



COPADATA
do it your way

zenon driver manual

ALLANBNT

v.7.50





©2016 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. ALLANBNT	5
3. ALLANBNT - Data sheet	6
4. Driver history	7
5. Requirements	8
5.1 PC	8
5.2 Control	8
6. Configuration	9
6.1 Creating a driver	9
6.2 Create driver RSLinx	11
6.3 Settings in the driver dialog	13
6.3.1 General	13
6.3.2 Settings	16
6.3.3 Unsolicited Messages configuration	17
6.3.4 ControlLogix Gateway Routing	20
7. Creating variables	25
7.1 Creating variables in the Editor	25
7.2 Addressing	29
7.3 Driver objects and datatypes	30
7.3.1 Driver objects	30
7.3.2 Mapping of the data types	31
7.4 Creating variables by importing	32
7.4.1 XML import	33
7.4.2 DBF Import/Export	33
7.5 Driver variables	38
8. Driver-specific functions	45
9. Driver commands	46

10. Error analysis.....	48
10.1 Analysis tool	48
10.2 Check list	49

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. ALLANBNT

This driver connects to one or more PLCs in a direct connection (COM interface or network).

3. ALLANBNT - Data sheet

General:	
Driver file name	ALLANBNT.exe
Driver name	Allen Bradley RS-Linx driver
PLC types	Allen Bradley PLC types: PLC2, PLC3, PLC5, PLC5250, SLC500;
PLC manufacturer	Allen-Bradley; Rockwell;

Driver supports:	
Protocol	DataHighway Plus; DeviceNet; DataHighway; DF1; TCP/IP; ControlNet;
Addressing: Address-based	X
Addressing: Name-based	--
Spontaneous communication	--
Polling communication	X
Online browsing	--
Offline browsing	--
Real-time capable	--
Blockwrite	--
Modem capable	--
Serial logging	--
RDA numerical	--
RDA String	--

Requirements:	
Hardware PC	Network or serial connections
Software PC	RSLinx (higher than v1.6) from Rockwell necessary. Licenced version of RS Linx(e.g. RS Linx Classic Gateway) RS Linx Classic Lite would not work.
Hardware PLC	--
Software PLC	--
Requires v-dll	--

Platforms:	
Operating systems	Windows 7, 8, 8.1, 10, Server 2008R2, Server 2012, Server 2012R2;
CE platforms	-;

4. Driver history

Date	Driver version	Change
8/14/2008	500	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 form 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

There must be connectors for network or serial connections as well as Serial, BUS and RJ45 cables.

SOFTWARE

Copy the driver file (*.exe) into the current program directory (if they do not already exist there) and enter into TREIBER_EN.XML with the tool driverinfo.exe.

Additionally the software RSLinx from Rockwell Automation is needed.

5.2 Control

HARDWARE

PLC2, PLC3, PLC5, PLC5250, SLC500 and network or serial connectors required.

SOFTWARE

PLC can be configured with tools (Allan Bradley).

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 Manage 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.



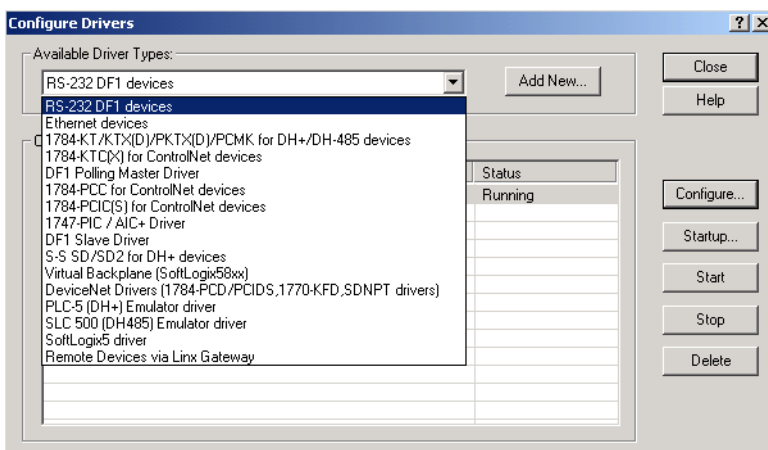
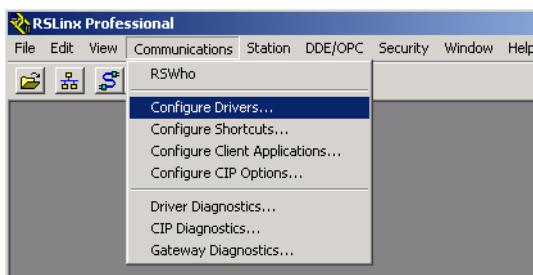
Information

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

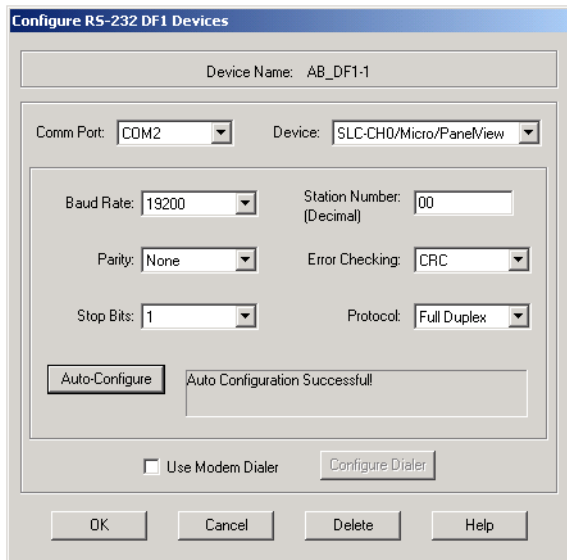
- ▶ Internal
- ▶ MathDr32
- ▶ SysDrv.

▶

6.2 Create driver RSLinx

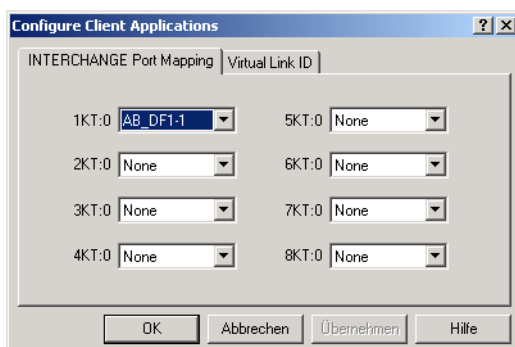
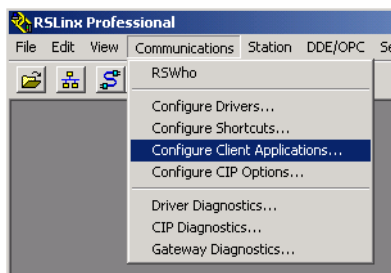


Select the according driver from the menu **Available Driver Types**. Add it with **Add New..**; this will also create a name for the driver. To open the driver configuration, select the driver in the list and press the button **Configure**.



If the PLC is connected, RSLinx will be able to read the correct parameters from it.

Otherwise, you will have to enter the connection parameters here.

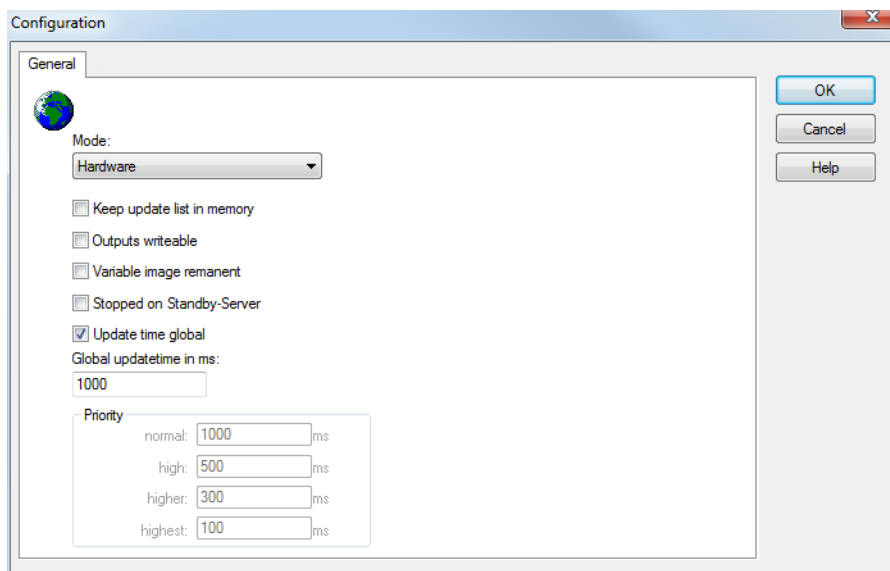


6.3 Settings in the driver dialog

You can change the following settings of the driver:

6.3.1 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>

	<p>Thus the communication load is distributed better.</p> <p>Attention: Priority classes are not supported by each driver For example, drivers that communicate spontaneously do not support it.</p>
--	--

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.

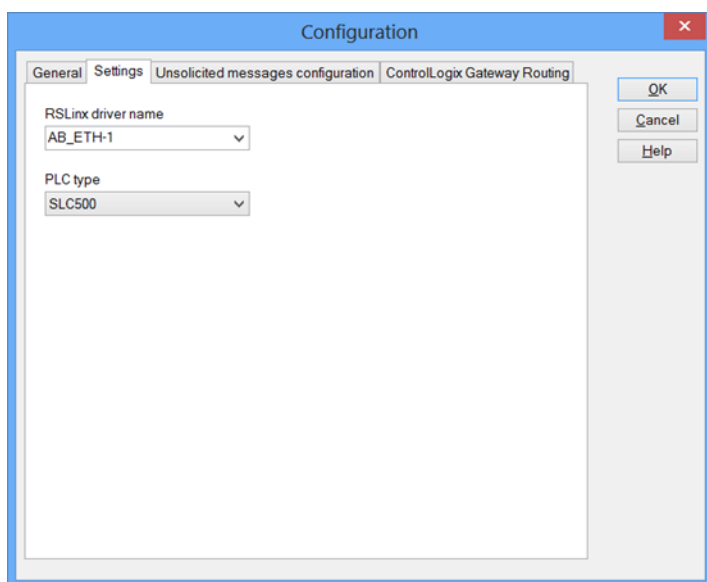
UPDATE TIME FOR CYCLICAL DRIVERS

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.

6.3.2 Settings

Configuration of RSLink driver name and PLC type.



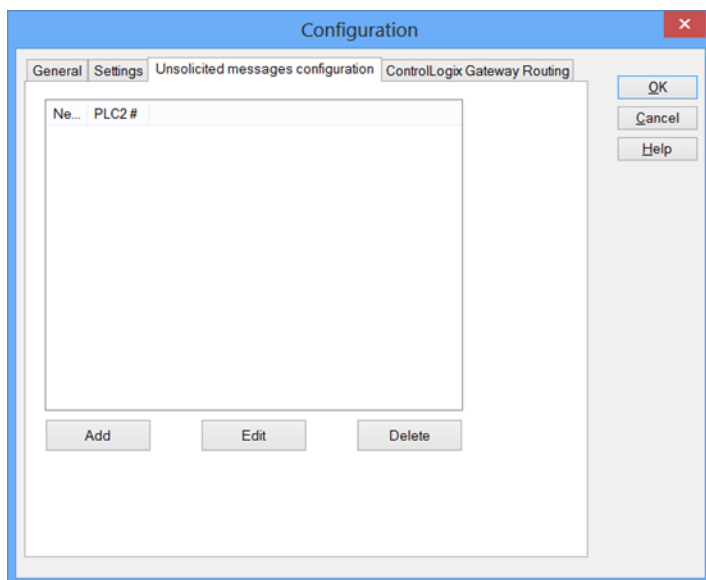
Parameters	Description
RSLinux driver name	Label of the RS-LINX driver. Enter it in the field manually or select it via the drop-down list. It contains the drivers available in RS-LINX.
PLC Type	Type of the used PLC. Select from drop-down list. Note: Is ignored when routing (on page 20) is used.
OK	Applies all changes in all tabs and closes the dialog.
Cancel	Discards all changes in all tabs and closes the dialog.
Help	Opens online help.

A PLC-2 unprotected write command contains only one setting for the addressing - the **data table address**. The mapping of this address to the net address of the variable is carried out in the **Unsolicited messages configuration** (on page 17).

6.3.3 Unsolicited Messages configuration

Messages of type `unsolicited` can be read and written by the zenon RS Linx driver.

To depict received data on the engineered variables, you must map the **data table address** of the PLC2 to the variable address. For each message one entry must be configured.



Parameters	Description
List field	Contains all configured allocations from variable net addresses to PLC2 addresses.
Add	Opens dialog in order to add a new allocation.
Edit	Opens dialog in order to edit the allocation highlighted in the list field.
Delete	Deletes the allocation highlighted in the list field from the list without confirmation request. If several allocations are highlighted, only the topmost allocation is deleted.
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.

ALLOCATION

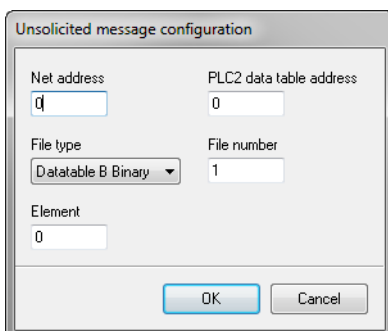
In the PLC program - triggered by the program - ein Unsolicited Message is sent with a **PLC2 unprotected write** command. Target address is the computer with zenon and RS-LINX.

For the allocation of PLC2 addresses to variables the following is true:

- ▶ The same PLC2 address can be allocated to different variables with different net addresses.
- ▶ A pair composed of a net address and a PLC2 address can only be allocated to one variable address.
- ▶ If several variables lie one after the other and if the write command contains data for several variables, these data are automatically depicted in the following variable.

To configure the allocation:

1. click on the button **Add** on tab `Unsolicited MessagesConfiguration`
2. the dialog for configuring an allocation is opened

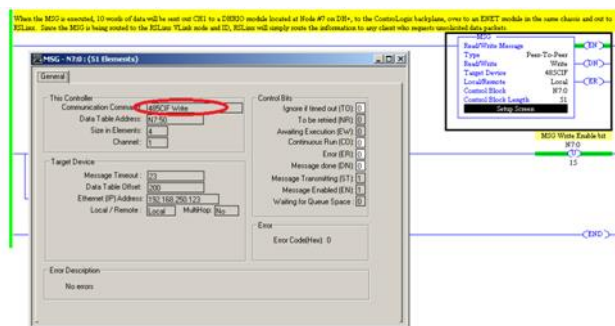


The dialog box titled "Unsolicited message configuration" contains the following fields and controls:

- Net address:** A text input field containing the character "Q".
- PLC2 data table address:** A text input field containing the number "0".
- File type:** A dropdown menu with "Datatable B Binary" selected.
- File number:** A text input field containing the number "1".
- Element:** A text input field containing the number "0".
- Buttons:** "OK" and "Cancel" buttons at the bottom right.

Parameters	Description
Net address	Network address of variables.
PLC2 data table address	Data table address of the PLC2.
File type	Selection of the type from the drop-down list.
File number	File number.
Element	Element number.
OK	Takes over settings, creates entry in list and closes the dialog.
Cancel	Rejects all setting changes and closes the dialog.

In the driver configuration you must create a corresponding unsolicited message for the fitting station number (configured in RS_LINX).



Information

For zenon versions < 6.51 option **Update time global** must be deactivated in tab General (on page 13) for the to be processed faster than the **Update time global**.

EXAMPLE

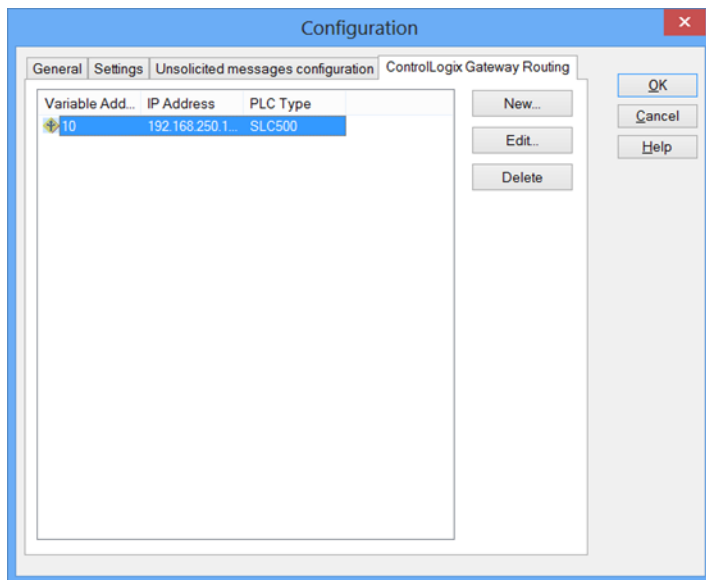
4 integer values - N7:50 to N7:53 - are copied as data in the message. As **PLC2 data table address** 200 is used.

As in the RS-LINX configuration the **Data table address** 200 is again depicted on integer values N7:50 and following:

- ▶ the corresponding variables (N7:50 - N7:53) can be polled (according to the defined update time)
- ▶ received spontaneous values which were received additionally via Unsolicited Messages can be depicted on this variable

6.3.4 ControlLogix Gateway Routing

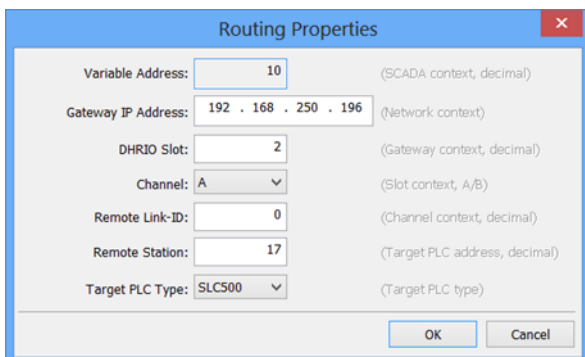
For variables, routing can be configured using DH+. For details, see the **Routing of variables via DH+** (on page 22) chapter.



Parameters	Description
Routings list field	Shows configured routings.
New	Opens the dialog for configuring a new routing.
Edit	Opens the dialog for editing the routing shown in the list.
Delete	Deletes the selected routing from the list.
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.

CREATING OR EDITING ROUTING

Clicking on the **New** or **Edit** button opens the dialog to configure a routing. The required values can be collected with the RSLinx software. You can find details in the RSLinx software documentation.



The image shows a screenshot of the 'Routing Properties' dialog box. It contains several input fields and dropdown menus for configuring a routing. The fields are as follows:

- Variable Address:** A text box containing the value '10'. To its right is the text '(SCADA context, decimal)'.
- Gateway IP Address:** A text box containing the value '192 . 168 . 250 . 196'. To its right is the text '(Network context)'.
- DHRIO Slot:** A text box containing the value '2'. To its right is the text '(Gateway context, decimal)'.
- Channel:** A dropdown menu with 'A' selected. To its right is the text '(Slot context, A/B)'.
- Remote Link-ID:** A text box containing the value '0'. To its right is the text '(Channel context, decimal)'.
- Remote Station:** A text box containing the value '17'. To its right is the text '(Target PLC address, decimal)'.
- Target PLC Type:** A dropdown menu with 'SLC500' selected. To its right is the text '(Target PLC type)'.

At the bottom of the dialog box, there are two buttons: 'OK' and 'Cancel'.

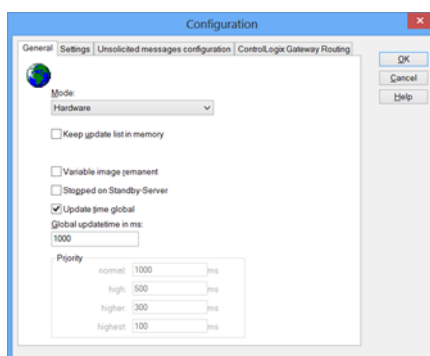
Parameters	Description
Variable Address	Net address of the variable.
Gateway IP Address	IP address of the gateway controller.
DHRIO Slot	Number of the slot in the rack in which the DHRIO module is located.
Channel	Selected channel.
Remote Link ID	ID of the station. ▶ 0: Local. ▶ >0: Link ID of the connection configuration in RSLinx
Remote Station	Number of the remote station.
Target PLC Type	Type of target PLC. Note: If a different PLC has been selected in the Settings (on page 16) tab, the PLC selected here is used for routing.
OK	Applies settings and closes the dialog.
Cancel	Discards all changes and closes the dialog.

Configuration of variable routing via DH+

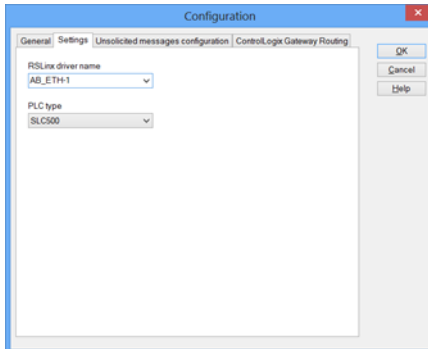
Configuration of the routing for communication with variables that are connected via DH+. You can find information on parameters from RSLinx in the module configuration in the RSLinx software.

To configure the routing:

1. In the driver configuration, select **Hardware** as the **mode** in the **General** tab.

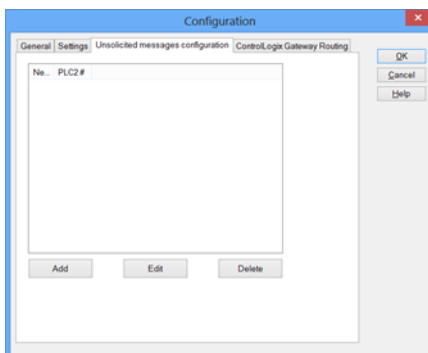


2. Select the **RSLinx driver name** in the Settings tab.

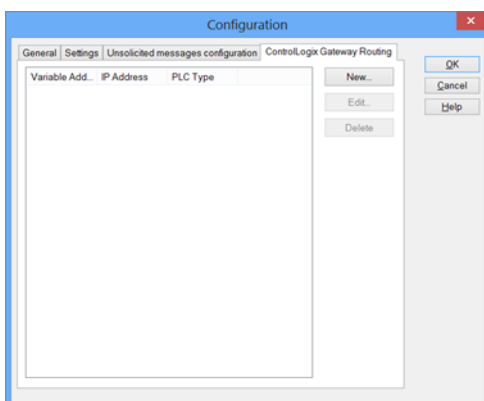


For routing, the PLC type option is taken from the settings in the **ControlLogix Gateway Routing** (on page 20) tab. Settings that are different to this in the **Settings** (on page 16) tab are ignored.

3. Configure the **Unsolicited Messages Configuration** (on page 17) tab.

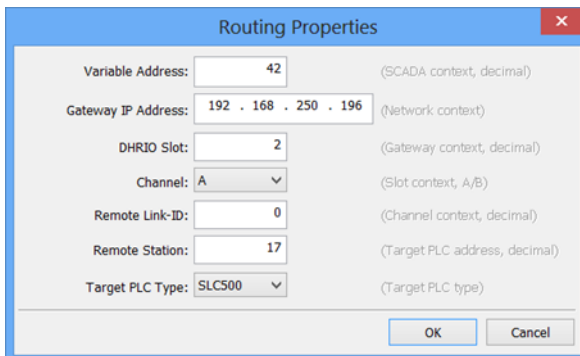


4. Configure the **ControlLogix Gateway Routing** tab.



5. Click on the **New** button to configure a new routing.

The dialog for the routing properties is opened. Configure the properties:

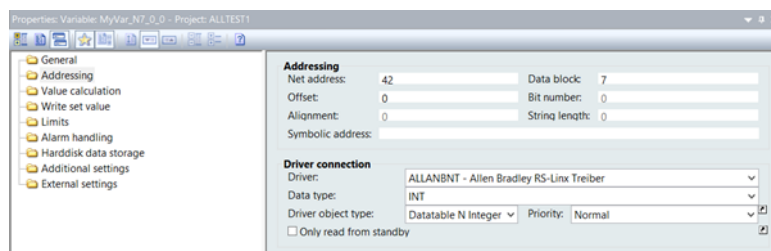


The 'Routing Properties' dialog box contains the following fields and values:

- Variable Address: 42 (SCADA context, decimal)
- Gateway IP Address: 192 . 168 . 250 . 196 (Network context)
- DHRIO Slot: 2 (Gateway context, decimal)
- Channel: A (Slot context, A/B)
- Remote Link-ID: 0 (Channel context, decimal)
- Remote Station: 17 (Target PLC address, decimal)
- Target PLC Type: SLC500 (Target PLC type)

Buttons: OK, Cancel

- a) Enter, for the **variable address**, the **Net address** of the variables in zenon.



The 'Properties' dialog box for 'Variable: MyVar_N7_0_0' shows the following settings:

- Addressing:**
 - Net address: 42
 - Offset: 0
 - Alignment: 0
 - Symbolic address:
 - Data block: 7
 - Bit number: 0
 - String length: 0
- Driver connection:**
 - Driver: ALLANBNT - Allen Bradley RS-Linx Treiber
 - Data type: INT
 - Driver object type: Databale N Integer
 - Priority: Normal
 - ☐ Only read from standby

- b) Enter the IP address of the gateway in RSLinx.

- c) Enter the slot in the rack in **DHRIO Slot**.

- d) Enter the **remote link ID**.

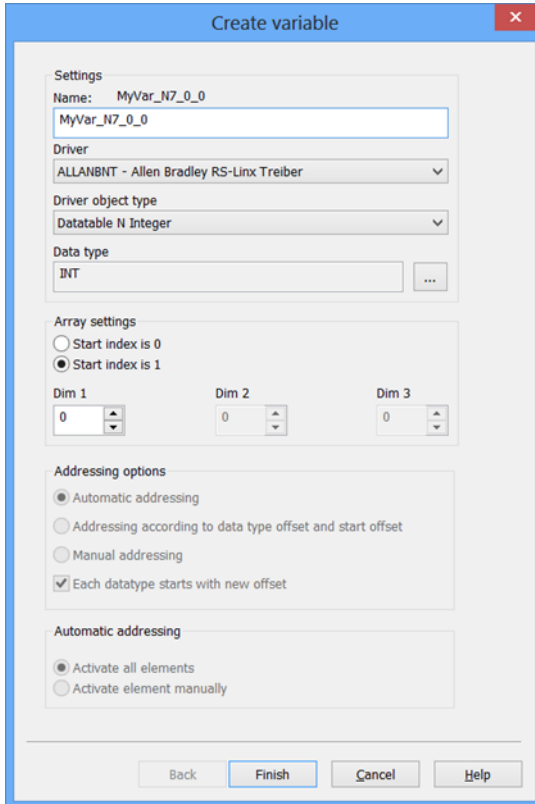
If it is the same station as that defined in the gateway IP address option, then set the value to 0.

If routing is via a different station, enter the **link ID** for the corresponding connection.

- e) Enter the address of the target PLC.

- f) Enter the type of the target PLC. This setting is preferred over the setting with the same name in the **Settings** (on page 16) tab.

6. Configure the corresponding variables:



Create variable

Settings

Name: MyVar_N7_0_0
MyVar_N7_0_0

Driver: ALLANBNT - Allen Bradley RS-Linx Treiber

Driver object type: Datatable N Integer

Data type: INT

Array settings

☐ Start index is 0
☒ Start index is 1

Dim 1: 0 Dim 2: 0 Dim 3: 0

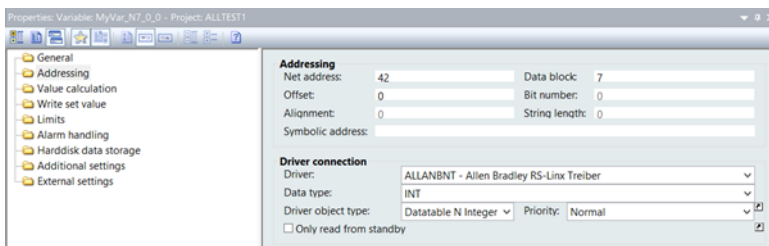
Addressing options

☒ 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

Back Finish Cancel Help



Properties: Variable: MyVar_N7_0_0 - Project: ALLTEST1

General

Addressing

Net address: 42 Data block: 7
Offset: 0 Bit number: 0
Alignment: 0 String length: 0
Symbolic address:

Driver connection

Driver: ALLANBNT - Allen Bradley RS-Linx Treiber
Data type: INT
Driver object type: Datatable N Integer Priority: Normal
☐ Only read from standby

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:

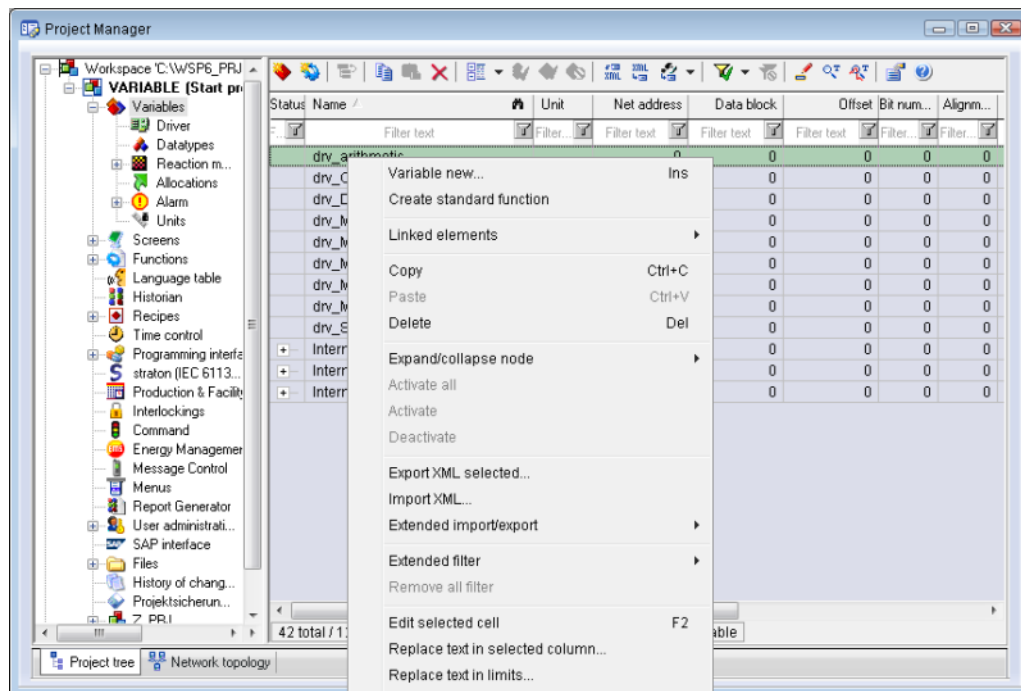
- ▶ as simple variables

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

VARIABLE DIALOG

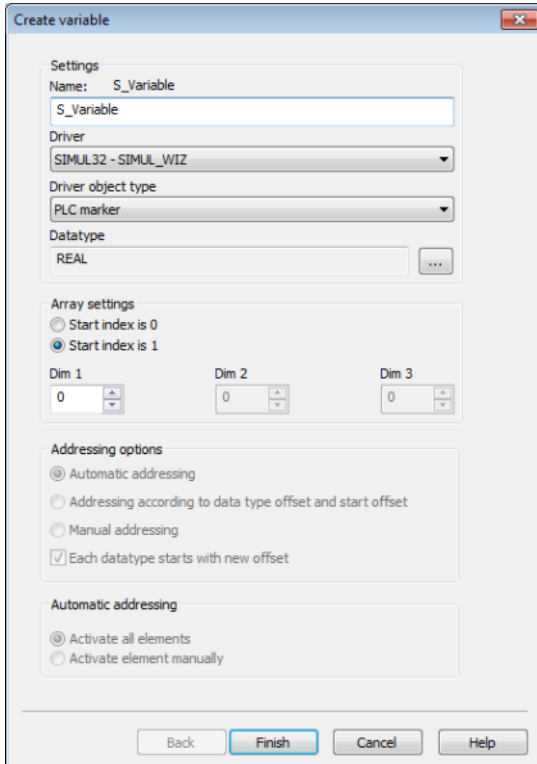
To create a new variable, regardless of which type:

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	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 character 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 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.

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.

INHERITANCE FROM DATA TYPE

Measuring range, **Signal range** and **Set value** are always:

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

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

7.2 Addressing

Property	Description
Name	Freely definable name <i>Attention:</i> the name must be unique within each control system project.
Identification	Any text can be entered here, e.g. for resource labels, comments ...
Net address	Bus address or net address of the variable. This address refers to the bus address in the connection configuration of the driver. This defines the PLC, on which the variable resides. Corresponds to the variable address in RSLinx.
Data block	Corresponds to the data tables in the PLC. For variables of object type Extended data block, enter the datablock number here. Configurable [0.. 4294967295]. Please look up the exact maximum range for data blocks in the manual of the PLC.
Offset	Offset of the variable; the memory address of the variable in the PLC. Configurable [0.. 4294967295]
Alignment	not used for this driver
Bit number	Number of the bit within the configured offset. Allowed entry [0.. 65535]
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	Data type of the variable, which is selected during the creation of the variable; the type can be changed here later. <i>Attention:</i> If you change the data type later, all other properties of the variable must be checked and adjusted, if necessary.
Driver connection/Priority	

ASSIGNMENT OF ADDRESSES OF ALLEN BRADLEY PLCs TO ZENON

AB control units administer addresses in this form: **Data_type_short+numeral:offset/bit number**.
For example: **B3:100/7**

Assignment of data types and addresses:

Data type	Short form	Digit
BINARY	B	3
TIMER (Only DF1)	T	4
COUNTER (Only DF1)	C	5
CONTROL (Only DF1)	R	6
INTEGER	N	7
FLOATING PT	F	8

To transfer these addresses in zenon correctly:

- ▶ Enter the **digit** (3, for example) in the **Data block** property
- ▶ The zenon **Offset** corresponds to the address within this bandwidth
- ▶ Bit number corresponds to **Bit number**

EXAMPLE

The AB address **B3:100/7** corresponds to the following in zenon:

- ▶ **Data block:** 3
- ▶ **Offset:** 100
- ▶ **Bit number:** 7

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:

OBJECTS FOR PROCESS VARIABLES IN ZENON

Driver object type	Channel type	Read / Write	Supported data types	Comment
Datatable B Binary	96	R / W	USINT, BOOL, SINT, UINT, INT	
Datatable F Float	97	R / W	REAL	
Datatable I Input	101	R / W	BOOL, INT, SINT, UINT, USINT	Sets Data block to 1.
Datatable N Integer	34	R / W	REAL, BOOL, DINT, UDINT, USINT, INT, UINT, SINT, STRING	
Datatable O Output	102	R / W	BOOL, INT, SINT, UINT, USINT	Sets Data block to 2.
Datatable ST String	98	R / W	STRING	Maximum 82 characters
Driver variable	35	R / W	BOOL, SINT, USINT, INT, UINT, DINT, UDINT, REAL, STRING	Variables for the statistical analysis of communication. Find out more in the chapter about the Driver variables (on page 38)

Object	Read	Write	Comment
B block (Word, Byte, Bool)	y	y	Binary - Data table
N block (DWORD, FLOAT, Word, String, Byte/Char, Bool)	y	y	Integer - Data table

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

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

7.4 Creating variables by importing

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



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

Identification	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)
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	L	1	time stamp in 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	L	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	L	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	L	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	L	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	L	1	Set blink attribute
BTB1	L	1	Logging in CEL
ALARM1	L	1	Alarm
DRUCKEN1	L	1	Printer output (for CEL or Alarm)
QUITTIER1	L	1	Must be acknowledged
LOESCHE1	L	1	Must be deleted
VARIABLE1	L	1	Dyn. limit value linking the limit is defined by an absolute value (see field GRENZWERTx).
FUNC1	L	1	Functions linking
ASK_FUNC1	L	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 value text
A_DELAY1	N	10	Time delay
INVISIBLE1	L	1	Invisible

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

7.5 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 implemented in the driver kit are available in the import file **drvvar.dbf** (on the installation medium in the \Predefined\Variables folder) 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.
Baudrate	UDINT	10	Baud rate of the serial interface.
Parity	SINT	11	Parity of the serial interface
ByteSize	USINT	14	Number of bits per character of the serial interface Value = 0 if the driver cannot establish any serial connection.
StopBit	USINT	13	Number of stop bits of the serial interface.
Autoconnect	BOOL	16	TRUE, if the modem connection should be established automatically for reading/writing
PhoneNumber	STRING	17	Current telephone number
ModemHwAdr	DINT	21	Hardware address of current telephone number
RxIdleTime	UINT	18	Modem is disconnected, if no data transfer occurs for this time in seconds (s)

WriteTimeout	UDINT	19	Maximum write duration for a modem connection in milliseconds (ms).
RingCountSet	UDINT	20	Number of ringing tones before a call is accepted
ReCallIdleTime	UINT	53	Waiting time between calls in seconds (s).
ConnectTimeout	UINT	54	Time in seconds (s) to establish a connection.

STATISTICS

Name from import	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:

INI ENTRIES

ZENON6.INI

None

PROJECT.INI

None

LIMITATIONS

Only data tables (datablocks) also existing in the PLC can be accessed.

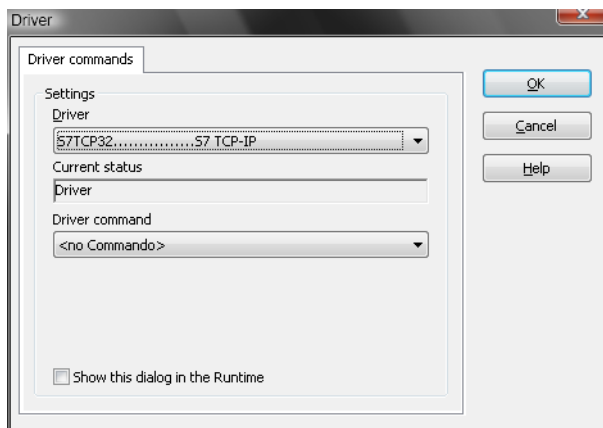
Supported data tables see driver objects.

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



Parameter	Description
Drivers	Drop-down list with all drivers which are loaded in the project.
Current status	Fixed entry which has no function in the current version.
Driver command	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.
▶ Driver - activate set setpoint value	Write set value to a driver is allowed.
▶ Driver - deactivate set setpoint value	Write set value to a driver is prohibited.
▶ Establish connecton with modem	Establish connection (for modem drivers) Opens the input fields for the hardware address and for the telephone number.
▶ Disconnect from modem	Terminate connection (for modem drivers)
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 7.50 -> Diagviewer.

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

```
%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 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

- ▶ Check configuration
- ▶ Test connection with external tools
- ▶ Data table numbers does not exist
- ▶ Offset in the data table does not exist
- ▶ Check driver communication error file

Block type (data table type) does not match object type.

In case of communication problems, the driver writes a detailed problem analysis into the driver communication error file. This file is stored in the project directory (RT\\FILES\\zenon\\custom\\log). The name of the file is _<drivename>.TXT.

The file can be opened with any text editor, e.g. Notepad.

For additional error analyses, please send a project backup and the "error file" to the support team responsible for you.