 -> 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 (main.chm::/12464.htm) manual.

10.2 Error numbers

Number	Description
X1000	Read. Command could not be sent to the PLC.
X1100	Time out while reading.
X1200	Read. Unknown command character in response.
X1300	Read response telegram is shorter than the minimum size.
X1400	Read. The checksum of the response is incorrect.
X1500	Read. Response telegram has an incorrect length.
X1600	Read. Error telegram has an incorrect length.
X1700	Read. PLC error message.
X1706	Read. PLC error message incorrect data range.
X1708	Read. PLC error message. Calendar or time data error.
X1800	Interface cannot be opened.
X1900	Write. Command could not be sent to the PLC.
X1A00	Write. Confirmation has an incorrect length.
X1B00	Write. PLC error message.
X1B06	Write. PLC error message incorrect data range.
X1B07	Write. PLC error message. Time/counter set value incorrect.
X1B08	Write. PLC error message. Calendar or time data error.
X1C00	Write. Error telegram has an incorrect length.
X1D00	Read. Command error has an incorrect length.



Number	Description
X1E00	Write. Unknown character in error telegram.
X1F00	Write. The checksum of the confirmation is incorrect.
X2000	Write. Telegram lenght of confirmation too short.
X2100	Write. Time for confirmation too long.
X22xx	Write. PLC-specific command error. Look up XX in the PLC documentation.
X2300	Write. Start sign of confirmation unknown.
X2400	Write. Command error message has an incorrect length.
X2500	Read. PLC-specific command error. Look up XX in the PLC documentation.
X2600	Read. Command error message has an incorrect length.

10.3 Check list

Is the connection to the PLC configured correctly (interface, character format)?

Is the PLC turned on?

Can the PLC use the defined ranges and offsets?

Are the offset and the net address in the variable correct?

Was the "Error - text file" analyzed? (which errors occurred?)

Send the zenon project to support@copadata.com