zenon
by COPA-DATA

zenon driver manual

DNP3_TG

v.8.10

Contents

1 Welcome to COPA-DATA help

ZENON VIDEO-TUTORIALS

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

GENERAL HELP

If you cannot find any information you require in this help chapter or can think of anything that you would like added, please send an email to documentation@copadata.com.

PROJECT SUPPORT

You can receive support for any real project you may have from our 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 DNP3_TG

Driver for the protocol in accordance with IEEE1815 Distributed Network Protocol (DNP3). The driver is Master at protocol level and supports serial communication with several Outstations, as well as IP communication via TCP.

DEFINITION OF TERMS

In order for you to understand this document better, please find the definition of important terms in the following list.

Term	Definition
Event Class	A type of grouping in the **Outstation**. There are: - **Event Classes** *1***,***2*, *3*: An **Event Class** is generally assigned to a buffer, in which value changes are stored for configured object groups with the variation configured in the **Outstation** . A master can request a read request, for **Class** *1* for example, for all value changes of this **Event Class** from the **Outstation**. In the **Outstation**, it is generally possible to configure which **Points** or which **Object Group** belong to which **Event Class**. There is no exact definition, however **Class** *1* is generally used for important messages, such as *Binary Inputs* for alarms. - **Static Class** *0*: The **Static Class** *0* returns, during a read request from the **Master**, the last respective value for virtually all **Points**.
Eventpoll	Read request from the **Master** to the **Outstation**, whereby the **Master** only requests that the **Outstation** sends the amended values for a certain **Event Class** or for all **Event Classes** (*1*, *2* and *3*).
Integrity Poll	Read requests from the **Master** to the **Outstation** for the **Static Class** *0*. The **Master** requests an initial display of all **Points** with this query.
Master	Controlling station. A **Master** sends read queries and control queries to the **Outstation**.
Objekt Group	Data type or data type in the **Outstation** with a defined functionality. Examples of assignments: - *Object Group 30* is as statistical analog entry. - A value change for an analog input is assigned to *Object Group 32*. - *Object Group 40* is for reading an analog output. - *Object Group 41* is for writing to an analog output.
Outstation	**SPS** or **RTU** in **DNP3**. An **Outstation** is a slave at protocol level and sends data to the **Master** on request.
Point	Equivalent of a variable in the Outstation. A **Point** is addressed with a **Point Number** (Offset) per **Objekt Group**, whereby the following is applicable: *Point 12* For **Object Group** *30* and *Point 12* for **Object Group** *32*, both have the same analog input as a basis, but *Point 12* for **Object Group** *1* is a completely independent binary input. The general term for a value, time or status

Term	Definition
	change is **DNP Object**.
Unsolicited Response	Message from value changes of an **Event Class** that is spontaneously sent from the **Outstation** to the **Master**. However, to do this, the **Master** must first activate, in the **Outstation**, **Unsolicited Responses**. The **Outstation** must support **Unsolicited Responses** and these must be configured for it.
Variation	Format in which the **Oustation** saves a static value or a value change in **Class** *0* or **Class** *1*, *2* or *3*. This can be configured in the **Outstation** for each **Object Group** or for each **Point**. The **Variation** defines (e.g.) ▸ a static value (**Class** *0*) or a value change, if this is an integer or a floating point. ▸ a time stamp is saved or not, or whether Object Flags are saved or not.

COMMUNICATION

The communication is mainly polling. The driver searches the Outstation for Events (value changes and status changes). The interval can be configed. These are assigned in the Outstation classes (*1*, *2* or *3*).

According to configuration in the Outstation more than one value for a single DNP object can be sent as an answer. (**Sequence Of Events Buffer** in contrast to **Latest Value-in** of the outstation).

The driver processes the received values and sends them to the Runtime.

If the Outstation supports this, you can activate **unsolicited responses** in the driver. In this case, the driver sends, at the end of the startup routine after the Integrity Poll has been concluded, the request to activate **unsolicited responses** for the respective class. The Outstation can send subsequently value changes without request of the Masters (Polling of eventclasses) to the Master. In this case, it is not unconditionally required, to poll Events regulary. The Polling cycle can then be adjusted higher or even to *0* (Polling for event classes deactivated). Polling for Events can also be instigated from Runtime manually (Class Poll variables).

Find out more information in the chapter **DNP3/IEEE1815-2012 standard**. You can acquire this documentation from the IEEE. You get also access if you join the **DNP3 users group** (http://www.dnp.org/ (http://www.dnp.org/)):

DEVICE PROFILE

A Device Profile is a standard document that describes, which functionality is supported with the DNP3 standard. For the DNP3_TG driver, you can find the XML Device Profile after installing zenon in the

following folder:
%ProgramData%\COPA-DATA\zenon8.10\CommunicationProfiles\Dnp3\Driver\DNP3_TG.xml.

INFORMATION ON THE DNP332, DNP3_NG AND DNP_NG DRIVER

The DNP332 driver and the DNP3_NG will be replaced by the DNP3_TG from version 7.20. For reasons of compatibility, the DNP3_NG and the older DNP332 drivers are still included in the setup, but are no longer displayed in the driver selection list by default. Existing projects that are converted use the driver that was originally configured as before. The old driver can continue to be used normally in converted projects. However a switch to the new DNP3_TG driver is also possible.

DISPLAYING THE DNP332/DNP3_NG IN THE DRIVER LIST

If you want to use the DNP332/DNP3_NG driver in a new project in 7.20, the driver must be added to the driver list again:

1. Start the program **Driverinfo.exe** from the zenon installation medium; subfolder *\AdditionalSoftware\COPA-DATA DriverXML Editor*.
2. Open the driver XML file using the program called **Driverinfo**.
   Example: **TREIBER_DE.XML** from the folder *%CD_PROGRAMDATA7200%*
   In doing so, **DE** is the code for the language in the Editor and **7200** for the installed version, version 7.20.
3. Go to the *DNP3* folder and select **New Driver** in the context menu.
4. Enter **DNP332** or **DNP3_NG** in all three fields of the dialog and confirm by clicking on **OK**.
5. Save the changes by clicking on the **Save** symbol in the toolbar.
   The driver can now be selected again in the Editor.
6. Repeat this step for each language that you use in the Editor.

SWITCH TO THE NEW DNP3_TG DRIVER

The DNP3_TG driver is compatible with the old DNP3_NG driver and DNP332 driver in principle. The **Replace driver** function in the Editor can also be used to switch from the DNP332 driver or from the DNP3_NG driver to the DNP3_NG driver. After the driver replacement, some settings need to be made again manually. Some functions in the DNP3_TG driver are implemented differently and require a change to the project configuration.

When planning to replace a driver, please note the following:

General:

- Back up your project first
- Note the driver configuration of the old driver. These must be entered again once the driver has been replaced.

- Writing to a Frozen Counter variable no longer leads to an Immediate Freeze No Ack. This function can be achieved using a Commando variable.

DNP332 to DNP3_TG:

- If you use Select Before Operate with the DNP332 driver for the Command Processing, deactivate the **Select Before Operate** property for the variable. Instead, select the *Auto-SBO* entry for *Binary Output* and *Analog Output* variables in the **Command Mode** property.
- If, you have used driver data types for **Analog Inputs** or **Counter** with DNP332 and these no longer exist in the DNP3_NG driver, then you must amend the data types of the variables before the change. Background: With the DNP332 driver, it was possible to select data types that were not envisaged by the DNP3 standard. These data types can no longer be used with DNP3_NG.

DNP3_NG to DNP3_TG:

- The **Command Mode** property for *Analog Output* and *Binary Output* variables must be set manually. Alternatively, it is possible to export the variables in XML format before the driver switch and to import them again after the switch. The DNP3_TG driver also supports, in addition to Direct Operate and Auto SBO , Direct Operate No Ack.
- The property variation for the variable has been removed. Instead of configuring the variation for the variable, the variable can be excluded from the class poll with the **Classless Read** check box and explicitly read with the desired version with a user-defined command.
- The "class scan" variable scan no longer exists for the DNP3_TG. It is best to delete variables of this type before the driver switch. To trigger a class poll, explicit reading or also a Cold Restart, create Command variables from the DNP3_TG driver variables.
-

3 DNP3_TG - data sheet

General:	
Driver file name	DNP3_TG.exe
Driver name	DNP3 third generation Treiber
PLC types	DNP3 / IEEE 1815 Outstations
PLC manufacturer	DNP3

Driver supports:	
Protocol	DNP3; IEEE Std 1815

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

Requirements:	
Hardware PC	Serial interface; standard network card
Software PC	--
Hardware PLC	--
Software PLC	--
Requires v-dll	X

Platforms:	
Operating systems	Windows 10; Windows 7; Windows 8; Windows 8.1; Windows

Platforms:	
	Server 2008 R2; Windows Server 2012; Windows Server 2012 R2; Windows Server 2016

4 Driver history

Date	Build number	Change
13.11.14	15670	Driver documentation was created newly.
16.03.15	20178	*Subset Level 4* Enhancements implemented and documented.
20.07.15	21164	*File transfer* was implemented and documented.

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
- Ethernet TCP/UDP

For Dual Endpoint, the configured Listening Socket in the Firewall must be configured accordingly.

SOFTWARE

If not already present, copy the driver **DNP3_TG.exe** to the zenon program folder and ensure that **DNP3_TGV.dll** is also present.

CE

Copy the driver **DNP3_TG.dll** to the zenon CE program directory.
The DN3_TGV.dll is not required for Runtime.

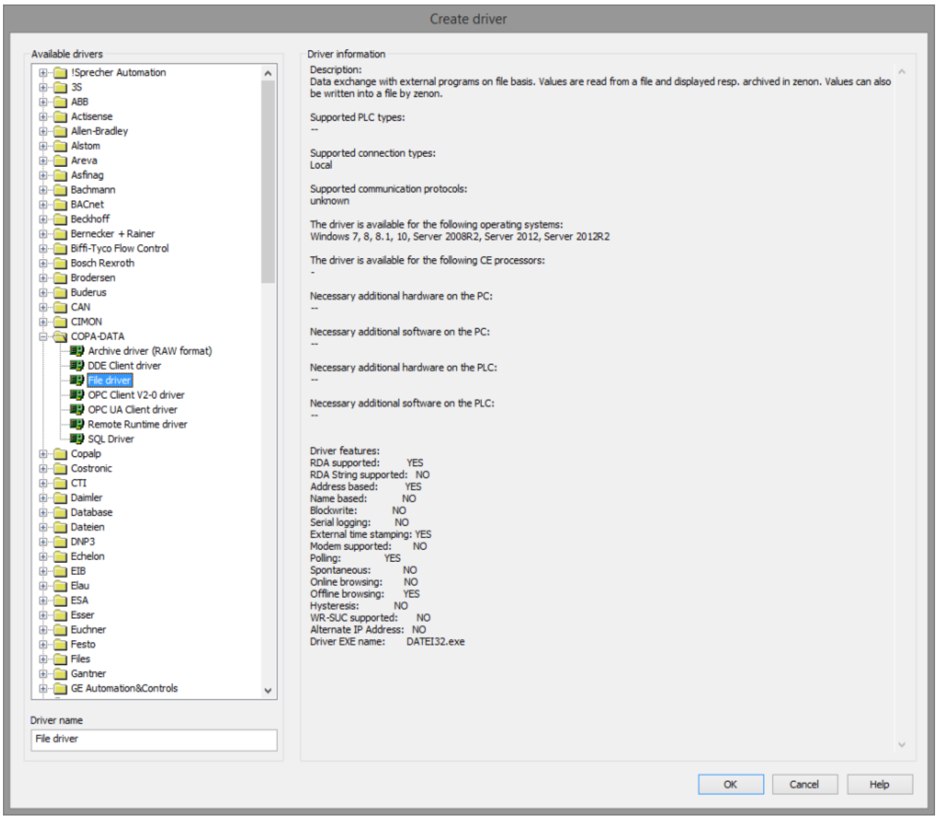
6 Configuration

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.



Driver name: File driver

```
Description:
Data exchange with external programs on file basis. Values are read from a file and displayed resp. archived in zenon. Values can also
be written into a file by zenon.

Supported PLC types:
--

Supported connection types:
Local

Supported communication protocols:
unknown

The driver is available for the following operating systems:
Windows 7, 8, 8.1, 10, Server 2008R2, Server 2012, Server 2012R2

The driver is available for the following CE processors:
-

Necessary additional hardware on the PC:
--

Necessary additional software on the PC:
--

Necessary additional hardware on the PLC:
--

Necessary additional software on the PLC:
--


Driver features:
RDA supported:          YES
RDA String supported:   NO
Address based:          YES
Name based:             NO
Blockwrite:             NO
Serial logging:         NO
External time stamping: YES
Modem supported:        NO
Polling:                YES
Spontaneous:            NO
Online browsing:        NO
Offline browsing:       YES
Hysteresis:             NO
WR-SUC supported:       NO
Alternate IP Address:   NO
Driver EXE name:        DATEI32.exe
```

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: *Empty* 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: *Empty* The information on the selected driver is shown in this area after selecting a driver.

CLOSE DIALOG

Option	Description
OK	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.

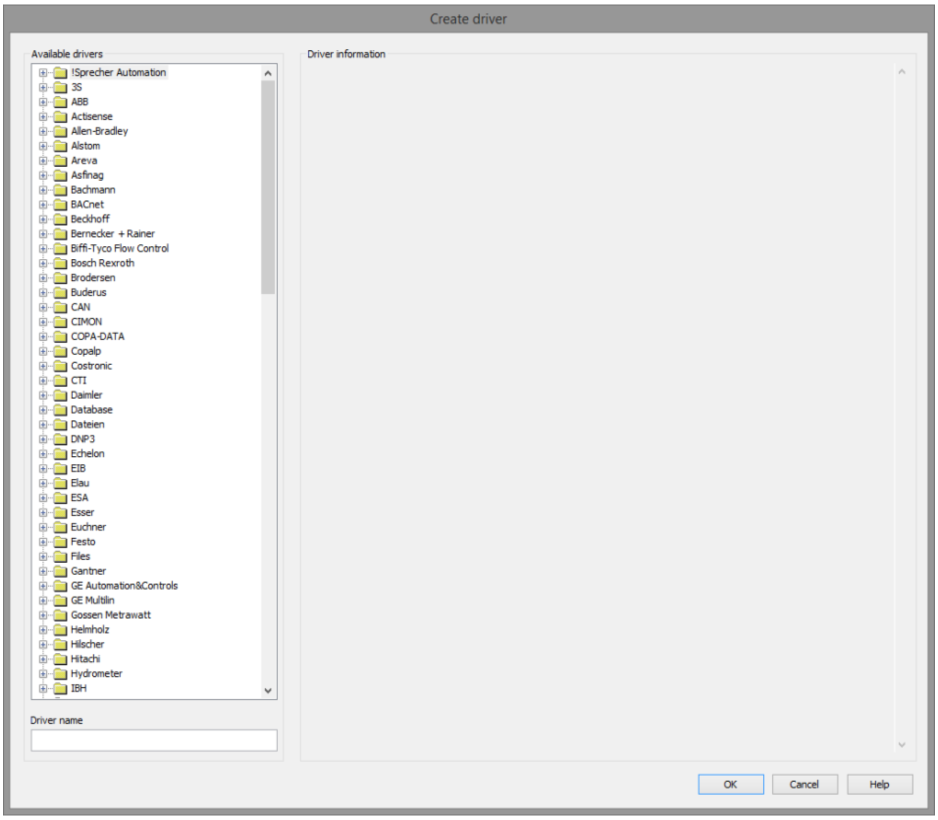
> ### Information
>
> The content of this dialog is saved in the file called Treiber_[Language].xml. You can find this file in the following folder: *C:\ProgramData\COPA-DATA\zenon[version number]*.

CREATE NEW DRIVER

In order to create a new driver:

1. 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.



3. 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 (_).

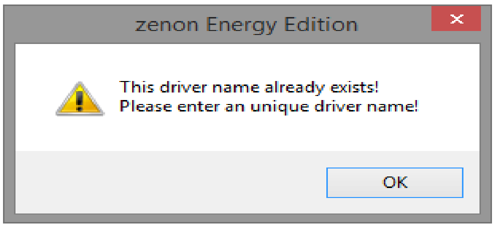
   - **Attention:** This name cannot be changed later on.

4. 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!

ZENON PROJECT

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

- **Intern**
- **MathDr32**
- **SysDrv**

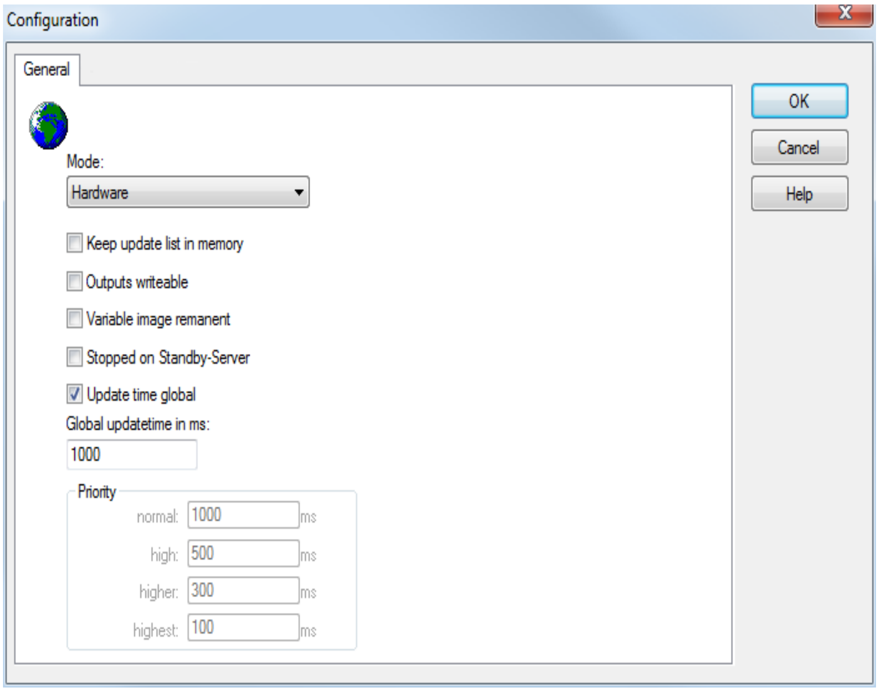
> **Information**
>
> Only the required drivers need to be present in a zenon project. Drivers can be added at a later time if required.

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.



Mode: Hardware

□ Keep update list in memory

□ Outputs writeable

□ Variable image remanent

□ Stopped on Standby-Server

☑ Update time global

Global updatetime in ms: 1000

Priority

normal:	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 *M1* (*0*) to *M8* (*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

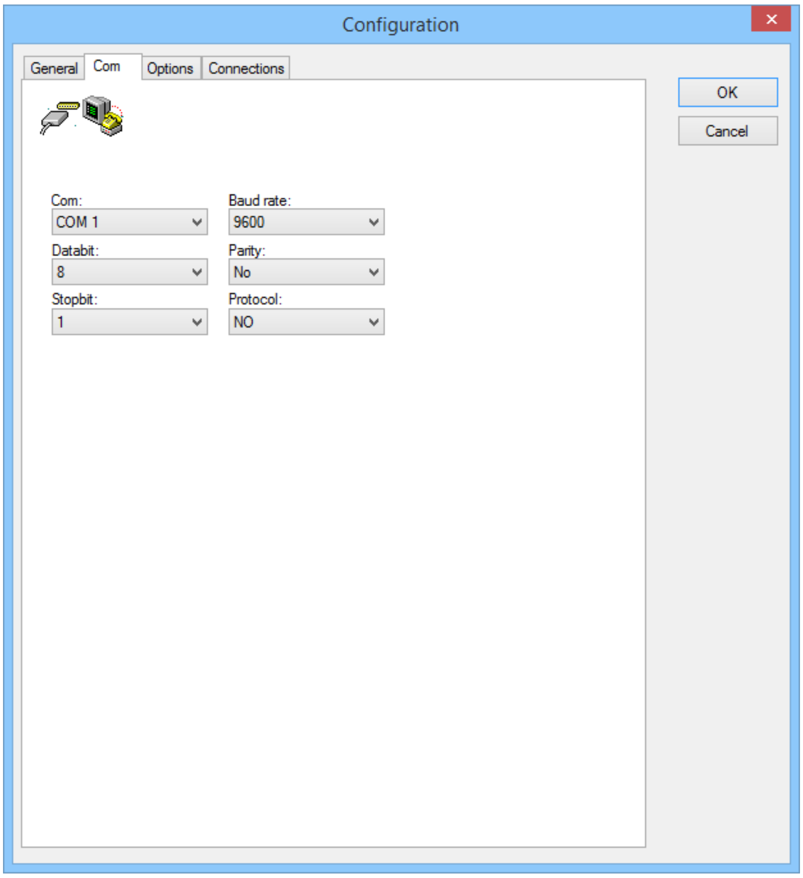
Option	Description
OK	**Applies all changes in all tabs and closes the dialog.**
Cancel	Discards all changes in all tabs and closes the dialog.
Help	Opens online help.

UPDATE TIME FOR SPONTANEOUS DRIVERS

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

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

6.2.2 Com



Com: COM 1

Baud rate: 9600

Databit: 8

Parity: No

Stopbit: 1

Protocol: NO

Parameter	Description
Com	Selection Com port. Default: *1*
Baud rate	Selection baud rate. Adapting to PLC. Default: *9600*
Data bit	Number of data bits. Adapting to PLC. Default: *8*
Stop bit	Selection stop bit. Adapting to PLC. Default: *1*
Parity	Selection parity. Adapting to PLC. Default: *No*
Protocol	Selection protocol. Adapting to PLC.

Parameter	Description
	Default: *No*

CLOSE DIALOG

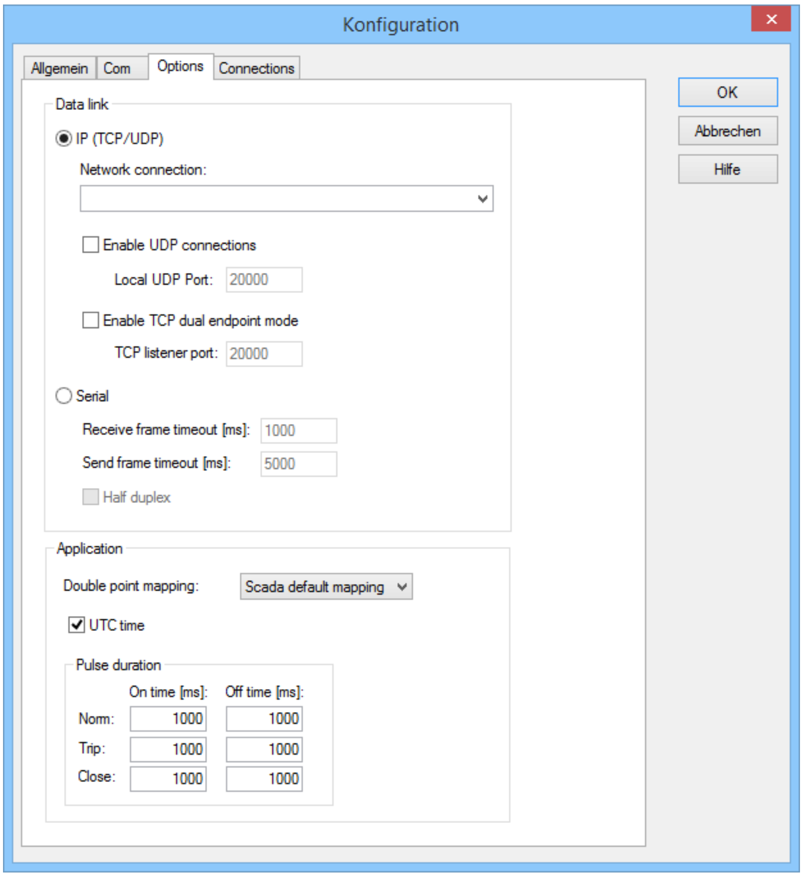
Option	Description
OK	**Applies all changes in all tabs and closes the dialog.**
Cancel	Discards all changes in all tabs and closes the dialog.
Help	Opens online help.

Info

The exact settings depend on the used PLCs. Take the valid values from the manual of your PLC.

6.2.3 Options

You configure the communication parameters in this tab.



Konfiguration

Allgemein | Com | Options | Connections

Data link

● IP (TCP/UDP)

Network connection:

□ Enable UDP connections

Local UDP Port: 20000

□ Enable TCP dual endpoint mode

TCP listener port: 20000

○ Serial

Receive frame timeout [ms]: 1000

Send frame timeout [ms]: 5000

□ Half duplex

Application

Double point mapping: Scada default mapping

☑ UTC time

Pulse duration

	On time [ms]:	Off time [ms]:
Norm:	1000	1000
Trip:	1000	1000
Close:	1000	1000

DATA LINK

Parameter	Description
IP (TCP/UDP)	*Active*: Communication is via TCP/UDP.
Network connection	Selection of the network connection that is used for communication. Entry in the field or selection from a drop-down list from: - Name or - Network connection or - IP address. The names of the network connections available on the local computer are shown in the drop-down list.

Parameter	Description
	For remote Runtime computers, enter the name of the network card or the IP address of the remote computer. This setting is active for UDP communication and TCP dual endpoint communication. If the opening of the LISTENING ports for UDP / TCP for the network configuration is unsuccessful, the driver creates a LISTENING port for all network cards in the system.
Enable UDP connections	*Active*: The UDP protocol can be used for communication with the controllers.
Local UDP port	Local UDP port. This is used for all UDP connections. Default: *20000*
Enable TCP dual endpoint mode	*Active*: TCP Dual-Endpoint Mode is used. In this mode, a controller can initiate a TCP connection if required.
TCP listener port	Port that is used for TCP Dual-Endpoint operation. **Note:** This port is used for all connections. Default: *20000*
Serial	*Active*: Communication is via the serial interface.
Recieve frame timeout [ms]	Timeout in milliseconds for the receipt of a complete telegram. The start time is the first character of the telegram. Default: *1000*
Send frame timeout [ms]	Timeout in milliseconds for the sending of a telegram. If the serial output buffer is filled for longer than this time period, this is considered a transmission error. Default: *5000*
Half duplex	*Active*: A telegram is sent again if a response has been received for the telegram that was last sent on the interface.

APPLICATION

Parameter	Description
Double Point Mapping	Selection of Double Point Mapping to an integer value. Select from drop-down list: - *SCADA default mapping* - *DNP3 binary mapping* - *Custom legacy mapping* Default: *SCADA default mapping*
UTC time	*Active*: All times from and to the PLC are treated as UTC and not as local time. Default: *active*
Puls duration	Setting of the pulse duration for **Norm**, **Trip** and **Close** for each connection. **Norm**, **Trip** and **Close** define which relay is switched
Norm	Corresponds to **NUL**: Pulse duration norm: - **On time:** Period of time, in milliseconds, in which the Output is *ON* Default: *1000* - **Off time:** Period of time, in milliseconds, in which the output is *OFF* Default: *1000*
Trip	Pulse duration trip: - **On time:** Period of time, in milliseconds, in which the output is *ON* Default: *1000* - **Off time:** Period of time, in milliseconds, in which the output is *OFF* Default: *1000*
Close	Pulse duration close: - **On time:**

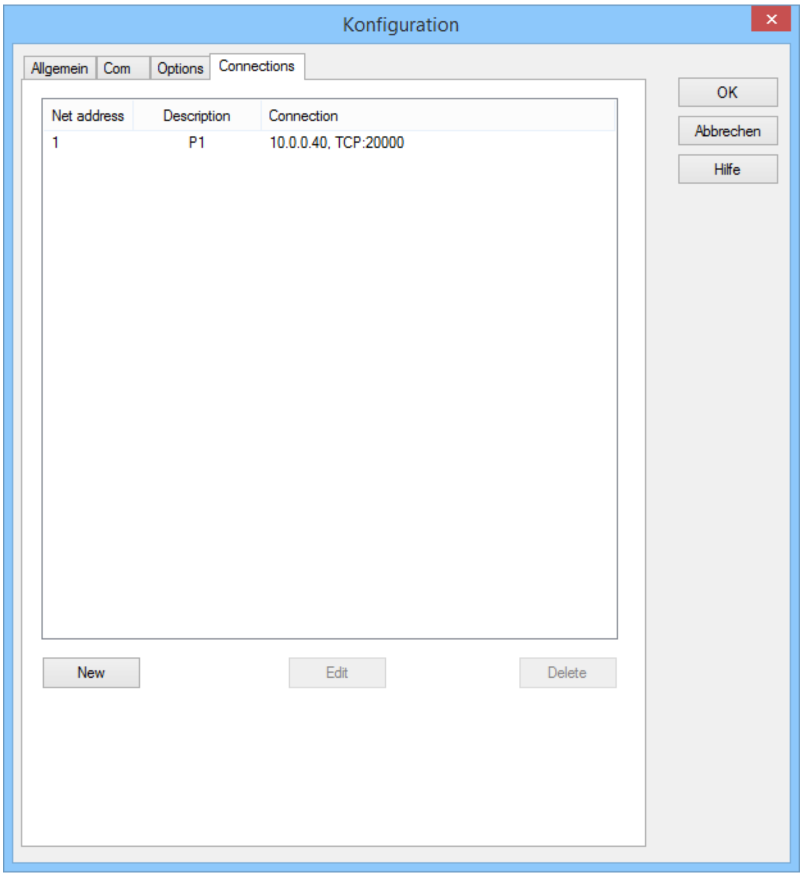
Parameter	Description
	Period of time, in milliseconds, in which the output is *ON* Default: *1000* - **Off time:** Period of time, in milliseconds, in which the output is *OFF* Default: *1000*

CLOSE DIALOG

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

6.2.4 Connections

You configure the connection in this tab.



Net address	Description	Connection
1	P1	10.0.0.40, TCP:20000

Parameter	Description
List of connections	Displays the configured connections.
New	Opens the dialog for creating a new connection (on page 29).
Edit	Opens dialog for editing the selected connection.
Delete	Deletes the selected connection.

CLOSE DIALOG

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

CREATE NEW CONNECTION

1. Click on the **New** button.
2. Enter the connection details.
3. Click on **OK**.

EDIT CONNECTION

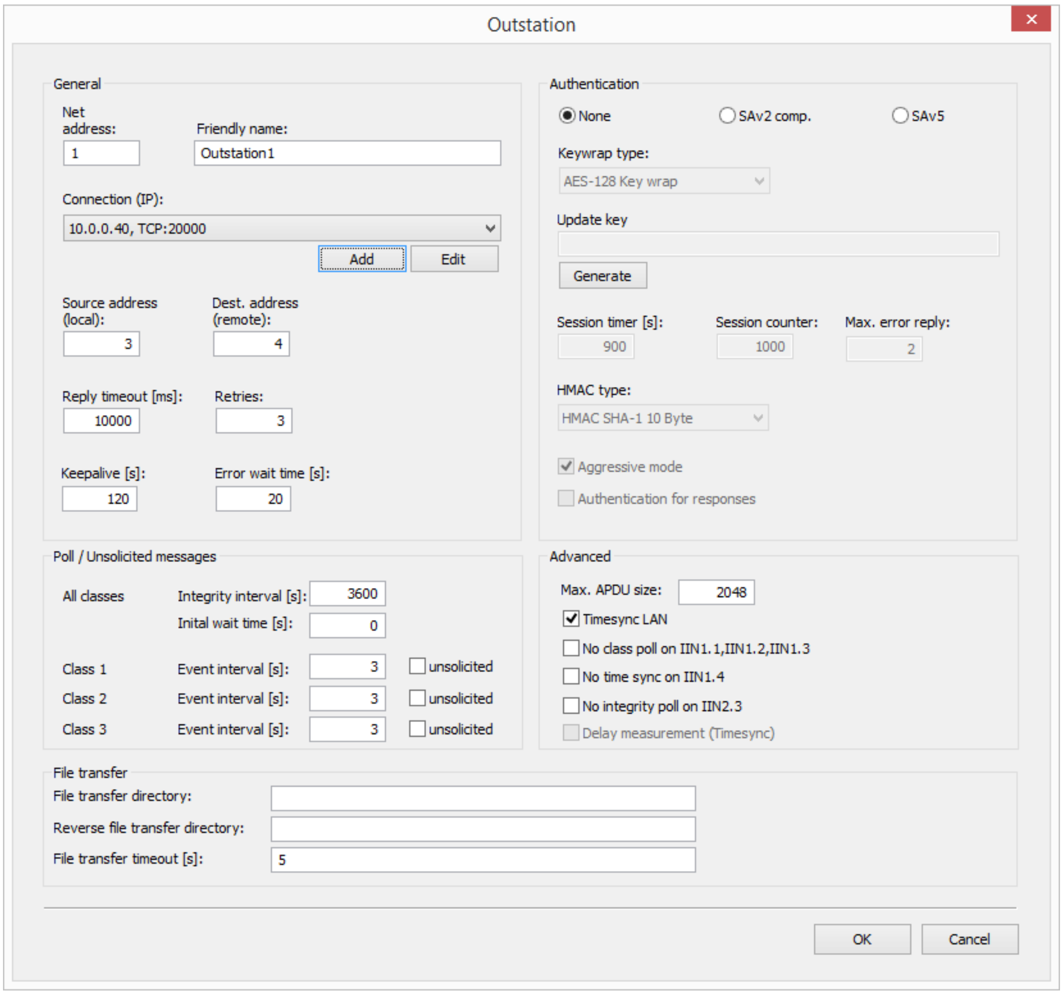
1. Select the connection in the connection list.
2. Click on the **Edit** button.
3. Change the connection parameters.
4. Click on **OK**.

DELETE CONNECTION

1. Select the connection in the connection list.
2. Click on the **Delete** button.
3. The connection will be removed from the list

6.2.4.1 Outstation

You configure connections in this dialog. The dialog is opened if you click, in the **Connections** (on page 27) tab, on the **New** or **Edit** button.



Outstation

General

Net address: 1

Friendly name: Outstation1

Connection (IP): 10.0.0.40, TCP:20000

Source address (local): 3

Dest. address (remote): 4

Reply timeout [ms]: 10000

Retries: 3

Keepalive [s]: 120

Error wait time [s]: 20

Authentication

● None ○ SAv2 comp. ○ SAv5

Keywrap type: AES-128 Key wrap

Update key

Session timer [s]: 900

Session counter: 1000

Max. error reply: 2

HMAC type: HMAC SHA-1 10 Byte

☑ Aggressive mode

□ Authentication for responses

Poll / Unsolicited messages

All classes Integrity interval [s]: 3600

Inital wait time [s]: 0

Class 1 Event interval [s]: 3 □ unsolicited

Class 2 Event interval [s]: 3 □ unsolicited

Class 3 Event interval [s]: 3 □ unsolicited

Advanced

Max. APDU size: 2048

☑ Timesync LAN

□ No class poll on IIN1.1,IIN1.2,IIN1.3

□ No time sync on IIN1.4

□ No integrity poll on IIN2.3

□ Delay measurement (Timesync)

File transfer

File transfer directory:

Reverse file transfer directory:

File transfer timeout [s]: 5

GENERAL

General settings.

Parameter	Description
Net address	Net address of the connection. Value between *0* and *255*. Corresponds to the **Net address** property for the variable.
Friendly name	Name of connection. Freely selectable.
Connection (IP)	Selection of the TCP/UDP channel via which

Parameter	Description
	communication with this Outstation takes place. With virtual Outstations, it is possible that several stations share one channel. (Not available for serial communication.)
Add	Adds a new TCP/UDP link configuration (on page 37). The TCP/UDP links dialog is opened.
Edit	Opens the dialog to edit (on page 36) all existing TCP/UDP links.
Source address (local)	The driver's own address (DNP3 Master).
Dest. address (remote)	Link address of the PLC (DNP3 Outstation). The address 65532 (self address) of the outstation can be used provided the outstation supports this. However, the responses from the outstation with its own address are not accepted by the driver.
Reply timeout [ms]	Time in milliseconds that is waited for a response from the controller. If this time has expired, a repetition is sent. If a fragment of a multi-part response is received. The interval starts over with each fragment received. Default: *10000 ms*
Retries	Number of communication retries after timeouts. - *0*: deactivated
Keepalive [s]	If data is received or sent beyond the given time, a LINK_STATUS_REQUEST telegram is sent. - *0*: deactivated
Error wait time [s]	Error waiting time in seconds. After an error, the given time is waited before another attempt at establishing a connection is made - *0*: deactivated

POLL/UNSOLICITED MESSAGES

Settings for Polling and **Unsolicited Events**.

Parameter	Description
All classes	Integration interval and event interval for all classes.
Integrity interval [s]	Integrity poll of the driver in seconds. Default: *3600*
Initial wait time [s]	Initial waiting time before another attempt to establish a connection Can be used to stagger the establishment of a connection to different Outstations. Default: 0
Class1 - Class 3	Settings for Class 1, Class 2 and Class 3. For each class, the options **Event interval [s]** and **unsolicited** are also configured.
Event interval [s]	Specific interval for the respective class for the Polling of Events in seconds. Value: *0 to 99999* - 0: *no polling* Default: *0*
unsolicited	*Active*: For this class, Unsolicited Events are accepted. (See also chapter adressing .)

AUTHENTICATION

Settings for authentication.

Parameter	Description
Authentication	Selection of authentication method: - **None** - **SAv2 compatible** - **SAv5**
None	*Active*: No authentication filter is used.
SAv2 compatible	*Active*: Authentication in compatibility mode for version 2.
SAv5	*Active*: Authentication for version 5.

Parameter	Description
Keywrap type	Keywrap algorithm to be applied. Select from drop-down list: - AES-128 - AES-256
Update Key	Authentication key for secure communication. Entry in the field or automatic generation by clicking on the **Generate** button. It is expected for: - AES-128 Keywrap: 32 hexadecimal digits - AES-256 Keywrap: 64 hexadecimal digits **Permitted characters:** - Digits: *0* to *9* - Letters: *a* to *f* and *A* to *F* - Special characters: Space, points and colon All remaining letters: Are interpreted as *0* and will possibly fail during generating the session key.
Generate	Creates a random **Update Key**.
Session timer	Validity of the session key in seconds. Value: *<700000 s* Default: *900 s*
Session counter	Select how often a session key may be used. Value: *<10000 s* Default: *1000*
Max. error reply	Specification how many mistakes are reported in the authentication. Value: *0 bis 10* Default: *2*
HMAC type	Type of HMAC (HASH function) used if the driver sends a challenge for a response Authentication Requests. Select from drop-down list:

Parameter	Description
	- HMAC SHA-1 4 byte (Index 1) Communication only permitted with the **SAv2 comp.** authentication method. Only applicable for serial connections. The Session timer must be selected as less than 1800 seconds (30 minutes). - HMAC SHA-1 8 Byte (Index 5) - HMAC SHA-1 10 Byte (Index 2) - HMAC SHA-256 8 Byte (Index 3) - HMAC SHA-256 16 Byte (Index 4)
Aggressive mode	*Active*: Authentication is carried out in the aggressive mode.
Authentication for responses	Only permitted with **SAv5** authentication method. - Active: The driver sends Authentication Requests for responses from the Outstation and Unsolicited Responses. Default: not active.

ADVANCED

Parameter	Description
Max. APDU size	Maximum size of a DNP3 Application Layer Fragment - Minimum: 249 - Maximum: 65535 Default: 2048
Timesync LAN	- *Active*: Time synchronization via the LAN. Can only be configured for TCP connection. If serial connection type (on page 23) is selected, the property in the driver dialog is automatically activated and grayed out. **Note:** Use variation 3 of the time object. This is not accepted by all stations.
No class poll on IIN1.1, IIN1.2, IIN1.3	Active: Driver ignores Internal Indication Flags IIN1.1, IIN1.2 and IIN 1.3 Compatibility setting. Ensure that polling for events is

Parameter	Description
	carried out regularly in order to avoid possible buffer overflows in the Outstation.
No time sync on IIN1.4	*Active*: The driver ignores the internal indication flag IIN1.4. Compatibility setting. Time synchronization required.
No integrity poll on IIN2.3	*Active*: Driver ignores the Internal Indication Flag IIN2.3 (Buffer Overflow). Compatibility setting.
Delay Measurement (Timesync)	If active, the driver carries out a Delay Measurement when time synchronizing. Active if: - Connection type Serial - IP connection type (TCP/UDP) **and Timesync LAN** not active. **Note:** With a serial connection, **Timesync_LAN** is active by default and grayed out, **Delay Measurement (Timesync)** can however be selected.

FILE TRANSFER

Settings for file transfer

Parameter	Description
File transfer directory	Folder for file transfer (read). Specify the folder on the Runtime computer in which files for *File transfer* are to be stored. The data is copied from the outstation to the master. The computer on which zenon Runtime runs is designated as the master. This path must be entered manually. Note that this path must also be present on the target system. If this path does not exist on the target system, the file transfer will not be successful. Only the entry of an absolute path is permitted. The entry is checked. A corresponding warning dialog is shown in the event of an incorrect configuration. Ensure that there are the corresponding access rights. **Note:** For **File transfer directory** and **Reverse file**

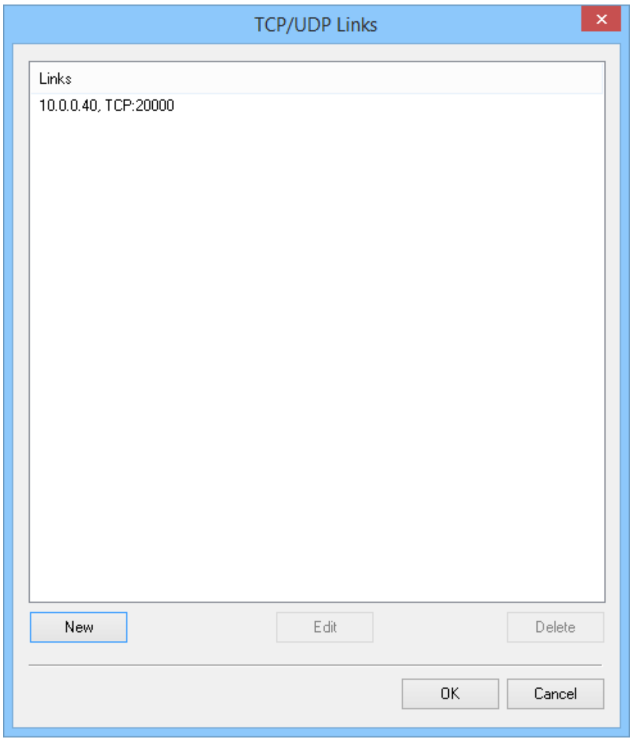
Parameter	Description
	transfer directory, the same directory can also be used.
Reverse file transfer directory	Specify the folder on the Runtime computer in which the files for transfer in write direction are stored. The files that are to be sent by the master (zenon) to the outstation are stored in this folder. This path must be entered manually. Note that this path must also be present on the target system. If this path does not exist on the target system, the file transfer will not be successful. Only the entry of an absolute path is permitted. The entry is checked. A corresponding warning dialog is shown in the event of an incorrect configuration. Ensure that there are the corresponding access rights. **Note:** For **File transfer directory** and **Reverse file transfer directory**, the same directory can also be used.
File transfer timeout [s]	Time period in which an attempt is made to conclude the file transfer. Once this time has expired, the file transfer is canceled with an error on the master. Default: *5 seconds*

CLOSE DIALOG

Option	Description
OK	**Applies settings and closes the dialog.**
Cancel	Discards all changes and closes the dialog.

6.2.4.1.1 TCP/UDP Links

Display and administration of all TCP/UDP links.



10.0.0.40, TCP:20000

Parameter	Description
Links	Displays the configured TCP/UDP channels.
New	Opens the dialog to create a new TCP/UDP channel.
Edit	Opens the dialog to edit the selected TCP/UDP channel.
Delete	Deletes selected TCP/UDP link.
OK	Applies changes and closes dialog.
Cancel	Discards all changes and closes the dialog.

CREATE A NEW TCP/UDP CHANNEL

1. Click on the **New** button.
2. Enter the connection details.
3. Click on **OK.**

EDIT TCP/UDP CHANNEL

1. Select the desired Link in the list.

2. Click on the **Edit** button.
3. Change the link parameter.
4. Finish with **OK**.

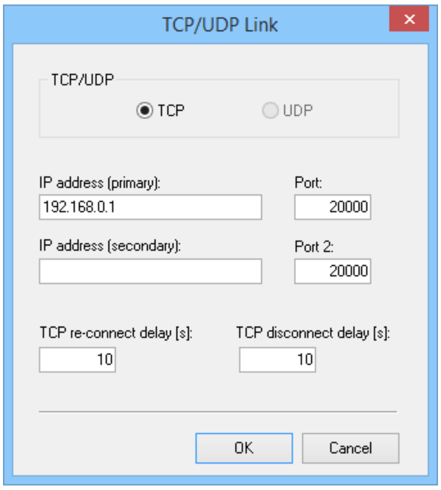
DELETE TCP/UDP CHANNEL

1. Select the desired Link in the list.
2. Click on the **Delete** button.
3. The connection will be removed from the list

> ⚠ **Attention**
>
> Only one outstation with a port number can be configured per IP address. If two outstations at the same IP address with different port numbers need to be addressed, use one more driver.

6.2.4.1.2TCP/UDP link configuration

Configuration of TCP/UDP links.



TCP/UDP Link

TCP/UDP

● TCP ○ UDP

IP address (primary): 192.168.0.1

Port: 20000

IP address (secondary):

Port 2: 20000

TCP re-connect delay [s]: 10

TCP disconnect delay [s]: 10

Parameter	Description
TCP/UDP	Selection of connection type.
TCP	*Active*: TCP connection
UDP	*Active*: UDP connection
IP address (primary)	Entry of the primary IP address.

Parameter	Description
Port	Primary UDP/TCP port of the PLC. Default: *20000*
IP address (secondary)	Backup address for redundant connection to the PLC. Only permitted for TCP connections. Not possible in combination with TCP Dual-Endpoint.
Port (secondary)	Port of the backup address.
TCP re-connect delay [s]	Minimum time period in seconds that must be between two TCP connection attempts.
TCP disconnect delay [s]	If no data is sent or received over the given time period, the TCP connection is disconnected automatically. Can also be used without TCP Dual-Endpoint. In this case, the Polling intervals must also be configured accordingly. - *0*: deactivated

CLOSE DIALOG

Option	Description
OK	**Applies settings and closes the dialog.**
Cancel	Discards all changes and closes the dialog.

Connection establishment

The configured settings connections for the primary connection **IP address (primary)** and also for the secondary connection **IP address (secondary)** are working as follows:

- During the start:
  - Is the primary connection not available, a connection attempt to the secondary connection is built up.
  - Are both connections not available, the driver attempts continuously to establish a connection to one of the two addresses. Thereby it attempts to establish a connection alternating from the primary to the secondary address.
- Connection failure during the running operation:
  - Is the primary connection not available anymore, the driver attempts to establish a connection to the secondary connection.

- Is the secondary connection not available anymore, the driver attempts to establish a connection to the primary connection.

7 Creating variables

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

7.1 Creating variables in the Editor

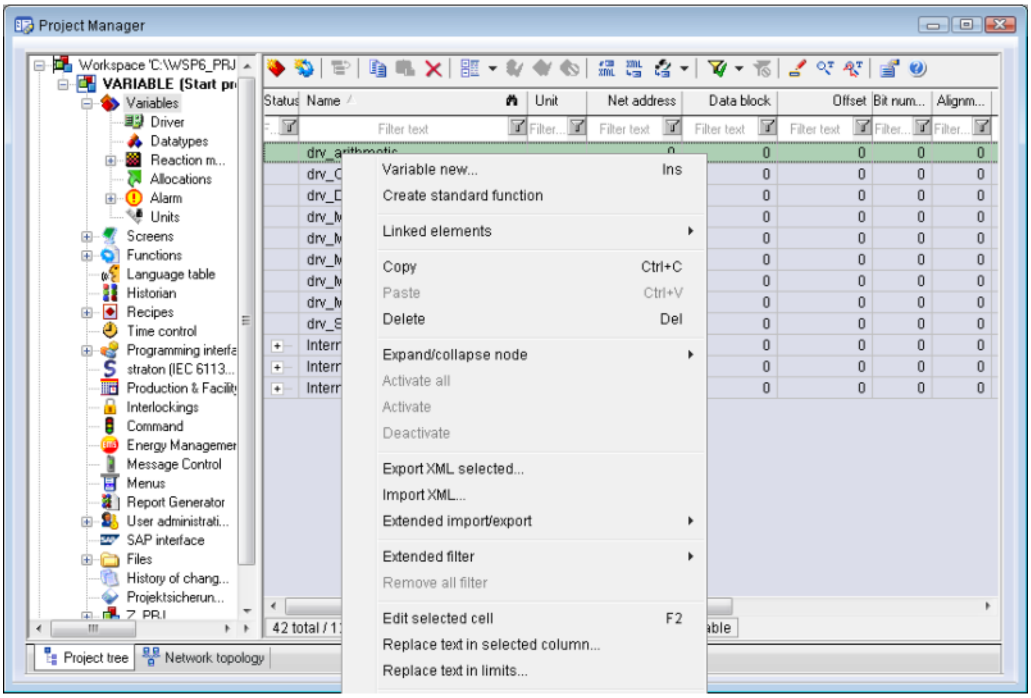
Variables can be created:

- 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:

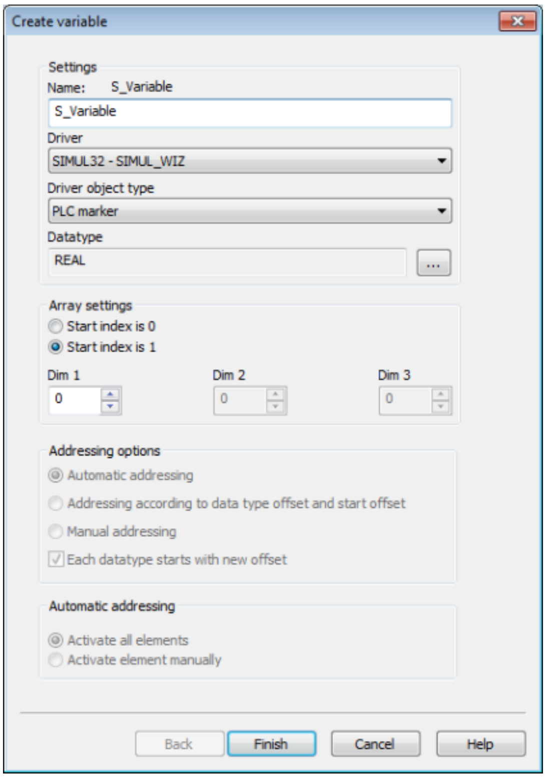
1. Select the **New variable** command in the **Variables** node in the context menu



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



Create variable

Settings

Name: S_Variable

S_Variable

Driver

SIMUL32 - SIMUL_WIZ

Driver object type

PLC marker

Datatype

REAL

Array settings

○ Start index is 0

● Start index is 1

Dim 1: 0

Dim 2: 0

Dim 3: 0

Addressing options

● Automatic addressing

○ Addressing according to data type offset and start offset

○ Manual addressing

☑ Each datatype starts with new offset

Automatic addressing

● Activate all elements

○ Activate element manually

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. 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.
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.htm**)	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

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 Addressing

The offset determines the DNP Point Nummer. In combination with the driver object type, the DNP object group is determined.

Group/Property	Description
General	Property group for general settings.
Name	Freely definable name. **Attention:** For every zenon project the name must be unambiguous.
Identification	Freely definable identification. E.g. for Resources label, comments, ...
Addressing	Properties of the variable addressing.
Net address	Network address of variables. This address refers to the bus address in the connection configuration of the driver. This defines the PLC, on which the variable resides. Attention: The DNP address of the Outstation is configured separately during the connection.
Data block	Only for variables of the *Device Attribute* driver object type. In this case, the data block of the index is configured.
Offset	For variables of all driver object types except **Device Attribute:** DNP Point in the Outstation. To see always in combination with the driver object type. For example: **Analog Input**, **Group** *20*, **Point** *0* here corresponds to **Offset** *0* with an *analog input* variable. For **Device Attribute** driver object type variables. Variation (type of the device attribute).
Alignment	not used for this driver
Bit number	not used for this driver
String length	Only available for String variables.

Group/Property	Description
	Maximum number of characters that the variable can take.
Driver connection	Driver-specific properties.
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.
Data Type	Data type of the variable. Is selected during the creation of the variable; the type can be changed here. **Attention:** If you change the data type later, all other properties of the variable must be checked and adjusted, if necessary.
Command Mode	Only for *Binary Output* and *Analog Output*. Determines whether the driver carries out a *Direct operate*, *Direct operate no-ack* or a *Select Before Operate*. Default:*Direct Operate*
Classless read	Assignment of the values to the variable with explicit read queries: - *Inactive*: Values that are received from an Integrity-Poll, an Event-Poll or unsolicited responses are assigned to the variable. Values that are received as a response to an explicit read request are not assigned - *Active*: The variables are only assigned values that are received as a response to an explicit read query.

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
Absolute time	77	X	--	*UDINT,REAL*	**Note:** This variable must be created manually. It is

Driver Object Type	Channel type	Read	Write	Supported data types	Comment
					not offered if variables in the driver are created with the **import variables from the driver** command. Ensure that the **Net address** is correct.
Analog input	68	X	--	*UDINT, DINT, REAL, LREAL, UINT, INT*	
Analog Input reporting deadbands	78	X	X	*UINT, UDINT, REAL*	**Note:** This variable must be created manually. It is not offered if variables in the driver are created with the **import variables from the driver** command. Ensure that the **Net address** is correct.
Analog output	69	X	X	*UDINT, DINT, REAL, LREAL, UINT, INT*	
Binary Input	64	X	--	*BOOL*	
Binary Output	65	X	X	*BOOL, USINT*	
Command	11	X	X	*STRING*	**Note:** This variable must be created manually. It is not offered if variables in the driver are created with the **import variables from the driver** command.
Counter	66	X	X	*BOOL, UDINT, DINT, USINT, UINT, INT, SINT*	

Driver Object Type	Channel type	Read	Write	Supported data types	Comment
Device attributes	72			*INT, INT, INT, DINT, EAL, REAL, STRING*	**Note:** This variable must be created manually. It is not offered if variables in the driver are created with the **import variables from the driver** command. Ensure that the **Net address** is correct.
Double-bit binary input	74			*USINT, SINT*	
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 74) chapter.
File transfer result	36	X	--	*STRING*	**Note:** This variable must be created manually. It is not offered if variables in the driver are created with the **import variables from the driver** command. Ensure that the **Net address** is correct.

Driver Object Type	Channel type	Read	Write	Supported data types	Comment
Frozen Counter	67	X	--	*BOOL, UDINT, DINT, USINT, UINT, INT, SINT*	
Internal indications	73	X	--	*BOOL*	
Legacy double bit binary input	71	X	--	*USINT*	**Note:** This variable must be created manually. It is not offered if variables in the driver are created with the **import variables from the driver** command. Ensure that the **Net address** is correct.
Security statistics	76	X	--	*DINT, UDINT*	
Statistics	9	X	--	*UDINT*	Local communication statistics
String	70	X	X	*STRING*	
Virtual terminal output block	75	X	X	*STRING*	

Key:

X: supported

--: not supported

7.3.2 Mapping of the data types

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.

DRIVER OBJECT TYPES AND SUPPORTED IEC DATA TYPES FOR VARIABLES IN ZENON

Driver object types	Channel type	Supported data types (DataType)	Read	Write	Comment
Binary Input *Static object group 1* *Event object group 2*	*64*	*BOOL*	X	--	
Legacy double-bit binary input	*71*	*USINT*	X	--	USINT-Mapping of 2 binary states with sequenced offsets. (No Couble-Bit Binary Input! Group 3 / 4.)
Double-bit binary input *Static object group 3* *Event object group 4*	*74*	*USINT*	X	--	
Binary Output *Static object group 10* *Event object Group 11* *CROB object group 12*	*65*	*BOOL*	--	X	Value 1: LATCH_ON Value 0: LATCH_OFF
Binary Output *Static object group 10* *Event object group 11* *CROB object group 12*	*65*	*USINT*	--	X	Value 1: PULSE_ON Value 2: PULSE_OFF Value 3: LATCH_ON Value 4: LATCH_OFF Value 65: CLOSE Value 129: TRIP Value 1 for PULSE_ON, LATCH_ON and CLOSE, value 0 for PULSE_OFF,

Driver object types	Channel type	Supported data types (DataType)	Read	Write	Comment
					LATCH_OFF and TRIP
Counters *Static object group 20* *Event object group 22*	*66*	*UINT, UDINT*	X	-	
Frozen Counters *Static object group 21* *Event object group 23*	*67*	*UINT, UDINT*	X	--	
Analog Input *Static object group 30* *Event object group 32*	*68*	*INT, DINT, REAL, LREAL*	X	--	
Analog Output *Static object group 40* *Event object group 42* *Analog Output Block object group 41*	*69*	*INT, DINT, REAL, LREAL*	--	X	written value is mirrored as a response after successful writing
Octet strings *Static object group 110* *Event object group 111*	*70*	*STRING*	X	X	sent values are not mirrored. Get the latest values via update.
Virtual terminal output block	*75*	*STRING*	X	X	sent values are not mirrored. Get the latest values via

Driver object types	Channel type	Supported data types (DataType)	Read	Write	Comment
Static object group 112 *Event object group 113*					update.
Device Attributes *Group 0*	72	INT, INT, INT, DINT, EAL, REAL, STRING	X	X	Device attributes can be written. IEEE 1815 stipulates which device attributes from the standard set can be written.
Internal indications *Object group 80*	73	BOOL	X	--	
Security statistics *Static object group 121* *Event object group 122*	76	DINT, UDINT	X	--	
Statistics	9	UDINT	X	--	Local communication statistics
Command	11	STRING	X	X	Initiates an explicit command e.g. explicit read, counter freeze, ...
Absolute time Object Group 50	77	LREAL, DINT	X	--	Is only read explicitly via command. "Classless Read" must be activated for the variable.
Analog Input reporting deadbands *Object Group 34*	78	REAL, UINT, UDINT	X	X	Is only read explicitly via command. "Classless Read" must be activated for the

Driver object types	Channel type	Supported data types (DataType)	Read	Write	Comment
					variable.
File transfer result	36	STRING	X	--	The result of the following file transfer actions is shown on this variable: - *GET_FILE_INFORMATION* - *READ_DIRECTORY*

BINARY OUTPUTS

The writing of *Binary Outputs* is always carried out by means of a **CROB** (group 12), with a choice of *Direct Operate*, *Select Before Operate*, or *Direct Operate No Ack*. In doing so, the setting for the variable-specific property **Command Mode** (accessible via XML export/import and VBA **COMMAND_MODE**) is used. Direct writing to group *10* is not supported by the driver.

After successful writing, the variable receives the value *0* for *Binary Outputs* with *PULSE_OFF*, *LATCH_OFF* and *TRIP*, and value *1* for *PULSE_ON*, *LATCH_ON* and *CLOSE*. The value is also updated if a *Binary Output Status* (group 10) or *Binary Output Status Event* (group 11) object is received.

Command processing with *Binary Output* variables:

- Only **dual command** or **switching commands** can be used.
- The property **Select Before Operate** must be deactivated for the variable; the DNP3_TG driver uses *Auto-SBO* if necessary.

Use of the **Qualifier of Command** is recommended for the Command Processing in order to determine the type:

Type	QoC
PULSE OFF/PULSE ON	*1*
LATCH OFF/LATCH ON	*0*
TRIP/CLOSE	*2*

The **Qualifier of Command** option can be used with both *Binary Output* variables of the *BOOL* data type as well as the *USINT* data type.

ANALOG OUTPUTS

The writing of *Analog Outputs* is always with a group 41, with the choice of *Direct Operate*, *Select Before Operate* or *Direct Operate No Ack*. In doing so, the setting for the variable-specific property **Command Mode** (accessible via XML export/import and VBA **COMMAND_MODE**) is used.

After successful writing, the variable first receives the value written in Runtime. The value is also updated if an *Analog Output Status* (group 40) or *Analog Output Status* (group 42) object is received.

Use the *write set value* action text for *Analog Outputs*. According to the **Command Mode** option for the variable, for *Analog Outputs* either a *direct operate* or a *select before operate* is carried out. The **Select Before Operate** option must be deactivated for the variable. The **Qualifier of Command** option is not available for the *Write set value* action and has no influence with *Analog Outputs*.

For two-stage command processing, the command is only sent to the outstation at the second stage of the command, including **Select** with *Auto-SBO*.

BEHAVIOR OF COMMAND MODE

The following is applicable for the **Command mode** property for a variable:

- *direct Operate*: The value is written directly. (Default)
- *automatic SBO*: When writing from the stack, a Select is sent first, which is then confirmed with an immediate *Operate* if the answer is positive.

This applies for binary and analog outputs.

ASSIGNMENT OF WRITE SET VALUE TO BINARY OUTPUT WITHOUT COMMAND PROCESSING

- For a Binary Output variable of BOOL datatype, *LatchON* is sent for *High* or *LatchOFF* for *Low*.
- For a USINT data type Binary Output variable, the set value is handled in accordance with the table below

value USINT	Action	Comment
0	*keine*	
1	*Pulse On*	
2	*Pulse Off*	Not fully compatible. Is not necessarily supported by the outstation.
3	*Latch On*	
4	*Latch Off*	

value USINT	Action	Comment
65	*Close*	
129	*Trip*	

The **Command Mode** is also taken into account for direct writing of set values without a command processing

RULES FOR SETTING THE STATUS BIT WHEN USING THE COMMAND PROCESSING

Status bits are set according to the rules in the table:

- As soon as a *Select* is sent, *SE* and *CoT_act* are set.
- If an error occurs when sending, or a negative answer has been received, *SE*, *P/N* and *CoT_actcon* are set (*4*).
- The status is *SE* and *CoT_actcon* (*1*, *2*, *3*) after a positive response.
  After that, the *Operate* is sent automatically and the status is set to *CoT_act*.
  If an error occurs when sending, or a negative response was received, *P/N* and *CoT_actcon* are set (*2*).
- If the response to the *Operate* is positive, the status is initially set to *CoT_actcon* and then to *CoT_actterm* (*1*).
- The status after a *DirectOperate* is also handled (*1*, *2*, *3*).
- In the event of an error in sending, the invalid bit is set (*2*, *4*).

Fall	Action	Status	Success	Status	Action	Status	Success	Status	Status
1	*select*	*act SE*	*ack*	*actcon SE*	*operate*	*act*	*ack*	*actcon*	*actterm*
2	*select*	*act SE*	*ack*	*actcon SE*	*operate*	*act*	*nack*	*actcon P/N*	
3	*select*	*act SE*	*ack*	*actcon SE*					
4	*select*	*act SE*	*nack*	*actcon SE P/N*					

Meaning of the terms in the **Success** column:

- ack: positive
- nack: negative
- t/o: Timeout

LEGACY DOUBLE-BIT BINARY INPUT

The driver carries out mapping with the *Legacy double-bit binary input* driver object type according to the settings in the driver configuration.

For example: Using USINT Offset 14 combine the binary states with offset 14 and 15.

Attention: The mapping of two *Binary inputs* to a *Legacy double-bit binary input* is a driver-internal function and should not be confused with the DNP3 *group 3/4*, driver object type*Double-bit binary input*.

RESULTS

Binary status bits SCADA default mapping	Binary status bits DNP3 binary logic mapping	Binary status bits Custom legacy mapping	Status in zenon USINT with *Binary Inputs Double)*
both bits: *off* (*false*)	First bit: (**Offset** 14): *off (false)* second Bit (**Offset** 15): *on (true)*	First bit: (**Offset** 14): *on (true)* second Bit (**Offset** 15): *off (false)*	*2 - intermediate*
both bits: *on* (*true*)	both bits: *on (true)*	both bits: *on (true)*	*3 - faulty*
First bit: (**Offset** 14): *on* (*true*) second Bit (**Offset** 15): *off* (*false*)	both bits: *off (false)*	both bits: *off (false)*	*0 - off*
First bit: (**Offset** 14): *off* (*false*) second Bit (**Offset** 15): *on* (*true*)	First bit: (**Offset** 14): *on (true)* second Bit (**Offset** 15): *off (false)*	first Bit (**Offset** 14): *off (false)* second Bit (**Offset** 15): *on (true)*	*1 - on*

DEVICE ATTRIBUTES

Device Attributes must be read via an explicit read command (see **Command**). The "READ_VAR 1 0 254" command reads all device attributes for the network address "1" The index is determined via the data block setting of the variable. The **Variation** matches the offset of the variable. The user must select the correct data type. When creating, the network address must be set according to the station number. The "classless reading" property must be activated. Device attributes can currently only be read.

LOCAL COMMUNICATION STATISTICS

The status of the local DNP3 controllers can be monitored by means of the communication statistics. The variables are addressed via **Net address** and **Offset**. In the following table, all available statistics variables are listed with their Offset.

If a physical data-link channel is used by several virtual outstations, all these virtual outstations share the variables of classes Data-link (phys) and TCP.

Addressing is carried out via the network address of any desired virtual Outstation.

TCP state (Offset 204): Value 0 means no connection, value 2 means that the connection is being established, value 3 means that the connection has been established

TCP flags (Offset 206): Value 5 means connection established via primary IP address, value 6 means connection established via secondary IP address

Class statistic	Detail type statistic	Offset
Data-link (virt)	*Bytes received*	*500*
Data-link (virt)	*Bytes sent*	*501*
Data-link (virt)	*Frames received*	*502*
Data-link (virt)	*Frames sent*	*503*
Data-link (virt)	*Frames discarded*	*504*
Data-link (virt)	*Link status timeouts*	*505*
Data-link (virt)	*Link status errors*	*506*
Data-link (virt)	*Frame errors*	*507*
Transport	*Fragments received*	*600*
Transport	*Segments received*	*601*
Transport	*Assembling errors*	*602*
Transport	*Fragments sent*	*603*

Class statistic	Detail type statistic	Offs et
Transport	*Segments sent*	*604*
Application layer (connection)	*Failures*	*700*
Application layer (connection)	*Sequence errors*	*701*
Application layer (connection)	*Retries*	*702*
Application layer (connection)	*Timeouts*	*703*
Application layer (connection)	*Invalid fragments*	*704*
Application layer (connection)	*Unsolicted responses*	*705*
Authentication	*Unexpected messages*	*800*
Authentication	*Authorization failures*	*801*
Authentication	*Authentication failures*	*802*
Authentication	*Reply timeouts*	*803*
Authentication	*Rekeys due to authentication failures*	*804*
Authentication	*Total messages sent*	*805*
Authentication	*Total messages received*	*806*
Authentication	*Critical messages sent*	*807*
Authentication	*Critical messages received*	*808*
Authentication	*Discarded messages*	*809*
Authentication	*Error messages sent*	*810*
Authentication	*Error messages received*	*811*
Authentication	*Successful authentications*	*812*
Authentication	*Session key changes*	*813*
Authentication	*Failed session key changes*	*814*
Authentication	*Rekeys due to restart*	*815*
Application layer (state)	*Integrity polls*	*900*
Application layer (state)	*Event class 1 polls*	*901*

Class statistic	Detail type statistic	Offs et
Application layer (state)	*Event class 2 polls*	*902*
Application layer (state)	*Event class 3 polls*	*903*
Application layer (state)	*Commands succeeded*	*904*
Application layer (state)	*Commands failed*	*905*
Application layer (state)	*Response object count*	*906*
Application layer (state)	*Unsolicted response object count*	*907*
Data-link (phys)	*Bytes received*	*100*
Data-link (phys)	*Bytes sent*	*101*
Data-link (phys)	*Invalid bytes received*	*102*
Data-link (phys)	*Valid frames received*	*103*
Data-link (phys)	*Frames sent*	*104*
Data-link (phys)	*Frame CRC erros*	*105*
Data-link (phys)	*Send timeouts*	*106*
Data-link (phys)	*Receive timeouts*	*107*
TCP	*Connection succeeded primary*	*200*
TCP	*Connection succeeded secondary*	*201*
TCP	*Connection failed primary*	*202*
TCP	*Connection failed secondary*	*203*
TCP	*State*	*204*
TCP	*State counter*	*205*
TCP	*Flags*	*206*

EXPLICIT COMMAND

Command driver object type variables are used to instigate a certain action once. The action is initiated by the writing of a string value to the command variable. The result of the action is assigned to the variables.

- Syntax: A command consists of the command name and parameters. The command name and the individual parameters are separated by spaces.
  Example: READ_VAR 1 0 254 Read all device attributes for network address 1 (Object Group 0, Variation 254)
  Example: READ 1 1 Read all binary inputs for network address 1 (Object Group 1, Variation 0 (default variation))

Attention: It is possible, with the compilation of parameters for an explicit command, that there are combinations that do not correspond to a valid command in the meaning of the IEEE 1815 standard. Commands are sent to the outstation without further checking and can, under certain circumstances, trigger incorrect actions here.

EXPLICIT READ, VARIATION 0

Command: READ

Read all points of a group: (Qualifier Code 06 - you can use this number of parameters for static Object Groups and Event Object Groups)

- Parameter 1: **Net address**
- Parameter 2: Group

Read n-points of a group: (Qualifier Code 07, 08, 09 - you can use this number of parameters for the Event Object Groups. The use of a number greater than 65535 (Qualifier Code 09) is not recommended))

- Parameter 1: **Net address**
- Parameter 2: Group
- Parameter 3: Number of the points to be read

Read certain points of a group: (Qualifier Code 00, 01 - you can use this number of parameters for static Object Groups, for example: Group 1 - Binary Input).

- Parameter 1: **Net address**
- Parameter 2: Group
- Parameter 3: Start index
- Parameter 4: Stop index

EXPLICIT READ

Command: READ_VAR

Read all points of a group: (Qualifier Code 06 - you can use this number of parameters for static Object Groups and Event Object Groups)

- Parameter 1: **Net address**

- Parameter 2: Group
- Parameter 3: Variation

Read n-points of a group: (Qualifier Code 07, 08, 09 - you can use this number of parameters for Event Object Groups. The use of a number greater than 65535 (Qualifier Code 09) is not recommended))

- Parameter 1: **Net address**
- Parameter 2: Group
- Parameter 3: Variation
- Parameter 4: Number of the points to be read

Read certain points of a group: (Qualifier Code 00, 01 - you can use this number of parameters for static Object Groups, for example: Group 1 - Binary Input).

- Parameter 1: **Net address**
- Parameter 2: Group
- Parameter 3: Variation
- Parameter 4: Start index
- Parameter 5: Stop index

COUNTER FREEZE

- Freeze counter - with response (Function Code 7)
  Command: FREEZE
  Parameter 1: **Net address**
- Freeze counter - without response (Function Code 8)
  Command: FREEZE_NR
  Parameter 1: Net address
- Freeze and reset counter - with response (Function Code 9)
  Command: FREEZE_CLEAR
  Parameter 1: Net address
- Freeze and reset counter - without response (Function Code 10)
  Command: FREEZE_CLEAR_NR
  Parameter 1: **Net address**

ASSIGN CLASS

Sends an Assign Class query to the outstation (Function Code 22)

Assign to Class 0, deactivates the events. After a successful call, the driver automatically sends a Read Request for Event Class 123.

- Assign class of a complete Object Group (Binary Input for example):
  - Command: ASSIGN_CLASS
  - Parameter 1: **Net address**
  - Parameter 2: Object Group
  - Parameter 3: Class (1 = Class0, 2 = Event Class1, 3 = Event Class2, 4 = Event Class3)
- Class of individual Points of an Object Group (for example: Assign Analog Input with the Point Index 4 to 10):
  - Command: ASSIGN_CLASS
  - Parameter 1: **Net address**
  - Parameter 2: Object Group
  - Parameter 3: Class (1 = Class0, 2 = Event Class1, 3 = Event Class2, 4 = Event Class3)
  - Parameter 4: Start-Index
  - Parameter 5: Stop-Index

ACTIVATE/DEACTIVATE UNSOLICITED RESPONSES

Command: ENABLE_UNSOL

- Parameter 1: **Net address**
- Parameter 2: 1 ... Enable / 0 ... Disable
- Parameter 3: Event-Class 1 (1 ... include, 0 ... exclude)
- Parameter 4: Event-Class 2 (1 ... include, 0 ... exclude)
- Parameter 5: Event-Class 3 (1 ... include, 0 ... exclude)

INSTIGATE CLASS POLL

Command: CLASS_POLL

- Parameter 1: **Net address**
- Parameter 1: Class-0 (1 ... include , 0 ... exclude)
- Parameter 2: Event-Class 1 (1 ... include, 0 ... exclude)
- Parameter 3: Event-Class 2 (1 ... include, 0 ... exclude)
- Parameter 4: Event-Class 3 (1 ... include, 0 ... exclude)

ACTIVATE/DEACTIVATE CLASS POLL

Command: CYCLIC_POLL

- Parameter 1: **Net address**
  The command is rejected if an invalid network address is entered.
- Parameter 2: Polling:
  - 0 - deactivated
  - 1 - activated

COLD RESTART

Sends an Cold Restart query to the outstation (Function Code 13)

The driver separates the connection after a successful response. The driver waits with new queries to the Outstation. The waiting time is the delay time reported by the Outstation (contained in the response). The connection is re-established again after this waiting time.

Command: COLD_RESTART

- Parameter 1: **Net address**

INSTIGATE TIME SYNCHRONIZATION

Time synchronization is carried out in accordance with the method selected in the driver settings (on page 29):

- **Timesync LAN Net address**active
- **Timesync LAN** inactive
  - **Delay measurement (Timesync)** active
  - **Delay measurement (Timesync)** inactive

DELAY MEASUREMENT

- Delay Measurement active:
  If the Delay measurement option is active, the request for **Delay Measurement (FC23)** is sent with the **TIMESYNC** command.
  Command: TIMESYNC
  Parameter 1: **Net address**

TRIGGER TCP REDUNDANCY SWITCHING

Attention: Only has an effect if there is precisely one active connection (statistics variable tcp state, offset 206, value 3). If there is no connection currently active because the connection was disconnected due to the idle disconnect, the command is ignored. In this case, the next attempt to make a connection goes to the last IP address that was used.

Using this command, it is possible to actively switch from the primary IP address to the secondary IP address, or vice versa. The active connection is separated and the respective alternative IP address is used the next time an attempt to establish a connection is made.

COMMAND: RED_SWITCH

- Parameter 1: **Net address**

Attention: In the case of virtual Outstations, the redundancy switching is applicable for all Outstations that share an IP channel.

Attention: It is assumed that the outstations connected via the primary IP address and secondary IP address have the same points configuration and their own channel, or are synchronized with the process.

RESET CHANNEL STATISTICS

Resets the Data-link (phys) and TCP statistics for a communication channel.

Command: RESET_LINK_STAT

- Parameter 1: **Net address**

Attention: In the case of virtual Outstations, the resetting of statistics is applicable for all Outstations that share an IP channel.

RESET OUTSTATION STATISTICS

Command: RESET_OUTSTATION_STAT

- Parameter 1: **Net address**

COMMAND STATUS CODES

Status code	Description
3	Command has been sent
4	Command completed successfully
5	Command could not be processed
6	Command received a negative response
7	Error when executing the command (Timeout, Authentication failed)
8	Execution of the command / file transfer was interrupted.

Status code	Description
10	File Transfer: Timeout occurred
11	File Transfer: Saving of local file failed
12	File Transfer: Abort failed
13	File Transfer: Cancel, more data received than the reported file size
16	Syntax error
17	Unknown command
18	**Net address** not existing
19	File transfer: is already being carried out
20	File Transfer: Directory error or directory not configured
21	File Transfer: Abort failed, no file transfer active

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	Type	Field size	Comment
KANALNAME	Char	128	Variable name. The length can be limited using the **MAX_LAENGE** entry in the **project.ini** file.
KANAL_R	C	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 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	C	128	Identification. The length can be limited using the **MAX_LAENGE** entry in the **project.ini** file.
EINHEIT	C	11	Technical unit
DATENART	C	3	Data type (e.g. bit, byte, word, ...) corresponds to the data type.
KANALTYP	C	3	Memory area in the PLC (e.g. marker area, data area, ...) corresponds to the driver object type.

Identification	Type	Field size	Comment
HWKANAL	Num	3	Net 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 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	N	2	Driver-specific ID number of the primitive object comprises TREIBER-OBJEKTTYP and DATENTYP
SIGMIN	Float	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	N	7	Only for compatibility reasons
HDRATE	F	19	HD update rate for historical values (in sec, one decimal possible)

Identification	Typ e	Field size	Comment
HDTIEFE	N	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	N	16	Priority of the variable
REAMATRIZE	C	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	C	128	Resources label. Free string for export and display in lists. The length can be limited using the MAX_LAENGE entry in **project.ini**.
ADJWVBA	R	1	Non-linear value adaption: *0*: Non-linear value adaption is used *1*: Non-linear value adaption is not used
ADJZENON	C	128	Linked VBA macro for reading the variable value for non-linear value adjustment.
ADJWVBA	C	128	ed VBA macro for writing the variable value for non-linear value adjustment.
ZWREMA	N	16	Linked counter REMA.
MAXGRAD	N	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	Type	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 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
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	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

Identification	Type	Field size	Comment
MIN_MAX1	C	3	Minimum, Maximum
FARBE1	N	10	Color as Windows coding
GRENZTXT1	C	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

RULES FOR THE DNP3_TG ONLINE IMPORT

For Online-Import:

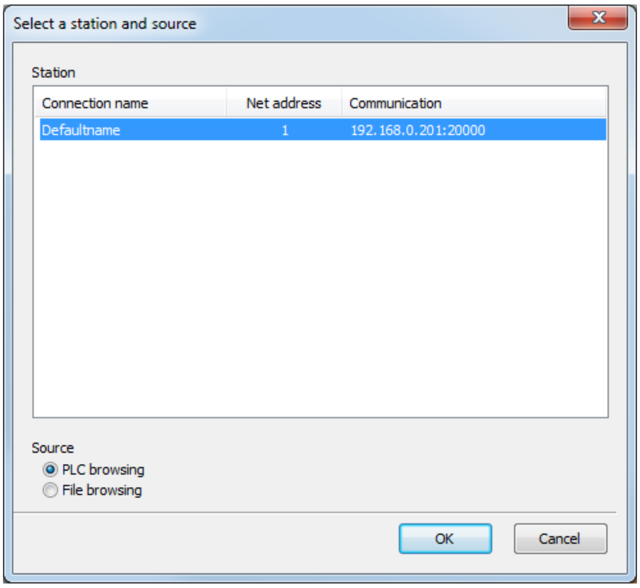
- The response to an Integritypoll is evaluated.
- The variable name is created from the network address, connection name, group number and index.
- The identification contains the network address, index and description of the object type.
- Ensure that Runtime is not active when you start an online import. The Outstation supports, under certain circumstances, only one Master or only one connection from the same computer.
- Ensure that the response Timeout in the driver configuration is set higher accordingly if you are using an Outstation with a large Point Database and a slow (serial) connection.

ONLINE IMPORT IN GENERAL

To import variables online from the PLC:

1. Select the driver.
2. Select **Import variables from driver** in the toolbar or in the context menu

3. The dialog for the import is opened:



Select a station and source

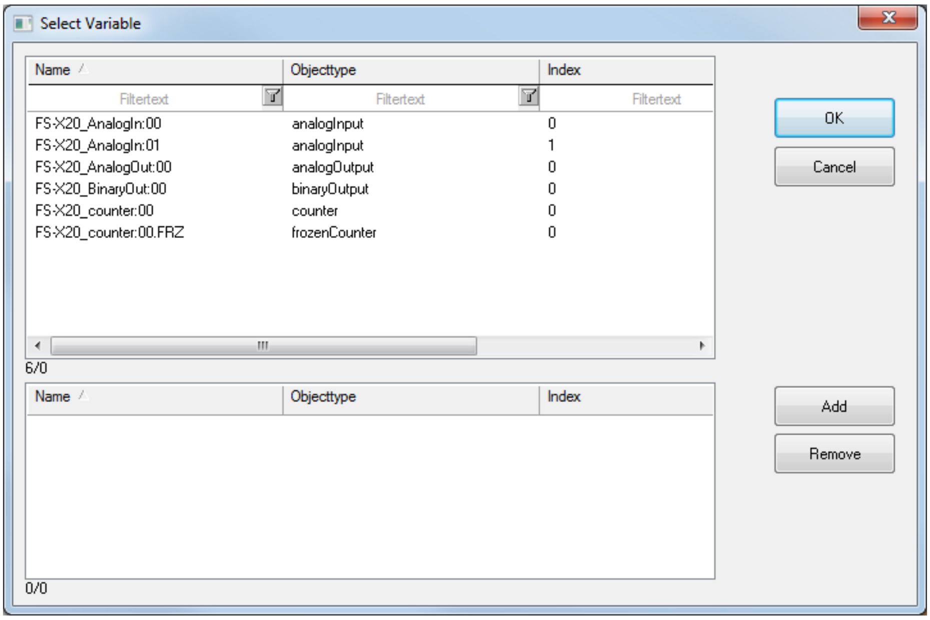
Station

Connection name	Net address	Communication
Defaultname	1	192.168.0.201:20000

Source

● PLC browsing
○ File browsing

4. Select the desired connection.
5. Select **Selection of PLC**.
6. Confirm the selection by clicking **OK**.
7. The dialog for variable selection is opened:



Select Variable

Name	Objecttype	Index
Filtertext	Filtertext	Filtertext
FS-X20_AnalogIn:00	analogInput	0
FS-X20_AnalogIn:01	analogInput	1
FS-X20_AnalogOut:00	analogOutput	0
FS-X20_BinaryOut:00	binaryOutput	0
FS-X20_counter:00	counter	0
FS-X20_counter:00.FRZ	frozenCounter	0

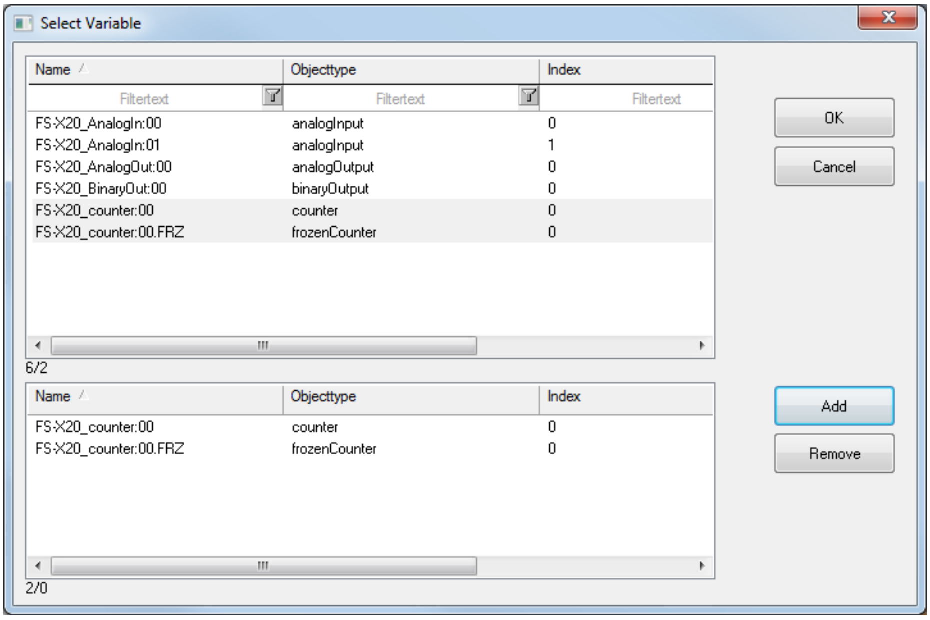
6/0

Name	Objecttype	Index

0/0

8. Select the desired variables (multiple selection is possible).

9. Add selected variables via click on button **Add** to the list of the variables to be imported.



Name	Objecttype	Index
Filtertext	Filtertext	Filtertext
FS-X20_AnalogIn:00	analogInput	0
FS-X20_AnalogIn:01	analogInput	1
FS-X20_AnalogOut:00	analogOutput	0
FS-X20_BinaryOut:00	binaryOutput	0
FS-X20_counter:00	counter	0
FS-X20_counter:00.FRZ	frozenCounter	0

6/2

Name	Objecttype	Index
FS-X20_counter:00	counter	0
FS-X20_counter:00.FRZ	frozenCounter	0

2/0

10. You can also deselect variables again by clicking on **Remove**.

11. Start the import by clicking on the **OK** button.

The selected variables are generated automatically during import in the zenon project and are assigned the selected driver. The net adress of the variables is configured according to the selected station in the driver configuration.

7.4.4 Offline import

The driver supports the offline import of variables from a **DNP3 XML Device Profile** file for the following versions:

- 2.07 (January 2012)
- 2.08 (July 2012)

RULES FOR THE DNP3_TG OFFLINE IMPORT

For Offline-Import:

- Variable definition must contain a name.

  The variable name is made up of a composite name comprising the XML device profile document in accordance with the following scheme:
  devicename_variablename
  It must be ensured that all DNP variables have a unique name in the document, including throughout the group.

- If the definition of the variable in the document contains a field **Description**, this information is stored in the **Identification** of the variable during import. The name of the variable and the variable ID can be changed after import.
- If a variable with the same name already exist in the project, you receive an error during a new import. The variable is not overwritten or merged. This error message can also be displayed:
  - if the document does not use unique names in the **XML Device Profile**
  - if the device name in the document is identical with already imported variable
- Only variables from the **XML Device Profile** document that are supported by the driver are offered for import.
- **Frozen Counters** are not explicitly present in the **XML Device Profile** file. If however the value for **frozenCounterExists** is set to *True* for a counter (*1*), the option to import variables also for **Frozen Counter** is offered.

VARIABLES THAT CANNOT BE IMPORTED

The following variables are not imported and must be created manually:

- *Device attributes*
- Control variables for class polls, classless reading and *Cold Restart*
- *Binary inputs double*

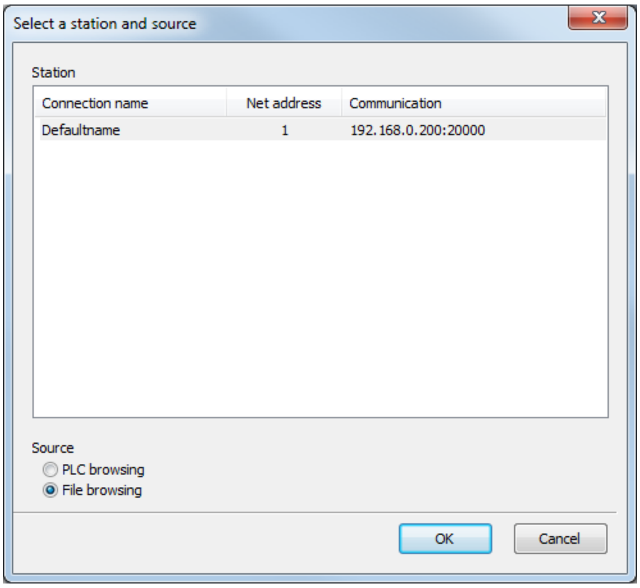
Note the correct **Net address** when creating variables manually

OFFLINE IMPORT IN GENERAL

To import variables from a **DNP3 XML Device Profile** file:

1. Select the driver.
2. Select **Import variables from driver** in the toolbar or in the context menu

3. The dialog for the import is opened:



Select a station and source

Station

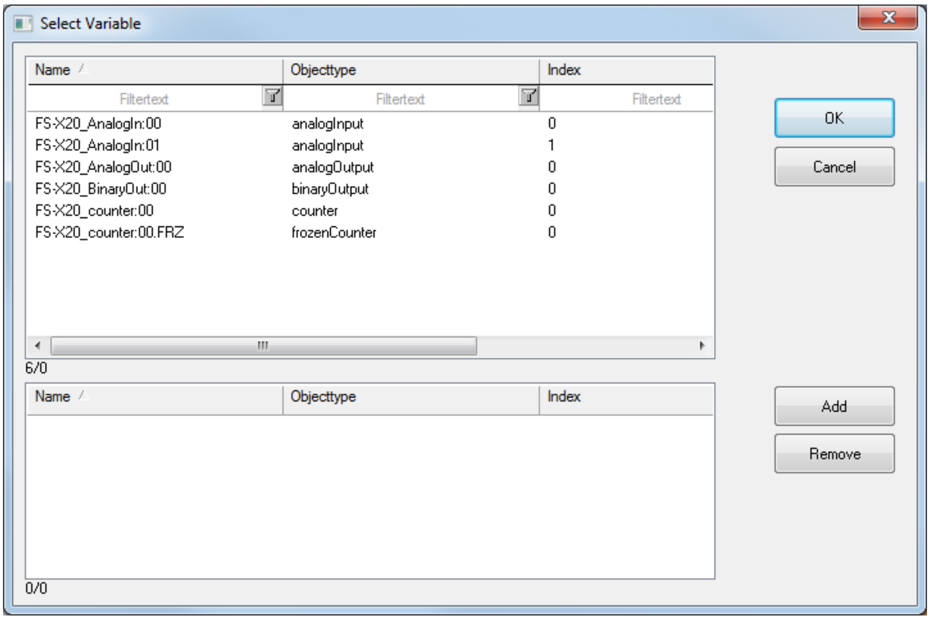
Connection name	Net address	Communication
Defaultname	1	192.168.0.200:20000

Source

○ PLC browsing

● File browsing

4. Select the desired connection.
5. Select **Import from file**.
6. The dialog for file selection will be opened.
7. Select the desired file and confirm this selection by clicking **OK**.
8. The dialog for variable selection is opened.



Select Variable

Name	Objecttype	Index
Filtertext	Filtertext	Filtertext
FS-X20_AnalogIn:00	analogInput	0
FS-X20_AnalogIn:01	analogInput	1
FS-X20_AnalogOut:00	analogOutput	0
FS-X20_BinaryOut:00	binaryOutput	0
FS-X20_counter:00	counter	0
FS-X20_counter:00.FRZ	frozenCounter	0

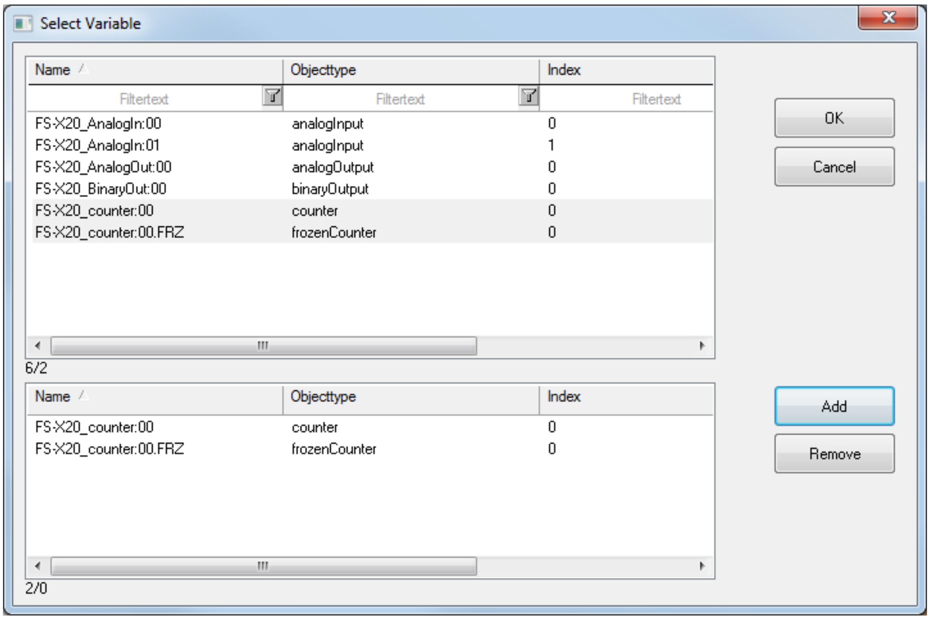
6/0

Name	Objecttype	Index

0/0

9. Select the desired variables (multiple selection is possible).

10. Add selected variables via click on button **Add** to the list of the variables to be imported.



Name	Objecttype	Index
Filtertext	Filtertext	Filtertext
FS-X20_AnalogIn:00	analogInput	0
FS-X20_AnalogIn:01	analogInput	1
FS-X20_AnalogOut:00	analogOutput	0
FS-X20_BinaryOut:00	binaryOutput	0
FS-X20_counter:00	counter	0
FS-X20_counter:00.FRZ	frozenCounter	0

6/2

Name	Objecttype	Index
FS-X20_counter:00	counter	0
FS-X20_counter:00.FRZ	frozenCounter	0

2/0

11. You can also deselect variables again by clicking on **Remove**.
12. Start the import by clicking on the **OK** button.

The selected variables are generated automatically during import in the zenon project and are assigned the selected driver. The net adress of the variables is configured according to the selected station in the driver configuration.

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.

INFORMATION

Name from import	Type	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

Name from import	Type	Offset	Description
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 terminated.
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.
ConnectionStates	*STRING*	61	Internal connection status of the driver to the PLC. Connection statuses: *0:* Connection OK *1:* Connection failure *2:* Connection simulated Formating: *<Netzadresse>:<Verbindungszustand>;…;…;* 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.

Name from import	Type	Offset	Description
			Otherwise there is no communication with the corresponding controller.

CONFIGURATION

Name from import	Type	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.
Baudrate	*UDINT*	10	Baud rate of the serial interface.
Parity	*SINT*	11	Parity of the serial interface

Name from import	Type	Offset	Description
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	Type	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.

Name from import	Type	Offset	Description
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	Type	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.

Name from import	Type	Offse t	Description
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:

Function	Description
Blockwrite	Not supported.
Redundancy	If the Outstation supports several Masters.
RDA	not supported **Sequence Of Events** is supported if activated in the Outstation.
Real time stamping	If the **Variation** is configured accordingly in the Outstation.
Browsen	Online and Offline.
Polling	Polling for event classes, in configurable cycles or controlled manually.
Spontaneous	Yes. Only value changes are transferred.
Number of PLCs	One driver can connect to any number of Outstations. For serial communications, several Outstations can be configured for a serial interface (bus). Mixed operation *serial/TCP* is also

Function	Description
	possible with a driver.

ERROR FILE

The driver supports central logging in the Diagnosis Server (on page 96).

DRIVER VARIABLES

Driver statistics variables are not set by the **DNP3_TG** driver. Communication monitoring is possible using the **INVALID** status bit.

INTEGRITY POLL

After the driver is started, an Integrity Poll is automatically sent by the driver to the Outstation (**Read Request Group** *60*, **Variation** *2,3,4* and *0*, **Range** *all*). The values received are are available in Runtime accordingly. The Outstation, under certain circumstances, sends a variation as a response to the Integrity Poll that is different to that of a normal value change. (with/without time stamp, with/without flags.)

An Integrity Poll can also be instigated explicitly in Runtime by means of a *class poll* type control variable. If the driver is stopped and started using the driver command function, this also triggers an Integrity Poll.

INTERNAL INDICATIONS (IIN)

The Outstation can inform the master of its status via **Internal Indication** status bits. These flags can be monitored using variables of the **IIN** driver object group.
The DNP3_TG driver evaluates **Internal Indication** bits as follows:

- IIN 1.1 CLASS_1_EVENTS:
  The Outstation sets this bit if the event buffer for class 1 contains other DNP objects that are not included in the current answer. The DNP3_TG Master reacts to this bit in that a read query for group 60, variation 2 is immediately sent to the Outstation. If the Outstation frequently sets this status bit, this can lead to a higher read cycle that that defined in the driver configuration.
- IIN 1.2 CLASS_2_EVENTS:
  The Outstation sets this bit if the event buffer for class 2 events contains further DNP objects that are not included in the current response. The DNP3_TG Master reacts to this bit in that a read query for group 60, variation 3 is immediately sent to the Outstation. If the Outstation frequently sets this status bit, this can lead to a higher read cycle that that defined in the driver configuration.
- IIN 1.3 CLASS_3_EVENTS:
  The Outstation sets this bit if the event buffer for class 3 contains other DNP objects that are not included in the current answer. The DNP3_TG Master reacts to this bit in that a read query

for group 60, variation 4 is immediately sent to the Outstation. If the Outstation frequently sets this status bit, this can lead to a higher read cycle that that defined in the driver configuration.

- IIN 1.4 NEED_TIME:
  The Outstation sets this bit if a time synchronization is demanded by the Master. The DNP3_TG Master reacts immediately and responds to the Outstation depending on the selected time synchronization option in the driver configuration.
- IIN 2.3 BUFFER_OVERFLOW:
  The Outstation sets this bit if there has been an overflow of the event buffer. The DNP3_TG Master reacts to this with an automatic Integrity Poll.
- IIN 1.7 DEVICE_RESTART:
  Is set by the Outstation in the event of a restart. The connection to the Outstation is reinitialized in this case. It is reset by the DNP3_TG Master in this process.

If the **No class poll on IIN1.1, IIN1.2, IIN1.3** option is set to active in the driver configuration for the station, the driver ignores the **Internal Indication** flags *IIN1.1*, *IIN1.2* and *IIN1.3*. The driver does not send an automatic read query for group 60, Variation 2, 3, 4. This option can be activated in order to circumvent compatibility problems if the Outstation of one of these flags is not reset in time, which leads to the driver only sending Event Polls.

In general, the Outstation sets these flags in order to command the Master to read again, because other data is available and this could therefore possibly avoid a buffer overflow in the Outstation. Note this if you activate this option for compatibility reasons, and ensure that the driver polls the Outstation for events at regular intervals.

The **No time sync on IIN1.4** and **No integrity poll on IIN2.3** options can be used to switch off automation for time synchronization and the Integrity Poll.

DNP3 OBJECT FLAGS MAPPING

The **ONLINE** and **COMM_LOST** DNP object flags are evaluated by the DNP3_TG driver. With **ONLINE** = *false* or **COMM_LOST** = *true*, the *INVALID* bit for the variable is set in Runtime. In the opposite scenario, either the *SPONTAN* bit of the *GA* bit is set. The *GA* bit is set for values from non event variations; the *SPONTAN* bit is set for values from Event-Variations.

Note that the variation configured in the outstation determines whether objects are sent with or without flags for an object group.

In addition, the following flags/IIN bits listed are transferred to the variable status bits:

DNP3 flag / IIN	Status bit
Restart	*NT* Bit
Remote forced	*SB* Bit
Local forced	*SB* Bit

DNP3 flag / IIN	Status bit
Rollover	*OV* Bit
IIN.1.4 Need time	*Time-inval* Bit

TIME SYNCHRONIZATION

The DNP3_TG driver supports time synchronization of Outstations with the time of the Master station. If the Outstation reports a need for time synchronization by means of an **Internal Indication** flag 1.4 an, the DNP3_NG driver sends the current system time in accordance with the options for UTC/local time and the LAN time synchronization. The time synchronization can also be triggered manually by means of an explicit command.

The Outstation can synchronize the time with another source, e.g. via GPS receiver. In this case the possibility for time synchronization is not used. In this case, ensure that the Runtime computer with the DNP3_TG Master station is synchronized with the same time source.

The DNP3 protocol does not allow the Master to synchronize its own time with the time of the Outstation.

SELECT AND CANCEL (COMMAND PROCESSING)

The DNP3_TG driver immediately responds positively to a Select and Cancel with a corresponding COT. The Execute is carried out after a Select with the COT amended. Automatic Select and Execute are given preferential treatment in the process.
Furthermore, Select, Cancel and Execute have an additional status bit. As a result of this, orderly Runtime monitoring of the configured routing is possible.

SELECT BEFORE OPERATE - DIRECT OPERATE

The DNP3_TG driver uses a choice of *Direct Operate* (Default), *Direct Operate NoAck* or *Select Before Operate* for the writing of *Binary Output Status* variables or *Analog Output* variables. Configuration is carried out using the driver-specific **Command Mode** variable property.

Ensure that the **Select Before Operate** property remains inactive for the variable! This property changes the behavior of the two-stage command processing and is not compatible with the DNP3 standard. If this property is active however:

- Block the command processing for further commands if *Direct Operate* is set for the variable
- Carry out the command at the first level and also for the second level if *Auto-SBO* is set for the variable

Note: With the DNP3 32 driver, a **Select Before Operate** can only be carried out via this option. This is however not compliant with the rules for **Select Before Operate** defined in the DNP3 standard.

HYSTERESIS

The driver supports hysteresis for spontaneous values (unsolicited responses). Hysteresis is not taken into account by:

- Values that are received as a response to a read request for an event class
- Variables that are explicitly read as classless variables

If an identical value, however with a more recent time stamp is received, then this value is sent by the driver as a new value in Runtime.

8.1 DNP3 File Transfer

The DNP3_TG driver supports file transfer in accordance with the IEEE1815 standard. The following functions are supported:

- Reading file or directory information from the outstation.
- Reading directory content from the outstation.
- Reading file from the outstation and saving locally.
- Removing file from the outstation.
- Writing local file to the outstation.
- Canceling an ongoing file transfer

It is recommended that a separate command string variable is used for the transferring files with File transfer. In addition to this command string variable, a *File Transfer Result* string variable of the corresponding length must be created for each net address in order to show the result of the queries to read file information or directory information.

File transfer actions are executed asynchronously. The driver can send a File transfer-command to the outstation if a cyclical poll is not called up or if a command is not currently being sent. Only one individual File transfer action can be active. No further queries are sent to the outstation until the initial response has been obtained from the outstation. The normal *Response Timeout* applies here. If there is an initial response with *Null Response*, the normal polling cycle continues. New commands, such as the explicit reading of events, can also be sent.

For each connection, the directory for reading a file from the outstation and the directory for sending a file to the outstation must be configured in order for file transfer to be able to be used. Only local directories can be entered. The path cannot be a relative path. The directories must be present on the computer on which the Runtime runs.

Information

If Runtime is to be used with Windows CE, the absolute path c:\temp may not exist under certain circumstances but must be addressed using \temp. Because the driver configuration requires the entry of an absolute path, the path must be entered into the driver configuration manually in this case.

The attendant *Timeout* for each connection can also be configured. This *Timeout* ensures that File transfer actions run asynchronously, and are also ended automatically if the outstation does not provide a response in time. This ensures that a new File transfer action can also be started again.

NULL RESPONSE

In accordance with IEEE1815, the outstation need not send a response in which the requested data is contained. It must also respond initially with *Null response* and then send the response with the required data later as an event. The outstation can send the response with the required data as an event in different types:

- As a response to an *Event Class Poll* (Class 1, Class 2 or Class 3)
- As a response to a *Read Request* for G70v4 or G70v7
- As a response to a *Read Request* for G70v0 (not officially supported in accordance with IEEE1815)
- As an *Unsolicited Response*

For a master that starts a File transfer action for an outstation - that initially provides a *Null Response* - this means that the master must explicitly query the outstation about these events after the initial *Null Response*.

The **DNP3_TG driver** does not do this query for these events automatically, however the **DNP3_TG driver** supports all the above-mentioned types and can explicitly request these regardless of the configured polling cycle using commands from the general command string variable. The person configuring the project must in this case configure mechanisms that send cyclically-corresponding requests to the outstation whilst waiting until either the response is received and until the File transfer action is running or the *File transfer timeout* has expired.

One possibility of implementing this is with the help of zenon Logic, where a command string variable can be monitored. If no response with data has been received after a File transfer command, the commands to read an event can be sent cyclically using the general command string variable. This is carried out until either a response is received and File transfer is completed or the query runs into the *File transfer timeout* and fails.

REQUEST FILE OR DIRECTORY INFORMATION

The DNP3_TG master sends a file transfer query to the outstation using the *GET_FILE_INFO* command. The response from the outstation is transferred to a "File Transfer Result" driver object variable that must be manually created with the corresponding net address. The result for the command is transferred to the command string variable.

Command: *GET_FILE_INFO*

- Parameter 1: **Net address**
- Parameter 2: Remote path

Remote path:

- The remote path must be listed between "".
- The "" are removed during a query to the outstation.
- The remote path must contain at least one character and must not consist of spaces.
- The remote path can contain spaces.

If the query is successful, the result is formatted and transferred to the "*File Transfer Result*" string variable:

Result: **<Type>;<Name>;<Size>;<Date and time>**

The type contains either "*F*" for a file or "*D*" for a Directory.

Example

Three examples for the value of the "**File Transfer Result**" string variable as a result of this action:

- F;hallo.txt;234;2015-03-20T13:58:34:22.387Z
  File with a size of 234 bytes
- D;Dir1;3;2015-03-20T13:58:34:22.387Z
  Directory with 3 objects (files or subdirectories)
- D;/;0;2015-03-20T13:58:34:22.387Z
  (empty root directory)

REQUEST CONTENTS OF DIRECTORY

The *READ_DIRECTORY* command is used so that the **DNP3_TG Master** sends a *File Open* query to the outstation and waits for the response. The content is then read using the *File Handle* received for the directory. In doing so, the **DNP3_TG driver** can also send several *Read Requests*. The response is transferred to a "**File Transfer Result**" driver object variable that must be manually created with the corresponding net address in the project. The result of the command is transferred to the command string variable.

Command: *READ_DIRECTORY*

- Parameter 1: **Net address**
- Parameter 2: Remote path

Remote path:

- The remote path must be listed between "".
- The "" are removed during a query to the outstation.
- The remote path must contain at least one character and must not consist of spaces.
- The remote path can contain spaces.

If the query is successful, the response is transferred to the "*File Transfer Result*" string variable:

Result:

<Type>;<Name>;<Size>;<Date and time>CR LF

..

<Type>;<Name>;<Size>;<Date and time>

The type contains either "*F*" for a file or "*D*" for a Directory.

The information is shown in its own line for each file and each directory. Each line is separated by a **Carriage Return / Line Feed**.

READ FILE

The *READ_FILE*command is used so that the **DNP3_TG Master** sends a *File Open* query to the outstation and waits for the response. Then the *File Handle* received is used to read the file in one or more *Read Requests* and save it in a configured directory.

There are two versions of this command:

Command: READ_FILE

- Parameter 1: **Net address**
- Parameter 2: Remote path

Command: READ_FILE

- Parameter 1: **Net address**
- Parameter 2: Remote path
- Parameter 3: local path

With the version with two parameters, only the remote path and the file name are specified. The local file name and the path are identical to the remote file name and path.

With the version with three parameters, the path can be explicitly specified for the local file.

Attention: For the parameter 3 (local path) the characters "..", "\\", and ":" are not permitted. This prevents files being saved locally outside the configured directory.

Information

If, for example, a ":" or a "\" is included in the remote path, the version with three parameters must be used.

Example: *READ_FILE 1 "c:\temp\text1.txt" "text1.txt"*

DELETE FILE

The **DNP3_TG Master** uses the *DELETE_FILE* command to send a query to the outstation to delete a file and wait for the response. The result of the command is transferred to the command string variable.

Command: DELETE_FILE

- Parameter 1: **Net address**
- Parameter 2: Remote path

Remote path:

- The remote path must be listed between "".
- The "" are removed during a query to the outstation.
- The remote path must contain at least one character and must not consist of spaces.
- The remote path can contain spaces.

WRITE FILE

The *WRITE_FILE*command is used so that the **DNP3_TG Master** sends a *File Open* query to the outstation and waits for the response. The driver then uses the *File Handle* received to write the file to one or more *Write Requests* in the remote outstation. There are two versions of the command:

Command: WRITE_FILE

- Parameter 1: **Net address**
- Parameter 2: local path
- Parameter 3: Outstation file name

Command: WRITE_FILE

- Parameter 1: **Net address**
- Parameter 2: local path
- Parameter 3: Outstation file name
- Parameter 4: Authorization
- Parameter 5: Timeout

With the version with two parameters, the *World Read, Group Read, Owner Read*" authorization is used and the default setting for the file transfer timeout.

With the version with five parameters, the authorization can be set using the table below and a timeout must also be defined in seconds.

World Execute: 1

World Write: 2

World Read: 4

Group Execute: 8

Group Write: 16

Group Read: 32

Owner Execute: 64

Owner Write: 128

Owner Read: 256

The numbers for the authorization (parameter 4) can be added in order to set the corresponding authorization. For example, value 438 for read/write for World, Group and Owner. The maximum value where all bits are set for the authorization is thus 511.

CANCEL FILE TRANSFER

Under certain circumstances, it may be necessary to cancel an ongoing *File transfer* action. Before a planned redundancy switching for example. With the cancellation, the outstation should unlock the *File Handle* again and be ready for the next *File transfer* action. It can make sense to use a separate command variable for the abort command. The result of the abort command is thus immediately portrayed by the "Abort" command string variable (successful, status 4). The commando string variable for the ongoing file transfer contains the value "8".

Command: FILE_TRANSFER_ABORT

- Parameter 1: **Net address**

If the *File transfer* has already been canceled for a different reason, for example because the connection was lost and the number of *Retries* was exceeded, the *File transfer* can no longer be canceled explicitly.

FILE-TRANSFER-SPECIFIC STATUS CODES

The command string variable contains the result of the command execution as a status code. In addition to the general commando status codes, there are specific status codes for *File transfer*, which portray the response from the outstation in the event of an error. 1000 is added to the *File transfer* status code of the response from the outstation that is defined in the standard.

Parameter	Description
1001	Permission Denied
1003	File not found
1004	File locked
1005	too many open handles
1006	invalid handle or handle timed out
1007	write block size error
1008	communication failed

Parameter	Description
1009	abort failed
1016	file not open
1017	file handle expired
1018	buffer overrun on writing a file
1019	general error with file transfer
1020	block / sequence error during file transfer
1255	other error during file transfer occurred

COMMAND STATUS CODES

Status code	Description
3	Command has been sent
4	Command completed successfully
5	Command could not be processed
6	Command received a negative response
7	Error when executing the command (Timeout, Authentication failed)
8	Execution of the command / file transfer was interrupted.
10	File Transfer: Timeout occurred
11	File Transfer: Saving of local file failed
12	File Transfer: Abort failed
13	File Transfer: Cancel, more data received than the reported file size
16	Syntax error
17	Unknown command
18	**Net address** not existing
19	File transfer: is already being carried out
20	File Transfer: Directory error or directory not configured

Status code	Description
21	File Transfer: Abort failed, no file transfer active

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.

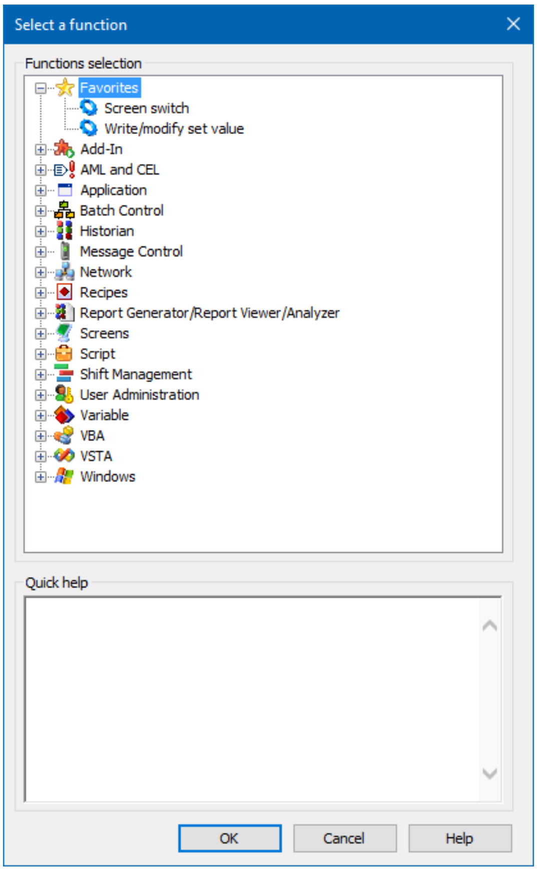
> ⚠ **Attention**
>
> The zenon **Driver commands** function is not identical to driver commands that can be executed in the Runtime with Energy drivers!

CONFIGURATION OF THE FUNCTION

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

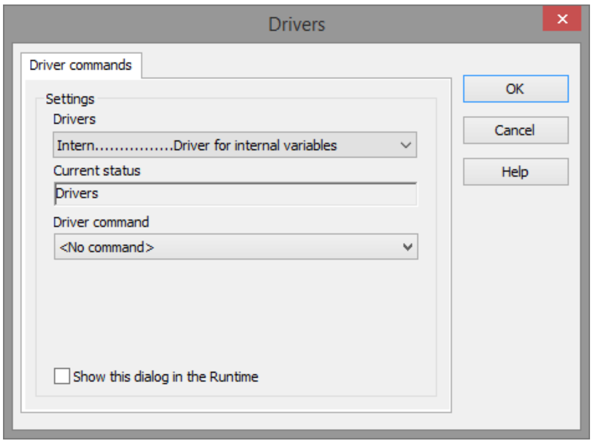
1. Create a new function in the zenon Editor.

The dialog for selecting a function is opened



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

   The dialog for configuration is opened



Drivers: Intern...............Driver for internal variables

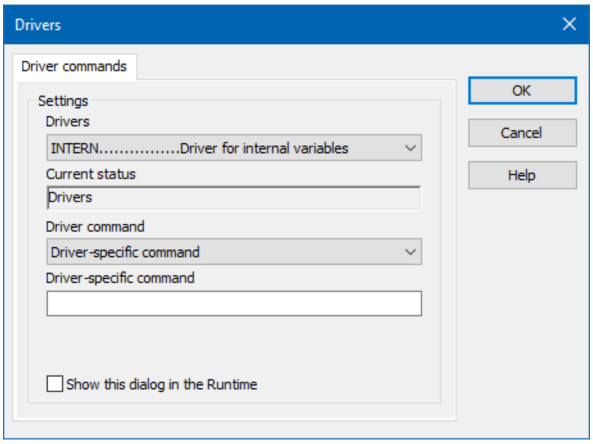
Current status: Drivers

Driver command: <No command>

□ 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



Settings

Drivers: INTERN...............Driver for internal variables

Current status: Drivers

Driver command: Driver-specific command

Driver-specific command:

☐ 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. **Note:** Only available if, for the **driver command** option, the *driver-specific command* has been selected.
Show this dialog in the Runtime	Configuration of whether the configuration can be changed in the Runtime: - *Active*: This dialog is opened in the Runtime before executing the function. The configuration can thus still be changed in the Runtime before execution. - *Inactive*: The Editor configuration is applied in the Runtime when executing the function. Default: *inactive*

CLOSE DIALOG

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

AVAILABLE DRIVER COMMANDS

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

Driver command	Description
<No command>	No command is sent. A command that already exists can thus be removed from a configured function.
Start driver (online mode)	Driver is reinitialized and started. **Note:** If the driver has already been started, it must be stopped. Only then can the driver be re-initialized and started.
Stop driver (offline mode)	Driver is stopped. No new data is accepted. **Note:** If the driver is in offline mode, all variables that were created for this driver receive the status *switched off* (*OFF*; Bit *20*).
Driver in simulation mode	Driver is set into simulation mode. The values of all variables of the driver are simulated by the driver. No values from the connected hardware (e.g. PLC, bus system, ...) are displayed.
Driver in hardware mode	Driver is set into hardware mode. For the variables of the driver the values from the connected hardware (e.g. PLC, bus system, ...) are displayed.
Driver-specific command	Entry of a driver-specific command. Opens input field in order to enter a command.
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)

Driver command	Description
	Opens the input fields for the hardware address and for the telephone number.
Disconnect from modem	Terminate connection (for modem drivers)
Driver in counting simulation mode	Driver is set into counting simulation mode. All values are initialized with *0* and incremented in the set update time by *1* each time up to the maximum value and then start at *0* again.
Driver in static simulation mode	No communication to the controller is established. All values are initialized with *0*.
Driver in programmed simulation mode	The values are calculated by a freely-programmable simulation project. The simulation project is created with the help of the zenon Logic Workbench and runs in the zenon Logic Runtime.

DRIVER COMMAND FUNCTION IN THE NETWORK

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

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

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

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 Check list

Questions and hints for fault isolation:

GENERAL TROUBLESHOOTING

- Is the PLC connected to the power supply?
- Analysis with the Diagnosis Viewer (on page 96):
  -> Which messages are displayed?
- Are the participants available in the **TCP/IP** network?
- Can the PLC be reached via the *Ping* command?

  Ping: **Open command line -> ping <IP address > (e.g.: ping 192.168.0.100) -> Press the Enter key**.

  Do you receive an answer with a time or a timeout?
- Can the PLC be reached at the respective port via *TELNET*?

  Telnet: **Command line: enter: telent <IP address port number> (e. g. telnet 192,168,0,100 20000) -> press the enter key**.

  If the monitor turns black and the cursor blinks, a connection could be established.
- Analysis by using a network monitoring program (Sniffer, e.g. Wireshark, Microsoft Network Monitor / Microsoft Message Analyzer)
- Are you using the correct cable which is recommended by the manufacturer for the connection between the PLC and the PC?
- Did you select the right COM port?
- Do the communication parameters match (Baud rate, parity, start/stop bits,...)?
- Is the COM port blocked by another application?
- Did you configure the Net address in the address properties of the variable correctly?
  - Does the addressing match with the configuration in the driver dialog?
  - Does the net address match the address of the target station?
- Did you use the right object type for the variable?
  Example: Driver variables based on driver object type **Communication details** are purely statistics variables. They do not communicate with the PLC.
  You can find detailed information on this in the Communication details (Driver variables) (on page 74) chapter.
- Does the offset addressing of the variable match the one in the PLC?

SOME VARIABLES REPORT INVALID.

- INVALID bits always refer to a net address.
- At least one variable of the net address is faulty.

VALUES ARE NOT DISPLAYED, NUMERIC VALUES REMAIN EMPTY

Under circumstances, no answer can be received from the PLC for a reading-request.

Driver is not working. Check the:

- Installation of zenon
- the driver installation
- The installation of all components
  -> Pay attention to error messages during the start of the Runtime.

VARIABLES ARE DISPLAYED WITH A BLUE DOT

The communication in the network is faulty:

- With a network project:
  Is the network project also running on the server?
- With a stand-alone project or a network project which is also running on the server:
  Deactivate the property Only read from Standby Server in node Driver connection/Addressing.

VALUES ARE DISPLAYED INCORRECTLY

Check the information for the calculation in node Value calculation of the variable properties.

Check the configuration of the Outstation, if the desired variation is sent (p. e.: **Float** with decimal places).

VALUES ARE NOT DISPLAYED PROMPTLY

Check in the Outstation, if events are generated for the selected values. Also check in which class they are generated. Check the polling of this event class in the driver configuration.

Check the configuration of **unsolicited responses** in the outstation.

THE TIME STAMP OF THE VARIABLE IS INCORRECT.

Check the configuration of the Outstation, to see whether the DNP3 objects are actually sent with a time stamp. (corresponding variation selected with time stamp.)

Check to see if the Outstation uses local time or UTC and set the option in the driver configuration accordingly,

Check the time of the local computer and the time of the Outstation, including settings for the time zone.

DRIVER FAILS OCCASIONALLY

Analysis with the Diagnosis Viewer (on page 96):
-> Which messages are displayed?

Check the timeout of the response time in the driver configuration, especially with outstations with many data points and a slow connection.

With *Secure Authentication*, check whether the **pre-shared Update Key** in the driver and in the Outstation is identical.