



COPADATA
do it your way

zenon driver manual

DNP3_NG

v.7.20





©2015 Ing. Punzenberger COPA-DATA GmbH

All rights reserved.

Distribution and/or reproduction of this document or parts thereof in any form are permitted solely with the written permission of the company COPA-DATA. Technical data is only used for product description and are not guaranteed qualities in the legal sense. Subject to change, technical or otherwise.

Contents

1. Welcome to COPA-DATA help	5
2. DNP3_NG	5
3. Driver history	8
4. DNP3_NG - Data sheet	9
5. Requirements.....	11
5.1 PC	11
6. Configuration	11
6.1 Creating a driver.....	12
6.2 Settings in the driver dialog	13
6.2.1 General	14
6.2.2 Com	17
6.2.3 Connections.....	19
7. Creating variables.....	25
7.1 Creating variables in the Editor.....	26
7.2 Addressing.....	28
7.3 Driver objects and datatypes	30
7.3.1 Driver objects	30
7.3.2 Mapping of the data types	34
7.4 Creating variables by importing	40
7.4.1 XML import.....	40
7.4.2 DBF Import/Export	41
7.4.3 Online import	46
7.4.4 Offline import	48
7.5 Driver variables	51
8. Driver-specific functions	58
9. Driver commands	62

10. Error analysis.....	65
10.1 Analysis tool	65
10.2 Check list	66

1. Welcome to COPA-DATA help

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 (<mailto: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 (<mailto: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 (<mailto:sales@copadata.com>).

2. DNP3_NG

Driver for the protocol in accordance with IEEE1815 Distributed Network Protocol (DNP3).

The driver is the 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	<p>A type of grouping in the outstation.</p> <p>There are Event Classes 1,2, 3 and the Static Class 0. An Event Class is assigned as a rule a buffer, in which value changes are stored for configured object groups with the variation configured in the Outstation .</p> <p>A master can request all value changes of this Event Class of the Outstation by means of a harvest inquiry for for example Class 1.</p> <p>The Static Class 0 has an own task. It redelivers the last value in a harvest inquiry of the master respectively for nearly all Points . In the Outstation can be configured as a rule, which Points belong and/or which Object Group to which Event Class . There is no exact definition, however Class 1 is generally used for important messages, such as binary inputs for alarms.</p>
Event poll	Read query from the master to the outstation , whereby the master requests the outstation to only send the amended values for a certain event class or for all event classes (1, 2 and 3).
Integrity Poll	Read query 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. Object Group 30 stands for a static analog input, for example. A value change for an analog input is assigned to object group 32. Object group 40 serves, for example, for the reading of an analog output, object group 41 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 one Point Number (Offset) per object Group with e.g. Point 12 for object group 30 and Point 12 for Object Group 32 both have the same analog input as a basis, however point 12 for object group 1 is a completely independent binary input. The general term for a value, time or status change is DNP object .
Unsolicited Response	Message from value changes of an event class that is spontaneously sent from the outstation to the master . to do this, the master must however first activate unsolicited responses in the outstation . The outstation must support unsolicited responses and these must be configured for it.
Variation	Format, in which the outstation saves a static value or a value change in class 0 or class 1, 2 or 3. It can be configured in the outstation for each object group or for each point . The variation determines if a static value (class 0) or a value change is an integer or a floating point. Or whether a time stamp is saved or not, or whether object flags are saved or not.

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

DEVICE PROFILE

Device profile is a standard document that describes, which functionality is supported with the DNP3 standard. For the DNP3_NG driver, you find the XML Device profile after zenon installation in the folder `C:\ProgramData\COPA-DATA\zenonxxx\CommunicationProfiles\Dnp3\Driver\DNP3_NG.xml` (xxx corresponds the current zenon version number).

INFORMATION ON THE DNP332, DNP3_NG AND DNP3_TG 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\Edit DriverXML`.
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 tool bar.
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_TG 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 command variable.

DNP332 to DNP3_TG:

- ▶ If you use Select Before Operate with the DNP332 driver for the command, deactivate the **Select Before Operate** property for the variable. Instead, select **Auto-SBO** for **Binary Output** and **Analog Output** type 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 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 a cold restart, create "command" variables from the DNP3_TG driver.
- ▶
- ▶

3. Driver history

Date	Driver version	Change
11/9/2012	3754	Created driver documentation

DRIVER VERSIONING

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

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



Example

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

4. DNP3_NG - Data sheet

General:	
Driver file name	DNP3_NG.exe
Driver name	DNP3 Driver Next Generation
PLC types	DNP3 / IEEE 1815 Outstations
PLC manufacturer	DNP3; GE Harris;

Driver supports:	
Protocol	TCP/IP; DNP3; IEEE Std 1815;
Addressing: Address-based	x
Addressing: Name-based	-
Spontaneous communication	x
Polling communication	x
Online browsing	x
Offline browsing	x
Real-time capable	x
Blockwrite	-
Modem capable	-
Serial logging	x
RDA numerical	-
RDA String	-

Requirements:	
Hardware PC	Serial interface; Standard LAN Adapter
Software PC	-
Hardware PLC	-
Software PLC	-
Requires v-dll	x

Platforms:	
Operating systems	Windows CE 6.0, Embedded Compact 7; Windows 7, 8, 8.1 Server 2008R2, Server 2012, Server 2012R2;
CE platforms	x86; ARM;

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/IP

For dual endpoint, the configured listening socket must be configured in the firewall accordingly.

SOFTWARE

If not already present, copy the driver **DN3_NG.exe** to the zenon program folder and ensure that **DN3_NGV.dll** is also present.

CE

Copy the driver **DN332.dll** to the zenon CE program directory. (The **DN3_NGV.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.



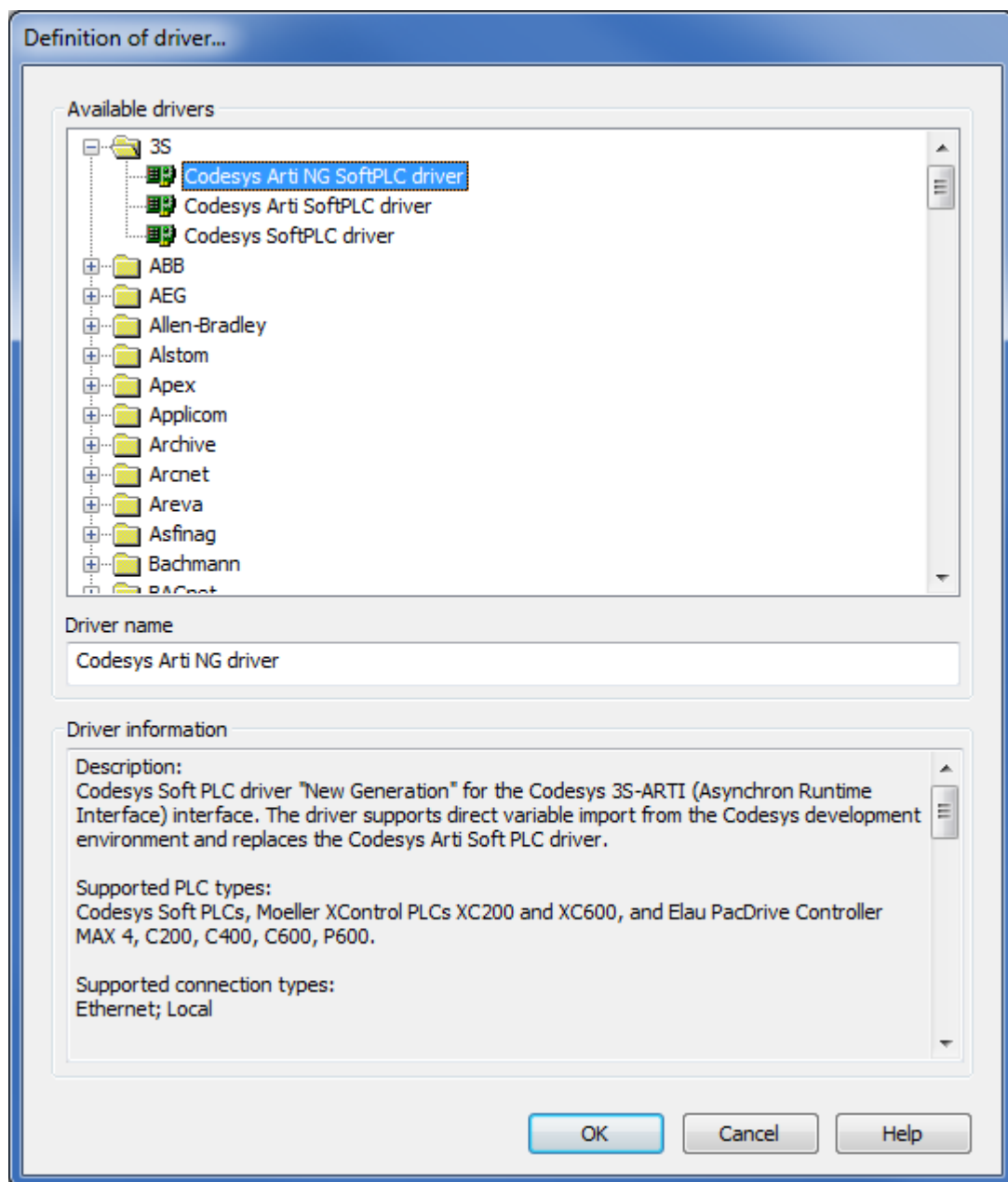
Information

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 order to create a new driver:

1. Right-click on **Driver** in the Project Manager and select **Driver new** in the context menu.
2. In the following dialog the control system offers a list of all available drivers.



3. Select the desired driver and give it a name:
 - The driver name has to be unique, i.e. if one and the same driver is to be used several times in one project, a new name has to be given each time.

- The driver name is part of the file name. Therefore it may only contain characters which are supported by the operating system. Invalid characters are replaced by an underscore (_).
 - **Attention:** This name cannot be changed later on.
4. Confirm the dialog with **OK**. In the following dialog the single configurations of the drivers are defined.

Only the respective required drivers need to be loaded for a project. Later loading of an additional driver is possible without problems.



Information

For new projects and for existing projects which are converted to version 6.21 or higher, the following drivers are created automatically:

- ▶ Internal
- ▶ MathDr32
- ▶ SysDrv.

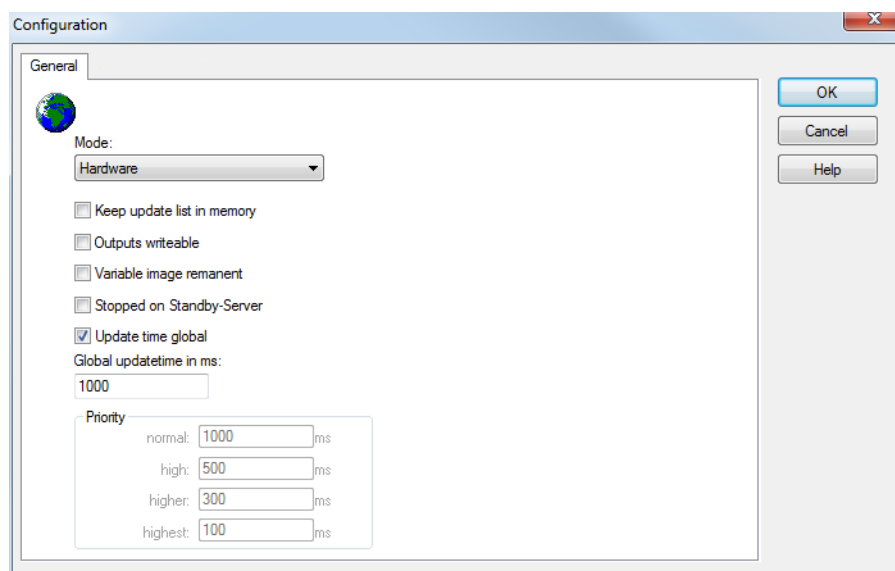
▶

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.



Parameters	Description
Mode	<p>Allows to switch between hardware mode and simulation mode</p> <ul style="list-style-type: none"> ▶ Hardware: <p>A connection to the control is established.</p> ▶ Simulation static <p>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.</p> ▶ Simulation - counting <p>No communication between to the control is established, the values are simulated by the driver. In this modus the driver increments the values within a value range automatically.</p> ▶ Simulation - programmed <p>N communication is established to the PLC. The values are calculated by a freely programmable simulation project. The simulation project is created with the help of the zenon Logic Workbench and runs in a zenon Logic Runtime which is integrated in the driver. For details see chapter Driver simulation (main.chm::/25206.htm).</p>
Keep update list in the memory	<p>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.</p>
Output can be written	<p>Active: Outputs can be written.</p> <p>Inactive: Writing of outputs is prevented.</p> <p>Note: Not available for every driver.</p>
Variable image remanent	<p>This option saves and restores the current value, time stamp and the states of a data point.</p> <p>Fundamental requirement: The variable must have a valid value and time stamp.</p>

	<p>The variable image is saved in mode hardware if:</p> <ul style="list-style-type: none"> ▶ one of the states S_MERKER_1(0) up to S_MERKER8(7), REVISION(9), AUS(20) or ERSATZWERT(27) is active <p>The variable image is always saved if:</p> <ul style="list-style-type: none"> ▶ the variable is of the object type Driver variable ▶ the driver runs in simulation mode. (not programmed simulation) <p>The following states are not restored at the start of the Runtime:</p> <ul style="list-style-type: none"> ▶ SELECT(8) ▶ WR-ACK(40) ▶ WR-SUC(41) <p>The mode Simulation - programmed at the driver start is not a criterion in order to restore the remanent variable image.</p>
Stop on Standby Server	<p>Setting for redundancy at drivers which allow only on communication connection. For this the driver is stopped at the Standby Server and only started at the upgrade.</p> <p>Attention: If this option is active, the gapless archiving is no longer guaranteed.</p> <p>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.</p> <p>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.</p>
Global Update time	<p>Active: The set Global update time in ms is used for all variables in the project. The priority set at the variables is not used.</p> <p>Inactive: The set priorities are used for the individual variables.</p>
Priority	<p>The polling times for the individual priority classes are set here. All variables with the according priority are polled in the set time.</p> <p>The allocation to the variables takes place separately in the settings of the variable properties.</p> <p>The communication of the individual variables are graduated in respect of importance or necessary topicality using the priorities.</p>

Thus the communication load is distributed better.

Attention: Priority classes are not supported by each driver. For example, drivers that communicate spontaneously do not support it.

CLOSE DIALOG

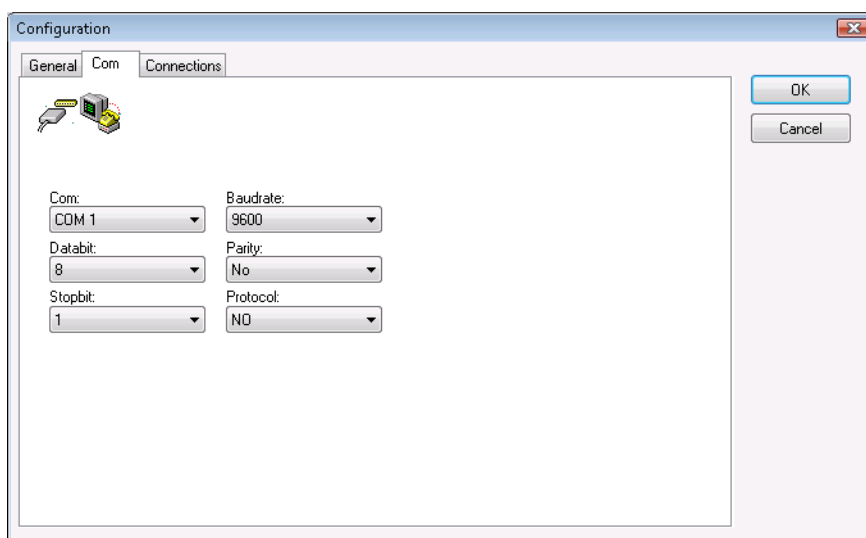
Parameters	Description
OK	Applies all changes in all tabs and closes the dialog.
Cancel	Discards all changes in all tabs and closes the dialog.
Help	Opens online help.

UPDATE TIME FOR CYCLICAL DRIVERS

The following applies for cyclical drivers:

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

6.2.2 Com



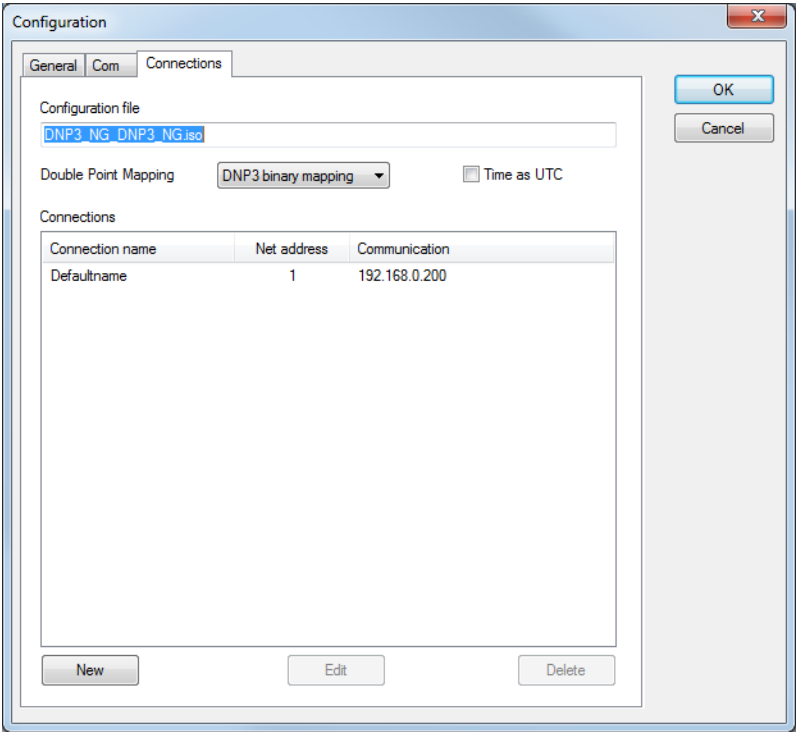
Parameters	Description
Com	Selection Com port. Default: 1
Baud rate	Selection baud rate. Adapting to PLC. Default: 9600
Data bits	Number of data bits. Adapting to PLC. Default: 8
Stop bit	Selection stop bit. Adapting to PLC. Default: 1
Parity	Selection parity. Adapting to PLC. Default: No
Protocol	Selection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



Information

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

6.2.3 Connections



Parameters	Description
Configuration file	Name of the file, in which the connection information is stored. For your information only. Cannot be changed.
Double Point Mapping	Type of double point mapping (on page 25) to an integer value. Select from drop-down list: <ul style="list-style-type: none"> ▸ SCADA default mapping ▸ DNP3 binary mapping ▸ Custom legacy mapping Default: DNP3 binary mapping
Time as UTC	All times from and to the control unit are treated as UTC and not as local time.
Connections	Displays the configured connections.
New	Opens the dialog for creating a new connection (on page 21).
Edit	Opens dialog for editing the selected connection.
Delete	Deletes the selected connection.
OK	Accepts changes in all tabs and closes dialog.
Cancel	Discards all changes and closes the dialog.
Help	Opens online help.



Information

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

CREATE NEW CONNECTION

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

EDIT CONNECTION

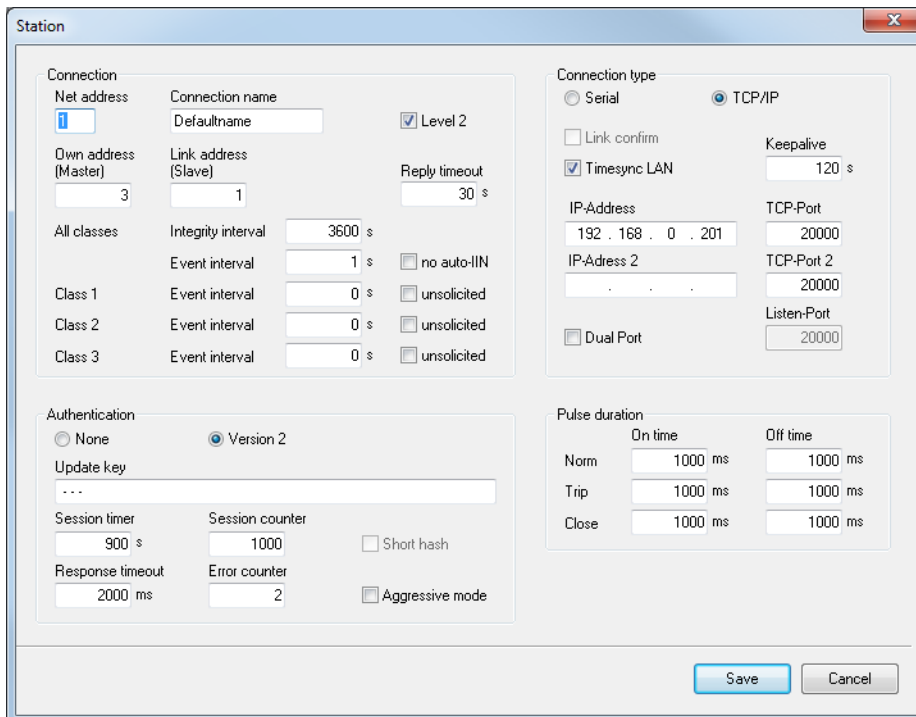
1. select the connection in the connection list
2. click on the button **Edit**
3. change the connection parameters
4. finish with **Save**

DELETE CONNECTION

1. select the connection in the connection list
2. click on the button **Delete**
3. the connection will be removed from the list

Configuration stations

Click on the button **New** in the configuration dialog opens the dialog for the station configuration.



The image shows a 'Station' configuration dialog box with the following sections:

- Connection**
 - Net address:
 - Connection name:
 - ☒ Level 2
 - Own address (Master):
 - Link address (Slave):
 - Reply timeout: s
 - All classes: Integrity interval: s
 - Class 1: Event interval: s ☐ no auto-IIN
 - Class 2: Event interval: s ☐ unsolicited
 - Class 3: Event interval: s ☐ unsolicited
- Connection type**
 - ☐ Serial ☒ TCP/IP
 - ☐ Link confirm
 - ☒ Timesync LAN
 - Keepalive: s
 - IP-Address:
 - TCP-Port:
 - IP-Address 2:
 - TCP-Port 2:
 - ☐ Dual Port
 - Listen-Port:
- Authentication**
 - ☐ None ☒ Version 2
 - Update key:
 - Session timer: s
 - Session counter:
 - ☐ Short hash
 - Response timeout: ms
 - Error counter:
 - ☐ Aggressive mode
- Pulse duration**

	On time	Off time
Norm	<input type="text" value="1000"/> ms	<input type="text" value="1000"/> ms
Trip	<input type="text" value="1000"/> ms	<input type="text" value="1000"/> ms
Close	<input type="text" value="1000"/> ms	<input type="text" value="1000"/> ms

Buttons: **Save** **Cancel**

Parameters	Description
Connection	Connection settings
Net address	Net address of the connection. Value between 0 and 255.
Connection name	Name of connection. Freely selectable.
Level 2	Active: Commands and functions for Levels 2 and level 1 can be used.
Own address (master)	Own address (DNP3 master driver).
Link address (slave)	Link address of the PLC (of the DNP3 slave).
SBO time-out	Time in seconds which is waited for an answer from the PLC after the <code>Select</code> . If the time expires, the <code>Select</code> is considered as denied. Default: 30 s
All classes	Integrationinterval and eventinterval for all classes.
Integrity interval	Integrity poll of the driver in seconds. Default: 3600
Event interval	Prompting the driver for new events in seconds. <ul style="list-style-type: none"> ▶ Value: 0 to 99999 ▶ 0: no Polling Default: 1
no Auto-IIN	Compatibility: Driver ignores Internal Indication Flags IIN1.1, IIN1.2 and IIN 1.3 Make sure to poll for events regularly to avoid bufferoverflows in the Outstation.
Class 1	
Event interval	Specific Intervals for event-polling in seconds for this class <ul style="list-style-type: none"> ▶ Value: 0 to 99999 ▶ 0: no Polling Default: 0
unsolicited	Active: For this class, unsolicited Events are accepted. (See also chapter addressing (on page 28) .)
Class 2	
Event interval	Specific Intervals for event-polling in seconds for this class <ul style="list-style-type: none"> ▶ Value: 0 to 99999 ▶ 0: no Polling Default: 0

unsolicited	Active: For this class, unsolicited Events are accepted. (See also chapter addressing (on page 28) .)
Class 3	
Event interval	<p>Specific Intervals for event-polling in seconds for this class</p> <ul style="list-style-type: none"> ▶ Value: 0 to 99999 ▶ 0: no Polling <p>Default: 0</p>
unsolicited	Active: For this class, unsolicited Events are accepted. (See also chapter addressing (on page 28) .)
Connection type	<p>Type of connection. Possible selection via Radiobuttons:</p> <ul style="list-style-type: none"> ▶ serial ▶ TCP/IP
Serial	Active: serial connection is used
TCP/IP	Active: Connection via TCP/IP is used
Link Confirm	<p>Active: Link Layer Confirmation is active.</p> <p>Only available for serial communication.</p>
Link Timeout	<p>Time in seconds for connection timeout</p> <p>Default: 30s</p>
Timesync LAN	<p>Active: Use suitable time synchronization.</p> <p>Only available for TCP-connection.</p> <p>Use variation 3 of the timeobject. This is not accepted of all stations (for example Brodersen RTU) and can be deselected.</p>
IP address	IP address of PLC.
TCP-port	<p>Port that is used for communication.</p> <p>Default: 20000</p>
IP ADDRESS 2	Backup address for redundant connection to the PLC.
TCO Port 2	TCP Port on the backup address
Dual Port	Active: Dual Endpoint communication is permitted.
List Port	<p>TCP Port that is used as local Dual Endpoint.</p> <p>Only active if property Dual Port is active.</p>
Authentication	<p>Selection of the authentication:</p> <ul style="list-style-type: none"> ▶ None ▶ Verion 2
None	Active: No authentication filter is used.

Verion 2	Active: Authentication version 2.
Update Key	<p>The authentications key for the secured communication via version 2. 32 Hexadecimal Digits expected.</p> <p><u>Permitted characters:</u></p> <ul style="list-style-type: none"> ▶ Digits: 0 to 9 ▶ Letters: a to f and A to F ▶ Special characters: Space, points and colon ▶ at the start: Header 0x or 0X permitted ▶ All remaining letters are interpreted as 0 and will possibly fail during generating the session key.
Session Timer	<p>Validity of the sessionkey in seconds.</p> <p>Value: <700000 s</p> <p>Default: 900 s</p>
Session counter	<p>Select how often a session key may be used.</p> <p>Value: <10000 s</p> <p>Default: 1000</p>
Answer Timeout	<p>Maximal time to the reply of an authentication inquiry in milliseconds.</p> <p>Value: 100 to 120000</p> <p>Default: 2000 ms</p>
Mistake counter	<p>Specification how many mistakes are reported in the authentication.</p> <p>Value: 0 to 10</p> <p>Default: 2</p>
Short Hash	<p>Active: In serial communication, a shortened Hash value is used in the authentication.</p> <p>Note: Only for serial connection and only with session Timer until 1800 s (30 minutes) allowed.</p>
Aggressive Mode	Active: Authentication is carried out in the aggressive mode.
Pulse duration	Defines pulse duration for Norm , Trip and Close for each connection. Norm , Trip and Close define which relays are switched.
Norm	<p>Equals NUL.</p> <p>Pulse duration norm:</p> <ul style="list-style-type: none"> ▶ On time: Period of time in milliseconds in which the output is ON <p>Default: 1000</p>

	<ul style="list-style-type: none"> ▶ Time off Period of time in milliseconds in which the output is OFF Default: 1000
Trip	Pulse duration trip: <ul style="list-style-type: none"> ▶ On time: Period of time in milliseconds in which the output is ON Default: 1000 ▶ Time off Period of time in milliseconds in which the output is OFF Default: 1000
Close	Pulse duration close: <ul style="list-style-type: none"> ▶ On time: Period of time in milliseconds in which the output is ON Default: 1000 ▶ Time off Period of time in milliseconds in which the output is OFF Default: 1000
Save	Saves parameters for connection and deactivates editing mode.
Cancel	Cancels changes and deactivates editing mode without saving.

Double Point Mapping

Double Point Mapping, dependent on the selection in the configuration dialog (on page 19)

Offset 15	Offset 14	DNP3 binary	Custom Legacy	SCADA default
0	0	0	0	2
0	1	1	2	0
1	0	2	1	1
1	1	3	3	3

7. Creating variables

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

7.1 Creating variables in the Editor

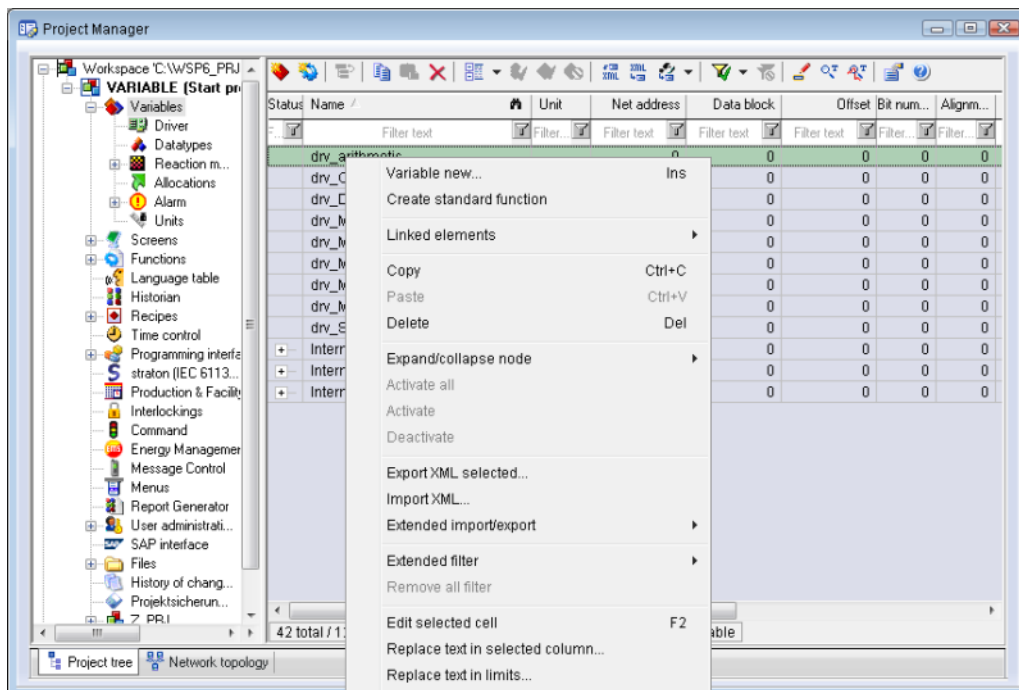
Variables can be created:

- ▶ as simple variables
- ▶ in arrays (main.chm::/15262.htm)
- ▶ as structure variables (main.chm::/15278.htm)

VARIABLE DIALOG

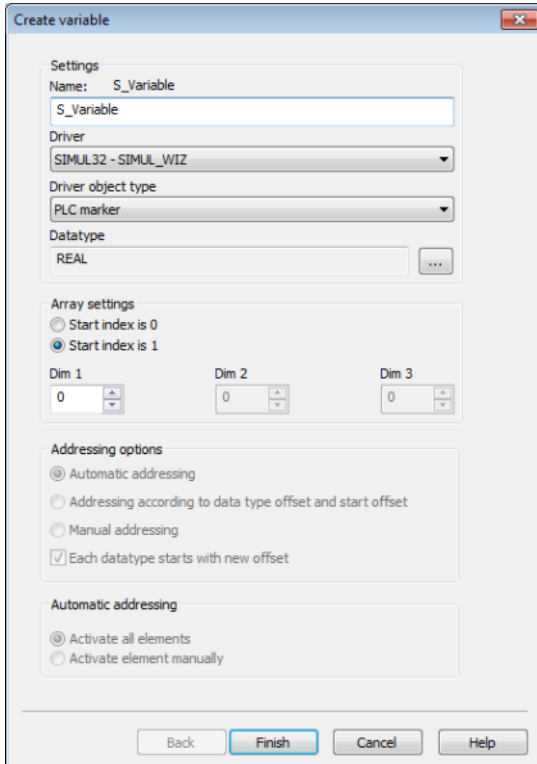
To create a new variable, regardless of which type:

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



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

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



The screenshot shows the 'Create variable' dialog box with the following settings:

- Settings**
 - Name: 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

Buttons at the bottom: Back, Finish, Cancel, Help.

Property	Description
Name	<p>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.</p> <p>Maximum length: 128 Zeichen</p> <p>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.</p> <p>Note: For some drivers, the addressing is possible over the property Symbolic address, as well.</p>
Driver	<p>Select the desired driver from the drop-down list.</p> <p>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.</p>
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 selection dialog.
Array settings	Expanded settings for array variables. You can find details in the Arrays chapter.
Addressing options	Expanded settings for arrays and structure variables. You can find details in the respective section.
Automatic element activation	Expanded settings for arrays and structure variables. You can find details in the respective section.

INHERITANCE FROM DATA TYPE

Measuring range, Signal range and Set value are always:

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

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

7.2 Addressing

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

Group/Property	Description
General	
Name	<p>Freely definable name.</p> <p>Attention: For every zenon project the name must be unambiguous.</p>
Identification	Freely assignable identification, e.g. for resources label, comment ...
Addressing	
Net address	<p>Bus address or Net address of the variable.</p> <p>This address refers to the station address in the connection configuration of the driver. This defines the PLC, on which the variable resides.</p> <p>Attention: The DNP address of the outstation is configured separately during the connection.</p>
Data block	not used for this driver
Offset	DNP Point in the Outstation. To see always in combination with the driver object type. For example: Analog Input, Group 20, Point 0 corresponds Offset 0 with a variable of the type <code>Analog Input</code> .
Alignment	not used for this driver
Bit number	not used for this driver
String length	Only available for String variables: Maximum number of characters that the variable can take.
Driver connection/Driver object type	Depending on the employed driver, an object type is selected during the creation of the variable; the type can be changed here later.
Driver connection/Data type	<p>Data type of the variable. Is selected during the creation of the variable; the type can be changed here later.</p> <p>ATTENTION: If you change the data type later, all other properties of the variable must be checked and adjusted, if necessary.</p>
Command Mode	<p>Only for <code>Binary Output</code> and <code>Analogy Output</code>. Determines if the driver does a <code>Direct operate</code> or <code>Select Before Operate</code>.</p> <p>Default: <code>direct operate</code></p>

Variation	<p>Only for variables in report direction, they are not either available in the classpoll or are supposed to be read for a special variation.</p> <p>If you want a different variation than the reported variation from the outstation, the Permanently read variable property must be activated in the variable.</p> <p>Variables are read explicit classless, not optimized:</p> <ul style="list-style-type: none"> ▶ after every Integrity poll or ▶ manually activated via offset 9 of the class poll variables
------------------	--

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** contrary to **Latest Value in the Outstation**).

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

If the Outstation supports this, you can activate **unsolicited** (on page 21) 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 master (Polling of eventclasses) to the master. In this case, it is not unconditionally required, to poll events regularly. The Polling cycle can then generally be adjusted to 0 or higher (Polling for eventklasses deactivated). The polling for events can be activated also manually by the Runtime (Class poll variables).

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
Analog inputs	68	R	REAL	
Analog Output Statuses	69	R / W	UDINT, DINT, REAL, UINT, INT	
Binary inputs	64	R	BOOL	
Binary Inputs Double	71	R	USINT	
Binary Output Statuses	65	R / W	BOOL, USINT	
Frozen Counters	67	R / W	BOOL, UDINT, DINT, USINT, UINT, INT, SINT	
Running Counters	66	R / W	BOOL, UDINT, DINT, USINT, UINT, INT, SINT	
String Data	70	R / W	STRING	
Driver variable	35	R / W	BOOL, SINT, USINT, INT, UINT, DINT, UDINT, REAL, STRING	<p>Variables for the static analysis of the communication; is transferred between driver and Runtime (not to the PLC).</p> <p>Note: The addressing and the behavior is the same for most zenon drivers.</p> <p>Find out more in the chapter about the Driver variables (on page 51)</p>

COMMANDS

BINARY OUTPUTS

Command with `binary output` variables:

- Only double messages can be used.
- The **Select Before Operate** property must be deactivated for the variable; the DNP3_NG driver uses `Auto-SBO` if necessary.

Use of the **qualifier of command** is recommended for the command 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

Use the `write set value` action text for analog outputs. Depending on the **command mode** option for the variable, either a `direct operate` or a `select before operate` is also carried out for analog outputs. 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 commands, 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

- ▶ 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	None	
1	Pulse On	
2	Pulse Off	Not fully compatible. Is not necessarily supported by the outstation.
3	Latch On	
4	Latch Off	
65	Close	
129	Trip	

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

RULES FOR SETTING THE STATUS BIT WHEN USING THE COMMAND

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).
- ▶ If a timeout follows instead of an answer, `SE`, `P/N` and `CoT_actterm` are set (5).
- ▶ The status is `SE` and `CoT_actcon` (1, 2, 3) after a positive response. The `Operate` is now 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, instead of a response, a timeout follows or if the answer a timeout of `Select`, `P/N` and `CoT_actterm` are set (3).
- ▶ 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).

Fa ll	Action	Status	Succ ess	Status	Action	Status	Succe ss	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	operate	act	t/o	actterm P/N	
4	select	act SE	nack	actcon SE P/N					
5	select	act SE	t/o	actterm SE P/N					

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

- ▶ **ack** = positive
- ▶ **nack** = negative
- ▶ **t/o** = Timeout

7.3.2 Mapping of the data types

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

Control	zenon	Data type
	BOOL	8
	USINT	9
	SINT	10
	UINT	2
	INT	1
	UDINT	4
	DINT	3
	ULINT	27
	LINT	26
	REAL	5
	LREAL	6
	STRING	12
	WSTRING	21
	DATE	18
	TIME	17
	DATE_AND_TIME	20
	TOD (Time of Day)	19

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 Group 1 Event Group 2	64	BOOL	Y	N	
Binary Inputs Double	71	USINT	Y	N	USINT-Mapping of 2 binary states with sequenced offsets. (No Couple-Bit Binary Input! Group 3 / 4)
Binary Output status Group 10	65	BOOL	N	Y	Value 1: LATCH_ON Value 0: LATCH_OFF

Event Group 11 Command Group 12					
Binary Output status Group 10 Event Group 11 Comamnd Group 12	65	USINT	N	Y	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, LATCH_OFF and TRIP
Running Counter Group 20 Event Group 22	66	UINT, UDINT	Y	N	
Frozen Counter Group 21 Event Group 23	67	UINT, UDINT	Y	Y	Writing freezes the counter
Analog Input Group 30 Event Group 32	68	INT, DINT, REAL, LREAL	Y	N	
Analog Output status Group 40 Event Group 41 Command Group 42	69	INT, DINT, REAL, LREAL	N	Y	written value is mirrored as a response after successful writing
String Data Group 110 Event Group 111	70	STRING	Y	Y	sent values are not mirrored. Get the latest values via update.
Class Scans	73	BOOL	N	Y	status INVALID until Integrity Poll has finished after driver start. Classpoll value "1" when successful Cold Restart: Value "1" during Cold Restart

Device Attributes Group 0	72	UINT, STRING	Y	N	

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

EXAMPLE BINARY INPUTS DOUBLE

The driver does mapping via driver object type `Binary Input Double` according to the settings in the driver configuration.

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

The mapping of 2 `Binary Inputs` to one `Binary Input Double` is a driver internal function. Do not confuse this with the actual DNP3 Group 3/4, Double-Bit Binary Inputs which is not supported by the driver.

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)
<ul style="list-style-type: none"> both bits: off (false) 	<ul style="list-style-type: none"> first Bit (Offset 14): off (false) second Bit (Offset 15): off (false) 	<ul style="list-style-type: none"> first Bit (Offset 14): on (true) second Bit (Offset 15): off (false) 	2 - intermediate
<ul style="list-style-type: none"> both bits: on (true) 	<ul style="list-style-type: none"> both Bits: on (true) 	<ul style="list-style-type: none"> both Bits: on (true) 	3 - faulty
<ul style="list-style-type: none"> first Bit (Offset 14): on (true) second Bit (Offset 15): off (false) 	<ul style="list-style-type: none"> beide Bits: off (false) 	<ul style="list-style-type: none"> beide Bits: off (false) 	0 - off
<ul style="list-style-type: none"> first Bit (Offset 14): off (false) second Bit (Offset 15): on (true) 	<ul style="list-style-type: none"> first Bit (Offset 14): on (true) second Bit (Offset 15): off (false) 	<ul style="list-style-type: none"> first Bit (Offset 14): off (false) second Bit (Offset 15): off (false) 	1 - on

DEVICE ATTRIBUTES

Currently device attributes are only read from the driver. Index 1 is always used (standard set of device attributes). The **Variation** matches the offset of the variable. The user must select the correct data type. When creating manually, the network address must be set according to the station number.

Variation 254 (all Device Attributes) can be read. Notice: This value is not send directly to the variable in the runtime. Instead, existing **device attributes** are automatically updated with the response to the read request from **variation** 254.

BINARY OUTPUTS

Writing of Binary Outputs allways uses **CROB** (Group 12). Either via **Direct Operate** or **Select Before Operate**. In doing so, the setting for the variable-specific command **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 for **PULSE_OFF**, **LATCH_OFF** and **TRIP**, and the 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.

Binary output command events (group 13) are currently parsed by the driver, but they are not sent to Runtime.

ANALOG OUTPUTS

Writing of analog outputs always is always carried out by means of a group 41 object, either by `Direct Operate` or `Select Before Operate`. In doing so, the setting for the variable-specific command **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.

Analog output command events (group 43) are currently parsed by the driver, but they are not sent to Runtime.

CLASS SCANS

Variables of the `class scan` driver object type are control variables to influence driver behavior and are not read by the outstation. `Class scan` variables must be created manually. In doing so, the network address must correspond to the station address in the driver configuration. The offset of the variable determines the function that is executed by the driver.

`Class scan` variables have the status `INVALID` in Runtime for as long as the integrity poll has not yet successfully been concluded by the driver. As soon as the integrity poll has been concluded, the variables are given the status `SPONT`.

The following offsets are currently supported:

- ▶ Offset 0:
An integrity poll will be triggered when writing to this variable (Read Request Group 60 Variation 2,3,4,1). If reading is successful, the variable receives the return value 1. Any classless variables that may exist are then read.
- ▶ Offset 1:
During writing to this variable a class 1 poll will be triggered (Read Request Group 60 Variation 2). If reading is successful, the variable receives the return value 1.
- ▶ Offset 2:
During writing to this variable a class 2 poll will be triggered (Read Request Group 60 Variation 3). If reading is successful, the variable receives the return value 1.
- ▶ Offset 3:
During writing to this variable a class 3 poll will be triggered (Read Request Group 60 Variation 4). If reading is successful, the variable receives the return value 1.
- ▶ Offset 9:
During writing to this variable a reading of all classless variables will be triggered. If reading is successful, the variable receives the return value 1.

► Offset 13:

A cold restart command is sent to the outstation when writing to this variable (Function Code 13). The object of group 52 contained in the response from the outstation (time delay) is evaluated by the driver and the driver sends no new requests to the outstation for the period specified in the time delay object. In Runtime, the value of this variable is "0" as long as Cold Restart is active; the value becomes 1 after that.

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

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

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



Attention

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

Example:

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

7.4.2 DBF Import/Export

Data can be exported to and imported from dBase.



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

Description	Type	Field size	Comment
KANALNAME	Char	128	Variable name. The length can be limited using the MAX_LAENGE entry in project.ini .
KANAL_R	C	128	The original name of a variable that is to be replaced by the new name entered under "VARIABLENNAME" (field/column must be entered manually). The length can be limited using the MAX_LAENGE entry in project.ini .
KANAL_D	Log	1	The variable is deleted with the 1 entry (field/column has to be created by hand).
TAGNR	C	128	Identification. The length can be limited using the MAX_LAENGE entry in project.ini .
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.
HWKANAL	Num	3	Bus address
BAUSTEIN	N	3	Datablock address (only for variables from the data area of the PLC)
ADDRESS	N	5	Offset
BITADR	N	2	For bit variables: bit address For byte variables: 0=lower, 8=higher byte For string variables: Length of string (max. 63 characters)
ARRAYSIZE	N	16	Number of variables in the array for index variables ATTENTION: Only the first variable is fully available. All others are only available for VBA or the Recipe Group Manager
LES_SCHR	R	1	Write-Read-Authorization

			0: Not allowed to set value. 1: Allowed to set value.
MIT_ZEIT	R	1	time stamp in zenon zenon (only if supported by the driver)
OBJEKT	N	2	Driver-specific ID number of the primitive object comprises TREIBER-OBJEKTYP 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)
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 DEFINITION

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

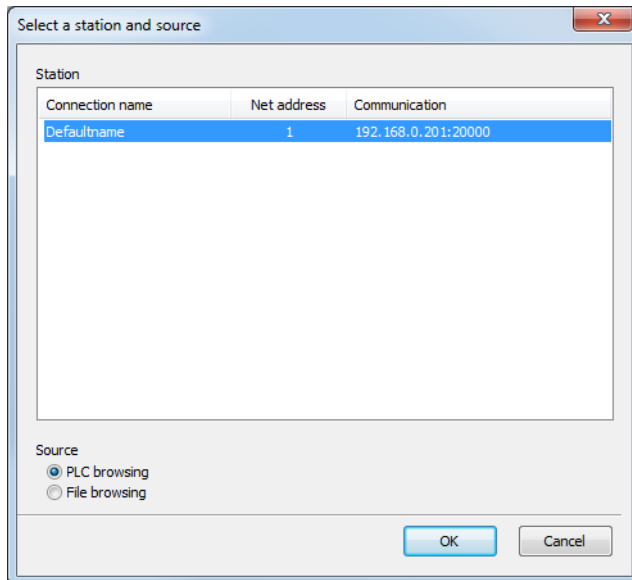
Description	Type	Field size	Comment
AKTIV1	R	1	Limit value active (per limit value available)
GRENZWERT1	F	20	technical value or ID number of a linked variable for a dynamic limit (see VARIABLEx) (if VARIABLEx is 1 and here it is -1, the existing variable linkage is not overwritten)
SCHWWERT1	F	16	Threshold value for limit
HYSTERESE1	F	14	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)
QUITTTIER1	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
MIN_MAX1	C	3	Minimum, Maximum
FARBE1	N	10	Color as Windows coding
GRENZTXT1	C	66	Limit 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.

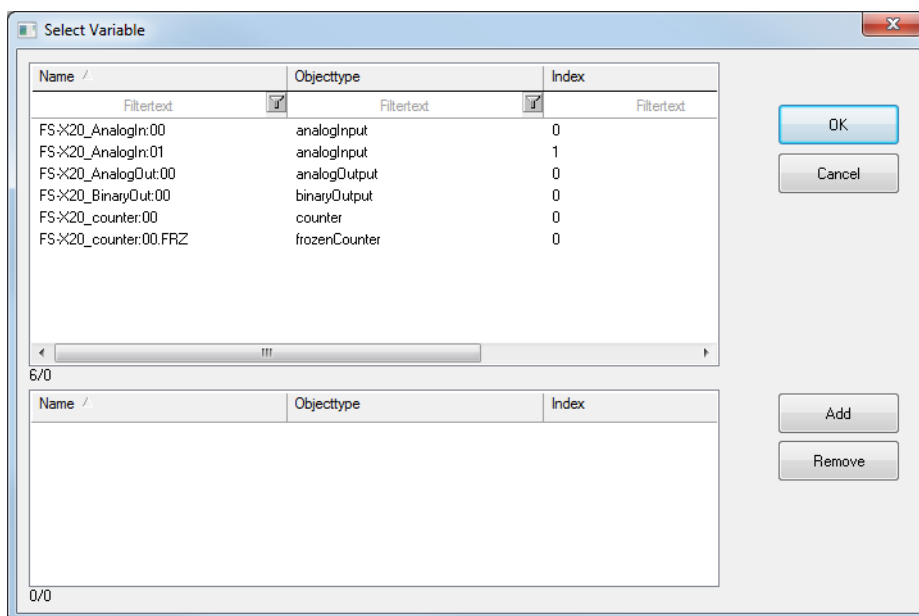
7.4.3 Online import

To import variables online from the PLC:

1. select the driver
2. Select **Import variables from driver** in the tool bar or in the context menu
3. The dialog for the import is opened

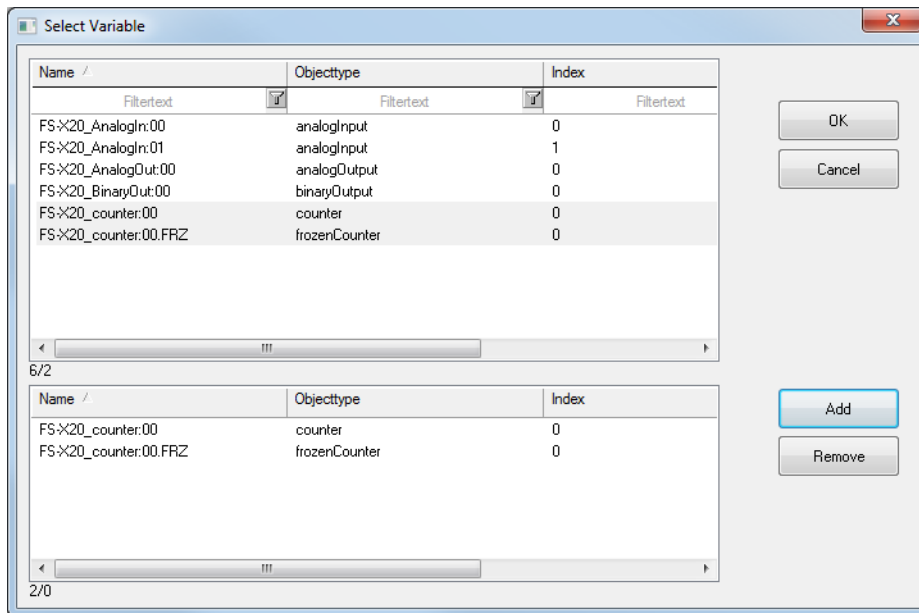


4. Select the desired connection
5. Select **Selection of PLC**
6. Confirm the selection by clicking **OK**
7. The dialog for variable selection is opened



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.



10. click on **removing**, deselect variables
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 address** of the variables is configured according to the selected station in the driver configuration (on page 21).

RULES FOR THE ONLINE IMPORT

For Online-Import:

- ▶ The response to an integrity poll is evaluated.
- ▶ The name is created from **Net address**, connection name, group number and index.
- ▶ The identification contains **Net address**, index and a description of the object type.
- ▶ Ensure that Runtime is not active if you start an online import; under certain circumstances, the outstation only supports a master or a 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.

7.4.4 Offline import

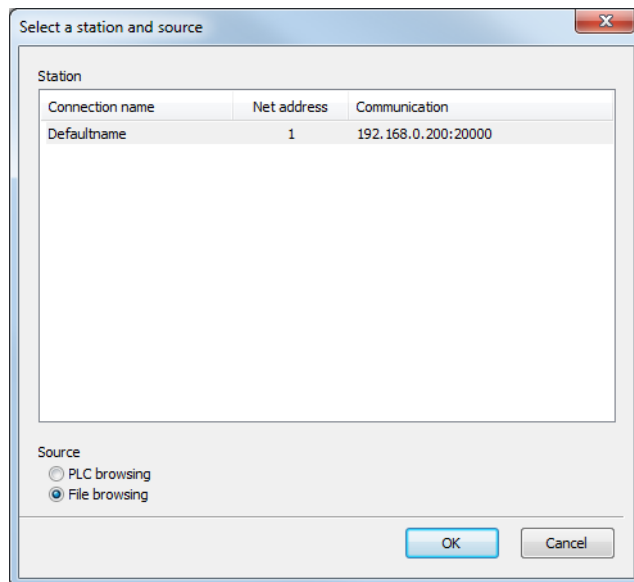
The driver supports the Offline import of variable out of a **DNP3 XML Device profile** file for the versions:

- ▶ 2.07 (January 2012)

► 2.08 (July 2012)

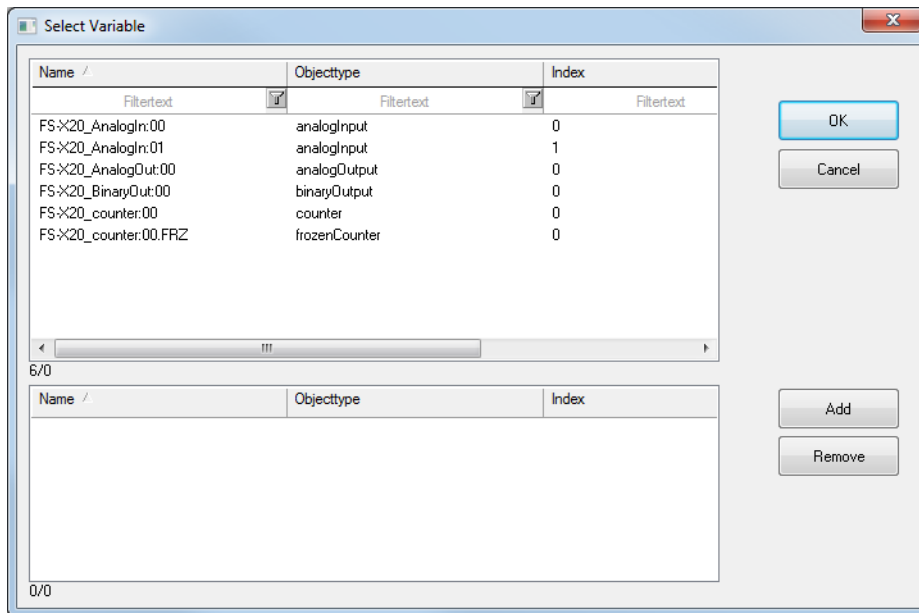
To import variables from a **DNP3 XML Device profile** file:

1. select the driver
2. Select **Import variables from driver** in the tool bar or in the context menu
3. The dialog for the import is opened

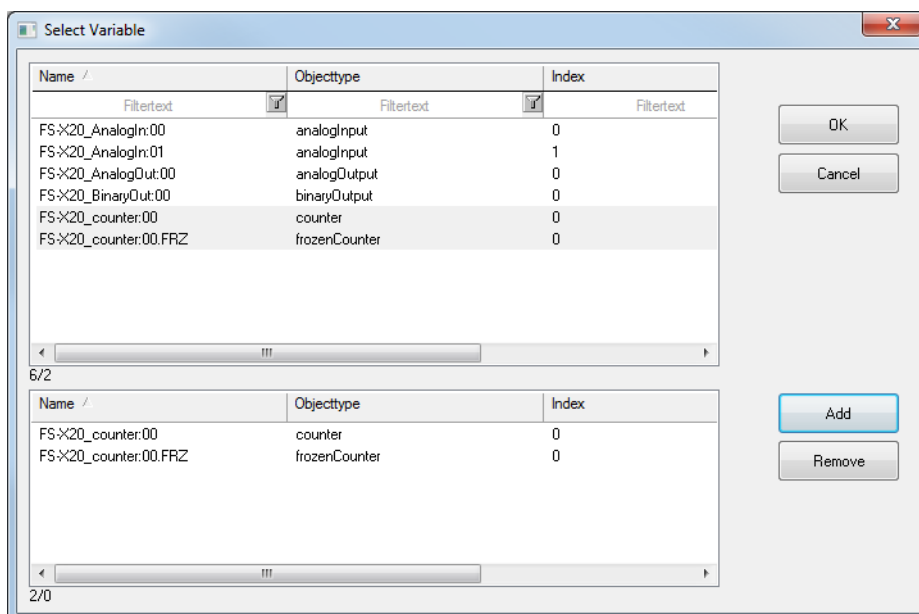


4. Select the desired connection
5. select **Import from file**
6. the dialog for file selection will be opened
7. select the desired file and confirm this selection by clicking **OK**

8. The dialog for variable selection is opened



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.



11. click on **removing** , deselect variables
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 address** of the variables is configured according to the selected station in the driver configuration (on page 21).

RULES FOR THE 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**. If however the value for **frozenCounterExists** is set to `true` for a counter (1), the option to also import variables for **frozen counter** is offered.

NOT IMPORTED VARIABLES

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

- ▶ Device attributes
- ▶ Control variables for classpoll, classless reading and cold restart.
- ▶ Binary inputs double

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

7.5 Driver variables

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

- ▶ Information
- ▶ Configuration
- ▶ Statistics and

► Error message

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

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



Information

Not every driver supports all driver variants.

For example:

- ▶ Variables for modem information are only supported by modem-compatible drivers
- ▶ Driver variables for the polling cycle only for pure polling drivers
- ▶ Connection-related information such as ErrorMessage only for drivers that only edit one connection at 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
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 <code>driver stop</code> , the variable has the value <code>TRUE</code> and an OFF bit. After the driver has started, the variable has the value <code>FALSE</code> and no OFF bit.
SimulRTState	UDINT	60	Informs the status of Runtime for driver simulation.

CONFIGURATION

Name from import	Type	Offset	Description
ReconnectInRead	BOOL	27	If <code>TRUE</code> , 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 <code>SrvDrvVarApplyCom</code> 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 <code>SrvDrvVarApplyModem</code> . This closes the current connection and opens a new one according to the settings PhoneNumberSet and ModemHwAdrSet .

PhoneNumberSet	STRING	38	Telephone number, that should be used
ModemHwAdrSet	DINT	39	Hardware address for the telephone number
GlobalUpdate	UDINT	3	Update time in milliseconds (ms).
BGlobalUpdaten	BOOL	4	TRUE, if update time is global
TreiberSimul	BOOL	5	TRUE, if driver in sin simulation mode
TreiberProzab	BOOL	6	TRUE, if the variables update list should be kept in the memory
ModemActive	BOOL	7	TRUE, if the modem is active for the driver
Device	STRING	8	Name of the serial interface or name of the modem
ComPort	UINT	9	Number of the serial interface.
Baud rate	UDINT	10	Baud rate of the serial interface.
Parity	SINT	11	Parity of the serial interface
ByteSize	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.
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.
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:

Parameters	Description
Blockwrite	Not supported.
Redundancy	if the outstation supports several masters
RDA	Not supported. Sequence of Events Buffer is supported if outstation is active.
Real time stamping	Yes, if the variation in the outstation is configured accordingly.
Browsing	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 possible with a driver.

ERROR FILE

The driver supports central logging in the diagnosis server.

DRIVER VARIABLES

Driver statistics variables are not set by the DNP3_NG driver. Communication monitoring is possible using the INVALID status bit.

EXTENDED ERROR FILE

The driver supports extended logging in the diagnosis server. Configuration is performed via the diagnosis server.

INTEGRITY POLL

After the driver has been started, an integrity poll of the driver is first sent to the outstation (**Read Request Group** 60, **Variation** 2, 3, 4 and 0, **Range** all). The values received are available in Runtime accordingly. In this case, ensure that the outstation sends a variation that is different to a normal value change as a response to the integrity poll. (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.

The DNP3_NG 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_NG master reacts to this bit in that a read query for group 60, variation 2,3,4 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 1.2 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_NG master reacts to this bit in that a read query for group 60, variation 2,3,4 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 1.3 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_NG master reacts to this bit in that a read query for group 60, variation 2,3,4 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_NG master reacts immediately and responds to the outstation depending on the selected time synchronization option in the driver configuration.
- ▶ `IIN 1.7 DEVICE_RESTART`: Is set by the outstation in the event of a restart. If a cold restart is sent to the outstation by the DNP3_NG master, the outstation sets this bit. It is reset by the DNP3_NG master in this process.
- ▶ `IIN 2.2 PARAMETER_ERROR`: Is set by the outstation, if, for example, an object that is not present in the outstation with this DNP3 index is explicitly read. In this case, the DNP3_NG sets the `INVALID` status bit for the variable in Runtime.

If the `NO_AUTO_IIN` 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.

DNP3 OBJECT FLAGS MAPPING

The DNP3 object flags **ONLINE** and **COMM_LOST** are currently evaluated by the DNP3_NG driver. With **ONLINE** = `false` or **COMM_LOST** = `true`, the **INVALID** bit of the variable is set. Note that the **Variation** configured in the outstation determines whether objects are sent with or without flags for an object group.

TIME SYNCHRONIZATION

The DNP3_NG 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.

It is not currently possible to send a time synchronization to the outstation cyclically without a request.

Ensure that the outstation gets the time from a different source in this case (such as a GPS receiver) and as a result may possibly not send any requirement for time synchronization to the master. In this case, ensure that the Runtime computer with the DNP3_NG 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

The DNP3_NG 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.

DNP3 SECURE AUTHENTICATION V2

SELECT BEFORE OPERATE - DIRECT OPERATE

The DNP3_NG driver uses **Direct operate** (by default) or **Select Before Operate**, for writing **Binary Output Status** or **Analog Output** type variables. Configuration is carried out using the driver-specific variable property. **Command Mode**.

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

- Block the command 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 DNP332 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.

DNP3 SEQUENTIAL FILE TRANSFER

`DNP3 sequential file transfer` is not currently supported by the DNP3_NG driver.

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.

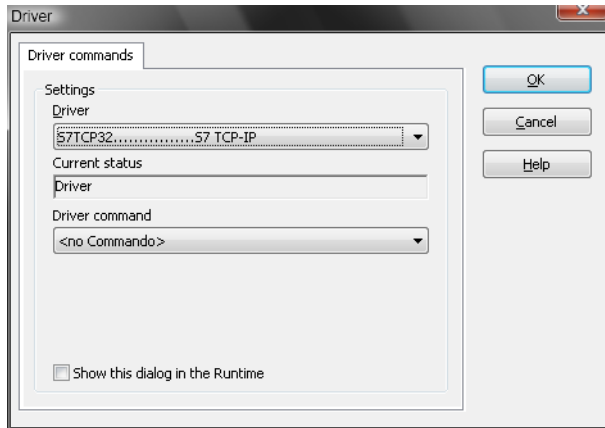
9. Driver commands

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

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

- ▶ create a new function
- ▶ select *Variables -> Driver commands*

- The dialog for configuration is opened



Parameters	Description
Drivers	Drop-down list with all drivers which are loaded in the project.
Current state	Fixed entry which has no function in the current version.
Driver commands	Drop-down list for the selection of the command.
▶ Start driver (online mode)	Driver is reinitialized 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 <code>switched off (OFF; Bit 20)</code> .
▶ 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	Enter driver-specific commands. Opens input field in order to enter a command.
▶ Activate driver write set value	Write set value to a driver is allowed.
▶ Deactivate driver write set value	Write set value to a driver is prohibited.
▶ Establish connection with modem	Establish connection (for modem drivers) Opens the input fields for the hardware address and for the telephone number.
▶ Disconnect from modem	Terminate connection (for modem drivers)
Show this dialog in the Runtime	The dialog is shown in Runtime so that changes can be made.

DRIVER COMMANDS IN THE NETWORK

If the computer, on which the **driver command** function is executed, is part of the zenon network, additional actions are carried out. A special network command is sent from the computer to the project server, which then executes the desired action on its driver. In addition, the Server sends the same driver command to the project standby. The standby also carries out the action on its driver.

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

10. Error analysis

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

10.1 Analysis tool

All zenon modules such as Editor, Runtime, drivers, etc. write messages to a joint log file. To display them correctly and clearly, use the Diagnosis Viewer (main.chm::/12464.htm) program that was also installed with zenon. You can find it under *Start/All programs/zenon/Tools 7.20 -> Diagviewer*.

zenon driver log all errors in the log files. The default folder for the log files is subfolder `LOG` in directory `ProgramData`, example:

`C:\ProgramData\COPA-DATA\LOG`. Log files are text files with a special structure.

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 currently created entries live
- ▶ customize the logging settings
- ▶ change the folder in which the log files are saved

Note:

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

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

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 65):
-> 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 Enter.
Do you receive an answer with a time or a time-out?
- ▶ Can the PLC be reached via **Telnet**?
Telnet: Command line Enter open, telnet <IP address port number> Input (e. g. telnet 192,168,0,100 20000) -> press enter.
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 are purely statistics variables. They do not communicate with the PLC. (See chapter Driver variable (on page 51).)
- ▶ 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.
- ▶ Class poll variables are INVALID after starting the driver as long as the integritypoll was not finished.

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 65):

-> Which messages are displayed?

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

With `secure authentication`, check to see if the `pre-shared update keys` in the driver and the outstation are identical.