Contents

1 Welcome to COPA-DATA help .............................................................................................................. 5
2 S7 Driver for S7-1500/1200 .............................................................................................................. 5
3 S7TIA - data sheet .............................................................................................................................. 6
4 Driver history ....................................................................................................................................... 7
5 Requirements ....................................................................................................................................... 7
  5.1 PC ................................................................................................................................................... 7
  5.2 PLC ................................................................................................................................................. 8
6 Configuration ....................................................................................................................................... 8
  6.1 Creating a driver ............................................................................................................................. 9
  6.2 Settings in the driver dialog .......................................................................................................... 12
      6.2.1 General .................................................................................................................................. 13
      6.2.2 Options .................................................................................................................................. 16
      6.2.3 Connections .......................................................................................................................... 19
7 Creating variables ............................................................................................................................... 23
  7.1 Creating variables in the Editor .................................................................................................... 23
  7.2 Addressing ..................................................................................................................................... 27
  7.3 Driver objects and datatypes ....................................................................................................... 28
      7.3.1 Driver objects ....................................................................................................................... 28
      7.3.2 Mapping of the data types .................................................................................................... 29
  7.4 Creating variables by importing ................................................................................................. 30
      7.4.1 XML import ......................................................................................................................... 31
      7.4.2 DBF Import/Export .............................................................................................................. 32
      7.4.3 Variable import .................................................................................................................... 37
      7.4.4 Optimize TIA projects ........................................................................................................ 43
  7.5 Communication details (Driver variables) .................................................................................. 45
8 Driver-specific functions ................................................................................................................... 51
9 Driver command function .................................................................................................................. 51
10 Error analysis ..................................................................................................................................... 56
  10.1 Analysis tool ............................................................................................................................... 56
10.2 Check list ................................................................................................................................. 57
1 Welcome to COPA-DATA help

ZENON VIDEO-TUTORIALS

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

GENERAL HELP

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

PROJECT SUPPORT

You can receive support for any real project you may have from our Support Team, who you can contact via email at support@copadata.com.

LICENSES AND MODULES

If you find that you need other modules or licenses, our staff will be happy to help you. Email sales@copadata.com.

2 S7 Driver for S7-1500/1200

The S7 Driver for S7-1500/1200 driver uses enhanced TIA communication via the TCP/IP transport protocol to the S7-1200 and S7-1500. Access is by means of variables or via the symbolic TIA access path. Optimized modules are supported.

Block-wise flat access to the controller memory - similar to with PUT/GET - is not possible with the TIA protocol.
# S7TIA - data sheet

## General:

<table>
<thead>
<tr>
<th>Driver file name</th>
<th>S7TIA.exe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Driver name</td>
<td>S7-Treiber für S7-1200/1500</td>
</tr>
<tr>
<td>PLC types</td>
<td>S7-1200; S7-1500;</td>
</tr>
<tr>
<td>PLC manufacturer</td>
<td>Siemens</td>
</tr>
</tbody>
</table>

## Driver supports:

<table>
<thead>
<tr>
<th>Protocol</th>
<th>TCP/IP - RFC1006</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addressing: Address-based</td>
<td>Name based</td>
</tr>
<tr>
<td>Addressing: Name-based</td>
<td>--</td>
</tr>
<tr>
<td>Spontaneous communication</td>
<td>--</td>
</tr>
<tr>
<td>Polling communication</td>
<td>X</td>
</tr>
<tr>
<td>Online browsing</td>
<td>X</td>
</tr>
<tr>
<td>Offline browsing</td>
<td>X</td>
</tr>
<tr>
<td>Real-time capable</td>
<td>--</td>
</tr>
<tr>
<td>Blockwrite</td>
<td>X</td>
</tr>
<tr>
<td>Modern capable</td>
<td>--</td>
</tr>
<tr>
<td>RDA numerical</td>
<td>--</td>
</tr>
<tr>
<td>RDA String</td>
<td>--</td>
</tr>
<tr>
<td>Hysteresis</td>
<td>X</td>
</tr>
<tr>
<td>extended API</td>
<td>--</td>
</tr>
<tr>
<td>Supports status bit</td>
<td>WR-SUC</td>
</tr>
<tr>
<td>alternative IP address</td>
<td>--</td>
</tr>
</tbody>
</table>
4 Driver history

<table>
<thead>
<tr>
<th>Date</th>
<th>Build number</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>03.09.2015</td>
<td>22100</td>
<td>Created driver documentation</td>
</tr>
<tr>
<td>8/16/2017</td>
<td>40130</td>
<td>Driver supports TIA 14 projects</td>
</tr>
</tbody>
</table>

5 Requirements

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

5.1 PC

This driver supports a connection via the standard network card of the computer.
Make sure that the PLC and the computer are in the same network range and that the subnet masks are set accordingly on both devices.

5.2 PLC

The **S7 Driver for S7-1500/1200** driver uses enhanced TIA communication via the TCP/IP transport protocol to the S7-1200 and S7-1500. Access is by means of variables or via the symbolic TIA access path. Optimized modules are supported.

Block-wise flat access to the controller memory - similar to with PUT/GET - is not possible with the TIA protocol.

6 Configuration

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

💡 Information

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

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

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Available drivers | List of all available drivers. The display is in a tree structure: 

  - `[]` expands the folder structure and shows the drivers contained therein.
  - `[-]` reduces the folder structure

  Default: **no selection**

| Driver name      | Unique **Identification** of the driver. 

  Default: **Empty**

  The input field is pre-filled with the pre-defined **Identification** after selecting a driver from the list of available drivers.

| Driver information | Further information on the selected driver. 

  Default: **Empty**

  The information on the selected driver is shown in this area after selecting a driver.
CLOSE DIALOG

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OK</td>
<td>Accepts all settings and opens the driver configuration dialog of the selected driver.</td>
</tr>
<tr>
<td>Cancel</td>
<td>Discards all changes and closes the dialog.</td>
</tr>
<tr>
<td>Help</td>
<td>Opens online help.</td>
</tr>
</tbody>
</table>

Information

The content of this dialog is saved in the file called Treiber_[Language].xml. You can find this file in the following folder:

\C:\ProgramData\COPA-DATA\zenon\[version number].

CREATE NEW DRIVER

In order to create a new driver:

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

3. Select the desired driver and name it in the **Driver name** input field. This input field corresponds to the **Identification** property. The name of the selected driver is automatically inserted into this input field by default. The following is applicable for the **Driver name**:

   - The **Driver name** must be unique. If a driver is used more than once in a project, a new name has to be given each time. This is evaluated by clicking on the **OK** button. If the driver is already present in the project, this is shown with a warning dialog.

   - The **Driver name** is part of the file name. Therefore it may only contain characters which are supported by the operating system. Invalid characters are replaced by an underscore (\_).

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

4. Confirm the dialog by clicking on the **OK** button. The configuration dialog for the selected driver is opened.

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

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

---

**ZENON PROJECT**

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

- Intern
- MathDr32
- SysDrv

**Information**

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

---

**6.2 Settings in the driver dialog**

You can change the following settings of the driver:
6.2.1 General

The configuration dialog is opened when a driver is created. In order to be able to open the dialog later for editing, double click on the driver in the list or click on the **Configuration** property.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mode</strong></td>
<td>Allows to switch between hardware mode and simulation mode</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>▶ <strong>Hardware</strong>: A connection to the control is established.</td>
</tr>
<tr>
<td></td>
<td>▶ <strong>Simulation - static</strong>: No communication between to the control is established, the values are simulated by the driver. In this mode 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.</td>
</tr>
<tr>
<td></td>
<td>▶ <strong>Simulation - counting</strong>: No communication between to the control is established, the values are simulated by the driver. In this mode the driver increments the values</td>
</tr>
</tbody>
</table>
### Configuration

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>within a value range automatically.</td>
</tr>
<tr>
<td></td>
<td>- Simulation - programmed: No communication is established to the PLC. The values are calculated by a freely programmable simulation project. The simulation project is created with the help of the zenon Logic Workbench and runs in a zenon Logic Runtime which is integrated in the driver. For details see chapter Driver simulation (main.chm::/25206.htm).</td>
</tr>
<tr>
<td>Keep update list in the memory</td>
<td>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.</td>
</tr>
<tr>
<td>Output can be written</td>
<td>- Active: Outputs can be written.</td>
</tr>
<tr>
<td></td>
<td>- Inactive: Writing of outputs is prevented.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong>: Not available for every driver.</td>
</tr>
<tr>
<td>Variable image remanent</td>
<td>This option saves and restores the current value, time stamp and the states of a data point. Fundamental requirement: The variable must have a valid value and time stamp.</td>
</tr>
<tr>
<td></td>
<td>The variable image is saved in hardware mode if one of these statuses is active:</td>
</tr>
<tr>
<td></td>
<td>- User status M1 (0) to M8 (7)</td>
</tr>
<tr>
<td></td>
<td>- REVISION(9)</td>
</tr>
<tr>
<td></td>
<td>- AUS(20)</td>
</tr>
<tr>
<td></td>
<td>- ERSATZWERT(27)</td>
</tr>
<tr>
<td></td>
<td>The variable image is always saved if:</td>
</tr>
<tr>
<td></td>
<td>- the variable is of the object type <strong>Driver variable</strong></td>
</tr>
</tbody>
</table>
|                            |     - the driver runs in simulation mode. (not
## Configuration

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
</table>
|        | programmed simulation) The following states are not restored at the start of the Runtime:  
- SELECT(8)  
- WR-ACK(40)  
- WR-SUC(41)  
The mode Simulation - programmed at the driver start is not a criterion in order to restore the remanent variable image. |
| **Stop on Standby Server** | Setting for redundancy at drivers which allow only one communication connection. For this the driver is stopped at the Standby Server and only started at the upgrade.  
**Attention:** If this option is active, the gapless archiving is no longer guaranteed.  
- **Active:**  
  Sets the driver at the not-process-leading Server automatically in a stop-like state. In contrast to stopping via driver command, the variable does not receive status switched off ([statusverarbeitung.chm::/24150.htm](statusverarbeitung.chm::/24150.htm)) but an empty value. This prevents that at the upgrade to the Server irrelevant values are created in the AML, CEL and Historian.  
  Default: inactive  
  **Note:** Not available if the CE terminal serves as a data server. You can find further information in the zenon Operator manual in the CE terminal as a data server chapter. |
| **Global Update time** | Setting for the global update times in milliseconds:  
- **Active:**  
  The set Global update time is used for all variables in the project. The priority set at the variables is not used.  
- **Inactive:**  
  The set priorities are used for the individual variables. |
### Option Description

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

### Priority

The polling times for the individual priority classes are set here. All variables with the according priority are polled in the set time.

The variables are allocated separately in the settings of the variable properties. The communication of the individual variables can be graded according to importance or required topicality using the priority classes. Thus the communication load is distributed better.

**Attention:** Priority classes are not supported by each driver, e.g. spontaneously communicating zenon drivers.

### CLOSE DIALOG

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OK</strong></td>
<td>Applies all changes in all tabs and closes the dialog.</td>
</tr>
<tr>
<td><strong>Cancel</strong></td>
<td>Discards all changes in all tabs and closes the dialog.</td>
</tr>
<tr>
<td><strong>Help</strong></td>
<td>Opens online help.</td>
</tr>
</tbody>
</table>

### UPDATE TIME FOR SPONTANEOUS DRIVERS

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


### 6.2.2 Options

A TIA project file is selected in the **Options** tab.
Note: The user interface of the dialog is only available in English.
Parameter | Description
--- | ---
Options for variable import: | Option field to state the source for the online import of variables. In addition, this setting of parameters is important for correct communication of TIA symbols, including to zenon Runtime.

- **Symbols from TIA project:**
  The symbolic addresses for communication are determined using a linked TIA project file.
  The variables can be imported into the zenon Editor during online import of this TIA project file.
  The corresponding TIA project file must be linked in the TIA project option field.
  **Note:** TIA projects up to version TIA 14 SP1 can be selected. Alternatively, select the Symbols from PLC option.

- **Symbols from precompiled file:**
  The symbolic addresses for communication are determined by means of a linked project file. The linked project file was optimized using the TIAtoAGL.exe command line tool (on page 43).
  The variables can be imported into the zenon Editor with this option during an import of an optimized TIA project file.
  **Note:** The optimization of TIA project files is carried out with the TIAtoAGL.exe command line tool. You can find further information about this in the Optimizing TIA projects (on page 43) chapter. **Note:** TIA projects up to version TIA 14 SP1 can be selected. Alternatively, select the Symbols from PLC option.

- **Symbols from PLC:**
  The symbolic addresses for communication are read by the PLC directly when the driver starts.
  With this option, the variables can be imported into the zenon Editor directly by the PLC during online import.
  If this option is activated, no TIA project file is required.

Also note the Variable import (on page 37) chapter for further information.
Configuration

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TIA project (must be present on Runtime machine)</strong></td>
<td>Path and name of the TIA project. clicking on ... opens the dialog to select the TIA project file. TIA projects up to version TIA 14 SP1 can be selected.</td>
</tr>
<tr>
<td>Default:</td>
<td></td>
</tr>
<tr>
<td>› Empty (with initial selection)</td>
<td></td>
</tr>
<tr>
<td>› Last-opened TIA project files</td>
<td></td>
</tr>
</tbody>
</table>

⚠️ **Attention**

The TIA project must be used on the Editor and Runtime computer in the same version that is currently on the PLC.

CLOSE DIALOG

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OK</strong></td>
<td>Applies settings and closes the dialog.</td>
</tr>
<tr>
<td><strong>Cancel</strong></td>
<td>Discards all changes and closes the dialog.</td>
</tr>
<tr>
<td><strong>Help</strong></td>
<td>Opens online help.</td>
</tr>
</tbody>
</table>

6.2.3 Connections

The connection to the PLC is configured in the Connections tab.
CONFIDENCE

Note: This dialog is only available in English.

CONNECTIONS

Overview of the configured connections to the PLC.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| Connection list | List with all configured connections:  
  - **CPU name:**  
    Name of the CPU in the TIA project as configured in the area **Edit connection**.  
  - **Net address:**  
    Net address of the connection as configured in the area **Edit connection**.  

  Displays configured connection names with the corresponding net address. The connection parameters in the **Edit connection** area are displayed when selecting the connection name.  

| New | Create new connection.  
  Creates a new connection and unlocks fields in the **Edit connection** area for editing. |
| Delete | Deletes the selected connection from the connection list.  
  **Note:** The selected connection is deleted without requesting confirmation. |
| Edit | Edit existing connection |
The configuration of the selected connection can be amended in the Edit connection area. Clicking on the Edit button unlocks the fields in the Edit connection area for editing.

**EDIT CONNECTION**

Configuration of a connection to the PLC.
If there is no connection open for creation or editing, the input fields are grayed out.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Net address</strong></td>
<td>Corresponds to the Net address property in variable configuration. Each net address can only be issued once per driver. Input range: 0 - 255</td>
</tr>
<tr>
<td><strong>CPU name</strong></td>
<td>Name of the CPU in the TIA project. Corresponds to the naming of the station in the TIA project. Default: CPU_1 Input range: 32 characters</td>
</tr>
</tbody>
</table>

**Note:** If the CPU name does not correspond to the station name used in TIA project, the variables cannot be imported.

A distinction between capital letters and small letters should be made. If the Symbols from PLC option field has been activated in the Options tab, the name has no effect on the import.

Ensure that no spaces are configured in a CPU name. If this is the case, validation fails even when entry is correct. No error message is displayed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Variable prefix</strong></td>
<td>Prefix for the variables names for online import (on page 37). Characters configured here are placed in front of the variable name in the zenon project configuration during an online import.</td>
</tr>
<tr>
<td><strong>IP address</strong></td>
<td>IP address of the controller. Incorrect entries are suppressed (for letters) or are automatically corrected to the value 255 when the input field is left.</td>
</tr>
<tr>
<td><strong>Save</strong></td>
<td>Saves configuration of the new or amended connections.</td>
</tr>
<tr>
<td><strong>Cancel</strong></td>
<td>Discards all configurations and cancels the editing.</td>
</tr>
</tbody>
</table>
CLOSE DIALOG

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OK</td>
<td>Applies all changes in all tabs and closes the dialog.</td>
</tr>
<tr>
<td>Cancel</td>
<td>Discards all changes in all tabs and closes the dialog.</td>
</tr>
<tr>
<td>Help</td>
<td>Opens online help.</td>
</tr>
</tbody>
</table>

**Note:** If an existing connection is open for amendment, the OK button is not active. Save the connection first by clicking on the Save button.

CREATE A NEW CONNECTION

To create a new connection:

1. Click on the New button. The properties in the Edit connection area are made available.
2. Configure the connection settings.
3. Save the connection by clicking on the Save button. The connection is shown in the connection list.

EDIT EXISTING CONNECTION

To edit an existing connection:

1. Select the connection in the connection list.
2. Click on the Edit button. The properties in the Edit connection area are unlocked for editing.
3. Amend the connection settings.
4. Secure the amended connection parameters by clicking on the Save button.

DELETE EXISTING CONNECTION

To delete an existing connection:

1. Select the connection in the connection list.
2. Click on the Delete button.
3. The connection will be deleted from the list without requesting confirmation.

WARNING DIALOG

An input test is carried out by clicking on the Save button. Incorrect inputs are visualized with a corresponding warning dialog.
Creating variables

Net address already issued:

![Net address already issued dialog]

Note: This dialog is only available in English.

Net address invalid:

![Net address invalid dialog]

CPU name empty:

![CPU name empty dialog]

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:
Creating variables

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

![Image of variable configuration dialog]

The dialog for configuring variables is opened

2. Configure the variable

3. The settings that are possible depends on the type of variables
## CREATE VARIABLE DIALOG

![Create variable dialog](image)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td>Distinct name of the variable. If a variable with the same name already exists in the project, no additional variable can be created with this name. Maximum length: 128 characters</td>
</tr>
<tr>
<td></td>
<td><strong>Attention:</strong> the characters # and @ are not permitted in variable names. If non-permitted characters are used, creation of variables cannot be completed and the <strong>Finish</strong> button remains inactive. <strong>Note:</strong> For some drivers, the addressing is possible over the property <strong>Symbolic address</strong>, as well.</td>
</tr>
<tr>
<td><strong>Drivers</strong></td>
<td>Select the desired driver from the drop-down list.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If no driver has been opened in the project, the driver for internal variables (<a href="">Intern.exe</a>) is automatically loaded.</td>
</tr>
<tr>
<td><strong>Driver Object Type</strong></td>
<td>Select the appropriate driver object type from the drop-down list.</td>
</tr>
<tr>
<td>(<a href="">cti.chm::/28685.htm</a>)</td>
<td></td>
</tr>
<tr>
<td><strong>Data Type</strong></td>
<td>Select the desired data type. Click on the ... button to open the</td>
</tr>
</tbody>
</table>
Creating variables

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>selection dialog</td>
<td></td>
</tr>
<tr>
<td>Array settings</td>
<td>Expanded settings for array variables. You can find details in the Arrays chapter.</td>
</tr>
<tr>
<td>Addressing options</td>
<td>Expanded settings for arrays and structure variables. You can find details in the respective section.</td>
</tr>
<tr>
<td>Automatic addressing</td>
<td>Expanded settings for arrays and structure variables. You can find details in the respective section.</td>
</tr>
</tbody>
</table>

SYMBOLIC ADDRESS

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

Maximum length: 1024 characters.

The following drivers support the Symbolic address:

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

INHERITANCE FROM DATA TYPE

Measuring range, Signal range and Set value are always:

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

**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

<table>
<thead>
<tr>
<th>Group/Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General</strong></td>
<td>Property group for general settings.</td>
</tr>
<tr>
<td><strong>Name</strong></td>
<td>Freely definable name.</td>
</tr>
<tr>
<td><strong>Identification</strong></td>
<td>Freely definable identification.</td>
</tr>
<tr>
<td></td>
<td>E.g. for Resources label, comments, ...</td>
</tr>
<tr>
<td><strong>Symbolic address</strong></td>
<td>The symbolic address contains the TIA access path.</td>
</tr>
<tr>
<td></td>
<td>The <strong>Symbolic address</strong> property can be used for addressing as an alternative to the <strong>Name</strong> or <strong>Identification</strong> 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.</td>
</tr>
<tr>
<td></td>
<td>Maximum length: 1024 characters.</td>
</tr>
<tr>
<td><strong>Net address</strong></td>
<td>Network address of variables.</td>
</tr>
<tr>
<td></td>
<td>This address refers to the bus address in the connection configuration of the driver. This defines the PLC, on which the variable resides.</td>
</tr>
<tr>
<td><strong>Data block</strong></td>
<td>not used for this driver</td>
</tr>
<tr>
<td><strong>Offset</strong></td>
<td>not used for this driver</td>
</tr>
<tr>
<td><strong>Alignment</strong></td>
<td>not used for this driver</td>
</tr>
<tr>
<td><strong>Bit number</strong></td>
<td>not used for this driver</td>
</tr>
<tr>
<td><strong>String length</strong></td>
<td>Only available for String variables.</td>
</tr>
<tr>
<td></td>
<td>Maximum number of characters that the variable can take.</td>
</tr>
<tr>
<td><strong>Driver connection/Driver</strong></td>
<td>Object type of the variables. Depending on the driver used, is selected when the variable is created and can be changed here.</td>
</tr>
<tr>
<td><strong>Object Type</strong></td>
<td>Data type of the variable. Is selected during the creation of the</td>
</tr>
</tbody>
</table>
Creating variables

<table>
<thead>
<tr>
<th>Group/Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>connection/Data Type</td>
<td>variable; the type can be changed here.</td>
</tr>
<tr>
<td></td>
<td><strong>Attention:</strong> If you change the data type later, all other properties of the variable must be checked and adjusted, if necessary.</td>
</tr>
</tbody>
</table>

**Attention**

The option **Accessible from HMI** can be set for variables in TIA portal.

If this option is deactivated, the corresponding variable from zenon cannot be described. The variable can be described by the TIA portal.

If the option is deactivated and an additional change is subsequently made in the module, the **Accessible from HMI** option is activated when reloading. Zenon cannot access the variable with write permission.

### 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:

<table>
<thead>
<tr>
<th>Driver Object Type</th>
<th>Channe l type</th>
<th>Read</th>
<th>Write</th>
<th>Supported data types</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symbolic variable</td>
<td>8</td>
<td>X</td>
<td>X</td>
<td>BOOL, SINT, USINT, INT, UINT, DINT, UDINT, LINT, ULINT, REAL, LREAL, STRING, WSTRING, DATE_AND_TIME</td>
<td>TIA variable from the S7, symbolically addressed</td>
</tr>
<tr>
<td>Communication details</td>
<td>35</td>
<td>X</td>
<td>X</td>
<td>BOOL, SINT, USINT, INT, UINT, DINT, UDINT, REAL, STRING</td>
<td>Variables for the static analysis of the communication; is transferred between driver and Runtime (not</td>
</tr>
</tbody>
</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.

<table>
<thead>
<tr>
<th>PLC</th>
<th>zenon</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>S7_Bool</td>
<td>BOOL</td>
<td>8</td>
</tr>
<tr>
<td>S7_USInt</td>
<td>USINT</td>
<td>9</td>
</tr>
<tr>
<td>S7_Char</td>
<td>USINT</td>
<td>9</td>
</tr>
<tr>
<td>S7_Byte</td>
<td>USINT</td>
<td>9</td>
</tr>
<tr>
<td>S7_SInt</td>
<td>SINT</td>
<td>10</td>
</tr>
<tr>
<td>S7_UInt</td>
<td>UINT</td>
<td>2</td>
</tr>
<tr>
<td>WChar</td>
<td>UINT</td>
<td>2</td>
</tr>
<tr>
<td>S7_Int</td>
<td>INT</td>
<td>1</td>
</tr>
<tr>
<td>S7_UDInt</td>
<td>UDINT</td>
<td>4</td>
</tr>
<tr>
<td>S7_Time_Of-Day</td>
<td>UDINT</td>
<td>4</td>
</tr>
</tbody>
</table>

Note: The addressing and the behavior is the same for most zenon drivers.

You can find detailed information on this in the Communication details (Driver variables) (on page 45) chapter.

Key:

- X: supported
- --: not supported
Creating variables

<table>
<thead>
<tr>
<th>PLC</th>
<th>zenon</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>S7_DInt</td>
<td>DINT</td>
<td>3</td>
</tr>
<tr>
<td>S7_Time</td>
<td>DINT</td>
<td>3</td>
</tr>
<tr>
<td>S7_ULInt</td>
<td>ULINT</td>
<td>27</td>
</tr>
<tr>
<td>S7_LInt</td>
<td>LINT</td>
<td>26</td>
</tr>
<tr>
<td>S7_Real</td>
<td>REAL</td>
<td>5</td>
</tr>
<tr>
<td>S7_LReal</td>
<td>LREAL</td>
<td>6</td>
</tr>
<tr>
<td>S7_String</td>
<td>STRING</td>
<td>12</td>
</tr>
<tr>
<td>S7_WString</td>
<td>WSTRING</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>DATE</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>TIME</td>
<td>17</td>
</tr>
<tr>
<td>S7_DTL, S7_Date_And_Time</td>
<td>DATE_AND_TIME</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>TOD (Time of Day)</td>
<td>19</td>
</tr>
</tbody>
</table>

DATA TYPE

The term data type is the internal numerical identification of the data type. It is also used for the extended DBF import/export of the variables.

7.4 Creating variables by importing

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

Information

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

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

- **Import:**
  The element is imported as a new element.

- **Overwrite:**
  The element is imported and overwrites a pre-existing element.

- **Do not import:**
  The element is not imported.

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

**REQUIREMENTS**

The following conditions are applicable during import:

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

- **Consistency**
  The XML file to be imported has to be consistent. There is no plausibility check on importing the file. If there are errors in the import file, this can lead to undesirable effects in the project. Particular attention must be paid to this, primarily if not all properties exist in the XML file and these are then filled with default values. E.g.: A binary variable has a limit value of 300.

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

**Hint**

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

Data can be exported to and imported from dBase.

**Information**

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

**IMPORT DBF FILE**

To start the import:

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

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

**Information**

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

**EXPORT DBF FILE**

To start the export:

1. right-click on the variable list
2. in the drop-down list of Extended export/import... select the Export dBase... command
3. follow the export assistant
Attention

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

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

Information

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

FILE STRUCTURE OF THE DBASE EXPORT FILE

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

Attention

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

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

STRUCTURE

<table>
<thead>
<tr>
<th>Identification</th>
<th>Type</th>
<th>Field size</th>
<th>Comment</th>
</tr>
</thead>
</table>
| KANALNAME      | Char | 128        | Variable name.  
The length can be limited using the MAX_LAENGE entry in the project.ini file. |
| KANAL_R        | C    | 128        | The original name of a variable that is to be replaced by the new name entered under "VARIABLENNAME" (variable name) (field/column must be entered |
### Creating variables

<table>
<thead>
<tr>
<th>Identification</th>
<th>Type</th>
<th>Field size</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>manually). The length can be limited using the <strong>MAX_LAENGE</strong> entry in the <strong>project.ini</strong> file.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>KANAL_D</strong></td>
<td>Log</td>
<td>1</td>
<td>The variable is deleted with the 1 entry (field/column has to be created by hand).</td>
</tr>
<tr>
<td><strong>TAGNR</strong></td>
<td>C</td>
<td>128</td>
<td>Identification. The length can be limited using the <strong>MAX_LAENGE</strong> entry in the <strong>project.ini</strong> file.</td>
</tr>
<tr>
<td><strong>EINHEIT</strong></td>
<td>C</td>
<td>11</td>
<td>Technical unit</td>
</tr>
<tr>
<td><strong>DATENART</strong></td>
<td>C</td>
<td>3</td>
<td>Data type (e.g. bit, byte, word, ...) corresponds to the data type.</td>
</tr>
<tr>
<td><strong>KANALTYP</strong></td>
<td>C</td>
<td>3</td>
<td>Memory area in the PLC (e.g. marker area, data area, ...) corresponds to the driver object type.</td>
</tr>
<tr>
<td><strong>HWKANAL</strong></td>
<td>Num</td>
<td>3</td>
<td>Net address</td>
</tr>
<tr>
<td><strong>BAUSTEIN</strong></td>
<td>N</td>
<td>3</td>
<td>Datablock address (only for variables from the data area of the PLC)</td>
</tr>
<tr>
<td><strong>ADRESSE</strong></td>
<td>N</td>
<td>5</td>
<td>Offset</td>
</tr>
<tr>
<td><strong>BITADR</strong></td>
<td>N</td>
<td>2</td>
<td>For bit variables: bit address For byte variables: 0=lower, 8=higher byte For string variables: Length of string (max. 63 characters)</td>
</tr>
<tr>
<td><strong>ARRAYSIZE</strong></td>
<td>N</td>
<td>16</td>
<td>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</td>
</tr>
<tr>
<td><strong>LES_SCHR</strong></td>
<td>L</td>
<td>1</td>
<td>Write-Read-Authorization 0: Not allowed to set value. 1: Allowed to set value.</td>
</tr>
<tr>
<td><strong>MIT_ZEIT</strong></td>
<td>R</td>
<td>1</td>
<td>time stamp in zenon (only if supported by the driver)</td>
</tr>
<tr>
<td><strong>OBJEKT</strong></td>
<td>N</td>
<td>2</td>
<td>Driver-specific ID number of the primitive object comprises TREIBER-OBJEKTYP and DATENTYP</td>
</tr>
<tr>
<td>Identification</td>
<td>Type</td>
<td>Field size</td>
<td>Comment</td>
</tr>
<tr>
<td>------------------</td>
<td>------</td>
<td>------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SIGMIN</td>
<td>Float</td>
<td>16</td>
<td>Non-linearized signal - minimum (signal resolution)</td>
</tr>
<tr>
<td>SIGMAX</td>
<td>F</td>
<td>16</td>
<td>Non-linearized signal - maximum (signal resolution)</td>
</tr>
<tr>
<td>ANZMIN</td>
<td>F</td>
<td>16</td>
<td>Technical value - minimum (measuring range)</td>
</tr>
<tr>
<td>ANZMAX</td>
<td>F</td>
<td>16</td>
<td>Technical value - maximum (measuring range)</td>
</tr>
<tr>
<td>ANZKOMMA</td>
<td>N</td>
<td>1</td>
<td>Number of decimal places for the display of the values (measuring range)</td>
</tr>
<tr>
<td>UPDATERATE</td>
<td>F</td>
<td>19</td>
<td>Update rate for mathematics variables (in sec, one decimal possible)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>not used for all other variables</td>
</tr>
<tr>
<td>MEMTIEFE</td>
<td>N</td>
<td>7</td>
<td>Only for compatibility reasons</td>
</tr>
<tr>
<td>HDRATE</td>
<td>F</td>
<td>19</td>
<td>HD update rate for historical values (in sec, one decimal possible)</td>
</tr>
<tr>
<td>HDTIEFE</td>
<td>N</td>
<td>7</td>
<td>HD entry depth for historical values (number)</td>
</tr>
<tr>
<td>NACHSORT</td>
<td>R</td>
<td>1</td>
<td>HD data as postsorted values</td>
</tr>
<tr>
<td>DRRATE</td>
<td>F</td>
<td>19</td>
<td>Updating to the output (for zenon DDE server, in [s], one decimal possible)</td>
</tr>
<tr>
<td>HYST_PLUS</td>
<td>F</td>
<td>16</td>
<td>Positive hysteresis, from measuring range</td>
</tr>
<tr>
<td>HYST_MINUS</td>
<td>F</td>
<td>16</td>
<td>Negative hysteresis, from measuring range</td>
</tr>
<tr>
<td>PRIOR</td>
<td>N</td>
<td>16</td>
<td>Priority of the variable</td>
</tr>
<tr>
<td>REAMATRIZE</td>
<td>C</td>
<td>32</td>
<td>Allocated reaction matrix</td>
</tr>
<tr>
<td>ERSATZWERT</td>
<td>F</td>
<td>16</td>
<td>Substitute value, from measuring range</td>
</tr>
<tr>
<td>SOLLMIN</td>
<td>F</td>
<td>16</td>
<td>Minimum for set value actions, from measuring range</td>
</tr>
<tr>
<td>SOLLMAX</td>
<td>F</td>
<td>16</td>
<td>Maximum for set value actions, from measuring range</td>
</tr>
<tr>
<td>VOMSTANDBY</td>
<td>R</td>
<td>1</td>
<td>Get value from standby server; the value of the variable is not requested from the server but from the Standby Server in redundant networks</td>
</tr>
<tr>
<td>RESOURCE</td>
<td>C</td>
<td>128</td>
<td>Resources label. Free string for export and display in lists.</td>
</tr>
</tbody>
</table>
Creating variables

<table>
<thead>
<tr>
<th>Identification</th>
<th>Type</th>
<th>Field size</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>AKTIV1</td>
<td>R</td>
<td>1</td>
<td>Limit value active (per limit value available)</td>
</tr>
<tr>
<td>GRENZWERT1</td>
<td>F</td>
<td>20</td>
<td>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)</td>
</tr>
<tr>
<td>SCHWERT1</td>
<td>F</td>
<td>16</td>
<td>Threshold value for limit value</td>
</tr>
<tr>
<td>HYSTERESE1</td>
<td>F</td>
<td>14</td>
<td>Is not used</td>
</tr>
<tr>
<td>BLINKEN1</td>
<td>R</td>
<td>1</td>
<td>Set blink attribute</td>
</tr>
<tr>
<td>BTB1</td>
<td>R</td>
<td>1</td>
<td>Logging in CEL</td>
</tr>
<tr>
<td>ALARM1</td>
<td>R</td>
<td>1</td>
<td>Alarm</td>
</tr>
</tbody>
</table>

**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:

<table>
<thead>
<tr>
<th>Identification</th>
<th>Type</th>
<th>Field size</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>AKTIV1</td>
<td>R</td>
<td>1</td>
<td>Limit value active (per limit value available)</td>
</tr>
<tr>
<td>GRENZWERT1</td>
<td>F</td>
<td>20</td>
<td>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)</td>
</tr>
<tr>
<td>SCHWERT1</td>
<td>F</td>
<td>16</td>
<td>Threshold value for limit value</td>
</tr>
<tr>
<td>HYSTERESE1</td>
<td>F</td>
<td>14</td>
<td>Is not used</td>
</tr>
<tr>
<td>BLINKEN1</td>
<td>R</td>
<td>1</td>
<td>Set blink attribute</td>
</tr>
<tr>
<td>BTB1</td>
<td>R</td>
<td>1</td>
<td>Logging in CEL</td>
</tr>
<tr>
<td>ALARM1</td>
<td>R</td>
<td>1</td>
<td>Alarm</td>
</tr>
</tbody>
</table>
Creating variables

<table>
<thead>
<tr>
<th>Identification</th>
<th>Type</th>
<th>Field size</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>DRUCKEN1</td>
<td>R</td>
<td>1</td>
<td>Printer output (for CEL or Alarm)</td>
</tr>
<tr>
<td>QUITTIER1</td>
<td>R</td>
<td>1</td>
<td>Must be acknowledged</td>
</tr>
<tr>
<td>LOESCHE1</td>
<td>R</td>
<td>1</td>
<td>Must be deleted</td>
</tr>
<tr>
<td>VARIABLE1</td>
<td>R</td>
<td>1</td>
<td>Dyn. limit value linking the limit is defined by an absolute value (see field GRENZWERTx).</td>
</tr>
<tr>
<td>FUNC1</td>
<td>R</td>
<td>1</td>
<td>Functions linking</td>
</tr>
<tr>
<td>ASK_FUNC1</td>
<td>R</td>
<td>1</td>
<td>Execution via Alarm Message List</td>
</tr>
<tr>
<td>FUNC_NR1</td>
<td>N</td>
<td>10</td>
<td>ID number of the linked function (if &quot;-1&quot; is entered here, the existing function is not overwritten during import)</td>
</tr>
<tr>
<td>A_GRUPPE1</td>
<td>N</td>
<td>10</td>
<td>Alarm/Event Group</td>
</tr>
<tr>
<td>A_KLASSE1</td>
<td>N</td>
<td>10</td>
<td>Alarm/Event Class</td>
</tr>
<tr>
<td>MIN_MAX1</td>
<td>C</td>
<td>3</td>
<td>Minimum, Maximum</td>
</tr>
<tr>
<td>FARBE1</td>
<td>N</td>
<td>10</td>
<td>Color as Windows coding</td>
</tr>
<tr>
<td>GRENZTXT1</td>
<td>C</td>
<td>66</td>
<td>Limit value text</td>
</tr>
<tr>
<td>A_DELAY1</td>
<td>N</td>
<td>10</td>
<td>Time delay</td>
</tr>
<tr>
<td>INVISIBLE1</td>
<td>R</td>
<td>1</td>
<td>Invisible</td>
</tr>
</tbody>
</table>

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 Variable import

Carry out the following steps to import variables into the zenon Editor:

1. Start the online import.
   Select Import variables from driver in the toolbar or in the driver's context menu.
   The import wizard is opened with a dialog for prefiltering (on page 39).
   **Note:** When importing variables from a PLC, a dialog to select the configured connections is shown.

2. Configure prefiltering:
a) Issue filter criteria in the **Substring filter criteria for variable name** input field.  
In the following variable selection dialog (on page 41), only variables that correspond to the syntax entered are offered.

b) Activate the **only variables with "Accessible from HMI"** checkbox.  
In the following variable selection dialog, only variables that have **Accessible from HMI** activated in the TIA project configuration are offered.

c) Select a node from the existing TIA project configuration.  
In the following variable selection dialog (on page 41), only variables that are assigned to the selected node in the TIA project configuration are offered.

- **Please note:**  
If an optimized TIA project file is processed for the online import, there is no prefiltering. Online import then starts directly with the variable selection dialog (on page 41).
  - TIA projects are optimized with the **TIAtoAGL.exe** command line tool
  - In the driver configuration, activate the use of optimized TIA project file in the **Options** tab by activating the **Symbols from precompiled file** option field.

1. Click on **OK**.  
The variable selection dialog (on page 41) is opened.

2. Select the variables to be imported from the selection window by clicking with the mouse.  
Multiple selection is possible. Also use the filter and sorting possibilities to further limit the display in the selection window.
   a) Apply the selected variables by clicking on the **Add** button in the import window.  
The selected variables are shown in the import window.
   b) If necessary, select further variables and transfer these to the import window with the **Add** button.
   c) You can also deselect variables again by clicking on **Remove**.

3. Click on **OK**.  
The import of variables is started. The selected variables are created in the zenon Editor.  
Pre-existing variables are not created again.

If an error occurs, a corresponding message is displayed.

**GENERAL NOTES**

- The variable selection dialog is limited to 500,000 variables. If no prefilter is used and the maximum number is exceeded, this is shown with a corresponding dialog:
Creating variables

Only 500,000 variables are shown for selection in the following variable selection dialog (on page 41).

- *Char* and *WChar* data type variables are supported.  
  **Note:** Because these are not IEC61131 data types, variables of these data types are converted to the corresponding IEC data types:
  - *Char* to *USInt*
  - *WChar* to *UInt*

**NAME OF THE VARIABLE IN ZENON**

The naming of variables via online import in the zenon project configuration comprises:

- **Prefix**  
  As configured in the driver configuration in the *Connections* tab in the *Variable prefix* input field.

- **Datablock name**  
  Is only used for Datablock variables form a TIA project configuration.

- Variable name as configured for the PLC or in the TIA project configuration.  
  This can include more than one section with structure variables.

**7.4.3.1 Prefilter**

In this dialog, a prefilter can be configured for variable selection.
This dialog is not offered if an optimized TIA project file (on page 43) has been configured.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Substring filter criteria for variable name (empty=unfiltered)</td>
<td>Input field to enter a character chain that must be included in the variable name on the PLC in order for this to be shown in the variable selection dialog (on page 41).</td>
</tr>
<tr>
<td>only variables with &quot;Accessible from HMI&quot;</td>
<td>Checkbox for the activation of prefiltering with the Accessible from HMI TIA project property.</td>
</tr>
<tr>
<td></td>
<td>In the subsequent variable selection dialog (on page 41), only variables that have this property active in the TIA project configuration are offered.</td>
</tr>
<tr>
<td>Node to import</td>
<td>Display of the TIA project configuration in the tree structure. Preselection with a mouse click.</td>
</tr>
<tr>
<td></td>
<td>In the subsequent variable selection dialog (on page 41), only the variables that correspond to the selected level in the TIA project configuration and lower levels are offered for selection on the PLC.</td>
</tr>
</tbody>
</table>
### 7.4.3.2 Variable selection dialog

In this dialog, variables can be selected from the PLC (from the linked TIA project) for creation of these new variables in the zenon Editor.

The TIA project file is assigned in the driver configuration in the Options (on page 16) tab. This TIA project must be present on the computer.

**Information**

The height and width of the dialog is freely scalable.

#### SELECTION WINDOW

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Selection window</strong></td>
<td>All variables available for online import are displayed in the selection window. The origin of this list is the variables on the PLC and the preselection in the prior filter dialog.</td>
</tr>
<tr>
<td></td>
<td><strong>Symbolic address:</strong> Name of the symbolic address of a variable from the PLC.</td>
</tr>
<tr>
<td></td>
<td><strong>S7 datatype:</strong></td>
</tr>
</tbody>
</table>
### Creating variables

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Parameter</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td></td>
<td>S7 data type of the variable.</td>
</tr>
</tbody>
</table>
|                 |   - *Datatype:*  
|                 |     Data type of the variable.                                                                                                               |
|                 | The column width can be increased or decreased with the mouse button held down.  
|                 | The list can be sorted and filtered:                                                                                                        |
|                 |   - Click on the corresponding column overview for sorting.  
|                 |     Another click reverses the sorting order. The respective sorting order is visualized with a cursor symbol.  
|                 |   - To filter, enter the desired filter criteria into the corresponding input field directly underneath the column heading. An input always correspond to a “contains”.  
|                 |     Additional wild cards are not required.                                                                                                  |

### Import window

| Import window | All variables selected for online import are displayed in the import window.  
|---------------|-----------------------------------------------------------------------------|
|               | The column width can be increased or decreased with the mouse button held down.  
|               | The import window list can be sorted:                                                                                                     |
|               |   - Click on the corresponding column overview for sorting.  
|               |     Another click reverses the sorting order. The respective sorting order is visualized with a cursor symbol.  
|               | Default: *empty*                                                                                                                          |

### Footer

| Footer         | For both the selection window and the import window, a footer visualizes the number of available and selected variables:  
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[total number of available variables]/[total number of selected variables]</td>
</tr>
</tbody>
</table>

### Add

<table>
<thead>
<tr>
<th>Add</th>
<th>Transfers all variables highlighted in the selection window to the import window.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Multiple selection is possible using the Ctrl key + mouse click or the Shift key + mouse click.</td>
</tr>
</tbody>
</table>
Creating variables

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remove</td>
<td>Deletes marked entries from the import window. Multiple selection is possible using the Ctrl key + mouse click or the Shift key + mouse click.</td>
</tr>
<tr>
<td>OK</td>
<td>Starts variable import into the zenon Editor.</td>
</tr>
<tr>
<td>Cancel</td>
<td>Closes the import window without carrying out the import. The selection that was previously made is lost.</td>
</tr>
</tbody>
</table>

💡 Information

You can find further information in the Variables (main.chm::/15247.htm) manual.

**WARNING DIALOG**

If the required TIA project is not found, a corresponding warning dialog is displayed:

When importing variables from a PLC directly and there is no connection, the following warning dialog is shown:

Note: This dialog is only available in English.

7.4.4 Optimize TIA projects

The TIAToAGL.exe command line tool is used for conversion and optimization. The optimized file can be used if the Symbols from precompiled file option has been activated.
Creating variables

This application is supplied with zenon.

A TIA project can thus be converted to a memory-optimized binary file. The “original” TIA project is then no longer needed for communication. The .ap13 file must be present for the driver configuration.

The following is applicable for conversion:

- The TIAtoAGL.exe program contains fixed, preset prefilters for the Accessible from HMI variable property. This filter cannot be removed.
- The conversion program is available as a 32-bit or 64-bit version. The 64-bit version can be used for conversion with extensive TIA projects. The files that are generated are compatible with a 32-bit conversion.
- Before conversion, all modules in the TIA project must be “translated”. Non-translated modules lead to errors in conversion. No successful communication can be set up as a result.

**Attention**

Project optimization is only possible for TIA projects up to version TIA 13.

**EXECUTION**

**Information**

The TIA program must be “translated” in order to avoid errors.

To convert an existing TIA project configuration, carry out the following steps:

1. Start the command prompt as administrator.
   The command prompt window is opened.
2. Enter the following command:
   "[complete save location of the conversion tool]tiatoagl.exe" "[save location of the source file]\[source file name]"
   **Example:** "C:\Program Files (x86)\COPA-DATA\zenon 8.10 SP0\tiatoagl.exe"
   "C:\S7TIA\CPU1511_testX\cpu1511_testX.ap13"
3. The conversion is carried out.
   Further files are saved in the directory during the conversion process. The files are named with the name of the TIA project and differ through their file suffixes:
   - .age
     Is used for the variable import.
   - .agl
     is used for the communication of the variables.
Creating variables

- `.agr` is used for the communication of the variables.
- `.log` contains possible errors during conversion
- `.vars` Cache memory for the conversion routine

VARIABLE IMPORT IN THE ZENON EDITOR

When using the converted project file, no prefilter is shown when the variable is imported. The variable selection dialog appears directly.

**Note:** When converting by means of TIAtoAGL.exe, the filter for variables with the **Accessible from HMI** TIA property has already been carried out automatically.

7.5 Communication details (Driver variables)

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

- Information
- Configuration
- Statistics and
- Error message

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

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

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

**Information**

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

For example:

- Variables for modem information are only supported by modem-compatible drivers.
- Driver variables for the polling cycle are only available for pure polling drivers.
Creating variables

- Connection-related information such as ErrorMSG is only supported for drivers that only edit one connection at a time.

### INFORMATION

<table>
<thead>
<tr>
<th>Name from import</th>
<th>Type</th>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MainVersion</td>
<td>UINT</td>
<td>0</td>
<td>Main version number of the driver.</td>
</tr>
<tr>
<td>SubVersion</td>
<td>UINT</td>
<td>1</td>
<td>Sub version number of the driver.</td>
</tr>
<tr>
<td>BuildVersion</td>
<td>UINT</td>
<td>29</td>
<td>Build version number of the driver.</td>
</tr>
<tr>
<td>RTMajor</td>
<td>UINT</td>
<td>49</td>
<td>zenon main version number</td>
</tr>
<tr>
<td>RTMinor</td>
<td>UINT</td>
<td>50</td>
<td>zenon sub version number</td>
</tr>
<tr>
<td>RTSp</td>
<td>UINT</td>
<td>51</td>
<td>zenon Service Pack number</td>
</tr>
<tr>
<td>RTBuild</td>
<td>UINT</td>
<td>52</td>
<td>zenon build number</td>
</tr>
<tr>
<td>LineStateIdle</td>
<td>BOOL</td>
<td>24.0</td>
<td>TRUE, if the modem connection is idle</td>
</tr>
<tr>
<td>LineStateOffering</td>
<td>BOOL</td>
<td>24.1</td>
<td>TRUE, if a call is received</td>
</tr>
<tr>
<td>LineStateAccepted</td>
<td>BOOL</td>
<td>24.2</td>
<td>The call is accepted</td>
</tr>
<tr>
<td>LineStateDialtone</td>
<td>BOOL</td>
<td>24.3</td>
<td>Dialtone recognized</td>
</tr>
<tr>
<td>LineStateDialing</td>
<td>BOOL</td>
<td>24.4</td>
<td>Dialing active</td>
</tr>
<tr>
<td>LineStateRingBack</td>
<td>BOOL</td>
<td>24.5</td>
<td>While establishing the connection</td>
</tr>
<tr>
<td>LineStateBusy</td>
<td>BOOL</td>
<td>24.6</td>
<td>Target station is busy</td>
</tr>
<tr>
<td>LineStateSpecialInfo</td>
<td>BOOL</td>
<td>24.7</td>
<td>Special status information received</td>
</tr>
<tr>
<td>LineStateConnected</td>
<td>BOOL</td>
<td>24.8</td>
<td>Connection established</td>
</tr>
<tr>
<td>LineStateProceeding</td>
<td>BOOL</td>
<td>24.9</td>
<td>Dialing completed</td>
</tr>
<tr>
<td>LineStateOnHold</td>
<td>BOOL</td>
<td>24.10</td>
<td>Connection in hold</td>
</tr>
<tr>
<td>LineStateConferenced</td>
<td>BOOL</td>
<td>24.11</td>
<td>Connection in conference mode.</td>
</tr>
<tr>
<td>LineStateOnHoldPendConf</td>
<td>BOOL</td>
<td>24.12</td>
<td>Connection in hold for conference</td>
</tr>
<tr>
<td>LineStateOnHoldPendTransfer</td>
<td>BOOL</td>
<td>24.13</td>
<td>Connection in hold for transfer</td>
</tr>
<tr>
<td>LineStateDisconnected</td>
<td>BOOL</td>
<td>24.14</td>
<td>Connection terminated.</td>
</tr>
</tbody>
</table>
Creating variables

<table>
<thead>
<tr>
<th>Name from import</th>
<th>Type</th>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LineStateUnknown</td>
<td>BOOL</td>
<td>24.15</td>
<td>Connection status unknown</td>
</tr>
<tr>
<td>ModemStatus</td>
<td>UDINT</td>
<td>24</td>
<td>Current modem status</td>
</tr>
<tr>
<td>TreiberStop</td>
<td>BOOL</td>
<td>28</td>
<td>Driver stopped</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>For <em>driver stop</em>, the variable has the value <em>TRUE</em> and an <em>OFF</em> bit. After the driver has started, the variable has the value <em>FALSE</em> and no <em>OFF</em> bit.</td>
</tr>
<tr>
<td>SimulRTState</td>
<td>UDINT</td>
<td>60</td>
<td>Informs the status of Runtime for driver simulation.</td>
</tr>
<tr>
<td>ConnectionStates</td>
<td>STRING</td>
<td>61</td>
<td>Internal connection status of the driver to the PLC.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Connection statuses:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0: Connection OK</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1: Connection failure</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2: Connection simulated</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Formatting:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>&lt;Netzadresse&gt;:&lt;Verbindungszustand&gt;;…;…;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A connection is only known after a variable has first signed in. In order for a connection to be contained in a string, a variable of this connection must be signed in once.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The status of a connection is only updated if a variable of the connection is signed in. Otherwise there is no communication with the corresponding controller.</td>
</tr>
</tbody>
</table>

**CONFIGURATION**

<table>
<thead>
<tr>
<th>Name from import</th>
<th>Type</th>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReconnectInRead</td>
<td>BOOL</td>
<td>27</td>
<td>If TRUE, the modem is automatically reconnected for reading</td>
</tr>
<tr>
<td>ApplyCom</td>
<td>BOOL</td>
<td>36</td>
<td>Apply changes in the settings of the serial interface. Writing to this variable</td>
</tr>
</tbody>
</table>
Creating variables

<table>
<thead>
<tr>
<th>Name from import</th>
<th>Type</th>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Immediately results in the method SrvDrvVarApplyCom being called (which currently has no further function).</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ApplyModem</td>
<td>BOOL</td>
<td>37</td>
<td>Apply changes in the settings of the modem. Writing this variable immediately calls the method SrvDrvVarApplyModem. This closes the current connection and opens a new one according to the settings PhoneNumberSet and ModemHwAdrSet.</td>
</tr>
<tr>
<td>PhoneNumberSet</td>
<td>STRING</td>
<td>38</td>
<td>Telephone number, that should be used</td>
</tr>
<tr>
<td>ModemHwAdrSet</td>
<td>DINT</td>
<td>39</td>
<td>Hardware address for the telephone number</td>
</tr>
<tr>
<td>GlobalUpdate</td>
<td>UDINT</td>
<td>3</td>
<td>Update time in milliseconds (ms).</td>
</tr>
<tr>
<td>BGlobalUpdaten</td>
<td>BOOL</td>
<td>4</td>
<td>TRUE, if update time is global</td>
</tr>
<tr>
<td>TreiberSimul</td>
<td>BOOL</td>
<td>5</td>
<td>TRUE, if driver in sin simulation mode</td>
</tr>
<tr>
<td>TreiberProzab</td>
<td>BOOL</td>
<td>6</td>
<td>TRUE, if the variables update list should be kept in the memory</td>
</tr>
<tr>
<td>ModemActive</td>
<td>BOOL</td>
<td>7</td>
<td>TRUE, if the modem is active for the driver</td>
</tr>
<tr>
<td>Device</td>
<td>STRING</td>
<td>8</td>
<td>Name of the serial interface or name of the modem</td>
</tr>
<tr>
<td>ComPort</td>
<td>UINT</td>
<td>9</td>
<td>Number of the serial interface.</td>
</tr>
<tr>
<td>Baudrate</td>
<td>UDINT</td>
<td>10</td>
<td>Baud rate of the serial interface.</td>
</tr>
<tr>
<td>Parity</td>
<td>SINT</td>
<td>11</td>
<td>Parity of the serial interface</td>
</tr>
<tr>
<td>ByteSize</td>
<td>USINT</td>
<td>14</td>
<td>Number of bits per character of the serial interface</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Value = 0 if the driver cannot establish any serial connection.</td>
</tr>
<tr>
<td>StopBit</td>
<td>USINT</td>
<td>13</td>
<td>Number of stop bits of the serial interface.</td>
</tr>
<tr>
<td>Autoconnect</td>
<td>BOOL</td>
<td>16</td>
<td>TRUE, if the modem connection should be established automatically for reading/writing</td>
</tr>
<tr>
<td>PhoneNumber</td>
<td>STRING</td>
<td>17</td>
<td>Current telephone number</td>
</tr>
</tbody>
</table>
## Creating variables

<table>
<thead>
<tr>
<th>Name from import</th>
<th>Type</th>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ModemHwAdr</td>
<td>DINT</td>
<td>21</td>
<td>Hardware address of current telephone number</td>
</tr>
<tr>
<td>RxIdleTime</td>
<td>UINT</td>
<td>18</td>
<td>Modem is disconnected, if no data transfer occurs for this time in seconds (s)</td>
</tr>
<tr>
<td>WriteTimeout</td>
<td>UDINT</td>
<td>19</td>
<td>Maximum write duration for a modem connection in milliseconds (ms).</td>
</tr>
<tr>
<td>RingCountSet</td>
<td>UDINT</td>
<td>20</td>
<td>Number of ringing tones before a call is accepted</td>
</tr>
<tr>
<td>ReCallIdleTime</td>
<td>UINT</td>
<td>53</td>
<td>Waiting time between calls in seconds (s).</td>
</tr>
<tr>
<td>ConnectTimeout</td>
<td>UINT</td>
<td>54</td>
<td>Time in seconds (s) to establish a connection</td>
</tr>
</tbody>
</table>

## STATISTICS

<table>
<thead>
<tr>
<th>Name from import</th>
<th>Type</th>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxWriteTime</td>
<td>UDINT</td>
<td>31</td>
<td>The longest time in milliseconds (ms) that is required for writing.</td>
</tr>
<tr>
<td>MinWriteTime</td>
<td>UDINT</td>
<td>32</td>
<td>The shortest time in milliseconds (ms) that is required for writing.</td>
</tr>
<tr>
<td>MaxBlkReadTime</td>
<td>UDINT</td>
<td>40</td>
<td>Longest time in milliseconds (ms) that is required to read a data block.</td>
</tr>
<tr>
<td>MinBlkReadTime</td>
<td>UDINT</td>
<td>41</td>
<td>Shortest time in milliseconds (ms) that is required to read a data block.</td>
</tr>
<tr>
<td>WriteErrorCount</td>
<td>UDINT</td>
<td>33</td>
<td>Number of writing errors</td>
</tr>
<tr>
<td>ReadSucceedCount</td>
<td>UDINT</td>
<td>35</td>
<td>Number of successful reading attempts</td>
</tr>
<tr>
<td>MaxCycleTime</td>
<td>UDINT</td>
<td>22</td>
<td>Longest time in milliseconds (ms) required to read all requested data.</td>
</tr>
<tr>
<td>MinCycleTime</td>
<td>UDINT</td>
<td>23</td>
<td>Shortest time in milliseconds (ms) required to read all requested data.</td>
</tr>
<tr>
<td>WriteCount</td>
<td>UDINT</td>
<td>26</td>
<td>Number of writing attempts</td>
</tr>
<tr>
<td>ReadErrorCount</td>
<td>UDINT</td>
<td>34</td>
<td>Number of reading errors</td>
</tr>
</tbody>
</table>
### Creating variables

<table>
<thead>
<tr>
<th>Name from import</th>
<th>Type</th>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxUpdateTimeNormal</td>
<td>UDINT</td>
<td>56</td>
<td>Time since the last update of the priority group Normal in milliseconds (ms).</td>
</tr>
<tr>
<td>MaxUpdateTimeHigher</td>
<td>UDINT</td>
<td>57</td>
<td>Time since the last update of the priority group Higher in milliseconds (ms).</td>
</tr>
<tr>
<td>MaxUpdateTimeHigh</td>
<td>UDINT</td>
<td>58</td>
<td>Time since the last update of the priority group High in milliseconds (ms).</td>
</tr>
<tr>
<td>MaxUpdateTimeHighest</td>
<td>UDINT</td>
<td>59</td>
<td>Time since the last update of the priority group Highest in milliseconds (ms)</td>
</tr>
<tr>
<td>PokeFinish</td>
<td>BOOL</td>
<td>55</td>
<td>Goes to 1 for a query, if all current pokes were executed</td>
</tr>
</tbody>
</table>

### ERROR MESSAGE

<table>
<thead>
<tr>
<th>Name from import</th>
<th>Type</th>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ErrorTimeDW</td>
<td>UDINT</td>
<td>2</td>
<td>Time (in seconds since 1.1.1970), when the last error occurred.</td>
</tr>
<tr>
<td>ErrorTimeS</td>
<td>STRING</td>
<td>2</td>
<td>Time (in seconds since 1.1.1970), when the last error occurred.</td>
</tr>
<tr>
<td>RdErrPrimObj</td>
<td>UDINT</td>
<td>42</td>
<td>Number of the PrimObject, when the last reading error occurred.</td>
</tr>
<tr>
<td>RdErrStationsName</td>
<td>STRING</td>
<td>43</td>
<td>Name of the station, when the last reading error occurred.</td>
</tr>
<tr>
<td>RdErrBlockCount</td>
<td>UINT</td>
<td>44</td>
<td>Number of blocks to read when the last reading error occurred.</td>
</tr>
<tr>
<td>RdErrHwAdresse</td>
<td>DINT</td>
<td>45</td>
<td>Hardware address when the last reading error occurred.</td>
</tr>
<tr>
<td>RdErrDatablockNo</td>
<td>UDINT</td>
<td>46</td>
<td>Block number when the last reading error occurred.</td>
</tr>
<tr>
<td>RdErrMarkerNo</td>
<td>UDINT</td>
<td>47</td>
<td>Marker number when the last reading error occurred.</td>
</tr>
<tr>
<td>RdErrSize</td>
<td>UDINT</td>
<td>48</td>
<td>Block size when the last reading error occurred.</td>
</tr>
<tr>
<td>DrvError</td>
<td>USINT</td>
<td>25</td>
<td>Error message as number</td>
</tr>
</tbody>
</table>
8 Driver-specific functions

**BLOCKREAD**
A list of variables is queried with each TCP request when reading.

**BLOCKWRITE**
A list of variables is sent with each TCP request when writing.

9 Driver command function

The zenon **Driver commands** function is to influence drivers using zenon. You can do the following with a driver command:

- Start
- Stop
- Shift a certain driver mode
- Instigate certain actions

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

**Attention**
The zenon **Driver commands** function is not identical to driver commands that can be executed in the Runtime with Energy drivers!
CONFIGURATION OF THE FUNCTION

Configuration is carried out using the **Driver commands** function.

To configure the function:

1. Create a new function in the zenon Editor.
   The dialog for selecting a function is opened.

2. Navigate to the node **Variable**.

3. Select the **Driver commands** entry.
The dialog for configuration is opened

4. Select the desired driver and the required command.
5. Close the dialog by clicking on OK and ensure that the function is executed in the Runtime. Heed the notices in the **Driver command function in the network** section.

**DRIVER COMMAND DIALOG**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Driver</td>
<td>Selection of the driver from the drop-down list. It contains all drivers loaded in the project.</td>
</tr>
<tr>
<td>Current condition</td>
<td>Fixed entry that is set by the system. Has no function in the current version.</td>
</tr>
<tr>
<td>Driver command</td>
<td>Selection of the desired driver command from a drop-down list. For details on the configurable driver commands, see the available driver commands section.</td>
</tr>
<tr>
<td>Driver-specific command</td>
<td>Entry of a command specific to the selected driver.</td>
</tr>
</tbody>
</table>
Driver command function

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Show this dialog in the Runtime** | Configuration of whether the configuration can be changed in the Runtime:  
  - **Active**: This dialog is opened in the Runtime before executing the function. The configuration can thus still be changed in the Runtime before execution.  
  - **Inactive**: The Editor configuration is applied in the Runtime when executing the function.  
  Default: *inactive* |

CLOSE DIALOG

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OK</td>
<td>Applies settings and closes the dialog.</td>
</tr>
<tr>
<td>Cancel</td>
<td>Discards all changes and closes the dialog.</td>
</tr>
<tr>
<td>Help</td>
<td>Opens online help.</td>
</tr>
</tbody>
</table>

AVAILABLE DRIVER COMMANDS

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

<table>
<thead>
<tr>
<th>Driver command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;No command&gt;</td>
<td>No command is sent. A command that already exists can thus be removed from a configured function.</td>
</tr>
</tbody>
</table>
| **Start driver (online mode)** | Driver is reinitialized and started.  
  **Note**: If the driver has already been started, it must be stopped. Only then can the driver be re-initialized and started. |
| **Stop driver (offline mode)** | Driver is stopped. No new data is accepted.  
  **Note**: If the driver is in offline mode, all variables that were created for this driver receive the status *switched off* (*OFF*, Bit 20). |
<table>
<thead>
<tr>
<th>Driver command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Driver in simulation mode</td>
<td>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.</td>
</tr>
<tr>
<td>Driver in hardware mode</td>
<td>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.</td>
</tr>
<tr>
<td>Driver-specific command</td>
<td>Entry of a driver-specific command. Opens input field in order to enter a command.</td>
</tr>
<tr>
<td>Activate driver write set value</td>
<td>Write set value to a driver is possible.</td>
</tr>
<tr>
<td>Deactivate driver write set value</td>
<td>Write set value to a driver is prohibited.</td>
</tr>
<tr>
<td>Establish connection with modem</td>
<td>Establish connection (for modem drivers)</td>
</tr>
<tr>
<td>Disconnect from modem</td>
<td>Terminate connection (for modem drivers)</td>
</tr>
<tr>
<td>Driver in counting simulation mode</td>
<td>Driver is set into counting simulation mode. All values are initialized with 0 and incremented in the set update time by 1 each time up to the maximum value and then start at 0 again.</td>
</tr>
<tr>
<td>Driver in static simulation mode</td>
<td>No communication to the controller is established. All values are initialized with 0.</td>
</tr>
<tr>
<td>Driver in programmed simulation mode</td>
<td>The values are calculated by a freely-programmable simulation project. The simulation project is created with the help of the zenon Logic Workbench and runs in the zenon Logic Runtime.</td>
</tr>
</tbody>
</table>

**DRIVER COMMAND FUNCTION IN THE NETWORK**

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

- A special network command is sent from the computer to the project server. It then executes the desired action on its driver.
- In addition, the Server sends the same driver command to the project standby. The standby also carries out the action on its driver.
This makes sure that Server and Standby are synchronized. This only works if the Server and the Standby both have a working and independent connection to the hardware.

10 Error analysis

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

10.1 Analysis tool

All zenon modules such as Editor, Runtime, drivers, etc. write messages to a joint log file. To display them correctly and clearly, use the Diagnosis Viewer (main.chm::/12464.htm) program that was also installed with zenon. You can find it under Start/All programs/zenon/Tools 8.10 -> Diagviewer.

dzenon driver log all errors in the LOG files. LOG files are text files with a special structure. The default folder for the LOG files is subfolder LOG in the folder ProgramData. For example:

%ProgramData%\COPA-DATA\LOG

Attention: With the default settings, a driver only logs error information. With the Diagnosis Viewer you can enhance the diagnosis level for most of the drivers to "Debug" and "Deep Debug". With this the driver also logs all other important tasks and events.

In the Diagnosis Viewer you can also:

- Follow newly-created entries in real time
- Customize the logging settings
- Change the folder in which the LOG files are saved

Note:

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

**Attention**

In Windows CE errors are not logged per default due to performance reasons.

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

10.2 Check list

Questions and hints for fault isolation:

**GENERAL TROUBLESHOOTING**

- Is the PLC connected to the power supply?
- Analysis with the Diagnosis Viewer (on page 56):
  - Which messages are displayed?
- Are the participants available in the TCP/IP network?
- Can the PLC be reached via the Ping command?
  
  **Ping:** **Open command line** - > **ping** `<IP address>` (e.g.: **ping 192.168.0.100**) - > **Press the Enter key.**
  
  Do you receive an answer with a time or a timeout?
- Can the PLC be reached at the respective port via **TELNET**?
  
  **Telnet:** **Command line:** enter: **telnet** `<IP address port number>`  (for example for **Modbus:** **telnet** 192.168.0.100 502) - > **Press the Enter key.**
  
  If the monitor display turns black, a connection could be established.
- 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 CPU name in the connection correspond to that from the TIA project?
- 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 45).)
- Is the TIA project available in the configured path?
- Does the TIA project match the one in the PLC?
SOME VARIABLES REPORT INVALID.

- Does the symbolic address match the address in the PLC?
- Does the data type of the variable match the data type in the PLC?

VALUES ARE NOT DISPLAYED, NUMERIC VALUES REMAIN EMPTY

Driver is not working. Check the:

- Installation of zenon
- the driver installation
- Installation of all components

Note: Please notice the error messages at 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 Read from Standby Server only in node Driver connection/Addressing.

VALUES ARE DISPLAYED INCORRECTLY

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

DRIVER FAILS OCCASIONALLY

Analysis with the Diagnosis Viewer (on page 56):
- Which messages are displayed?