# zenon driver manual

## IEC870

**v.7.60**
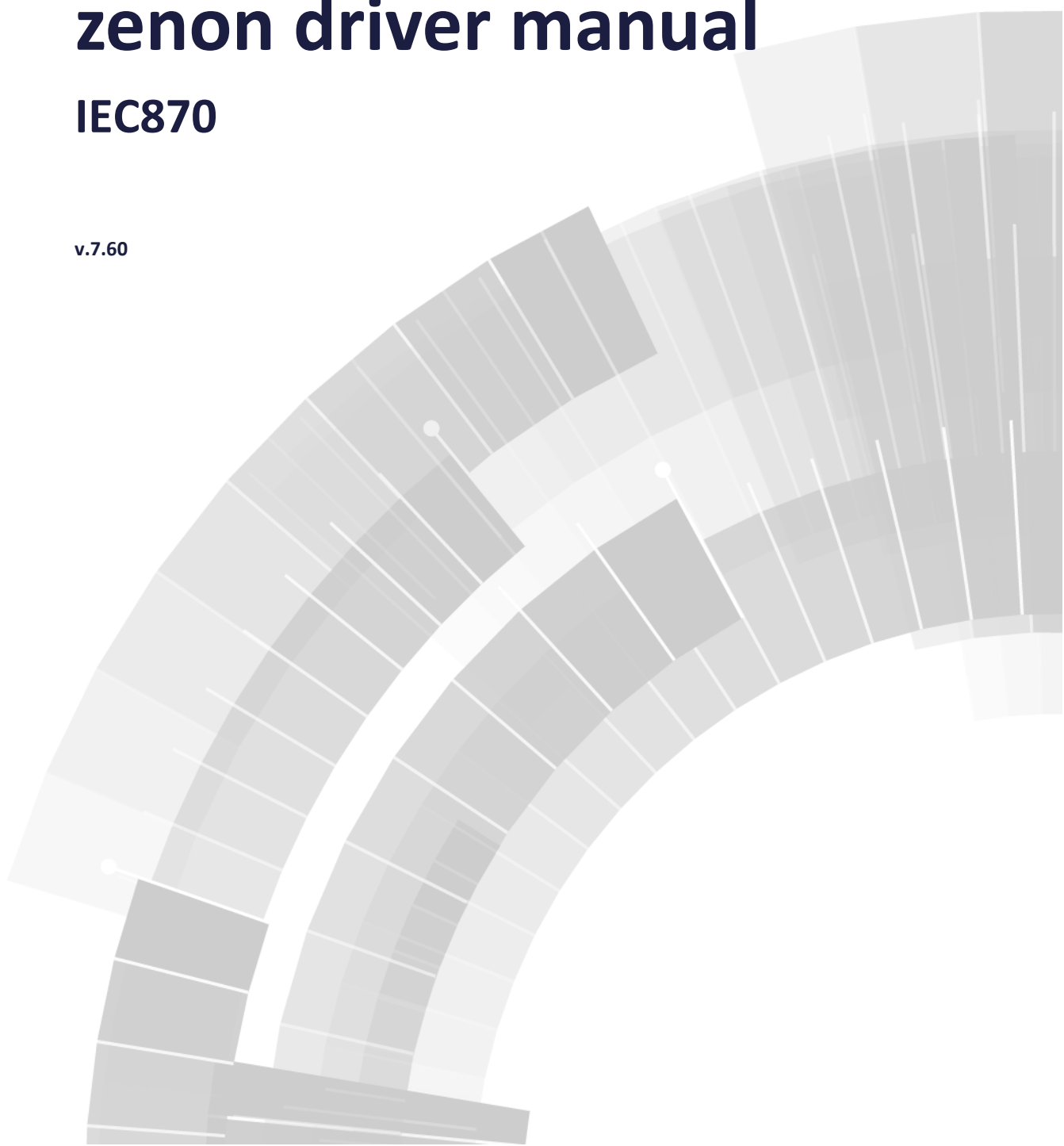
**COPADATA**
do it your way

# Contents

# 1. Welcome to COPA-DATA help

**ZENON VIDEO-TUTORIALS**

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

**GENERAL HELP**

If you cannot find any information you require in this help chapter or can think of anything that you would like added, please send an email to documentation@copadata.com (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. IEC870

Driver for the IEC 60870-5-101 (serial) and the IEC 60870-5-104 (TCP/IP) protocol.

Communication between zenon and the PLC is based on

▶ The serial IEC 60870-5-101 protocol

Here, zenon acts as a master in unbalanced communication mode. The communication channel can be shared between a 60870 Master (zenon) and several 60870 Slaves (PLC).

▶ or the TCP/IP protocol IEC 60870-5-104

Here, zenon acts as the master on protocol level and as a client on TCP level.

> ⚠ **Attention**
>
> *Only one variable per address can be created.*
> *If several variables are created, this can lead to unwanted behavior in Runtime.*

**JOINT USE OF SERIAL AND TCP/IP CONNECTION**

*Connections in accordance with **870-104** (TCP/IP) and **870-101** (serial) should not be created together in a driver instance.    The slower serial connection (101) would also slow down the TCP/IP connection (104). It could even lead to a TCP/IP timeout.*

*Hint: If you need both connections, create the connections for 101 and 104 in two separate instances.*

# 3.  IEC870 - Data sheet

| General: | |
|---|---|
| Driver file name | IEC870.exe |
| Driver name | IEC 60870-5-101_104 |
| PLC types | PLCs supporting the IEC    60870-5-101 or the    IEC 60870-5-104 protocol. |
| PLC manufacturer | Alstom; Siemens; IEC; SAT; Sprecher Automation; Areva; |

| Driver supports: | |
|---|---|
| Protocol | IEC 60870-5-104; IEC 60870-5-101; |
| Addressing: Address-based | X |
| Addressing: Name-based | -- |
| Spontaneous communication | X |

| | |
|---|---|
| Polling communication | X |
| Online browsing | X |
| Offline browsing | -- |
| Real-time capable | X |
| Blockwrite | -- |
| Modem capable | -- |
| Serial logging | X |
| RDA numerical | -- |
| RDA String | -- |
| Hysteresis | X |
| extended API | -- |
| Supports status bit **WR-SUC** | X |
| alternative IP address | X |

| Requirements: | |
|---|---|
| Hardware PC | RS 232 or standard network card |
| Software PC | -- |
| Hardware PLC | -- |
| Software PLC | -- |
| Requires v-dll | X |

| Platforms: | |
|---|---|
| Operating systems | Windows CE 6.0, Embedded Compact 7; Windows 7, 8, 8.1, 10, Server 2008R2, Server 2012, Server 2012R2, Server 2016; |
| CE platforms | x86; ARM; |

# 4.  Driver history

| Date | Driver version | Change |
|---|---|---|
| 07.07.08 | 4500 | Created driver documentation |
| 9/26/2008 | 4700 | Updated interoperability list. |
| 23.03.2010 | 6000 | Values from Buffered Reports are transferred to the archive at Runtime start. |

**DRIVER VERSIONING**

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

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

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

# 5.  Requirements

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

# 6.  Configuration

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

> 💡 **Information**
>
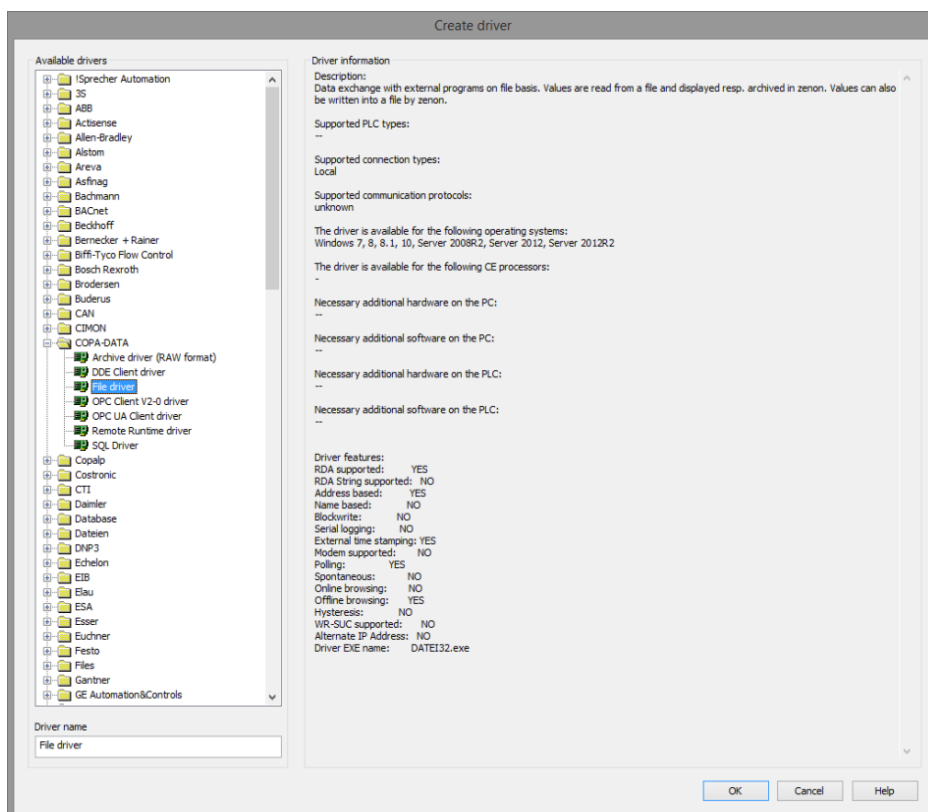> *Find out more about further settings for zenon variables in the chapter Variables (main.chm::/15247.htm) of the online manual.*

## 6.1 Creating a driver

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

| Parameter | Description |
|---|---|
| **Available drivers** | List of all available drivers.<br><br>The display is in a tree structure:<br>`[+]` expands the folder structure and shows the drivers contained therein.<br>[-] reduces the folder structure<br><br>Default: `no selection` |
| **Driver name** | Unique **Identification** of the driver.<br><br>Default: `Empty`<br>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.<br>Default: `Empty`<br>The information on the selected driver is shown in this area after selecting a driver. |

**CLOSE DIALOG**

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

**Information**

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

**CREATE NEW DRIVER**

In order to create a new driver:

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

2. The dialog offers a list of all available drivers.



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

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

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

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

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

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

DRIVER NAME **DIALOG ALREADY EXISTS**

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



**<CD_PRODUCNTAME> 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.

| Option | Description |
|---|---|
| **Mode** | Allows to switch between hardware mode and simulation mode<br><br>▸ `Hardware:`<br>A connection to the control is established.<br><br>▸ Simulation - static:<br>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.<br><br>▸ `Simulation - counting:`<br>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.<br><br>▸ `Simulation - programmed:`<br>No communication is established to the PLC. The values are calculated by a freely programmable simulation project. The simulation project is created with the help of the zenon Logic Workbench and runs in a zenon Logic Runtime which is integrated in the driver. For details see chapter Driver simulation (main.chm::/25206.htm). |
| **Keep update list in the memory** | Variables which were requested once are still requested from the control even if they are currently not needed.<br>This has the advantage that e.g. multiple screen switches after the screen was opened for the first time are executed faster because the variables need not be requested again. The disadvantage is a higher load for the communication to the control. |
| **Output can be written** | `Active`: Outputs can be written.<br><br>`Inactive`: Writing of outputs is prevented.<br><br>**Note**: Not available for every driver. |
| **Variable image remanent** | This option saves and restores the current value, time stamp and the states of a data point.<br><br>Fundamental requirement: The variable must have a valid value and time stamp.<br><br>The variable image is saved in mode hardware if:<br><br>▸ one of the states S_MERKER_1(0) up to S_MERKER8(7), REVISION(9), AUS(20) or ERSATZWERT(27) is active<br><br>The variable image is always saved if: |

| | |
|---|---|
| | ▸ the variable is of the driver object type **Communication details** |
| | ▸ the driver runs in simulation mode. (not 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.<br><br>Attention: If this option is active, the gapless archiving is no longer guaranteed.<br><br>`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.<br><br>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** | `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.<br>`Inactive`: The set priorities are used for the individual variables. |
| **Priority** | The polling times for the individual priority classes are set here. All variables with the according priority are polled in the set time.<br><br>The variables are allocated separately in the settings of the variable properties.<br>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.<br><br>Attention: Priority classes are not supported by each driver For example, drivers that communicate spontaneously do not support it. |

**CLOSE DIALOG**

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

**UPDATE TIME FOR CYCLICAL DRIVERS**

The following applies for cyclical drivers:

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

## 6.2.2    Com
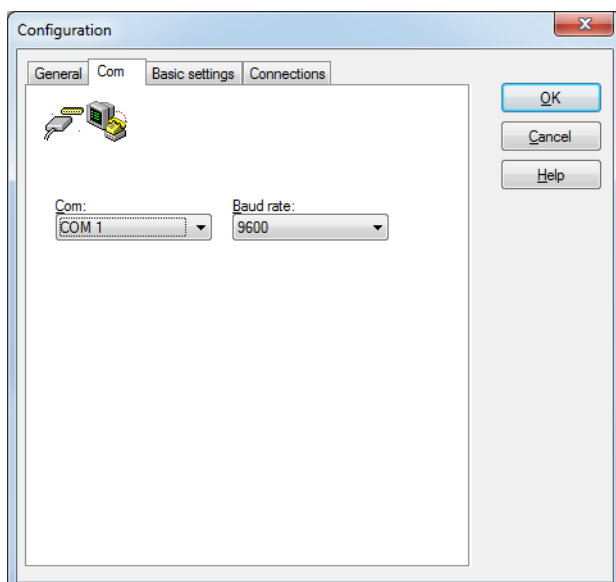


For this driver, only COM port and baud rate for the communication can be changed. All other communication parameters are defined and fixed in accordance with the IEC 870-5-101 standard.

Values:

- ▶ **Parity:** `even`
- ▶ **Stopbit**: `1`
- ▶ **Start bit:** `1`
- ▶ **Data bit:** `8`

For details see: 870-5-1 "Part 5: Transmission protocols, Section One - Transmission frame formats".

## 6.2.3    Basic setting

Note: This dialog is only available in English.



| Parameters | Description |
|---|---|
| **Configuration file name** | Enter the name of the configuration file here. This file is required for the definition of the driver connection. |
| **Deactivate standard DPI/DCS value mapping** | `Inactive`(default): The values of double messages (DPI and DCS) are adjusted to the operating elements of zenon. Use this configuration if you want to use the modules of zenon Energy.<br><br>The driver converts the values of double messages (DPI and DCS) for the Runtime:<br><br>▸ `intermediate | off | on | fault`<br>to: `2 | 0 | 1 | 3`,<br>so that in Runtime, for example, the value `0` means OFF and `1` means ON.<br><br>`Active`: The values of double messages are forwarded to zenon exactly as they are:<br><br>▸ `intermediate | off | on | fault`<br>= `0 | 1 | 2 | 3`).<br>However, in this case you cannot use the command processing, for example, to write double messages. |

| | |
|---|---|
| | **Note:** The driver only converts a value of the variable that corresponds to the DPI/DCS value range (on page 48). DPI/DCS consists, according to the standard, of 2 bits; all other bits of the variable are not transferred. |
| **Directory for file transfer** | Here, enter the folder in the Runtime computer in which the "file transfer" files are to be stored. <br> In this folder, there are the files that are received by the IEC60870 slave `(GET)`. |
| **Directory for file transfer in Control Direction** | Path of the directory in which the files for file transfer are located. <br> In this folder, the files that are sent to the IEC60870 slave are stored `(PUT)`. |
| **Maximum time difference for clock synchronisation** | These parameters define a maximum time limit up to which the time synchronizations in reverse direction will be performed automatically by the system. If the difference is bigger than the configured duration, the synchronization will not be performed. <br><br> Default: `60 s` |

## 6.2.4    Connections

The connections can be defined here.

▶    Devices

► Sectors
(**attention**: configured sectors must be present in the PLC)
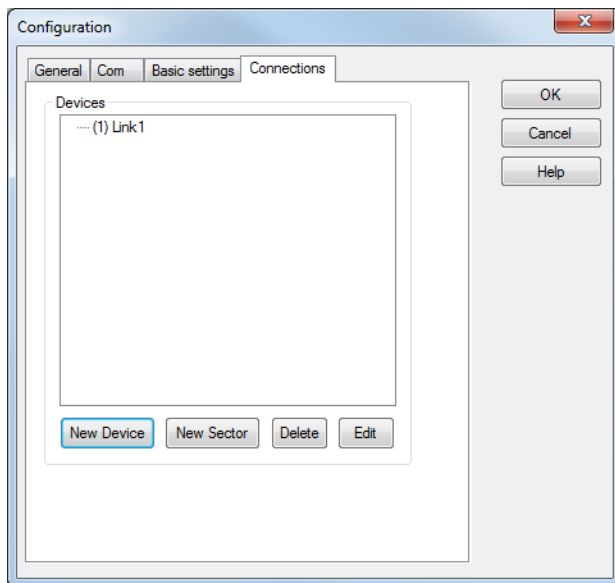


| Parameter | Description |
|---|---|
| **Devices** | List of configured devices and sectors. |
| **New Device** | Opens the dialog to configure new devices (on page 20). |
| **New Sector** | Opens the dialog to configure new sectors (on page 25). |
| **Delete** | Deletes selected entry from the list. |
| **Edit** | Opens the selected entry for editing. |

## CLOSE DIALOG

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

⚠ **Attention**

*Only one variable per address can be created.*
*If several variables are created, this can lead to unwanted behavior in Runtime.*

## Device

A connection to a Device consists of a physical device connection and several sectors (Common Address of ASDU) within this device. Each device is identified by the **Net address** by Runtime side and by the **link address** or **IP address** by the protocol.

Note: This dialog is only available in English.



### DEVICE

Net address and connection name.

| Parameter | Description |
|---|---|
| **Net address** | Net address of Runtime. Serial number of the configured connection. <br> **Attention:** Each connection should have a unique number. An automatic check of whether a number has already been issued is not carried out by zenon. |
| **Device name** | Approved name of the device: Name of the device that is displayed in the list. <br> **Note:** Use short names. This name becomes part of the variable name during online import. |

**APPLICATION LAYER**

Application level of the driver. Match the configuration in this area to that of your PLC. There will be no communication with the PLC if the configuration is incorrect.

| Parameter | Description |
|---|---|
| **ASDU COT size** | Defines the length of the COT (cause of transmission).<br><br>Selection of address size from drop-down list. Valid:<br><br>▸ 1 octet<br><br>▸ 2 octets<br><br>Note: If the **60870-5-104 (TCP/IP)** connection type is selected for link layer, the value `2 octets` is expected in accordance with the standard. Only change the default value for a connection with a non-compliant PLC. |
| **ASDU COA size** | Defines the length of the COA (Common Object Address/Common Address of ASDU).<br>Selection of address size from drop-down list:<br><br>▸ 1 octet<br><br>▸ 2 octets<br><br>Note: If the **60870-5-104 (TCP/IP)** connection type is selected for link layer, the value `2 octets` is expected in accordance with the standard. Only change the default value for a connection with a non-compliant PLC. |
| **ASDU IOA size** | Defines the length of the IOA (Information Object Address).<br><br>Selection of address size from drop-down list. Valid:<br><br>▸ 1 octet<br><br>▸ 2 octets<br><br>▸ 3 octets<br><br>Note: If the **60870-5-104 (TCP/IP)** connection type is selected for link layer, the value `3 octets` is expected in accordance with the standard. Only change the default value for a connection with a non-compliant PLC. |
| **Originator** | Numerical identification of the IEC60870 master. The identification is sent to the controller in commands. The PLC can thus distinguish commands from different masters.<br><br>The value of the originator is only sent if, for **ASDU COT-Size**, the value `2 octets` is configured.<br><br>The entry is validated. An error dialog is called up if the entry is invalid.<br><br>Default: `0`<br>Input range: `0 – 255`<br><br>Grayed out if, for **ASDU COT-Size**, the value `1 octet` is configured. |
| **Timestamps are UTC** | Checkbox to select the expected time format.<br>If this checkbox is activated, the time stamp in UTC format is |

| | expected or used for the command (e.g. `T103` time synchronization). A checkbox that is not activated means that the time stamp is interpreted as local time in accordance with the standard.<br><br>Default: `Inactive`<br><br>**Attention**: All components should use local time in accordance with the IEC60870 standard. |
|---|---|

## LINK LAYER

Configuration of the physical connection for communication with serial interface or via TCP/IP.

## SERIAL COMMUNICATION

| Parameter | Description |
|---|---|
| **870-101** | `Selected`: Serial connection is made in accordance with **60870-5-101**. |
| **Link adress** | Serial address on protocol side.<br><br>Default: `1`<br><br>**Note:** Only active if **Link layer** is `60870-5-104 (TCP/IP)`. |
| **Link address size** | Selection of address size from drop-down list:<br><br>▸ `1 octet`<br>▸ `2 octets`<br><br>Default: `1 octet`<br><br>**Note:** Only active if **Link layer** is `60870-5-104 (TCP/IP)`. |

## COMMUNICATION VIA TCP/IP NETWORK

| Parameter | Description |
|---|---|
| **870-104** | `Selected`: Connection is made via the network in accordance with **60870-5-104**.<br><br>**Note:** Only active if **Link layer** is `60870-5-101 (Serial)`. |
| **Primary connection** | Addressing of the primary connection of the controller. |
| **IP address** | IP address of PLC.<br><br>**Note:** Only active if **Link layer** is `60870-5-101 (Serial)`. |
| **Port** | Portnumber for primary IP_address.<br><br>Default: 2404 |

| | |
|---|---|
| | **Note:** Only active if **Link layer** is `60870-5-101 (Serial)`. |
| **Secondary connection** | Addressing of the alternative connection of the controller. |
| **IP address** | IP address on the protocol side for redundant controllers. Alternative connection in case the primary connection fails. <br><br> **Note:** Only active if **Link layer** is `60870-5-101 (Serial)`. |
| **Port** | Portnumber for redundant IP_address. <br><br> Default: 0 |
| **Redundancy according to 870-104 ed. 2.0** | Active: Redundancy is implemented in accordance with **60870-104 edition 2**. <br><br> Activate this property only if you are sure that the controller supports the guidelines of Edition 2 for redundancy. <br><br> **Note:** Only active if **Link layer** is `60870-5-101 (Serial)`. |

**TIMING PARAMETERS**

**T0** corresponds to the timeout when establishing a connection and cannot be set.
The properties of this group are only active if `60870-5-104 (TCP/IP)` is selected as a connection type.

| Parameter | Description |
|---|---|
| **T1** | Timeout for frame confirmation by the master. <br><br> Value range: `0 - 4294967295` <br><br> Default: `15000` |
| **T2** | Timeout, within which the master should confirm if no data is exchanged. <br><br> Value range: `0 - 4294967295` <br><br> Default: `10000` |
| **T3** | Time after which a U-frame is sent to the master if no data is to be transferred. <br><br> Value range: `0 - 4294967295` <br><br> Default: `20000` |
| **K** | Number of maximum I-frames not yet confirmed by the master. <br><br> Value range: `0 - 4294967295` <br><br> Default: `12` |

| W | Number of I-frames received after a confirmation is sent. |
| | Value range: `0 - 4294967295` |
| | Default: `8` |

| OK | Applies all changes and closes the dialog. |
|----|--------------------------------------------|
| Cancel | Discards all changes and closes the dialog. |

§§Trenner

> ⚠ **Attention**
>
> *Only one variable per address can be created.*
> *If several variables are created, this can lead to unwanted behavior in Runtime.*

## Sectors

Every device can contain several sectors.    Every sector is a structural element by definition and can contain several data points. The sectors must be defined for special commands (e.g. "`„general interrogation`") in the driver configuration.

Note: This dialog is only available in English.
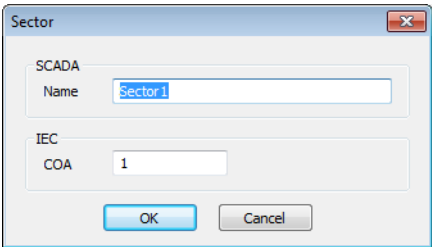
| Parameters | Description |
|---|---|
| **SCADA** | Settings in the SCADA . |
| **Name** | Sector name. |
| **IEC** | Settings on protocol side. |
| **COA** | The COMMON ADDRESS OF ASDUs (IEC 60870-5-101 7.2.4) by which the sector is addressed. This number must be unique on the device (1 ... 254). |

**CLOSE DIALOG**

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

💡 **Information**

*All sectors configured in the driver dialog (COAs) must be present in the PLC!*

# 7. Creating variables

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

## 7.1 Creating variables in the Editor

Variables can be created:

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

**VARIABLE DIALOG**

To create a new variable, regardless of which type:
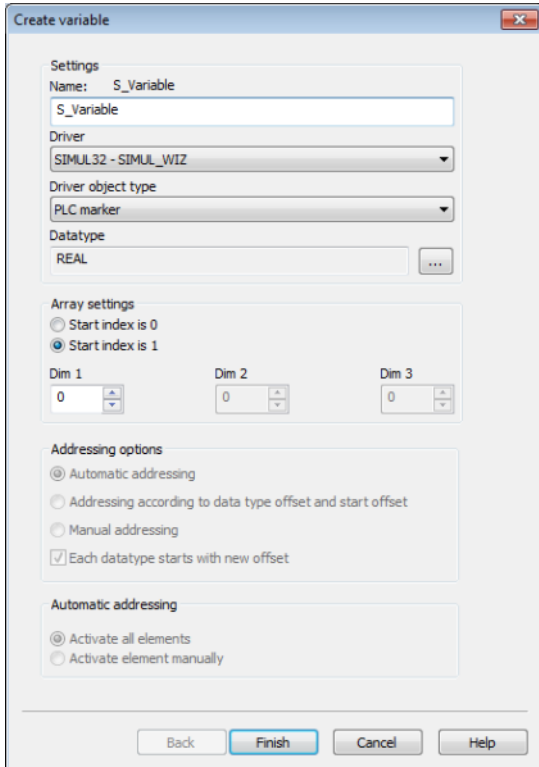
1. Select the **New variable** command in the **Variables** node in the context menu
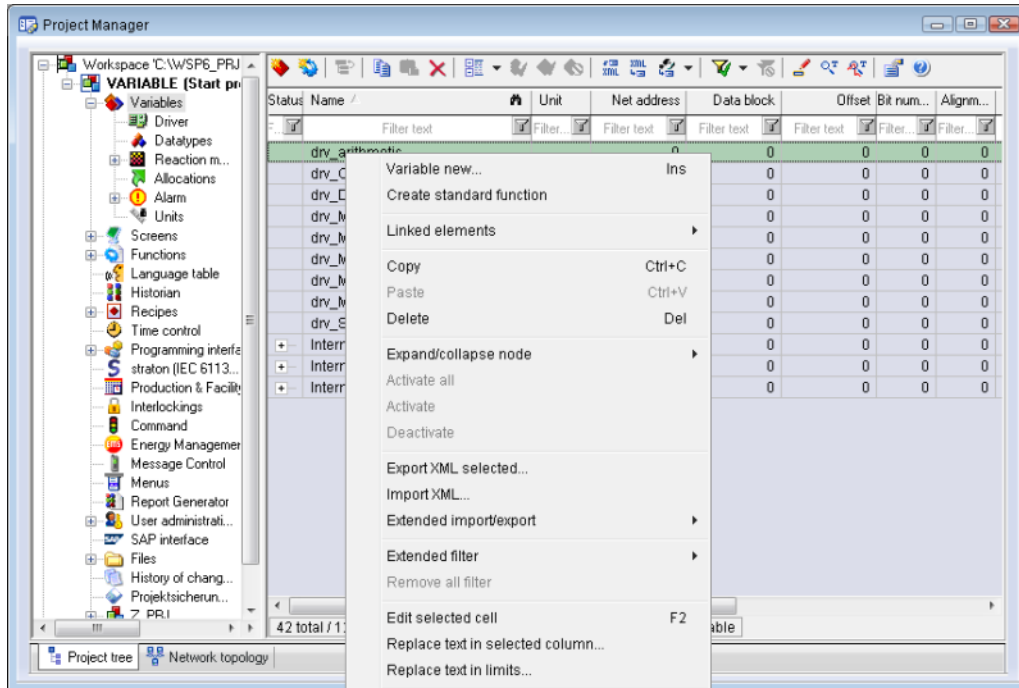


The dialog for configuring variables is opened

2. Configure the variable

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



| 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.<br><br>Maximum length: 128 characters<br><br>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.<br>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.<br><br>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

The data points are addressed via a COA (common object address), an IOA (information object address) and the IEC870 Type. The type defines the function of the variable (see interoperability (on page 58) list). The COA corresponds to the sector of the device in which the variable resides. The IOA determines the offset in that sector.

## SETTINGS FOR THE UNIQUE ADDRESSING OF VARIABLES

| Property | Description |
|---|---|
| **Name** | Freely definable name. Attention: For every zenon project the name must be unambiguous. |
| **Identification** | Freely definable identification. E.g. for Resources label, comments, ... |
| **Net address** | Network address of variable. This address is used to define the allocation to the device specified in the driver configuration. The Net address used there must be entered in the variable configuration accordingly. |
| **Data block** | not used for this driver |
| **Offset** | not used for this driver |
| **Alignment** | not used for this driver |
| **Bit number** | not used for this driver |
| **String length** | Only available for String variables. Maximum number of characters that the variable can take. |
| **Driver connection /Driver Object Type** | Object type of the variables. Depending on the driver used, is selected when the variable is created and can be changed here. |
| **Driver connection/Data Type** | Data type of the variable. Is selected during the creation of the variable; the type can be changed here. Attention: If you change the data type later, all other properties of the variable must be checked and adjusted, if necessary. |
| IEC870 type | Defines the type and function of the variable according to the IEC870 specification. |
| IEC870 COA1 | Corresponds to the sector of the device in which the variable resides. |
| IEC870 IOA1 | Address of the variable within a sector |
| **Driver connection/Priority** | not used for this driver The driver does not support cyclically-poling communication in priority classes. |

The current connection status can be requested via a USINT `internal state (T00)` variable. If the value of this variable is 5, this means that a connection is active and that the general request was finished successfully.

The communication is spontaneous, This means that all value changes of the PLC are processed by the driver.
Example:
The PLC sends three value changes at intervals of 5 ms, which are passed on to zenon. If the changes occur within the update time, in which the driver cyclically reads telegrams, the single telegram contains

all value changes.

E.g. if you use a reaction matrix that reacts to every value change and triggers a log entry in the Chronological Event List (CEL), the CEL will contain three entries at intervals of 5 ms. The same applies to spontaneous archiving and alarms. None of the values are lost.

## 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 driver object types are available in this driver:

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

| Driver Object Type | Channel type | Read | Write | Supported data types | Comment |
|---|---|---|---|---|---|
| **PLC marker** | 8 | X | X | `BOOL, SINT, USINT, INT, UINT, DINT, UDINT, REAL, STRING` | |
| **Communication details** | 35 | X | X | `BOOL, SINT, USINT, INT, UINT, DINT, UDINT, REAL, STRING` | Variables for the statistical analysis of communication. <br><br>You can find detailed information on this in the Communication details (Driver variables) (on page 42) chapter. |

Key:

X => supported

-- => not supported

## 7.3.2 Mapping of the data types

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

**MAPPING OF THE DATA TYPES FROM THE PLC TO ZENON DATA TYPES**

| Control | ASDU Type | zenon | Comment | Datatype |
|---|---|---|---|---|
| M_SP_NA_1 | 1 | BOOL | SPI <0..1> | 8 |
| M_SP_TA_1 | 2 | BOOL | SPI [1] | 8 |
| M_SP_TB_1 | 30 | BOOL | SPI [2] | 8 |
| M_DP_NA_1 | 3 | USINT | DPI <0..3> | 9 |
| M_DP_TA_1 | 4 | USINT | DPI [1] | 9 |
| M_DP_TB_1 | 31 | USINT | DPI [2] | 9 |
| M_ST_NA_1 | 5 | USINT | Corresponds to whole VTI (IEC60870-5-101 7.2.6.5). Bit 8 is the Transient bit. | 9 |
| M_ST_TA_1 | 6 | USINT | VTI [1] | 9 |
| M_ST_TB_1 | 32 | USINT | VTI [2] | 9 |
| M_BO_NA_1 | 7 | UDINT | BSI (32 bits) | 4 |
| M_BO_TA_1 | 8 | UDINT | BSI [1] | 4 |
| M_BO_TB_1 | 33 | UDINT | BSI [2] | 4 |
| M_ME_NA_1 | 9 | REAL | NVA $<-1..+1-2^{-15}>$, in practice $<-1..0,9999>$ | 5 |
| M_ME_TA_1 | 10 | REAL | NVA [1] | 5 |
| M_ME_TD_1 | 34 | REAL | NVA [2] | 5 |
| M_ME_NB_1 | 11 | INT | SVA $<-2^{15}..+2^{15}-1> = <-32768..32767>$ | 1 |
| M_ME_TB_1 | 12 | INT | SVA [1] | 1 |
| M_ME_TE_1 | 35 | INT | SVA [2] | 1 |
| M_ME_NC_1 | 13 | REAL | R32 | 5 |
| M_ME_TC_1 | 14 | REAL | R32 [1] | 5 |
| M_ME_TF_1 | 36 | REAL | R32 [2] | 5 |
| M_IT_NA_1 | 15 | DINT | BCR.Counter reading $<-2^{31}..+2^{31}-1>$ | 3 |
| M_IT_TA_1 | 16 | DINT | BCR.Counter reading [1] | 3 |
| M_IT_TB_1 | 37 | DINT | BCR.Counter reading [2] | 3 |
| C_SC_NA_1 | 45 | BOOL | | 8 |
| C_SC_TA_1 | 58 | BOOL | | 8 |
| C_DC_NA_1 | 46 | USINT | | 9 |
| C_DC_TA_1 | 59 | USINT | | 9 |
| C_RC_NA_1 | 47 | USINT | | 9 |

| | | | | | |
|---|---|---|---|---|---|
| C_RC_TA_1 | 60 | USINT | | | 9 |
| C_SE_NA_1 | 48 | REAL | | | 5 |
| C_SE_TA_1 | 61 | REAL | | | 5 |
| C_SE_NB_1 | 49 | INT | | | 1 |
| C_SE_TB_1 | 62 | INT | | | 1 |
| C_SE_NC_1 | 50 | REAL | | | 5 |
| C_SE_TC_1 | 63 | REAL | | | 5 |
| C_BO_NA_1 | 51 | UDINT | | | 4 |
| C_BO_TA_1 | 64 | UDINT | | | 4 |
| C_IC_NA_1 | 100 | BOOL | 1 during execution | | 8 |
| C_CS_NA_1 | 103 | BOOL | 1 during execution | | 8 |
| F_SC_NA_1 | 122 | STRING | Command for file transfer, e.g. "DIR" or "GET" | | 12 |
| F_DR_TA_1 | 126 | STRING | response variable for file transfer | | 12 |

1) Time tag CP24Time2a    only contains mm:ss.ms; is used for the time stamp of the variable, whereby the driver uses the PC clock to supplement the missing information. If the minute value is higher than that of the PC clock, the driver automatically sets the time back one hour.
  Time tag CP56Time2a is used for the time stamp of the variable.

ASDU Type: IEC60870-5-101 Type identification, corresponds to the **IEC870 Type identification** property of a variable.

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

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.

### 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 cannot 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 export file are imported into the project.

---

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

---

## 7.4.2 DBF Import/Export

Data can be exported to and imported from dBase.

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

## IMPORT DBF FILE

To start the import:

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

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

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

## EXPORT DBF FILE

To start the export:

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

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

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

> 💡 **Information**
>
> *dBase does not support structures or arrays (complex variables) at export.*

### FILE STRUCTURE OF THE DBASE EXPORT FILE

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

> ⚠️ **Attention**
>
> dBase does not support structures or arrays (complex variables) at export.
>
> DBF files must:
>
> ▸ conform with their name to the 8.3 DOS format (8 alphanumeric characters for name, 3 characters for extension, no space)
>
> ▸ Be stored close to the root directory    (Root)

## STRUCTURE

| Identification | Type | Field size | Comment |
|---|---|---|---|
| **KANALNAME** | Char | 128 | Variable name.<br><br>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).<br><br>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.<br><br>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 | Net address |
| **BAUSTEIN** | N | 3 | Datablock address (only for variables from the data area of the PLC) |
| **ADRESSE** | N | 5 | Offset |
| **BITADR** | N | 2 | For bit variables: bit address<br>For byte variables: 0=lower, 8=higher byte<br>For string variables: Length of string (max. 63 characters) |
| **ARRAYSIZE** | N | 16 | Number of variables in the array for index variables<br>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<br>0: Not allowed to set value.<br>1: Allowed to set value. |
|---|---|---|---|
| MIT_ZEIT | R | 1 | time stamp in zenon (only if supported by the driver) |
| OBJEKT | N | 2 | Driver-specific ID number of the primitive object<br>comprises TREIBER-OBJEKTTYP and DATENTYP |
| SIGMIN | Float | 16 | Non-linearized signal - minimum (signal resolution) |
| SIGMAX | F | 16 | Non-linearized signal - maximum (signal resolution) |
| ANZMIN | F | 16 | Technical value - minimum (measuring range) |
| ANZMAX | F | 16 | Technical value - maximum (measuring range) |
| ANZKOMMA | N | 1 | Number of decimal places for the display of the values (measuring range) |
| UPDATERATE | F | 19 | Update rate for mathematics variables (in sec, one decimal possible)<br>not used for all other variables |
| MEMTIEFE | N | 7 | Only for compatibility reasons |
| HDRATE | F | 19 | HD update rate for historical values (in sec, one decimal possible) |
| HDTIEFE | N | 7 | HD entry depth for historical values (number) |
| NACHSORT | R | 1 | HD data as postsorted values |
| DRRATE | F | 19 | Updating to the output (for zenon DDE server, in [s], one decimal possible) |
| HYST_PLUS | F | 16 | Positive hysteresis, from measuring range |
| HYST_MINUS | F | 16 | Negative hysteresis, from measuring range |
| PRIOR | N | 16 | Priority of the variable |
| REAMATRIZE | C | 32 | Allocated reaction matrix |
| ERSATZWERT | F | 16 | Substitute value, from measuring range |
| SOLLMIN | F | 16 | Minimum for set value actions, from measuring range |
| SOLLMAX | F | 16 | Maximum for set value actions, from measuring range |
| VOMSTANDBY | R | 1 | Get value from standby server; the value of the variable is not requested from the server but from the Standby Server in redundant networks |
| RESOURCE | C | 128 | Resources label.<br>Free string for export and display in lists.<br><br>The length can be limited using the MAX_LAENGE entry in **project.ini**. |
| ADJWVBA | R | 1 | Non-linear value adaption:<br>0: Non-linear value adaption is used |

| | | | |
|---|---|---|---|
| | | | `1`: Non-linear value adaption is not used |
| **ADJZENON** | C | 128 | Linked VBA macro for reading the variable value for non-linear value adjustment. |
| **ADJWVBA** | C | 128 | ed VBA macro for writing the variable value for non-linear value adjustment. |
| **ZWREMA** | N | 16 | Linked counter REMA. |
| **MAXGRAD** | N | 16 | Gradient overflow for counter REMA. |

⚠ **Attention**

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

## LIMIT VALUE DEFINITION

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

| Identification | Type | Field size | Comment |
|---|---|---|---|
| **AKTIV1** | R | 1 | Limit value active (per limit value available) |
| **GRENZWERT1** | F | 20 | technical value or ID number of a linked variable for a dynamic limit value (see VARIABLEx)<br>(if VARIABLEx is 1 and here it is −1, the existing variable linkage is not overwritten) |
| **SCHWWERT1** | F | 16 | Threshold value for limit value |
| **HYSTERESE1** | F | 14 | Is not used |
| **BLINKEN1** | R | 1 | Set blink attribute |
| **BTB1** | R | 1 | Logging in CEL |
| **ALARM1** | R | 1 | Alarm |
| **DRUCKEN1** | R | 1 | Printer output (for CEL or Alarm) |
| **QUITTIER1** | R | 1 | Must be acknowledged |
| **LOESCHE1** | R | 1 | Must be deleted |
| **VARIABLE1** | R | 1 | Dyn. limit value linking<br>the limit is defined by an absolute value (see field GRENZWERTx). |
| **FUNC1** | R | 1 | Functions linking |
| **ASK_FUNC1** | R | 1 | Execution via Alarm Message List |
| **FUNC_NR1** | N | 10 | ID number of the linked function<br>(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** | R | 1 | Invisible |

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

### 7.4.3 Online import

Variables can only be create by driver online import in message direction (T01..T39). You must create variables for the commands (T45..T64) manually, because the commands are `write-only` in the IEC 60870 protocol. It is not possible to read their addressing using the controller.

You can find the online import in the driver list in the context menu of the driver.

First select a component, whose variables you want to browse.

After that, a "general interrogation command" (GI) is sent to the selected device and all contained variables are displayed in a list.

After the GI is finished, you will see a dialog for selecting the variables to be imported:
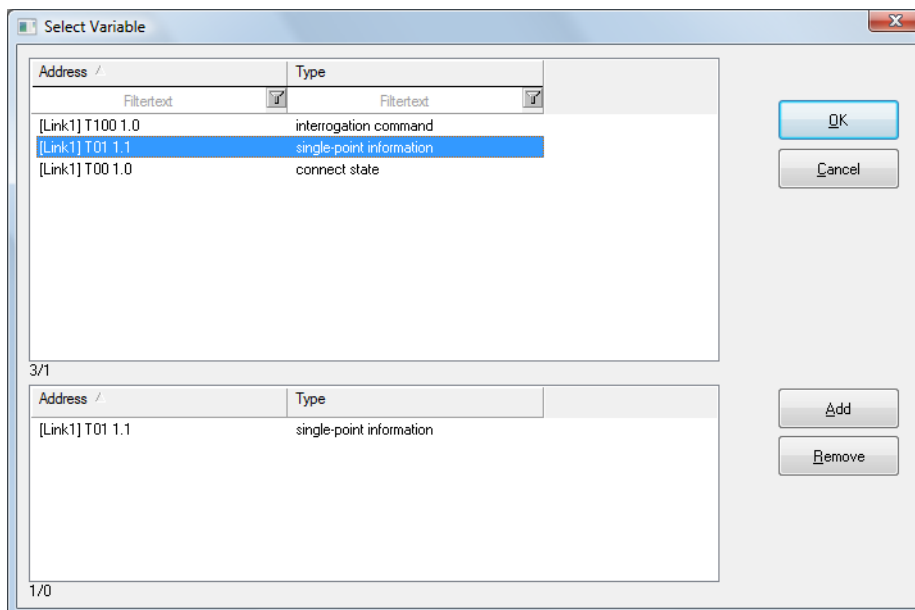


Add a variable to the variables to be imported by double-clicking on it or by selecting it and pressing the "Add" button. Press the button OK to create all selected variables in zenon. The addressing is taken over from the device.

## 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** (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 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
>
> ▸   Variables for the polling cycle only for pure polling drivers
>
> ▸   Connection-related information such as ErrorMSG only for drivers that only edit one connection at a a time

**INFORMATION**

| Name from import | Type | Offset | Description |
|---|---|---|---|
| MainVersion | UINT | 0 | Main version number of the driver. |
| SubVersion | UINT | 1 | Sub version number of the driver. |
| BuildVersion | UINT | 29 | Build version number of the driver. |
| RTMajor | UINT | 49 | zenon main version number |
| RTMinor | UINT | 50 | zenon sub version number |
| RTSp | UINT | 51 | zenon Service Pack number |
| RTBuild | UINT | 52 | zenon build number |
| LineStateIdle | BOOL | 24.0 | TRUE, if the modem connection is idle |
| LineStateOffering | BOOL | 24.1 | TRUE, if a call is received |
| LineStateAccepted | BOOL | 24.2 | The call is accepted |
| LineStateDialtone | BOOL | 24.3 | Dialtone recognized |
| LineStateDialing | BOOL | 24.4 | Dialing active |
| LineStateRingBack | BOOL | 24.5 | While establishing the connection |
| LineStateBusy | BOOL | 24.6 | Target station is busy |
| LineStateSpecialInfo | BOOL | 24.7 | Special status information received |
| LineStateConnected | BOOL | 24.8 | Connection established |
| 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<br><br>For `driver stop`, the variable has the value `TRUE` and an **OFF** bit. After the driver has started, the variable has the value `FALSE` and no **OFF** bit. |
| SimulRTState | UDINT | 60 | Informs the status of Runtime for driver simulation. |

| ConnectionStates | STRING | 61 | Internal connection status of the driver to the PLC. |
|---|---|---|---|
| | | | Connection statuses: |
| | | | `0:` Connection OK |
| | | | `1:` Connection failure |
| | | | `2:` Connection simulated |
| | | | Formating: |
| | | | `<Netzadresse>:<Verbindungszustand>;…;…;` |
| | | | A connection is only known after a variable has first signed in. In order for a connection to be contained in a string, a variable of this connection must be signed in once. |
| | | | The status of a connection is only updated if a variable of the connection is signed in. Otherwise there is no communication with the corresponding controller. |

**CONFIGURATION**

| Name from import | Type | Offset | Description |
|---|---|---|---|
| ReconnectInRead | BOOL | 27 | If TRUE, the modem is automatically reconnected for reading |
| ApplyCom | BOOL | 36 | Apply changes in the settings of the serial interface. Writing to this variable immediately results in the method SrvDrvVarApplyCom being called (which currently has no further function). |
| ApplyModem | BOOL | 37 | Apply changes in the settings of the modem. Writing this variable immediately calls the method SrvDrvVarApplyModem. This closes the current connection and opens a new one according to the settings **PhoneNumberSet** and **ModemHwAdrSet**. |
| PhoneNumberSet | STRING | 38 | Telephone number, that should be used |
| ModemHwAdrSet | DINT | 39 | Hardware address for the telephone number |

| GlobalUpdate | UDINT | 3 | Update time in milliseconds (ms). |
|---|---|---|---|
| BGlobalUpdaten | BOOL | 4 | TRUE, if update time is global |
| TreiberSimul | BOOL | 5 | TRUE, if driver in sin simulation mode |
| TreiberProzab | BOOL | 6 | TRUE, if the variables update list should be kept in the memory |
| ModemActive | BOOL | 7 | TRUE, if the modem is active for the driver |
| Device | STRING | 8 | Name of the serial interface or name of the modem |
| ComPort | UINT | 9 | Number of the serial interface. |
| Baudrate | UDINT | 10 | Baud rate of the serial interface. |
| Parity | SINT | 11 | Parity of the serial interface |
| ByteSize | USINT | 14 | Number of bits per character of the serial interface<br><br>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:

**GENERAL QUERY**

As envisaged in the IEC 60870 standard, the driver automatically sends the general queries (C_IC_NA_1) to the PLCs straight after a connection is established.

A general request can be re-executed at a later time. In order to do so, a BOOL variable of type C_IC_NA_1 (T100) must be written to. The IOA of these variables must be 0. The value of this variable is 1 whilst the general request is running. If the time general request could not be carried out correctly, this variable will get the status INVALID. Variables of type T100 can be created separately for every sector (COA). This allows you to check whether the sectors of a device can be reached.

## CONNECTION STATUS

The current status of the connection to the PLC can be requested via a USINT variable of the type "internal state" (T00) and COA 0 and IOA 0. If the value of this variable is 5, this means that a connection is active and that the general request was finished successfully.

## STATUS OF THE DL LAYER

The status of the data security layer connection to the PLC can be read with the help of "internal state" (T00) and COA 0 and IOA 1 variables: The variable relates to the Data Link Layer and at the same time to both connections to redundancy if the secondary connection was defined in the driver configuration:

| Bit | Description |
|---|---|
| Bit 0 | Connection status of the primary connection |
| Bit 1 | Displays whether the primary connection is active (is used for communication) |
| Bit 4 | Connection status of the secondary connection |
| Bit 5 | Displays whether the secondary connection is active (is used for communication) |

## MANUAL SWITCHING OF THE CONNECTION

> 💡 **Information**
>
> *Only available if you have activated the **Redundancy according to 60870-104 edition 2.0** option when creating the driver.*

You trigger manual switching to the redundant connection with the help of the driver-specific command **104_MANUAL_SWITCH 'net address'**. The 'net address' value stands for the corresponding network address of the connection.

## FILETRANSFER

There are three functions implemented for the file transfer:

1. Request folder information (DIR)
2. Get file from the PLC (GET)
3. Send file to the PLC (PUT)
4. Delete file from the PLC (DEL)

## 1. REQUEST DIRECTORY INFORMATION (DIR)

To request the folder information:

1. Create two string variables in your zenon project:

    a) The first variable is a "**call directory, call file**" type variable (T122) (hereinafter called a command variable). It can also be used to obtain and delete files.

    b) The second variable of „**directory** type (T126) is only used for the result of the folder query. It receives the folder content as legible text. For this reason, its size (string length) should correspond to the maximum size of the file name (including the folder) in the PLC.

2. For the command variable (T122), set the value "DIR" (for the root directory) or "DIR <IOA>.<NOF>"

If the folder has been successfully received:

▶ The command variable changes its value to "DIR OK"

▶ The folder variable (T126) contains the received folder content.
One line of this text has the following format: **<IOA>.<NOF>;<File length>;<Time stamp>;SOF**

## 2. GET FILE FROM THE PLC (GET)

To get a file from the PLC, set the value "GET <IOA>.<NOF>" for the command variable (T122).

If the file has been successfully received:

▶ If it is saved in the folder that was defined as **Directory for file transfer** in the driver dialog basic settings (on page 17)

▶ The command variable changes its value to "GET OK"

▶ A subfolder is created for each COA

An active file transfer can be canceled by setting the value "CANCEL" to the command variable (T122). The driver thus does not expect any further segments of data and will also not request any further sections of the file. A deactivation is not sent to the PLC however. The file for which the transmission was cancelled is not saved.

---

### Example

Get the file with IOA 1100 and NOF 'transparent' (1) from sector 151 if the folder was defined in the driver configuration as "C:\TEMP\IEC870":

▸ Send the target value "GET 1100.1" to a variable of type T122, COA 151 und IOA 0.

▸ The file is stored at C:\TEMP\IEC870\151\1100.1.

### 3. SEND FILE TO PLC (PUT)

To send a file to the PLC, set the value "`PUT <IOA>.<NOF>`" for the command variable (T122).

A file from the COA folder is sent. This folder is a subdirectory of the directory that is defined in the driver dialog basic settings (on page 17), in the **Directory for file transfer in Control Direction** input field. Note: This folder must be created manually by the user.

If the file has been successfully sent:

▸    The command variable changes its value to `PUT OK`

If the transfer is in progress, "`PUT BUSY`" is displayed. An active file transfer can be canceled by setting the value `CANCEL` to the command variable (T122).

### 4. DELETE FILE FROM THE PLC (DEL)

To delete a file on the PLC, set the value "`DEL <IOA>.<NOF>` for the command variable (T122).

If the file was deleted successfully, the command variable changes its value to "`DEL OK`"

### ERROR TREATMENT

If an error occurs when the file transfer is carried out, the command variable changes its value to "`XXX ERROR`" (XXX = DIR, GET, PUT or DEL) and the driver optionally writes an entry into the log file (see also error analysis (on page 74)).

### TIME SYNCHRONIZATION

By writing to a variable of Type C_CS_NA_1 (T103), the current time of the PC is sent to the PLC. The IOA of these variables must be 0.

Time synchronization in reverse direction:

The driver can accept and evaluate a T103 telegram in Reverse Direction from the device. If the device sends a T03 telegram, the time on the PC will be taken from the device, however only if the difference between the device time and the PC time does not exceed the configured **maximum difference** (see driver dialog basic setting).

### STATUS BIT WR-SUC (BIT 41)

If, for a **Write set value** or **Write recipe** action or command processing, a write confirmation has been requested (WR_ACK), this status bit is set accordingly after the command has been sent (**COT_act**). There is no wait for **COT_actcon**.

## STATUS BIT BL_870 (BIT 44)

Indicates IEC status `blocked`. The value is blocked for transferring and remains in the status it had before it was blocked. This status bit can be selected in Multi reaction matrices, in Combined elements and in the Interlocking formula.

In VBA the top 32 bits can be polled with `StatusExtValue()`. With `SetValueWithStatusEx()` all 64 status bits can be polled.

## STATUS BIT SB_870 (BIT 45)

Indicates IEC status `substituted`. The value was set by an operator or an automatic source. This status bit can be selected in Multi reaction matrices, in Combined elements and in the Interlocking formula.

In VBA the top 32 bits can be polled with `StatusExtValue()`. With `SetValueWithStatusEx()` all 64 status bits can be polled.

## STATUS BIT NT_870 (BIT 46)

Indicates IEC status `not topical`. The value was not updated or was not available for a certain period of time. This status bit can be selected in Multi reaction matrices, in Combined elements and in the Interlocking formula.

In VBA the top 32 bits can be polled with `StatusExtValue()`. With `SetValueWithStatusEx()` all 64 status bits can be polled.

## STATUS BIT OV_870 (BIT 47)

Indicates `Overflow`. The value lies outside the predefined bandwidth.

## STATUS BIT SE_870 (BIT 48)

Corresponds to IEC Qualifier S/E and is used in conjunction with the **Select before operate** functionality: serves to distinguish between `Select-` and `Execute-`status of a command.

## STATUS BITS COTX (BITS 32..37)

The received value of the Cause of Transmission - see `IEC60870-5-101 7.2.3`) is portrayed to the status bits `COTx`.

**STATUS BIT T_INVAL (BIT 49)**

The status bit T_INVAL (real time invalid) is set by driver IEC870 if the received real time stamp is marked as invalid. In this case, the local PC time is stamped.

Note: In the process gateway IEC870 slave, this status bit is forwarded in the direction of messaging in the time stamp.

**ASDUS WITH "TIME TAG CP24TIME2A"**

The time information of type **CP24Time2a** only contains minutes, seconds and milliseconds. Date information (year, month, day) and information about to the hour is not transferred. The driver reverts to the PC clock in order to complement the missing time information (year, month, day and hour). If a **CP24Time2a** with a difference of more than plus/minus 30 minutes is received, the driver corrects the time stamp automatically by one hour plus/minus. At this a possible date change is also considered.

Note: This does not influence **CP56Time2a** time stamps.

**HYSTERESIS HANDLING**

In general the driver supports hysteresis. The hysteresis is only considered for numeric data types if hysteresis<>0 was configured. For variables with hysteresis=0 and for variables of type BOOL or STRING, the driver sends all received values to the Runtime.

**SELECT & EXECUTE**

In order to be able to use "select & execute" a **command** must be executed. In addition, the **Select Before Operate** property must be activated for the command variables.     At the first stage of the action, the driver sends a `Select` command - an ASDU with IEC Qualifier S/E equals 1 (`select`). The `SE_870` status bit of the command variable is also set. There is then an `Execute` command - an ASDU with IEC Qualifier S/E equals 0 (`execute`), or a `Cancel` command. A `Cancel` (`deactivation`) can only be triggered from a two-stage action - if the user cancels the command during execution of the action, instead of confirming it.

If the value is set directly - not by a command processing action -    then:

▶ The driver ignores the **Select Before Operate** property of the command variable

▶ The driver uses the "`direct execute`" command and sends the command with the IEC Qualifier S/E = 0 (`execute`).

**LIMITATIONS**

▶ See IEC60870-5-101/104 interoperability list for information about supported communication parameters and type names.

▶ „select and execute can only be used for command processing. Otherwise "direct execute" is always used.

## 8.1 Mapping of double point values

Double Point Value Mapping    is a standard function of the zenon Energy driver. It only influences zenon Runtime and has no effect on the driver communication with a device. Configuration is carried out in the driver settings in the **Basic Settings** tab.

Note: It is recommended that you leave the **Deactivate standard double point value mapping** option in the driver configuration as the default, inactive.

The driver uses Double Point Value Mapping to convert values so that they are displayed in a user-friendly manner. However this only applies to the HMI.

The driver always communicates with one device with values for Double Points with 2-bit information. This corresponds to the definitions of the energy standard. That means:

| Parameters | Double Point | Value | Meaning |
|---|---|---|---|
| **Intermediate** | 00b | 0 | Switches are neither open nor closed, for example the End-Position has not yet been reached |
| **Off** | 01b | 1 | Switch open |
| **On** | 10b | 2 | Close switch/switch closed |
| **Fault** | 11b | 3 | Error |

Double Points are coded with 2-bits in the energy sector for historical reasons: The transmission of a telegram to a serial connection (RS232) with a series of values that only contain 0 was not safeguarded against transmission errors. In order to increase the certainty, it was decided in the first standards that the value for OFF    is not to be sent as 0 but as 01b, which corresponds to decimal 1. These Double Point Values also precisely reflect the type of how two sensors record the physical position of a switch.

However, the values sent this way may be confusing for people:

▶ OFF = 1

▶ ON = 2

Humans are used to all other devices and systems:

▶ OFF = 0

▶ ON =  1

At the same time, Single Point Values are also defined with OFF = 0 in the same standard.

The user must always be aware of the technical level on which they are acting and receiving or sending information. In stressful situations, this can very easily lead to serious mistakes. For example, if `ON` is sent instead of `OFF`.

In order to avoid this dangerous error, the zenon Energy driver offers its own Double Point Value Mapping.

**MAPPING VOR HMI**

With the Double Point Value Mapping, all Double Points in zenon have the following values:

- ▶ `Intermediate = 2`
- ▶ `Off = 0`
- ▶ `On = 1`
- ▶ `Fault = 3`

> 💡 **Information**
>
> This function can be deactivated in the driver settings. However some features such as Command Processing or ALC can no longer be used then.
>
> **Recommendation:** Do not use numerical elements and numerical values to display `OFF/ON` or `OPEN/CLOSE`. Use combined elements with graphic symbols or text elements instead.

**DCS MAPPING IN DOUBLE COMMAND**

In general, the following applies: If a set value is written to a command variable, the value is written to the PLC. The value is normally written to the controller exactly as it is.

However, if this command variable is:

- ▶ Type ID T46 or T59
- ▶ and has the value <3

communication to the controller is mapped:

Because DCS consists of two bits, all higher values are cut off and not sent. That means:

| zenon value | DCS value | Meaning |
|---|---|---|
| `0 = 00b` | `01b = `**1***  | **Off** |
| `1 = 01b` | `10b = `**2***  | **On** |
| `2 = 10b` | `00b = `**0***  | **Intermediate** |
| `3 = 11b` | `11b = 3` | **Fault** |
| `4 = 100b` | `00b = 0` | **Intermediate** |
| `5 = 101b` | `01b = 1` | **Off** |
| `6 = 110b` | `10b = 2` | **On** |
| `7 = 111b` | `11b = 3` | **Fault** |

* mapped value

Values `>4` are communicated to the controller as unmapped.

# 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
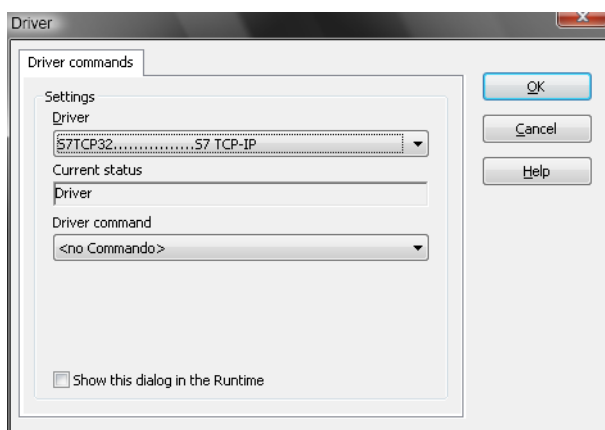
| Parameter | Description |
|---|---|
| **Drivers** | Drop-down list with all drivers which are loaded in the project. |
| **Current status** | Fixed entry which has no function in the current version. |
| Driver command | Drop-down list for the selection of the command. |
| ▶ `Start driver (online mode)` | 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 `switched off` (OFF; Bit 20). |
| ▶ `Driver in simulation mode` | Driver is set into simulation mode. The values of all variables of the driver are simulated by the driver. No values from the connected hardware (e.g. PLC, bus system, ...) are displayed. |
| ▶ `Driver in hardware mode` | Driver is set into hardware mode. For the variables of the driver the values from the connected hardware (e.g. PLC, bus system, ...) are displayed. |
| ▶ `Driver-specific command` | 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. Interoperability

This companion standard presents sets of parameters and alternatives from which subsets must be selected to implement particular telecontrol systems. Certain parameter values, such as the choice of 'structured' or 'unstructured' fields of the informationobject address of ASDUs represent mutually exclusive alternatives. This means that only one value of the defined parameters is admitted per system. Other parameters, such as the listed set of different process information incommand and in monitor direction allow the specification of the complete set orsubsets, as appropriate for given applications. This clause summarizes theparameters of the previous clauses to facilitate a suitable selection for aspecific application. If a system is composed of equipment stemming from different manufacturers, it is necessary that all partners agree on the selected parameters.

The interoperability list is defined as in IEC 60870-5-101 and extended with parameters used in this standard. The text descriptions of parameters which are not applicable to this companion standard are strike-through (corresponding check box is marked black).

NOTE      In addition, the full specification of a system may require individual selection of certain parameters for certain parts of the system, such as the individual selection of scaling factors for individually addressable measured values.

The selected parameters should bemarked in the white boxes as follows:

[    ]        Function or ASDU is not used
[ X ]        Function or ASDU is used as standardized (default)
[ R ]        Function or ASDU is used in reverse mode
[ B ]        Function or ASDU is used in standard and reverse mode

The possible selection (blank, X , R , or B ) is specified for each specific clause or parameter.

A black check box indicates that the option cannot be selected in this companion standard.

## 1. SYSTEM OR DEVICE

(system-specific parameter, indicate definition of a system or a device by marking one of the following with ' X ')
[    ]        System definition
[ X ]        Controlling station definition (Master)
[    ]        Controlled station definition (Slave)

## 2. NETWORK CONFIGURATION: 101 ONLY

| (network-specific parameter, all configurations that are used are to be marked ' x ') Configuration types | |
|---|---|
| [ X ]    Point-to-point | [ X ]    Multipoint-partyline |
| [ X ]    Multiple point-to-point | [    ]    Multipoint-star |

## 3. PHYSICAL LAYER: 101 ONLY

(network-specific parameter, all interfaces and data rates that are used are to be marked ' x ')

### TRANSMISSION SPEED (CONTROL DIRECTION)

| Unbalanced interchange Circuit V.24/V.28 Standard | Unbalanced interchange Circuit V.24/V.28 Recommended if >1 200 bit/s | Balanced interchange Circuit X.24/X.27 |
|---|---|---|
| [ X ]    100 bit/s | [ X ]    2400 bit/s | [    ]    2400 bit/s |
| [ X ]    200 bit/s | [ X ]    4800 bit/s | [    ]    4800 bit/s |
| [ X ]    300 bit/s | [ X ]    9600 bit/s | [    ]    9600 bit/s |
| [ X ]    600 bit/s | | [    ]    19200 bit/s |
| [ X ]    1200 bit/s | | [    ]    38400 bit/s |
| | | [    ]    56000 bit/s |
| | | [    ]    64000 bit/s |

**TRANSMISSION SPEED (MONITOR DIRECTION)**

| Unbalanced interchange Circuit V.24/V.28 Standard | Unbalanced interchange Circuit V.24/V.28 Recommended if >1 200 bit/s | Balanced interchange Circuit X.24/X.27 |
|---|---|---|
| [ X ]     100 bit/s | [ X ]     2400 bit/s | [     ]     2400 bit/s |
| [ X ]     200 bit/s | [ X ]     4800 bit/s | [     ]     4800 bit/s |
| [ X ]     300 bit/s | [ X ]     9600 bit/s | [     ]     9600 bit/s |
| [ X ]     600 bit/s | | [     ]     19200 bit/s |
| [ X ]     1200 bit/s | | [     ]     38400 bit/s |
| | | [     ]     56000 bit/s |
| | | [     ]     64000 bit/s |

**4. LINK LAYER: 101 ONLY**

(network-specific parameter, all options that are used are to be marked ' X '. Specify the maximum frame length. If a non-standard assignment of class 2 messages is implemented for unbalanced transmission, indicate the Type ID and COT of all messages assigned to class 2.)
Frame format FT 1.2, single character 1 and the fixed time out interval are used exclusively in this companion standard.

| Link transmission | Frame length | Address field of the link |
|---|---|---|
| [     ]     Balanced transmission | [255]    Maximum length L (number of octets) | [     ]     not present (balanced transmission only) |
| [ X ]     Unbalanced transmission | | [ X ]     One octet |
| | | [ X ]     Two octets |
| | | [ X ]     Structured |
| | | [ X ]     Unstructured |

When using an unbalanced link layer, the following ASDU types are returned in class 2 messages (low priority) with the indicated causes of transmission:

[     ]     The standard assignment of ASDUs to class 2 messages is used as follows:

| Type identification | Cause of transmission |
|---|---|
| 9, 11, 13, 21 | <1> |

[     ]     A special assignment of ASDUs to class 2 messages is used as follows:

| Type identification | Cause of transmission |
|---|---|
| any* | any |

*assignment of any ASDU type to class 2 as well to class 1 is supported

Note: (In response to a class 2 poll, a controlled station may respond with class 1 data when there is no class 2 data available).

## 5. APPLICATION LAYER

### TRANSMISSION MODE FOR APPLICATION DATA

Mode 1 (Least significant octet first), as defined in 4.10 of IEC 60870-5-4, is used exclusively in this companion standard.

### COMMON ADDRESS OF ASDU

(system-specific parameter, all configurations that are used are to be marked ' X ')

| [ X ]   One octet | [ X ]   Two octets |
|---|---|

### INFORMATION OBJECT ADDRESS

(system-specific parameter, all configurations that are used are to be marked ' X ')

| [ X ]   One octet | [ X ]   Structured |
|---|---|
| [ X ]   Two octets | [ X ]   Unstructured |
| [ X ]   Three octets | |

### CAUSE OF TRANSMISSION

(system-specific parameter, all configurations that are used are to be marked ' X ')

| [ X ]   One octet | [ X ]   Two octets (with originator address). Originator address is set to zero if not used |
|---|---|

**LENGTH OF APDU**

(system-specific parameter, all configurations that are used are to be marked ' X ')
The maximum length of the APDU is 253 (default). The maximum length may be reduced by the system.

| [ 253 ] | Maximum length of APDU per system |
|---------|-----------------------------------|

**SELECTION OF STANDARD ASDUS**
**PROCESS INFORMATION IN MONITOR DIRECTION**

(station-specific parameter, mark each Type ID ' X ' if it is only used in the standard direction, ' R ' if only used in the reverse direction, and ' B ' if used in both directions).

| [ X ] | <1> | := Single-point information | M_SP_NA_1 |
|---|---|---|---|
| [ X ] | <2> | := Single-point information with time tag | M_SP_TA_1 |
| [ X ] | <3> | := Double-point information | M_DP_NA_1 |
| [ X ] | <4> | := Double-point information with time tag | M_DP_TA_1 |
| [ X ] | <5> | := Step position information | M_ST_NA_1 |
| [ X ] | <6> | := Step position information with time tag | M_ST_TA_1 |
| [ X ] | <7> | := Bitstring of 32 bit | M_BO_NA_1 |
| [ X ] | <8> | := Bitstring of 32 bit with time tag | M_BO_TA_1 |
| [ X ] | <9> | := Measured value, normalized value | M_ME_NA_1 |
| [ X ] | <10> | := Measured value, normalized value with time tag | M_ME_TA_1 |
| [ X ] | <11> | := Measured value, scaled value | M_ME_NB_1 |
| [ X ] | <12> | := Measured value, scaled value with time tag | M_ME_TB_1 |
| [ X ] | <13> | := Measured value, short floating point value | M_ME_NC_1 |
| [ X ] | <14> | := Measured value, short floating point value with time tag | M_ME_TC_1 |
| [ X ] | <15> | := Integrated totals | M_IT_NA_1 |
| [ X ] | <16> | := Integrated totals with time tag | M_IT_TA_1 |
| [ ] | <17> | := Event of protection equipment with time tag | M_EP_TA_1 |
| [ ] | <18> | := Packed start events of protection equipment with time tag | M_EP_TB_1 |
| [ ] | <19> | := Packed output circuit information of protection equipment with time tag | M_EP_TC_1 |
| [ ] | <20> | := Packed single-point information with status change detection | M_SP_NA_1 |
| [ ] | <21> | := Measured value, normalized value without quality descriptor | M_ME_ND_1 |
| | | | |
| [ X ] | <30> | := Single-point information with time tag CP56Time2a | M_SP_TB_1 |
| [ X ] | <31> | := Double-point information with time tag CP56Time2a | M_DP_TB_1 |
| [ X ] | <32> | := Step position information with time tag CP56Time2a | M_ST_TB_1 |
| [ X ] | <33> | := Bitstring of 32 bit with time tag CP56Time2a | M_BO_TB_1 |
| [ X ] | <34> | := Measured value, normalized value with time tag CP56Time2a | M_ME_TD_1 |
| [ X ] | <35> | := Measured value, scaled value with time tag CP56Time2a | M_ME_TE_1 |
| [ X ] | <36> | := Measured value, short floating point value with time tag CP56Time2a | M_ME_TF_1 |

| | | | |
|---|---|---|---|
| [ X ] | <37> | := Integrated totals with time tag CP56Time2a | M_IT_TB_1 |
| [ ] | <38> | := Event of protection equipment with time tag CP56Time2a | M_EP_TD_1 |
| [ ] | <39> | := Packed start events of protection equipment with time tag CP56Time2a | M_EP_TE_1 |
| [ ] | <40> | := Packed output circuit information of protection equipment with time tag CP56Time2a | M_EP_TF_1 |

Either the ASDUs of the set <2>, <4>, <6>, <8>, <10>, <12>, <14>, <16>, <17>, <18>, <19> or of the set <30> – <40> are used.

## PROCESS INFORMATION IN CONTROL DIRECTION

(station-specific parameter, mark each Type ID ' X ' if it is only used in the standard direction, ' R ' if only used in the reverse direction, and ' B ' if used in both directions).

| | | | |
|---|---|---|---|
| [ X ] | <45> | := Single command | C_SC_NA_1 |
| [ X ] | <46> | := Double command | C_DC_NA_1 |
| [ X ] | <47> | := Regulating step command | C_RC_NA_1 |
| [ X ] | <48> | := Set point command, normalized value | C_SE_NA_1 |
| [ X ] | <49> | := Set point command, scaled value | C_SE_NB_1 |
| [ X ] | <50> | := Set point command, short floating point value | C_SE_NC_1 |
| [ X ] | <51> | := Bitstring of 32 bit | C_BO_NA_1 |
| | | | |
| [ X ] | <58> | := Single command with time tag CP56Time2a | C_SC_TA_1 |
| [ X ] | <59> | := Double command with time tag CP56Time2a | C_DC_TA_1 |
| [ X ] | <60> | := Regulating step command with time tag CP56Time2a | C_RC_TA_1 |
| [ X ] | <61> | := Set point command, normalized value with time tag CP56Time2a | C_SE_TA_1 |
| [ X ] | <62> | := Set point command, scaled value with time tag CP56Time2a | C_SE_TB_1 |
| [ X ] | <63> | := Set point command, short floating point value with time tag CP56Time2a | C_SE_TC_1 |
| [ X ] | <64> | := Bitstring of 32 bit with time tag CP56Time2a | C_BO_TA_1 |

Either the ASDUs of the set <45> – <51> or of the set <58> – <64> are used.

## SYSTEM INFORMATION IN MONITOR DIRECTION

(station-specific parameter, mark ' X ' if used)

| | | |
|---|---|---|
| [     ]     <70> | := End of initialization | M_EI_NA_1 |

## SYSTEM INFORMATION IN CONTROL DIRECTION

(station-specific parameter, mark each Type ID ' X ' if it is only used in the standard direction, ' R ' if only used in the reverse direction, and ' B ' if used in both directions).

| | | |
|---|---|---|
| [ X ]     <100> | := Interrogation command | C_IC_NA_1 |
| [     ]     <101> | := Counter interrogation command | C_CI_NA_1 |
| [     ]     <102> | := Read command | C_RD_NA_1 |
| [ B ]     <103> | := Clock synchronization command | C_CS_NA_1 |
| [     ]     <104> | := Test command | C_TS_NA_1 |
| [     ]     <105> | := Reset process command | C_RP_NA_1 |
| [     ]     <106> | := Delay acquisition command | C_CD_NA_1 |
| [     ]     <107> | := Test command with time tag CP56Time2a | C_TS_TA_1 |

## PARAMETER IN CONTROL DIRECTION

(station-specific parameter, mark each Type ID ' X ' if it is only used in the standard direction, ' R ' if only used in the reverse direction, and ' B ' if used in both directions).

| | | |
|---|---|---|
| [     ]     <110> | := Parameter of measured value, normalized value | P_ME_NA_1 |
| [     ]     <111> | := Parameter of measured value, scaled value | P_ME_NB_1 |
| [     ]     <112> | := Parameter of measured value, short floating point value | P_ME_NC_1 |
| [     ]     <113> | := Parameter activation | P_AC_NA_1 |

## FILE TRANSFER

(station-specific parameter, mark each Type ID ' X ' if it is only used in the standard direction, ' R ' if only used in the reverse direction, and ' B ' if used in both directions).

| | | | |
|---|---|---|---|
| [ B ] | <120> | := File ready | F_FR_NA_1 |
| [ B ] | <121> | := Section ready | F_SR_NA_1 |
| [ B ] | <122> | := Call directory, select file, call file, call section | F_SC_NA_1 |
| [ B ] | <123> | := Last section, last segment | F_LS_NA_1 |
| [ B ] | <124> | := Ack file, ack section | F_AF_NA_1 |
| [ B ] | <125> | := Segment | F_SG_NA_1 |
| [ X ] | <126> | := Directory {blank or X, only available in monitor (standard) direction} | F_DR_TA_1 |

**TYPE IDENTIFIER AND CAUSE OF TRANSMISSION ASSIGNMENTS**

(station-specific parameters)

Shaded boxes: option not required.
Black boxes: option not permitted in this companion standard
Blank: functions or ASDU not used.

Mark Type Identification/Cause of transmission combinations:
' X ' if only used in the standard direction;
' R ' if only used in the reverse direction;
' B ' if used in both directions.

| Type identification | | Cause of transmission | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 20 to 36 | 37 to 41 | 44 |
| <1> | M_SP_NA_1 | | | | | | | | | | | | | | | | |
| <2> | M_SP_TA_1 | | | | | | | | | | | | | | | | |
| <3> | M_DP_NA_1 | | | | | | | | | | | | | | | | |
| <4> | M_DP_TA_1 | | | | | | | | | | | | | | | | |
| <5> | M_ST_NA_1 | | | | | | | | | | | | | | | | |
| <6> | M_ST_TA_1 | | | | | | | | | | | | | | | | |
| <7> | M_BO_NA_1 | | | | | | | | | | | | | | | | |
| <8> | M_BO_TA_1 | | | | | | | | | | | | | | | | |
| <9> | M_ME_NA_1 | | | | | | | | | | | | | | | | |

| <10> | M_ME_TA_1 |
| <11> | M_ME_NB_1 |
| <12> | M_ME_TB_1 |
| <13> | M_ME_NC_1 |
| <14> | M_ME_TC_1 |
| <15> | M_IT_NA_1 |
| <16> | M_IT_TA_1 |
| <17> | M_EP_TA_1 |
| <18> | M_EP_TB_1 |
| <19> | M_EP_TC_1 |
| <20> | M_PS_NA_1 |
| <21> | M_ME_ND_1 |
| <30> | M_SP_TB_1 |
| <31> | M_DP_TB_1 |
| <32> | M_ST_TB_1 |
| <33> | M_BO_TB_1 |
| <34> | M_ME_TD_1 |
| <35> | M_ME_TE_1 |
| <36> | M_ME_TF_1 |
| <37> | M_IT_TB_1 |
| <38> | M_EP_TD_1 |
| <39> | M_EP_TE_1 |
| <40> | M_EP_TF_1 |
| <45> | C_SC_NA_1 |
| <46> | C_DC_NA_1 |
| <47> | C_RC_NA_1 |
| <48> | C_SE_NA_1 |

| <49> | C_SE_NB_1 |
|------|-----------|
| <50> | C_SE_NC_1 |
| <51> | C_BO_NA_1 |
| <58> | C_SC_TA_1 |
| <59> | C_DC_TA_1 |
| <60> | C_RC_TA_1 |
| <61> | C_SE_TA_1 |
| <62> | C_SE_TB_1 |
| <63> | C_SE_TC_1 |
| <64> | C_BO_TA_1 |
| <70> | M_EI_NA_1* |
| <100> | C_IC_NA_1 |
| <101> | C_CI_NA_1 |
| <102> | C_RD_NA_1 |
| <103> | C_CS_NA_1 |
| <104> | C_TS_NA_1 |
| <105> | C_RP_NA_1 |
| <106> | C_CD_NA_1 |
| <107> | C_TS_TA_1 |
| <110> | P_ME_NA_1 |
| <111> | P_ME_NB_1 |
| <112> | P_ME_NC_1 |
| <113> | P_AC_NA_1 |
| <120> | F_FR_NA_1 |
| <121> | F_SR_NA_1 |
| <122> | F_SC_NA_1 |
| <123> | F_LS_NA_1 |

| <124> | F_AF_NA_1 | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| <125> | F_SG_NA_1 | | | | | | | | | | | | | | |
| <126> | F_DR_TA_1* | | | | | | | | | | | | | | |

\* Blank or X only

## 5. BASIC APPLICATION FUNCTIONS

### STATION INITIALIZATION

(station-specific parameter, mark 'X' if function is used)

[      ]      Remote initialization

### CYCLIC DATA TRANSMISSION

(station-specific parameter, mark ' X ' if function is only used in the standard direction, ' R ' if only used in the reverse direction, and ' B ' if used in both directions)

[ X ]      Cyclic data transmission

### READ PROCEDURE

(station-specific parameter, mark ' X ' if function is only used in the standard direction, ' R ' if only used in the reverse direction, and ' B ' if used in both directions)

[      ]      Read procedure

### SPONTANEOUS TRANSMISSION

(station-specific parameter, mark ' X ' if function is only used in the standard direction, ' R ' if only used in the reverse direction, and ' B ' if used in both directions)

[ X ]      Spontaneous transmission

### DOUBLE TRANSMISSION OF INFORMATION OBJECTS WITH CAUSE OF TRANSMISSION SPONTANEOUS

(station-specific parameter, mark each information type ' X ' where both a Type ID without time and corresponding Type ID with time are issued in response to a single spontaneous change of a monitored object)

The following type identifications may be transmitted in succession caused by a single status change of an information object. The particular information object addresses for which double transmission is enabled are defined in a project-specific list.

[        ]      Single-point information M_SP_NA_1, M_SP_TA_1, M_SP_TB_1 and M_PS_NA_1

[        ]      Double-point information M_DP_NA_1, M_DP_TA_1 and M_DP_TB_1

[        ]      Step position information M_ST_NA_1, M_ST_TA_1 and M_ST_TB_1

[        ]      Bitstring of 32 bit M_BO_NA_1, M_BO_TA_1 and M_BO_TB_1 (if defined for a specific project)

[        ]      Measured value, normalized value M_ME_NA_1, M_ME_TA_1, M_ME_ND_1 and M_ME_TD_1

[        ]      Measured value, scaled value M_ME_NB_1, M_ME_TB_1 and M_ME_TE_1

[        ]      Measured value, short floating point number M_ME_NC_1, M_ME_TC_1 and M_ME_TF_1

### STATION INTERROGATION

(station-specific parameter, mark ' X ' if function is only used in the standard direction, ' R ' if only used in the reverse direction, and ' B ' if used in both directions).

| [ X ]    global | | |
|---|---|---|
| [    ]    group 1 | [    ]    group 7 | [    ]    group 13 |
| [    ]    group 2 | [    ]    group 8 | [    ]    group 14 |
| [    ]    group 3 | [    ]    group 9 | [    ]    group 15 |
| [    ]    group 4 | [    ]    group 10 | [    ]    group 16 |
| [    ]    group 5 | [    ]    group 11 | |
| [    ]    group 6 | [    ]    group 12 | Information object addresses assigned to each group must be shown in a separate table. |

### CLOCK SYNCHRONIZATION

(station-specific parameter, mark ' X ' if function is only used in the standard direction, ' R ' if only used in the reverse direction, and ' B ' if used in both directions).

[ B ]    Clock synchronization

optional, see 7.6

**COMMAND TRANSMISSION**

(station-specific parameter, mark ' X ' if function is only used in the standard direction, ' R ' if only used in the reverse direction, and ' B ' if used in both directions).

[ X ]      Direct command transmission
[ X ]      Direct set point command transmission
[ X*]      Select and execute command
[ X*]      Select and execute set point command
[ X*]      C_SE ACTTERM used

[ X ]      No additional definition
[ X*]      Short-pulse duration (duration determined by a system parameter in the outstation)
[ X*]      Long-pulse duration (duration determined by a system parameter in the outstation)
[ X*]      Persistent output

[ configurable* ]      Supervision of maximum delay in command direction of commands and set point commands
[ not limited*   ]      Maximum allowable delay of commands and set point commands

*only toggether with Command Input module

**TRANSMISSION OF INTEGRATED TOTALS**

(station-specific parameter, mark ' X ' if function is only used in the standard direction, ' R ' if only used in the reverse direction, and ' B ' if used in both directions).

[    ]    Mode A: Local freeze with spontaneous transmission
[    ]    Mode B: Local freeze with counter interrogation
[    ]    Mode C: Freeze and transmit by counter-interrogation commands
[    ]    Mode D: Freeze by counter-interrogation command, frozen values reported spontaneously

[    ]    Counter read
[    ]    Counter freeze without reset
[    ]    Counter freeze with reset
[    ]    Counter reset

[    ]    General request counter
[    ]    Request counter group 1
[    ]    Request counter group 2
[    ]    Request counter group 3
[    ]    Request counter group 4

## PARAMETER LOADING

(station-specific parameter, mark ' X ' if function is only used in the standard direction, ' R ' if only used in the reverse direction, and ' B ' if used in both directions).

[      ]      Threshold value
[      ]      Smoothing factor
[      ]      Low limit for transmission of measured values
[      ]      High limit for transmission of measured values

## PARAMETER ACTIVATION

(station-specific parameter, mark ' X ' if function is only used in the standard direction, ' R ' if only used in the reverse direction, and ' B ' if used in both directions).

[      ]      Act/deact of persistent cyclic or periodic transmission of the addressed object

## TEST PROCEDURE

(station-specific parameter, mark ' X ' if function is only used in the standard direction, ' R ' if only used in the reverse direction, and ' B ' if used in both directions).
[      ]      Test procedure

## FILE TRANSFER

(station-specific parameter, mark 'X' if function is used).
File transfer in monitor direction
[ X ]      Transparent file
[ X ]      Transmission of disturbance data of protection equipment
[ X ]      Transmission of sequences of events
[ X ]      Transmission of sequences of recorded analogue values

File transfer in control direction
[ X ]      Transparent file

## BACKGROUND SCAN

(station-specific parameter, mark ' X ' if function is only used in the standard direction, ' R ' if only used in the reverse direction, and ' B ' if used in both directions).
[ X ]      Background scan

**ACQUISITION OF TRANSMISSION DELAY**

(station-specific parameter, mark ' X ' if function is only used in the standard direction, ' R ' if only used in the reverse direction, and ' B ' if used in both directions).

[    ]    Acquisition of transmission delay

**DEFINITION OF TIME OUTS: 104 ONLY**

| Parameter | Default value | Remarks | Selected value |
|---|---|---|---|
| t0 | 30s | Time-out of connection establishment | setting |
| t1 | 15s | Time-out of send or test APDUs | setting |
| t2 | 10s | Time-out for acknowledges in case of no data messages; t2 < t1 | setting |
| t3 | 20s | Time-out for sending test frames in case of a long idle state; t3 > t1 | setting |

Maximum range of values for all time-outs: 1 to 255 s, accuracy 1 s.

**MAXIMUM NUMBER OF OUTSTANDING I FORMAT APDUS K AND LATEST ACKNOWLEDGE APDUS (W): 104 ONLY**

| Parameter | Default value | Remarks | Selected value |
|---|---|---|---|
| k | 12 APDUs | Maximum difference receive sequence number to send state variable | setting |
| w | 8 APDUs | Latest acknowledge after receiving w I format APDUs | setting |

Maximum range of values k: 1 to 32767 APDUs, accuracy 1 APDU
Maximum range of values w: 1 to 32767 APDUs, accuracy 1 APDU (Recommendation: w should not exceed two-thirds of k)

**PORTNUMBER: 104 ONLY**

| Parameter | Value | Remarks |
|---|---|---|
| Portnumber | 2404 | setting |

**REDUNDANT CONNECTIONS: 104 ONLY**

[ 2 ]　　Number N of redundant connections according Edition 2 used

**RFC 2200 SUITE**

RFC 2200 is an official Internet Standard which describes the state of standardization of protocols used in the Internet as determined by the Internet Architecture Board (IAB). It offers a broad spectrum of actual standards used in the Internet. The suitable selection of documents from RFC 2200 defined in this standard for given projects has to be chosen by the user of this standard.

[　　]　　Ethernet 802.3
[　　]　　Serial X.21 interface
[　　]　　Other selection from RFC 2200:


List of valid documents from RFC 2200
1. ......................................................................
2. ......................................................................
3. ......................................................................
4. ......................................................................
5. ......................................................................
6. ......................................................................
7. etc.


# 11. Error analysis

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


## 11.1　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 7.60 -> Diagviewer.*

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

**%ProgramData%\COPA-DATA\LOG**.

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

In the Diagnosis Viewer you can also:

- ▶ Follow newly-created entries in real time

- ▶ customize the logging settings

- ▶ change the folder in which the LOG files are saved

Note:

1. The Diagnosis Viewer displays all entries in UTC (coordinated world time) and not in local time.

2. The Diagnosis Viewer does not display all columns of a LOG file per default. To display more columns activate property **Add all columns with entry** in the context menu of the column header.

3. If you only use **Error-Logging**, the problem description is in the column **Error text**. For other diagnosis level the description is in the column **General text**.

4. For communication problems many drivers also log error numbers which the PLC assigns to them. They are displayed in **Error text** or **Error code** or **Driver error parameter** (**1** and **2**). Hints on the meaning of error codes can be found in the driver documentation and the protocol/PLC description.

5. At the end of your test set back the diagnosis level from **Debug** or **Deep Debug**. At **Debug** and **Deep Debug** there are a great deal of data for logging which are saved to the hard drive and which can influence your system performance. They are still logged even after you close the Diagnosis Viewer.

> ⚠  **Attention**
>
> In Windows CE errors are not logged per default due to performance reasons.

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


## 11.2    Check list

- ▶ Is the COM port in use by another application or are the settings incorrect?

- ▶ Is the device (PLC) that you are trying to communicate with connected to the power supply?

- ▶ Is the cable between PLC and PC/IPC connected correctly?

- ▶ Have you analyzed the error file (which errors did occur)?

- ▶ For additional error analyses, please send a project backup and the LOG file of the DiagViewer to the support team responsible for you.