zenon
by COPA-DATA

zenon Treiber Handbuch

KDNP1

v.8.20

© 2020 Ing. Punzenberger COPA-DATA GmbH

Alle Rechte vorbehalten.

Die Weitergabe und Vervielfältigung dieses Dokuments ist - gleich in welcher Art und Weise - nur mit schriftlicher Genehmigung der Firma COPA-DATA gestattet. Technische Daten dienen nur der Produktbeschreibung und sind keine zugesicherten Eigenschaften im Rechtssinn. Änderungen - auch in technischer Hinsicht - vorbehalten.

Inhaltsverzeichnis

1 Willkommen bei der COPA-DATA Hilfe 5

2 KDNP1 5

3 KDNP1 - Datenblatt 7

4 Treiber-Historie 8

5 Voraussetzungen 9

5.1 PC 9

6 Konfiguration 9

6.1 Anlegen eines Treibers 10

6.2 Einstellungen im Treiberdialog 13

6.2.1 Allgemein 14

6.2.2 Com 18

6.2.3 Options 20

6.2.4 Connections 23

7 Variablen anlegen 34

7.1 Variablen im Editor anlegen 34

7.2 Adressierung 38

7.3 Treiberobjekte und Datentypen 39

7.3.1 Treiberobjekte 39

7.3.2 Zuordnung der Datentypen 41

7.4 Variablen anlegen durch Import 46

7.4.1 XML Import 46

7.4.2 DBF Import/Export 47

7.5 Kommunikationsdetails (Treibervariablen) 53

8 Treiberspezifische Funktionen 59

9 Funktion Treiberkommandos 61

10 Fehleranalyse 67

10.1 Analysetool 67

10.2 Treiberüberwachung 68

10.3 Checkliste 69

1 Willkommen bei der COPA-DATA Hilfe

ZENON VIDEO-TUTORIALS

Praktische Beispiele für die Projektierung mit zenon finden Sie in unserem YouTube-Kanal (https://www.copadata.com/tutorial_menu). Die Tutorials sind nach Themen gruppiert und geben einen ersten Einblick in die Arbeit mit den unterschiedlichen zenon Modulen. Alle Tutorials stehen in englischer Sprache zur Verfügung.

ALLGEMEINE HILFE

Falls Sie in diesem Hilfekapitel Informationen vermissen oder Wünsche für Ergänzungen haben, wenden Sie sich per E-Mail an documentation@copadata.com.

PROJEKTUNTERSTÜTZUNG

Unterstützung bei Fragen zu konkreten eigenen Projekten erhalten Sie vom Customer Service, den Sie per E-Mail an support@copadata.com erreichen.

LIZENZEN UND MODULE

Sollten Sie feststellen, dass Sie weitere Module oder Lizenzen benötigen, sind unsere Mitarbeiter unter sales@copadata.com gerne für Sie da.

2 KDNP1

Der **KDNP1**-Treiber dient zur Kommunikation mit dem **KDNP1**-Protokoll der Firma Kepco, welches auf dem DNP3-Standard basiert. Der Treiber fungiert dabei als Master. Auf Protokollebene werden serielle Kommunikation sowie IP-Kommunikation über TCP zu mehrere Outstations unterstützt.

- Der **KDNP1**-Treiber pollt nach Objekten. Dieses Polling erfolgt zyklisch. Zusätzlich kann dieses Polling mit einem Treiberkommando aktiviert oder deaktiviert werden.

- Zusätzlich unterstützt der Treiber eine Kommunikation mittels Confirmed Data link layer. Dafür kann die Eigenschaft **Data link confirm** im Treiberdialog **Outstation** aktiviert werden. Für diese Eigenschaft können Timeout und Anzahl an Versuchen konfiguriert werden.

BEGRIFFSDEFINITIONEN

Zum besseren Verständnis dieses Dokuments finden Sie in der folgenden Liste die Definitionen wichtiger Begriffe.

Begriff	Definition
Master	Kontrollierende Station. Ein **Master** sendet Leseanfragen und Steueranfragen an die **Outstation**.
Objekt Group	Datentyp oder Datenart in der **Outstation** mit einer definierten Funktionalität. Beispiele für Zuordnungen: - *Object Group 30* steht für einen statischen Analogeingang. - Eine Wertänderung für einen Analogeingang ist *Object Group 32* zugeordnet. - *Object Group 40* dient zum Lesen eines Analogausgangs. - *Object Group 41* dient zum Schreiben auf einen Analogausgang.
Outstation	**SPS** oder **RTU** in **KDNP1**. Eine **Outstation** ist Slave auf Protokollebene und sendet auf Anfrage Daten an den **Master** .
Point	Äquivalent einer Variable in der Outstation. Ein **Point** wird angesprochen mit einer **Point Number** (Offset) pro **Objekt Group**, wobei z.B. gilt: *Point 12* für **Object Group** *30* und *Point 12* für **Object Group** *32* haben beide den gleichen Analogeingang als Basis, *Point 12* für **Object Group** *1* jedoch ist ein komplett unabhängiger Binäreingang. Der allgemeine Begriff für eine Wert-, Zeit- oder Statusänderung, ist **DNP Object**.
Variation	Format, in dem die **Oustation** einen statischen Wert oder eine Wertänderung in **Class** *0* bzw. **Class** *1*, *2* oder *3* speichert. Dass kann in der **Outstation** pro **Object Group** oder pro **Point** konfiguriert werden. Die **Variation** bestimmt z.B. ob: - ein statischer Wert (**Class** *0*) oder eine Wertänderung, eine Integer oder Floating Point ist - ein Zeitstempel gespeichert wird oder nicht, oder, ob Object Flags gespeichert werden oder nicht

KOMMUNIKATION

Die Kommunikation erfolgt vor allem pollend. Dabei führt der Treiber in konfigurierbaren Abständen ein entsprechendes Class Poll durch. Es können unterschiedliche Polling-Intervalle für Binary Inputs, Binary Outputs und Analog Inputs definiert werden.

Der Treiber verarbeitet die empfangenen Werte und sendet sie zur Runtime weiter.

3 KDNP1 - Datenblatt

Allgemein:	
Treiberdateiname	KDNP1.exe
Treiberbezeichnung	Kepco KDNP1 Treiber
Steuerungs-Typen	Outstations, die das KDNP1 Protokoll unterstützen
Steuerungs-Hersteller	KEPCO

Treiber unterstützt:	
Protokoll	KDNP1
Adressierung: Adress-basiert	Address based
Adressierung: Namens-basiert	--
Kommunikation spontan	--
Kommunikation pollend	X
Online Browsing	--
Offline Browsing	--
Echtzeitfähig	--
Blockwrite	--
Modemfähig	--
RDA numerisch	--
RDA String	--

Treiber unterstützt:	
Hysterese	X
erweiterte API	--
Unterstützung von Statusbit **WR-SUC**	X
alternative IP-Adresse	--

Voraussetzungen:	
Hardware PC	Serieller Anschluss, Standard Netzwerkkarte
Software PC	-
Hardware Steuerung	-
Software Steuerung	-
Benötigt v-dll	X

Plattformen:	
Betriebssysteme	Windows 10; Windows 7; Windows 8; Windows 8.1; Windows Server 2008 R2; Windows Server 2012; Windows Server 2012 R2; Windows Server 2016

4 Treiber-Historie

Datum	Build Nummer	Änderung
15.08.15	21913	Treiberdokumentation wurde neu erstellt

5 Voraussetzungen

Dieses Kapitel enthält Informationen zu den Voraussetzungen, die für die Verwendung des Treibers erforderlich sind.

5.1 PC

HARDWARE

- Serielle Schnittstelle
- Ethernet TCP

CE

Der **KDNP1**-Treiber ist nicht für CE verfügbar.

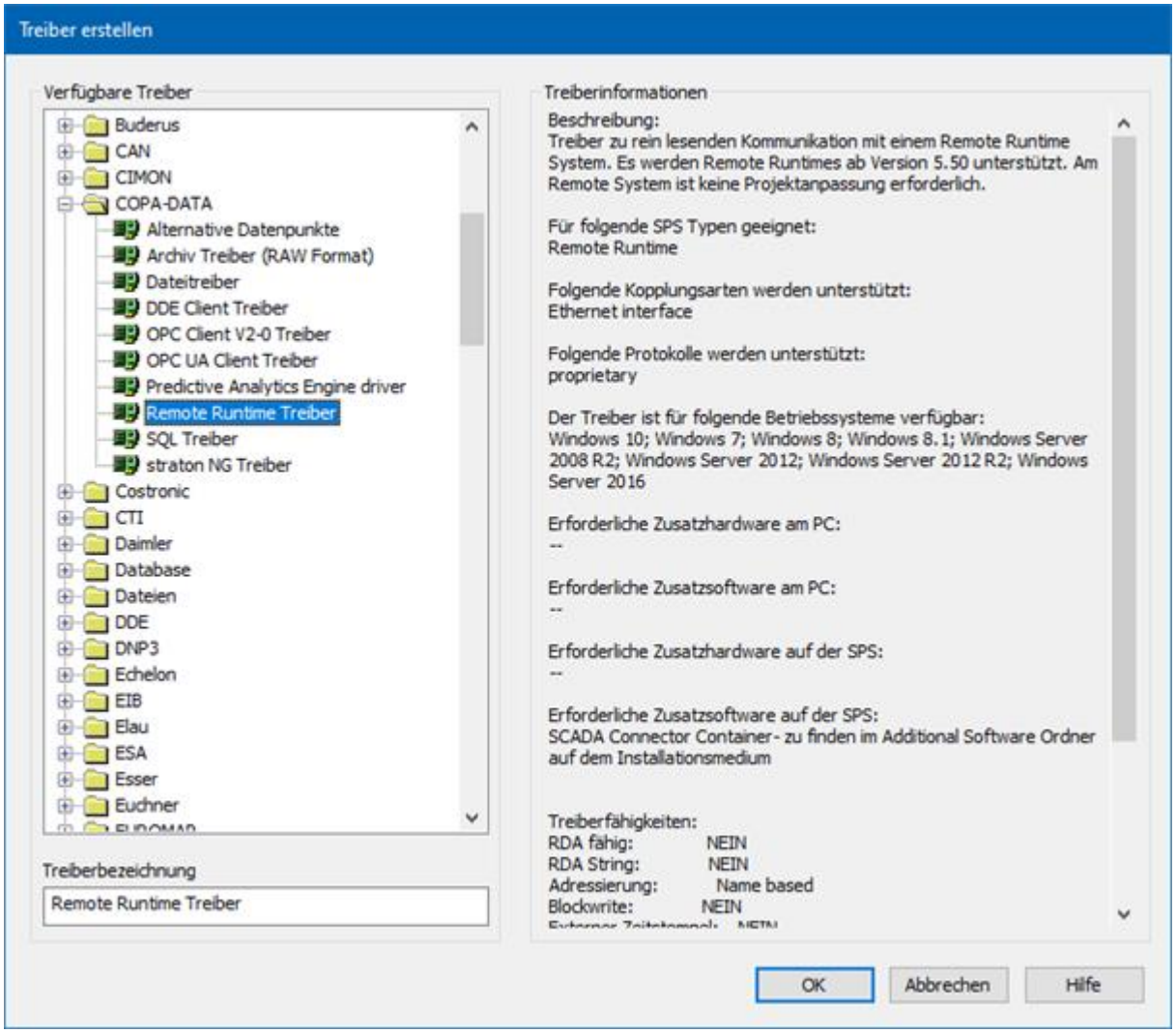
6 Konfiguration

In diesem Kapitel lesen Sie, wie Sie den Treiber im Projekt anlegen und welche Einstellungen beim Treiber möglich sind.

> **Info**
>
> Weitere Einstellungen, die Sie für Variablen in zenon vornehmen können, finden Sie im Kapitel Variablen der Online-Hilfe.

6.1 Anlegen eines Treibers

Im Dialog **Treiber erstellen** wählen Sie aus einer Liste jenen Treiber, den Sie neu anlegen wollen.



Parameter	Beschreibung
Verfügbare Treiber	Liste aller verfügbaren Treiber. Die Darstellung erfolgt in einer Baumstruktur: *[+]* erweitert die Ordnerstruktur und zeigt die darin enthaltenen Treiber. [-] reduziert die Ordnerstruktur Default: *keine Auswahl*
Treiberbezeichnung	Eindeutige **Bezeichnung** des Treibers. Default: *leer* Das Eingabefeld wird nach Auswahl eines Treibers

Parameter	Beschreibung
	aus der Liste der verfügbaren Treiber mit der vordefinierten **Bezeichnung** vorausgefüllt.
Treiberinformationen	Weiterführende Informationen über den gewählten Treiber. Default: *leer* Nach Auswahl eines Treibers werden in diesem Bereich die Informationen zum gewählten Treiber angezeigt.

DIALOG BEENDEN

Option	Beschreibung
OK	Übernimmt alle Einstellungen und öffnet den Treiberkonfigurationsdialog des ausgewählten Treibers.
Abbrechen	Verwirft alle Änderungen und schließt den Dialog.
Hilfe	Öffnet die Online-Hilfe.

> **Info**
>
> Die Inhalte dieses Dialogs sind in der Datei Treiber_[Sprachkürzel].xml gespeichert. Sie finden diese Datei im Ordner *C:\ProgramData\COPA-DATA\zenon[Versionsnummer]*.

TREIBER NEU ANLEGEN

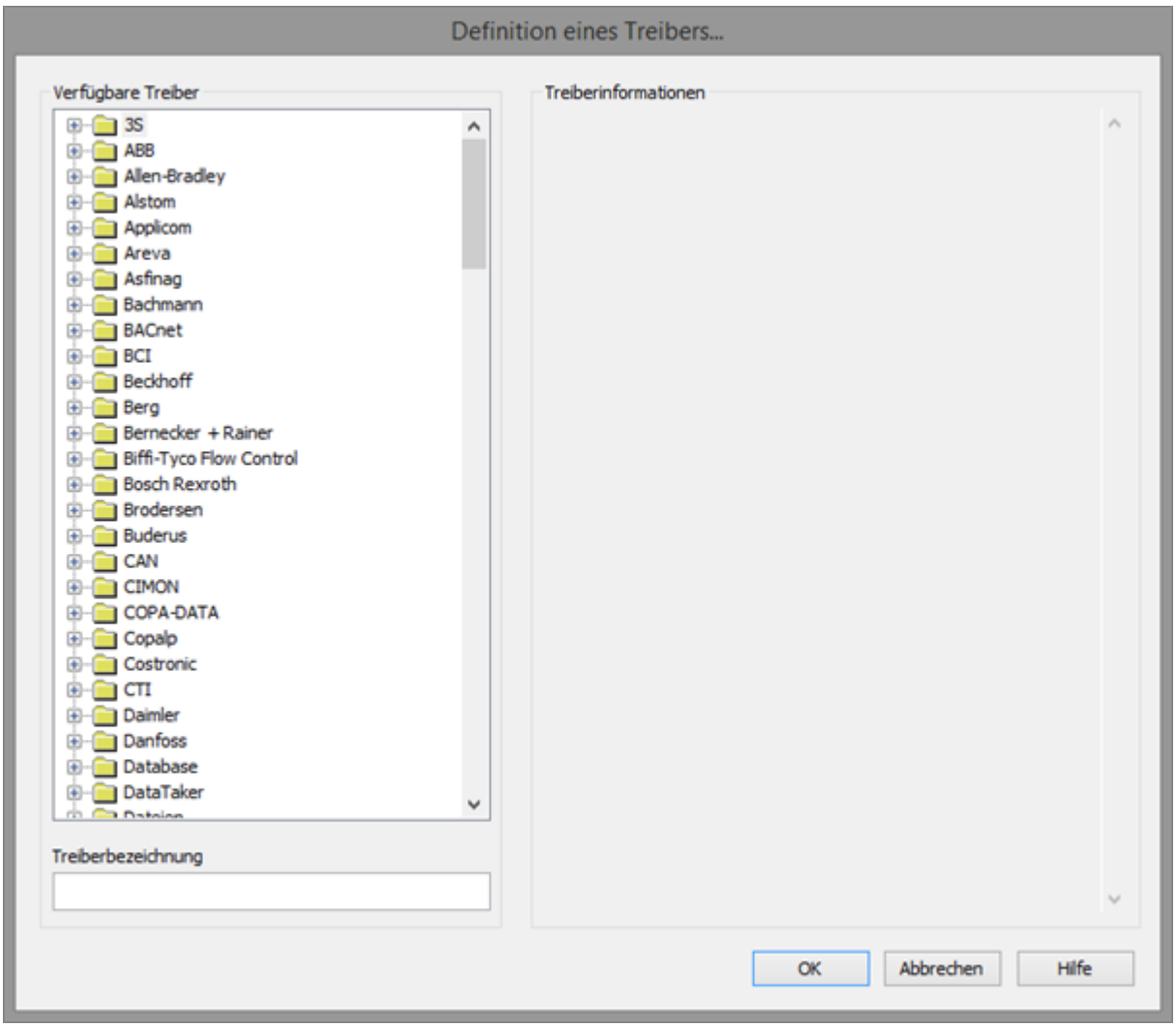
Um einen neuen Treiber anzulegen:

1. Klicken Sie mit der rechten Maustaste im Projektmanager auf **Treiber** und wählen Sie im Kontextmenü **Treiber neu** aus.

   Optional: Wählen Sie die Schaltfläche **Treiber neu** aus der Symbolleiste der Detailansicht der **Variablen**.

   Der Dialog **Treiber erstellen** wird geöffnet.

2. Der Dialog bietet eine Auflistung aller verfügbaren Treiber an.



3. Wählen Sie den gewünschten Treiber und benennen Sie diesen im Eingabefeld **Treiberbezeichnung**.
   Dieses Eingabefeld entspricht der Eigenschaft **Bezeichnung**. Per Default wird der Name des ausgewählten Treibers in diesem Eingabefeld automatisch eingefügt.
   Für die **Treiberbezeichnung** gilt:

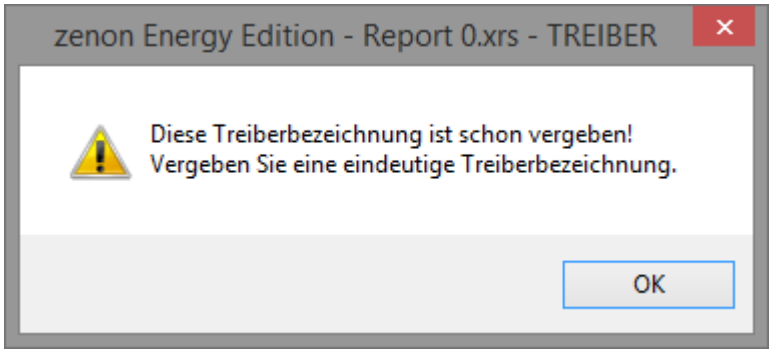
   - Die **Treiberbezeichnung** muss eindeutig sein.
     Wird ein Treiber mehrmals im Projekt verwendet, so muss jeweils eine neue Bezeichnung vergeben werden.
     Dies wird durch Klick auf die Schaltfläche **OK** evaluiert. Ist die Treiber im Projekt bereits vorhanden wird dies mit einem Warndialog angezeigt.
   - Die **Treiberbezeichnung** ist Bestandteil des Dateinamens.
     Daher darf Sie nur Zeichen enthalten, die vom Betriebssystem unterstützt werden. Nicht gültige Zeichen werden durch einen Unterstrich (_) ersetzt.
   - **Achtung:** Die Bezeichnung kann später nicht mehr geändert werden.

4. Bestätigen Sie den Dialog mit Klick auf die Schaltfläche **OK**.
   Der Konfigurationsdialog des ausgewählten Treibers wird geöffnet.

Hinweis: Treibernamen sind nicht sprachumschaltbar. Sie werden später immer in der Sprache angezeigt, in der sie angelegt wurden, unabhängig von der Sprache des Editors. Das gilt auch für Treiberobjekttypen.

DIALOG TREIBERBEZEICHNUNG BEREITS VORHANDEN

Ist ein Treiber bereits im Projekt vorhanden wird dies in einem Dialog angezeigt. Mit Klick auf die Schaltfläche **OK** wird der Warndialog geschlossen. Der Treiber kann korrekt benannt werden.



ZENON PROJEKT

Bei neu angelegten Projekten werden die folgenden Treiber automatisch angelegt:

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

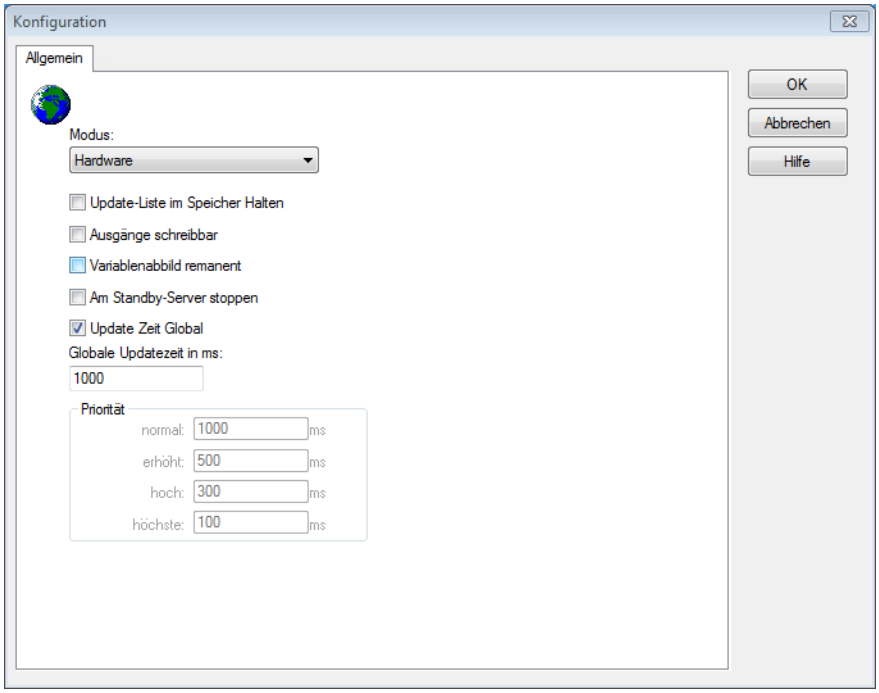
> **Info**
>
> In einem zenon Projekt müssen nur die benötigten Treiber vorhanden sein. Treiber können bei Bedarf zu einem späteren Zeitpunkt hinzugefügt werden.

6.2 Einstellungen im Treiberdialog

Folgende Einstellungen können Sie beim Treiber vornehmen:

6.2.1 Allgemein

Beim Anlegen eines Treibers wird der Konfigurationsdialog geöffnet. Um den Dialog später zum Bearbeiten zu öffnen, führen Sie einen Doppelklick auf den Treiber in der Liste aus oder klicken Sie auf die Eigenschaft **Konfiguration**.



Option	Beschreibung
Modus	Ermöglicht ein Umschalten zwischen Hardware und Simulationsmodus ▸ *Hardware*: Die Verbindung zur Steuerung wird hergestellt. ▸ Simulation - statisch: Es wird keine Kommunikation zur Steuerung aufgebaut, die Werte werden vom Treiber simuliert. In diesem Modus bleiben die Werte konstant oder die Variablen behalten die über zenon Logic gesetzen Werte. Jede Variable hat seinen eigenen Speicherbereich. Zum Beispiel zwei Variablen vom Typ Merker mit Offset 79, können zur Runtime unterschiedliche Werte haben und beeinflussen sich gegenseitig nicht. Ausnahme: Der Simulatortreiber. ▸ *Simulation - zählend*: Es wird keine Kommunikation zur Steuerung aufgebaut, die Werte werden vom Treiber simuliert.

Option	Beschreibung
	In diesem Modus zählt der Treiber die Werte innerhalb ihres Wertebereichs automatisch hoch. ▸ *Simulation - programmiert*: Es wird keine Kommunikation zur Steuerung aufgebaut, die Werte werden von einem frei programmierbaren Simulationsprojekt berechnet. Das Simulationsprojekt wird mit der zenon Logic Workbench erstellt und läuft in einer in den Treiber integrierten zenon Logic Runtime ab. Details siehe Kapitel Treibersimulation.
Update-Liste im Speicher Halten	Einmal angeforderte Variablen werden weiterhin von der Steuerung angefordert, auch wenn diese aktuell nicht mehr benötigt werden. Dies hat den Vorteil, dass z B. mehrmalige Bildumschaltungen nach dem erstmaligen Aufschalten beschleunigt werden, da die Variablen nicht neu angefordert werden müssen. Der Nachteil ist eine erhöhte Belastung der Kommunikation zur Steuerung.
Ausgänge schreibbar	▸ *Aktiv*: Ausgänge können beschrieben werden. ▸ *Inaktiv*: Das Beschreiben der Ausgänge wird unterbunden. **Hinweis**: Steht nicht für jeden Treiber zur Verfügungen.
Variablenabbild remanent	Diese Option speichert und restauriert den aktuellen Wert, den Zeitstempel und die Status eines Datenpunkts. Grundvoraussetzung: Die Variable muss einen gültigen Wert und Zeitstempel besitzen. Das Variablenabbild wird im Modus Hardware gespeichert, wenn einer dieser Status aktiv ist: ▸ Benutzerstatus *M1* (*0*) bis *M8* (*7*) ▸ *REVISION(9)* ▸ *AUS(20)* ▸ *ERSATZWERT(27)* Das Variablenabbild wird immer gespeichert wenn: ▸ die Variable vom Objekttyp

Option	Beschreibung
	Kommunikationsdetails ist ▸ der Treiber im Simulationsmodus läuft. (nicht programmierte Simulation) Folgende Status werden beim Start der Runtime nicht restauriert: ▸ *SELECT(8)* ▸ *WR-ACK(40)* ▸ *WR-SUC(41)* Der Modus **Simulation - programmiert** beim Treiberstart ist kein Kriterium, um das remanente Variablenabbild zu restaurieren.
Am Standby Server stoppen	Einstellung für Redundanz bei Treibern, die nur eine Kommunikationsverbindung erlauben. Dazu wird der Treiber am Standby Server gestoppt und erst beim Hochstufen wieder gestartet. **Achtung:** Ist diese Option aktiv, ist die lückenlose Archivierung nicht mehr gewährleistet. ▸ *Aktiv*: Versetzt den Treiber am nicht-prozessführenden Server automatisch in einen Stopp-ähnlichen Zustand. Im Unterschied zum Stoppen über Treiberkommando erhält die Variable nicht den Status **abgeschaltet**, sondern einen leeren Wert. Damit wird verhindert, dass beim Hochstufen zum Server nicht relevante Werte in AML , CEL und Archiv erzeugt werden. Default: *inaktiv* **Hinweis:** Nicht verfügbar, wenn CE Terminal als Datenserver dient. Weitere Informationen dazu erhalten Sie im Handbuch zenon Operator im Kapitel CE Terminal als Datenserver.
Update Zeit Global	Einstellung für globale Update-Zeiten in Millisekunden: ▸ *Aktiv*: Die eingestellte **Globale Update Zeit** wird für alle Variablen im Projekt verwendet. Die bei den Variablen eingestellte Priorität wird nicht

Option	Beschreibung
	verwendet. ▸ *Inaktiv*: Die eingestellten Prioritäten werden für die einzelnen Variablen verwendet. **Ausnahmen:** Spontane Treiber ignorieren diese Option. Sie nutzen in der Regel die kürzest mögliche Update Zeit. Details siehe Abschnitt **Update Zeit spontane Treiber**.
Priorität	Hier werden die Pollingzeiten der einzelnen Prioritätsklassen eingestellt. Alle Variablen mit der entsprechenden Priorität werden in der eingestellten Zeit gepollt. Die Zuordnung der Variablen erfolgt separat bei jeder Variablen über die Einstellungen in den Variableneigenschaften. Mit den Prioritätsklassen kann die Kommunikation der einzelnen Variablen auf die Wichtigkeit oder benötigte Aktualität abgestuft werden. Daraus ergibt sich eine verbesserte Verteilung der Kommunikationslast. **Achtung:** Prioritätsklassen werden nicht von jedem Treiber unterstützt, z.B. von spontan kommunizierenden zenon Treibern.

DIALOG BEENDEN

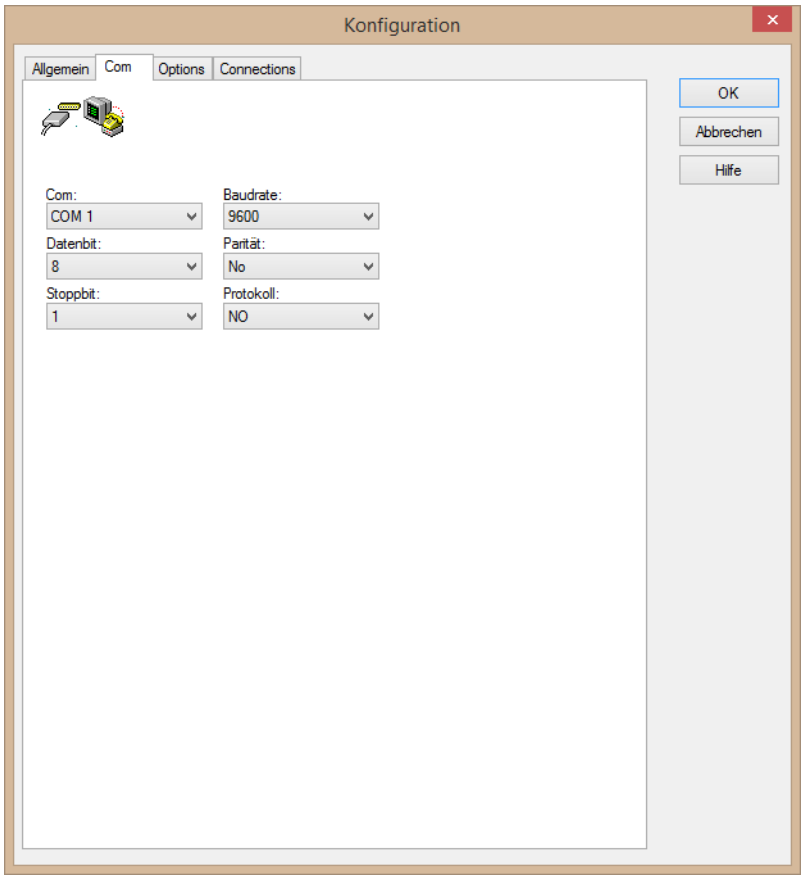
Option	Beschreibung
OK	Übernimmt alle Änderungen in allen Registerkarten und schließt den Dialog.
Abbrechen	Verwirft alle Änderungen in allen Registerkarten und schließt den Dialog.
Hilfe	Öffnet die Online-Hilfe.

UPDATE ZEIT SPONTANE TREIBER

Bei spontanen Treibern wird beim **Sollwert Setzen**, **Advisen** von Variablen und bei **Requests** sofort ein Lesezyklus ausgelöst - unabhängig von der eingestellten Update Zeit. Damit wird sicher gestellt, dass der Wert nach dem Schreiben in der Visualisierung sofort zur Verfügung steht. In der Regel beträgt die Updatezeit 100 ms.

Spontane Treiber sind **ArchDrv**, **BiffiDCM**, **BrTcp32**, **DNP3**, **Esser32**, **FipDrv32**, **FpcDrv32**, **IEC850**, **IEC870**, **IEC870_103**, **Otis**, **RTK9000**, **S7DCOS**, **SAIA_Slave**, **STRATON32** und **Trend32**.

6.2.2 Com



Parameter	Beschreibung
Com	Auswahl Com-Port. Auswahl aus Dropdownliste: *COM 1* bis *COM 64* Default: *1*
Baudrate	Auswahl Baudrate. Anpassen an Steuerung. Auswahl aus Dropdownliste: *110* bis *256000* Default: *9600*
Datenbit	Anzahl Datenbits. Anpassen an Steuerung. Auswahl aus Dropdownliste: *5, 6, 7, 8*

Parameter	Beschreibung
	Default: *8*
Stopbit	Auswahl Stopbit. Anpassen an Steuerung. Auswahl aus Dropdownliste: ▸ *1* ▸ *1.5* ▸ *2* Default: *1*
Parität	Auswahl der Parität. Anpassen an Steuerung. Auswahl aus Dropdownliste: ▸ *No* ▸ *Odd* ▸ *Even* Default: *No*
Protokoll	Auswahl Protokoll. Anpassen an Steuerung. Auswahl aus Dropdownliste: ▸ *No* ▸ *Xon/Xoff* ▸ *Rts/CTS* ▸ *DTR/DSR* Default: *No*

DIALOG BEENDEN

Option	Beschreibung
OK	Übernimmt alle Änderungen in allen Registerkarten und schließt den Dialog.
Abbrechen	Verwirft alle Änderungen in allen Registerkarten und schließt den Dialog.
Hilfe	Öffnet die Online-Hilfe.

Option	Beschreibung

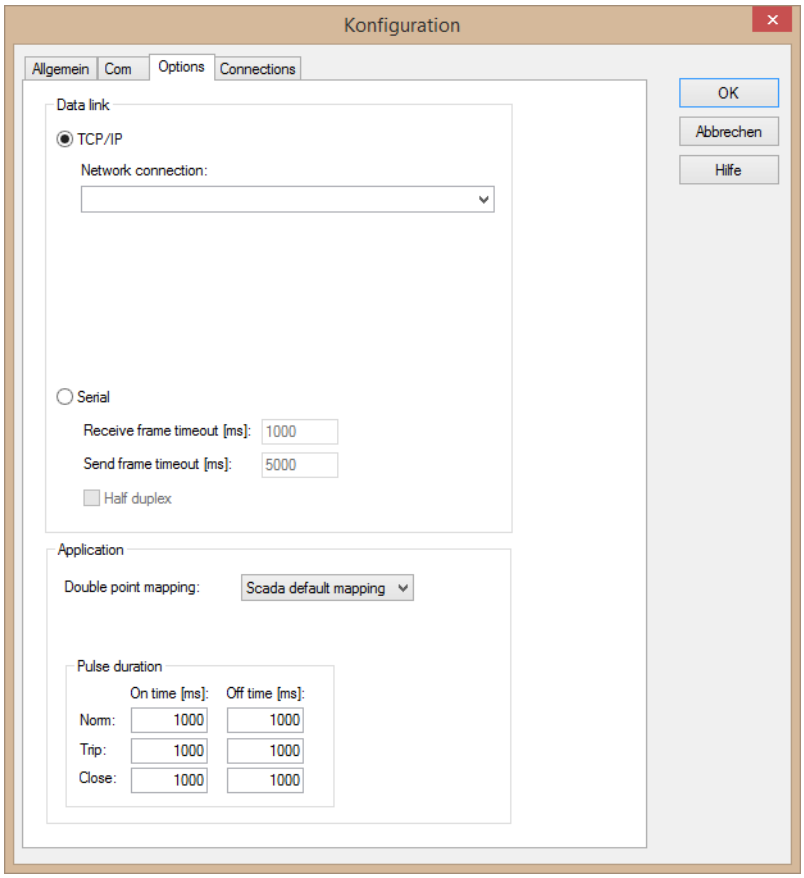
Info

Die exakten Einstellungen hängen von der verwendeten Steuerung ab. Entnehmen Sie die gültigen Werte dem Handbuch der Steuerung.

6.2.3 Options

In dieser Registerkarte konfigurieren Sie die Kommunikationsparameter.

Hinweis: Dieser Dialog steht nur in englischer Sprache zur Verfügung.



DATA LINK

Parameter	Beschreibung
TCP/IP	▸ *Aktiv*: Die Kommunikation erfolgt über TCP/IP.
Network connection	Auswahl der Netzwerkverbindung, die für die

Parameter	Beschreibung
	Kommunikation genutzt wird. Eingabe in Feld oder Auswahl aus Dropdownliste von: ▸ Name oder ▸ Netzwerkverbindung oder ▸ IP-Adresse. In der Dropdownliste werden die Namen der am lokalen Rechner verfügbaren Netzwerkverbindungen angezeigt. Für Remote Runtime Rechner geben Sie den Namen der Netzwerkkarte oder die IP-Adresse des Remote Rechners ein.
Serial	Aktiv: Kommunikation erfolgt über die serielle Schnittstelle. Default: *nicht aktiv*
Receive frame timeout [ms]	Timeout in Millisekunden für den Empfang eines kompletten Telegramms. Startzeit ist das erste Zeichen des Telegramms. Default: *1000* **Hinweis:** Nur aktiv, wenn Serial aktiviert ist.
Send frame timeout [ms]	Timeout in Millisekunden für das Senden eines Telegramms. Ist der serielle Ausgangspuffer länger als diese Zeitspanne gefüllt, wird das als Sendefehler betrachtet Default: *5000* **Hinweis:** Nur aktiv, wenn Serial aktiviert ist.
Half duplex	▸ *Aktiv*: Es wird erst wieder ein Telegramm versendet, wenn für das zuletzt auf der Schnittstelle gesendete Telegramm eine Antwort empfangen wurde. Default: *nicht aktiv* **Hinweis:** Nur aktiv, wenn Serial aktiviert ist.

APPLICATION

Parameter	Beschreibung
Double Point Mapping	Auswahl des Double Point Mappings auf einen Integerwert. Auswahl aus Dropdownliste: ▸ *SCADA default mapping* ▸ *DNP3 binary mapping* ▸ *Custom legacy mapping* Default: *SCADA default mapping*

PULSE DURATION

Einstellung der Pulsdauer für **Norm**, **Trip** und **Close** je Verbindung. **Norm**, **Trip** und **Close** legen fest, welches Relais geschaltet wird.

Parameter	Beschreibung
Norm	Entspricht **NUL**. Pulsdauer Norm: ▸ **On time:** Zeitraum, in dem der Output auf *EIN* ist, in Millisekunden Default: *1000* ▸ **Off time:** Zeitraum, in dem der Output auf *AUS* ist, in Millisekunden Default: *1000*
Trip	Pulsdauer Trip: ▸ **On time:** Zeitraum, in dem der Output auf *EIN* ist, in Millisekunden Default: *1000* ▸ **Off time:** Zeitraum, in dem der Output auf *AUS* ist, in Millisekunden Default: *1000*
Close	Pulsdauer Close: ▸ **On time:** Zeitraum, in dem der Output auf *EIN* ist, in

Parameter	Beschreibung
	Millisekunden Default: *1000* ▸ **Off time:** Zeitraum, in dem der Output auf *AUS* ist, in Millisekunden Default: *1000*

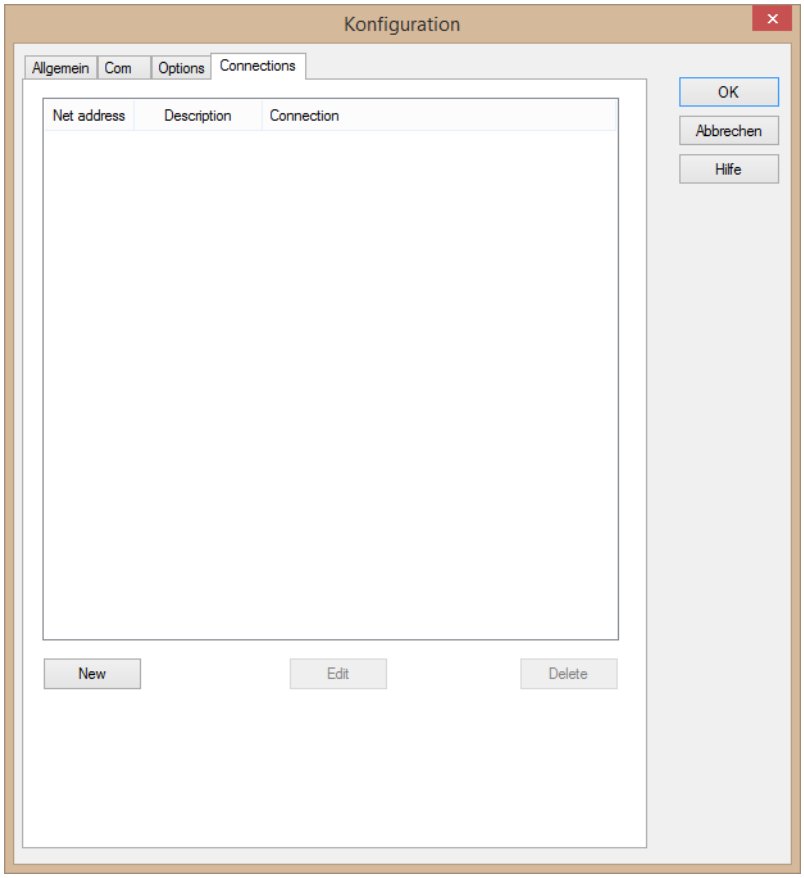
DIALOG BEENDEN

Option	Beschreibung
OK	Übernimmt alle Änderungen in allen Registerkarten und schließt den Dialog.
Abbrechen	Verwirft alle Änderungen in allen Registerkarten und schließt den Dialog.
Hilfe	Öffnet die Online-Hilfe.

6.2.4 Connections

In dieser Registerkarte konfigurieren Sie die Verbindungen.

Hinweis: Dieser Dialog steht nur in englischer Sprache zur Verfügung.



Parameter	Beschreibung
Liste Verbindungen	Zeigt die konfigurierten Verbindungen an: ▸ *Net address* Netzadresse der Outstation. Diese Adresse muss eindeutig sein und darf nur einmal vergeben sein. ▸ *Description* Beschreibung der Outstation. Entspricht dem **Friendly name**. ▸ *Connection* Verbindungs-Port. ▸ Bei TCP-Verbindung: TCP/IP-Adresse, TCP:Port ▸ Bei serieller Verbindung: Entspricht der in der Registerkarte **Com** konfigurierten Com-Einstellung.
New	Öffnet Dialog zum Anlegen einer neuen Verbindung (auf Seite 25).

Parameter	Beschreibung
Edit	Öffnet Dialog zum Bearbeiten der ausgewählten Verbindung.
Delete	Löscht ausgewählte Verbindung.

DIALOG BEENDEN

Option	Beschreibung
OK	Übernimmt alle Änderungen in allen Registerkarten und schließt den Dialog.
Abbrechen	Verwirft alle Änderungen in allen Registerkarten und schließt den Dialog.
Hilfe	Öffnet die Online-Hilfe.

NEUE VERBINDUNG ANLEGEN

1. Klicken Sie auf die Schaltfläche **New**.
2. Tragen Sie die Verbindungsdetails ein.
3. Klicken Sie auf **Save**.

VERBINDUNG BEARBEITEN

1. Wählen Sie in der Verbindungsliste die gewünschte Verbindung.
2. Klicken Sie auf die Schaltfläche **Edit**.
3. Ändern Sie die Verbindungsparameter.
4. Schließen Sie mit Klick auf die Schaltfläche **Save** ab.

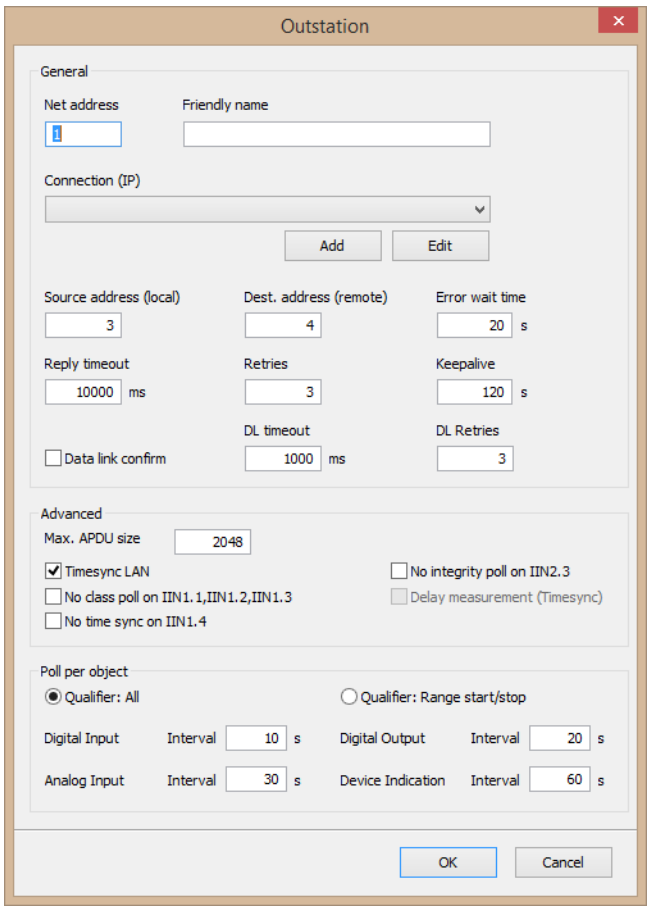
VERBINDUNG LÖSCHEN

1. Wählen Sie in der Verbindungsliste die gewünschte Verbindung.
2. Klicken Sie auf die Schaltfläche **Delete**.
3. Die Verbindung wird aus der Liste gelöscht.

6.2.4.1 Outstation

In diesem Dialog konfigurieren Sie Verbindungen. Der Dialog wird geöffnet, wenn Sie in der Registerkarte **Connections** auf die Schaltfläche **New** oder **Edit** klicken.

Hinweis: Dieser Dialog steht nur in englischer Sprache zur Verfügung.



GENERAL

Allgemeine Einstellungen.

Parameter	Beschreibung
Net address	Netzadresse der Verbindung. Wert zwischen *0* und *255*. Entspricht der Eigenschaft **Netzadresse** bei der Variable. Die Netzadresse muß eindeutig sein. Default: *zuletzt vergebene Netzadresse + 1*
Friendly name	Name der Verbindung. Frei wählbar.
Connection (IP)	Auswahl des TCP-Kanals, über den mit dieser Outstation kommuniziert wird. Im Fall von virtuellen Outstations ist es möglich, dass sich

Parameter	Beschreibung
	mehrere Stationen einen Kanal teilen. Auswahl aus Dropdownliste. Verwaltung der Verbindung mit den Schaltflächen **Add** und **Edit**. (Nicht verfügbar für serielle Kommunikation.)
Add	Fügt eine neue TCP Link-Konfiguration (auf Seite 31) hinzu. Der Dialog TCP-Links wird geöffnet.
Edit	Öffnet Dialog zum Editieren (auf Seite 33) aller vorhandenen TCP-Links.
Source address (local)	Eigene Adresse des Treibers (KDNP1 Master). Default: 3
Dest. address (remote)	Linkadresse der Steuerung (KDNP1 Slave). Default: 4
Error wait time	Wartezeit bei Kommunikationsfehler. Nach einem Fehler wird die angegebene Zeit gewartet, bevor erneut ein Verbindungsaufbau versucht wird. Default: 20 s
Reply timeout [ms]	Zeit in Millisekunden, die auf eine Antwort der Steuerung gewartet wird. Ist die Zeit abgelaufen wird eine Wiederholung gesendet. Wird ein Fragment einer mehrteiligen Antwort empfangen. Beginnt das Intervall mit jedem empfangenem Fragment von Neuem. Default: *10000 ms*
Retries	Anzahl der Kommunikationswiederholungen nach Timeouts. Default: 3
Keepalive	Werden über die angegebene Zeit keine Daten empfangen oder gesendet, wird ein LINK_STATUS_REQUEST Telegramm gesendet. Default: 120 s
Data link confirm	Ist die Eigenschaft **Data link confirm** aktiviert, werden in

Parameter	Beschreibung
	der Kommunikation Confirmed data link layer-Telegramme verwendet. Konfiguriert wird diese Überprüfung mit den Eigenschaften **DL timeout** und **DL Retries**. Default: *nicht aktiv* **Hinweis:** Diese Eigenschaft sollte nur dann aktiviert sein, wenn Confirmed data link layer auch auf der Outstation aktiviert ist.
DL timeout	Zeit in Millisekunden, die auf eine Data-Link Antwort der Steuerung gewartet wird. Ist die Zeit abgelaufen wird eine Wiederholung gesendet. Im Gegensatz zum **Reply timeout [ms]** das für ein Application Layer Telegram gilt, gilt das **DL timeout** für ein Data link layer-Telegramm. Ein Application Layer-Telegramm kann aus mehreren Data link layer-Telegrammen bestehen. Default: *1000 ms* Maximalzeit: *99999 ms* Nur aktiv, wenn **Data link confirm** aktiv ist.
DL Retries	Maximale Anzahl der Wiederholungen für eine Verbindungsüberprüfung nach einem Verbindungsfehler auf dem Data link layer. Default: *3* Nur aktiv, wenn **Data link confirm** aktiv ist.

ADVANCED

Parameter	Beschreibung
Max. APDU size	Maximale Größe eines KDNP Application Layer Fragments. ▸ Minimum: *249* ▸ Maximum: *65535* Default: *2048*
Timesync LAN	▸ *Aktiv*: Zeitsynchronisation über das LAN.

Parameter	Beschreibung
	Nur für TCP-Verbindung projektierbar. Default: *aktiviert* Ist als Verbindungsart Seriell (auf Seite 20) gewählt, ist die Eigenschaft im Treiberdialog automatisch aktiviert und ausgegraut. **Hinweis:** Verwendet die Variation 3 des Zeitobjekts. Diese wird nicht von allen Stationen akzeptiert.
No class poll on IIN1.1, IIN1.2, IIN1.3	Aktiv: Treiber ignoriert die Internal Indication Flags IIN1.1, IIN1.2 und IIN1.3. Kompatibilitätseinstellung: Stellen Sie sicher, dass regelmäßig für Events gepollt wird, um eventuelle Pufferüberläufe in der Outstation zu vermeiden.
No time sync on IIN1.4	*Aktiv*: Treiber ignoriert das Internal Indication Flag IIN1.4. Kompatibilitätseinstellung. Zeitsynchronisation nötig.
No integrity poll on IIN2.3	Aktiv: Treiber ignoriert das Internal Indication Flag IIN2.3 (Buffer Overflow). Kompatibilitätseinstellung.
Delay Measurement (Timesync)	Wenn aktiv, führt der Treiber bei der Zeitsynchronisierung ein Delay Measurement durch. Aktiv wenn: ▸ Verbindungstyp Serial ▸ Verbindungstyp IP (TCP/IP) **und Timesync LAN** nicht aktiv. **Hinweis:** Bei einer seriellen Verbindung ist zwar **Timesync_LAN** per default aktiv und ausgegraut, **Delay Measurement (Timesync)** aber wählbar.

POLL PER OBJECT

In diesem Bereich konfigureren Sie die Abfrageparameter zur Outstation.

Konfiguriert werden können:

- Art des Pollings (Abfrage)
- gepollte Objekte
- Zeitintervall des Pollings

Parameter	Beschreibung
Qualifier: All	Ist diese Eigenschaft aktiv, werden alle verfügbaren Datenpunkte für den jeweiligen Objekttyp der Outstation gelesen. Die Intervallzeit der Abfrage kann bei der jeweiligen Eigenschaft konfiguriert werden. Default: *aktiviert*
Qualifier: Range start/stop	Ist diese Eigenschaft aktiv, werden nur im Projekt vorhandene Variablen nach Objekttypen gepollt. Dies bedeutet: Der Treiber bestimmt den kleinsten und größten Offset und verwendet diesen für den Object poll. Default: *nicht aktiviert*
Binary Input	Polling Intervall für Variablen vom Objekttyp **Binary Input**. Default: *10 s* **Hinweis:** Wert *0* deaktiviert das Polling für diesen Objekttyp.
Binary Output	Polling Intervall für Variablen vom Objekttyp **Binary Output**. Default: *20 s* **Hinweis:** Wert *0* deaktiviert das Polling für diesen Objekttyp.
Analog Input	Polling Intervall für Variablen vom Objekttyp **Analog Input**. Default: *30 s* **Hinweis:** Wert *0* deaktiviert das Polling für diesen Objekttyp.
Device Indication	Polling Intervall für den Objekttyp **Device Indication**. Dieser Objekttyp gibt Auskunft über den Zustand der Module in der Outstation: ▸ *0 = ok* ▸ *1 = fault*

Parameter	Beschreibung
	Default: *60 s* **Hinweis:** Wert *0* deaktiviert das Polling für diesen Objekttyp.

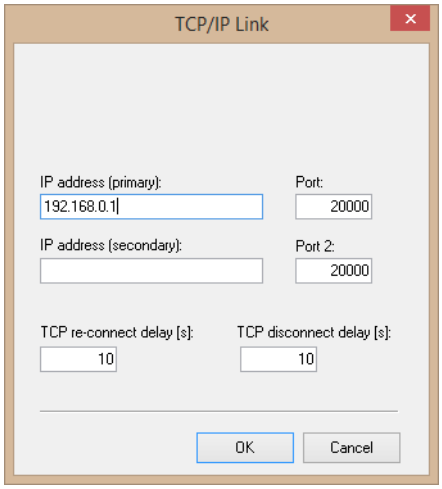
DIALOG BEENDEN

Option	Beschreibung
OK	Übernimmt Einstellungen und schließt den Dialog.
Abbrechen	Verwirft alle Änderungen und schließt den Dialog.

6.2.4.1.1 TCP/IP Link Konfiguration

Konfiguration von TCP Links.

Hinweis: Dieser Dialog steht nur in englischer Sprache zur Verfügung.



Parameter	Beschreibung
IP address (primary)	Eingabe der primären IP-Adresse.
Port	Primärer TCP-Port der Steuerung. Default: *20000*
IP address (secondary)	Backup-Adresse bei redundanter Anbindung dieser Steuerung.

Parameter	Beschreibung
Port 2	Port der Backup-Adresse. Default: *20000*
TCP re-connect delay [s]	Minimale Zeitspanne in Sekunden, die zwischen zwei TCP-Verbindungsversuchen liegen muss. Default: *10 s*
TCP disconnect delay [s]	Werden über die Verbindung über die angegebene Zeitspanne keine Daten gesendet oder empfangen, wird die TCP Verbindung automatisch getrennt. Default: *10 s* **Hinweis:** In diesem Fall müssen die Pollingintervalle mit einer entsprechend geringeren Zeitspanne konfiguriert werden. Ist die hier eingestellte Zeitspanne = 0 wird die TCP-Verbindung nicht getrennt.

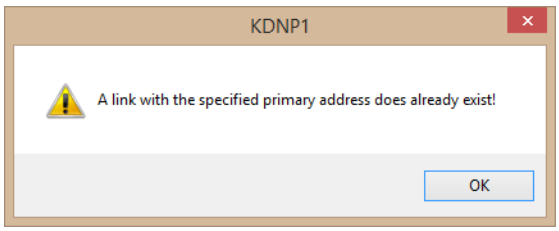
NAVIGATION

OK	Übernimmt Einstellungen und schließt den Dialog.
Cancel	Verwirft alle Änderungen und schließt den Dialog.

FEHLERDIALOG - ADRESSE BEREITS VORHANDEN

Es kann pro IP-Addresse nur eine Outstation mit einer Portnummer konfiguriert werden. Wenn zwei Outstations auf der gleiche IP Addresse mit unterschiedlichen Portnummern angesprochen werden müssen, verwenden Sie einen weiteren Treiber.

Tritt bei der Konfiguration einer Outstation ein IP-Adressenkonflikt auf, wird dieser mit einem Warndialog visualisiert.



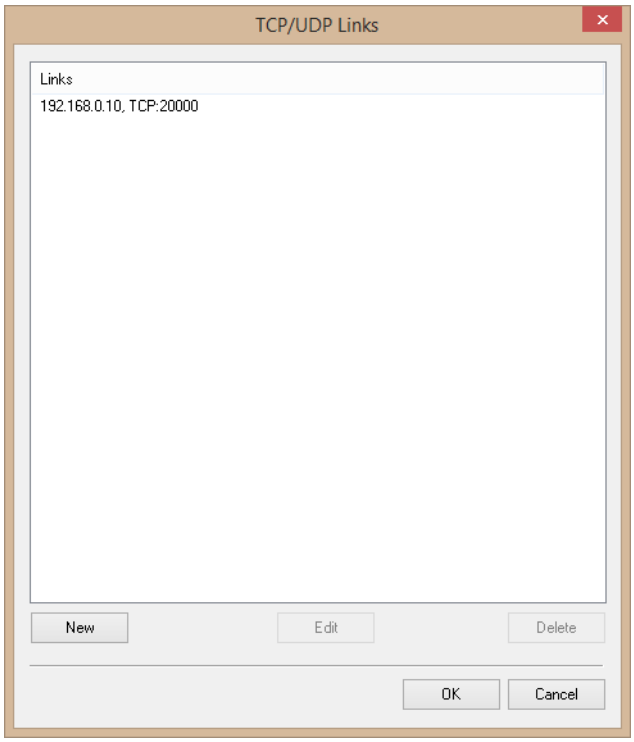
Klicken Sie die Schaltfläche **OK** um zum Konfigurationsdialog zurückzukehren.

Hinweis: Dieser Dialog steht nur in englischer Sprache zur Verfügung.

6.2.4.1.2TCP/UDP Links

Anzeige und Verwaltung aller TCP Links.

Hinweis: Dieser Dialog steht nur in englischer Sprache zur Verfügung.



Parameter	Beschreibung
Links	Zeigt die konfigurierten TCP Kanäle an. Angezeigt werden IP-Adresse sowie der TCP-Port: *[IP-Adresse], TCP:[Portnummer]*
New	Öffnet Dialog zum Anlegen eines neuen (auf Seite 31) TCP Kanals.
Edit	Öffnet Dialog zum Bearbeiten (auf Seite 31) des ausgewählten TCP Kanals.
Delete	Löscht ausgewählten TCP Link. Der TCP-Link wird ohne Rückfrage gelöscht.

NAVIGATION

Parameter	Beschreibung
OK	Übernimmt Einstellungen und schließt den Dialog.
Cancel	Verwirft alle Änderungen und schließt den Dialog.

NEUEN TCP KANAL ANLEGEN

1. Klicken Sie auf die Schaltfläche **New**.
   Der Dialog TCP/IP Link Konfiguration (auf Seite 31) wird geöffnet.
2. Tragen Sie die Verbindungsdetails ein.
3. Klicken Sie auf **OK.**
   Der Dialog wird geschlossen. Der konfigurierte TCP-Kanal wird in der Liste **Links** angezeigt.

TCP KANAL BEARBEITEN

1. Wählen Sie in der Liste den gewünschten Link.
2. Klicken Sie auf die Schaltfläche **Edit**.
   Der Dialog TCP/IP Link Konfiguration (auf Seite 31) wird geöffnet.
3. Ändern Sie die Linkparameter.
4. Schließen Sie mit **OK** ab.
   Der Dialog wird geschlossen. Der konfigurierte TCP-Kanal wird in der Liste **Links** angezeigt.

TCP KANAL LÖSCHEN

1. Wählen Sie in der Liste den gewünschten Link.
2. Klicken Sie auf die Schaltfläche **Delete**.
3. Die Verbindung wird ohne Rückfrage aus der Liste gelöscht.

7 Variablen anlegen

So werden Variablen im zenon Editor angelegt:

7.1 Variablen im Editor anlegen

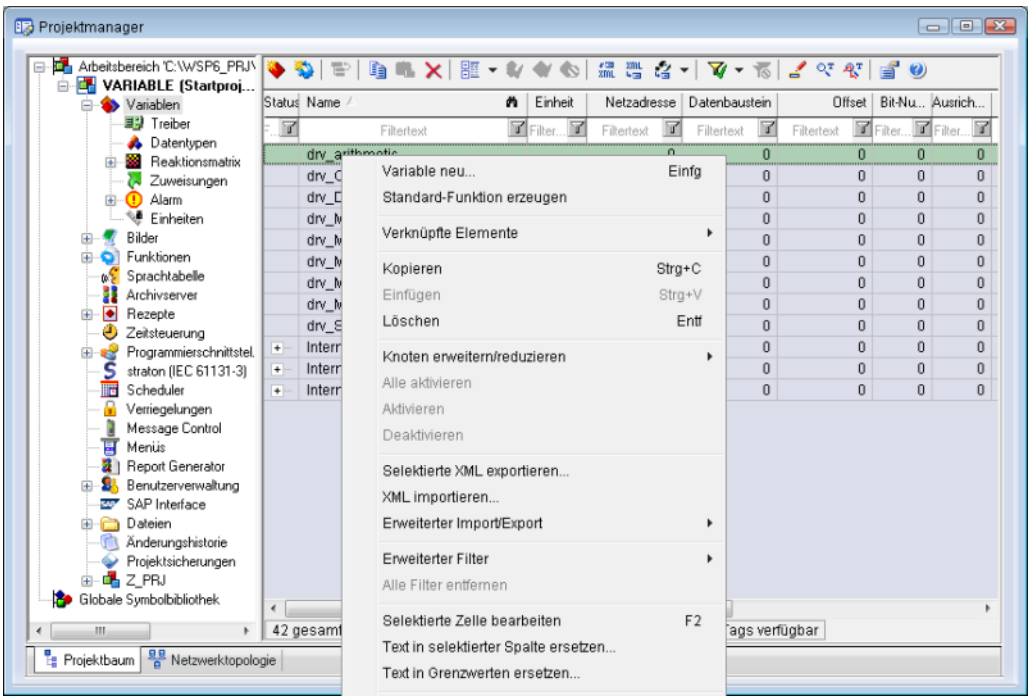
Variablen können angelegt werden:

- als einfache Variable
- in Arrays
- als Struktur-Variablen

DIALOG VARIABLE

Um eine neue Variable zu erstellen, gleich welchen Typs:

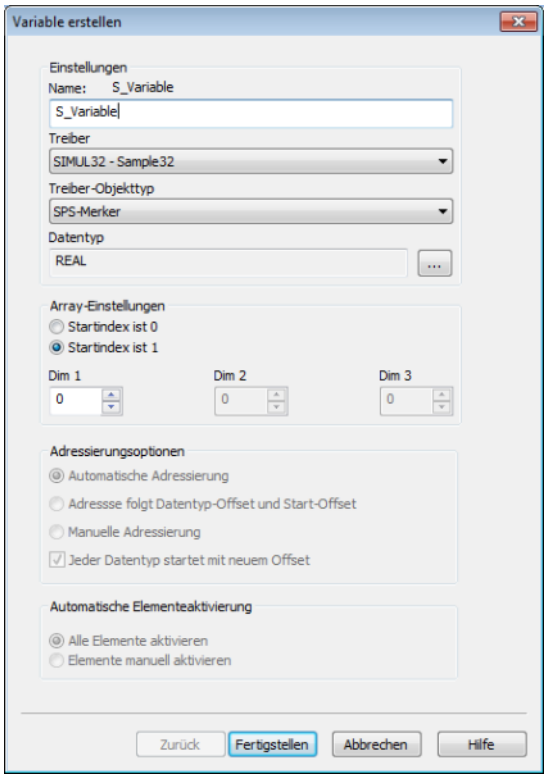
1. Wählen Sie im Knoten **Variablen** im Kontextmenü den Befehl **Variable neu**.



   Der Dialog zur Konfiguration der Variable wird geöffnet.

2. Konfigurieren Sie die Variable.
3. Welche Einstellungen möglich sind, hängt ab vom Typ der Variablen.

DIALOG VARIABLE ERSTELLEN



Eigenschaft	Beschreibung
Name	Eindeutiger Name der Variablen. Ist eine Variable mit gleichem Namen im Projekt bereits vorhanden, kann keine weitere Variable mit diesem Namen angelegt werden. Maximale Länge: 128 Zeichen **Achtung:** Die Zeichen **#** und **@** sind für Variablennamen nicht erlaubt. Bei Verwendung nicht zugelassener Zeichen kann die Variablenerstellung nicht abgeschlossen werden, die Schaltfläche **Fertigstellen** bleibt inaktiv. **Hinweis:** Manche Treiber erlauben die Adressierung auch über die Eigenschaft **Symbolische Adresse**.
Treiber	Wählen Sie aus der Dropdownliste den gewünschten Treiber. **Hinweis:** Sollte im Projekt noch kein Treiber angelegt sein, wird automatisch der Treiber für interne Variable (**Intern.exe**) geladen.
Treiberobjekttyp	Wählen Sie aus der Dropdownliste den passenden Treiberobjekttyp aus.
Datentyp	Wählen Sie den gewünschten Datentyp. Klick auf die Schaltfläche

Eigenschaft	Beschreibung
	... öffnet den Auswahl-Dialog.
Array-Einstellungen	Erweiterte Einstellungen für Array-Variablen. Details dazu lesen Sie im Abschnitt Arrays.
Adressierungsoptionen	Erweiterte Einstellungen für Arrays und Struktur-Variablen. Details dazu lesen Sie im jeweiligen Abschnitt.
Automatische Elementeaktivierung	Erweiterte Einstellungen für Arrays und Struktur-Variablen. Details dazu lesen Sie im jeweiligen Abschnitt.

SYMBOLISCHE ADRESSE

Die Eigenschaft **Symbolische Adresse** kann für die Adressierung alternativ zu **Name** oder **Kennung** der Variablen verwendet werden. Die Auswahl erfolgt im Treiberdialog, die Konfiguration in der Variableneigenschaft. Beim Import von Variablen unterstützter Treiber wird die Eigenschaft automatisch eingetragen.

Maximale Länge: *1024* Zeichen.

Folgende Treiber unterstützen die **Symbolische Adresse**:

- **3S_V3**
- **AzureDrv**
- **BACnetNG**
- **IEC850**
- **KabaDPServer**
- **OPCUA32**
- **Phoenix32**
- **POZYTON**
- **RemoteRT**
- **S7TIA**
- **SEL**
- **SnmpNg32**
- **PA_Drv**
- **EUROMAP63**

ABLEITUNG VOM DATENTYP

Messbereich, **Signalbereich** und **Sollwert Setzen** werden immer:

- vom Datentyp abgeleitet
- beim Ändern des Datentyps automatisch angepasst

Hinweis Signalbereich: Bei einem Wechsel auf einen Datentyp, der den eingestellten **Signalbereich** nicht unterstützt, wird der **Signalbereich** automatisch angepasst. Zum Beispiel wird bei einem Wechsel von **INT** auf **SINT** der **Signalbereich** auf *127* geändert. Die Anpassung erfolgt auch dann, wenn der **Signalbereich** nicht vom Datentyp abgeleitet wurde. In diesem Fall muss der **Messbereich** manuell angepasst werden.

7.2 Adressierung

Der Offset bestimmt die KDNP Point Nummer. In Kombination mit dem Treiberobjekttyp wird die KDNP-Objektgruppe bestimmt.

Gruppe/Eigenschaft	Beschreibung
Allgemein	Allgemeine Variableneigenschaften.
Name	Frei vergebbarer Name. **Achtung:** Je zenon Projekt muss der Name eindeutig sein.
Kennung	Frei vergebbare Kennung. Z. B. für **Betriebsmittelkennung**, Kommentar usw.
Adressierung	Eigenschaften der Variablenadressierung.
Netzadresse	Busadresse oder **Netzadresse** der Variablen. Diese Adresse bezieht sich auf die Stationsadresse der Verbindungsprojektierung im Treiber. Damit wird ausgewählt, auf welcher Steuerung sich die Variable befindet. **Achtung:** Die DNP Adresse der Outstation wird bei der Verbindung separat konfiguriert.
Datenbaustein	Wird für diesen Treiber nicht verwendet.
Offset	Für Variablen aller Treiberobjekttypen**:** KDNP Point in der Outstation. Immer in Kombination mit dem Treiberobjekttyp zu sehen.
Ausrichtung	Wird für diesen Treiber nicht verwendet.
Bitnummer	Wird für diesen Treiber nicht verwendet.
Stringlänge	Nur verfügbar bei String-Variablen. Maximale Anzahl von Zeichen, die die Variable aufnehmen kann.

Gruppe/Eigenschaft	Beschreibung
Treiber Anbindung	Treiberspezifische Eigenschaften.
Treiberobjekttyp	Objekttyp der Variablen. Wird abhängig vom verwendeten Treiber beim Erstellen der Variablen ausgewählt und kann hier geändert werden.
Datentyp	Datentyp der Variablen. Wird beim Erstellen der Variablen ausgewählt und kann hier geändert werden. **ACHTUNG:** Wenn der Datentyp nachträglich geändert wird, müssen alle anderen Eigenschaften der Variablen überprüft bzw. angepasst werden.

7.3 Treiberobjekte und Datentypen

Treiberobjekte sind in der Steuerung verfügbare Bereiche wie z. B. Merker, Datenbausteine usw. Hier lesen Sie, welche Treiberobjekte vom Treiber zur Verfügung gestellt werden und welche IEC-Datentypen dem jeweiligen Treiberobjekt zugeordnet werden können.

7.3.1 Treiberobjekte

Folgende Objekttypen stehen in diesem Treiber zur Verfügung:

Treiber-Objekttyp	Kanaltyp	Lesen	Schreiben	Unterstützte Datentypen	Kommentar
Analog input	68	X	--	*UDINT, DINT, REAL, LREAL, UINT, INT*	
Analog output	69	X	X	*UDINT, DINT, REAL, LREAL, UINT, INT*	
Binary Input	64	X	--	*BOOL*	
Binary Output	65	X	X	*BOOL, USINT*	
Command	11	X	X	*STRING*	
Device indication	79	X	--	*USINT*	
Kommunikationsde	35	X	X	*BOOL, SINT, USINT, INT,*	Variablen für die statische Analyse der

Treiber-Objekttyp	Kanaltyp	Lesen	Schreiben	Unterstützte Datentypen	Kommentar
tails				*UINT*, *DINT*, *UDINT*, *REAL*, *STRING*	Kommunikation. Werte werden nur zwischen Treiber und Runtime übertragen, nicht zur SPS! **Hinweis**: Die Adressierung und das Verhalten ist bei den meisten zenon Treibern gleich. Weitere Informationen dazu finden Sie im Kapitel Kommunikationsdetails (Treibervariablen) (auf Seite 53).

Legende:

X: wird unterstützt

--: wird nicht unterstützt

KANALTYP

Der Begriff "**Kanaltyp**" ist die interne numerische Bezeichnung des Treiberobjekttyps. Diese wird auch für den erweiterten DBF Import/Export der Variablen verwendet.

"**Kanaltyp**" wird für den erweiterten CSV Import/Export der Variablen in der Spalte "**HWObjectType**" verwendet.

7.3.2 Zuordnung der Datentypen

TREIBER OBJEKTTYPEN UND UNTERSTÜTZTE IEC DATENTYPEN FÜR VARIABLEN IN ZENON

Treiber Objekttypen	Kanaltyp	Unterstützte Datentypen (Datenart)	Lesen	Schreiben	Kommentar
Binary Input *Static object group 1* *Event object group 2*	*64*	*BOOL*	X	--	
Binary Output *Static object group 10* *Event object Group 11* *CROB object group 12*	*65*	*BOOL*	--	X	Wert 1 : LATCH_ON Wert 0 : LATCH_OFF
Binary Output *Static object group 10* *Event object group 11* *CROB object group 12*	*65*	*USINT*	--	X	Wert 1 : PULSE_ON Wert 2 : PULSE_OFF Wert 3 : LATCH_ON Wert 4 : LATCH_OFF Wert 65: CLOSE Wert 129: TRIP Wert 1 bei PULSE_ON, LATCH_ON und CLOSE, Wert 0 bei PULSE_OFF, LATCH_OFF und TRIP
Analog Input *Static object group 30* *Event object group 32*	*68*	*INT, DINT, REAL, LREAL*	X	--	

Treiber Objekttypen	Kanaltyp	Unterstützte Datentypen (Datenart)	Lesen	Schreiben	Kommentar
Analog Output *Static object group 40* *Event object group 42* *Analog Output Block object group 41*	*69*	*INT, DINT, REAL, LREAL*	--	X	geschriebener Wert wird nach erfolgreichem Schreiben als Rückmeldung gespiegelt
Command	11	STRING	X	X	Initiiert ein explizites Kommando z.B. explizites Lesen, ...
Device indication	79	USINT	x	--	Gibt Auskunft über den Zustand der Module der Outstation: ▸ *0 = ok* ▸ *1 = fault*

BINARY OUTPUTS

Das Schreiben von *Binary Outputs* erfolgt immer über ein **CROB** (Gruppe 12), wahlweise mit *Direct Operate* oder *Select Before Operate*, oder *Direct Operate No Ack*. Dabei wird die Einstellung bei der variablenspezifischen Eigenschaft **Command Mode** (über XML-Export/-Import und VBA zugänglich über **COMMAND_MODE**) verwendet. Direktes Schreiben auf Gruppe *10* wird vom Treiber nicht unterstützt.

Nach dem erfolgreichem Schreiben enthält die Variable den Wert *0* für *Binary Outputs* bei *PULSE_OFF*, *LATCH_OFF* und *TRIP*, und Wert *1* bei *PULSE_ON*, *LATCH_ON* und *CLOSE*. Der Wert wird ebenfalls aktualisiert wenn ein *Binary Output Status* (Gruppe 10) oder *Binary Output Status Event* (Gruppe 11) Objekt empfangen wird.

Befehlsgabe mit *Binary Output* Variablen:

- Es dürfen ausschließlich Befehle vom Typ **Doppelkommando** bzw. **Schaltbefehl** verwendet werden.
- Die Eigenschaft **Select Before Operate** bei der Variable muss deaktiviert sein; der KDNP1-Treiber verwendet bei Bedarf *Auto-SBO*.

Es wird empfohlen den **Qualifier of Command** bei der Befehlsgabe zu verwenden, um den Typ zu bestimmen:

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

Die Option **Qualifier of Command** kann sowohl bei *Binary Output* Variablen vom Datentyp *BOOL* als auch vom Datentyp *USINT* verwendet werden.

ANALOG OUTPUTS

Das Schreiben von *Analog Outputs* erfolgt immer über ein Objekt der Gruppe 41, wahlweise mit *Direct Operate*, *Select Before Operate* oder *Direct Operate No Ack*. Dabei wird die Einstellung bei der variablenspezifischen Eigenschaft **Command Mode** (über XML-Export/-Import und VBA zugänglich über **COMMAND_MODE**) verwendet.

Nach dem erfolgreichen Schreiben enthält die Variable zuerst den in der Runtime geschriebenen Wert. Der Wert wird ebenfalls aktualisiert wenn ein *Analog Output Status* (Gruppe 40) oder *Analog Output Status* (Group 42) Objekt empfangen wird.

Verwenden Sie die den Aktionstyp *Sollwert Setzen* für *Analog Outputs*. Entsprechend der Option **Command Mode** bei der Variable wird auch für *Analog Outputs* entweder ein *direct operate* oder ein *select before operate* durchgeführt. Die Option **Select Before Operate** bei der Variable muss deaktiviert sein. Die Option **Qualifier of Command** steht beim Aktionstyp *Sollwert Setzen* nicht zur Verfügung und hat keinen Einfluss bei *Analog Outputs*.

Bei zweistufiger Befehlsgabe wird erst bei der zweite Stufe der Befehl zur Outstation gesendet, inklusive **Select** bei *Auto-SBO*.

EXPLIZITES KOMMANDO (COMMAND)

Variablen vom Treiberobjekttyp *Command* werden verwendet, um einmalig eine bestimmte Aktion anzustoßen. Die Aktion wird durch das Schreiben eines Stringwerts auf die Kommandovariable initiiert. Das Resultat der Aktion wird der Variablen zugewiesen.

- Syntax:
  Ein Kommando besteht aus Kommandoname und Parametern. Kommandoname und die einzelnen Parameter werden durch Leerzeichen getrennt.
  Beispiel: READ 1 1 Lesen für Netzaddresse 1 alle Binary Inputs (Object Group 1, Variation 0 (default variation))

Achtung: Über das zusammensetzen von Parameter für ein Explizites Kommando kann es zu Kombinationen kommen die kein gültiges Kommando im Sinne des IEEE 1815 Standards entsprechen.

Kommandos werden ohne weitere Prüfung an die Outstation gesendet und können hier unter Umstände ein Fehlverhalten auslösen.

EXPLIZITES KOMMANDO (COMMAND)

Variablen vom Treiberobjekttyp *Command* werden verwendet, um einmalig eine bestimmte Aktion anzustoßen. Die Aktion wird durch das Schreiben eines Stringwerts auf die Kommandovariable initiiert. Das Resultat der Aktion wird der Variablen zugewiesen.

- Syntax: Ein Kommando besteht aus Kommandoname und Parametern. Kommandoname und die einzelnen Parameter werden durch Leerzeichen getrennt.
  Beispiel: READ_VAR 1 0 254 Lesen für Netzaddresse 1 alle Device Attributes (Object Group 0, Variation 254)
  Beispiel: READ 1 1 Lesen für Netzaddresse 1 alle Binary Inputs (Object Group 1, Variation 0 (default variation))

Achtung: Über das zusammensetzen von Parameter für ein Explizites Kommando kann es zu Kombinationen kommen die kein gültiges Kommando im Sinne des IEEE 1815 Standards entsprechen. Kommandos werden ohne weitere Prüfung an die Outstation gesendet und können hier unter Umstände ein Fehlverhalten auslösen.

EXPLIZITES LESEN VARIATION 0

Kommando: READ

Alle Points einer Gruppe lesen: (Qualifier Code 06 - Sie können diese Parameteranzahl für statische Object Groups und Event Object Groups verwenden)

- Parameter 1: **Netzadresse**
- Parameter 2: Gruppe

N-Points einer Gruppe lesen: (Qualifier Code 07, 08, 09 - Sie können diese Parameteranzahl für Event Object Groups verwenden. Die Nutzung einer Anzahl grösser als 65535 (Qualifier Code 09) wird nicht empfohlen))

- Parameter 1: **Netzadresse**
- Parameter 2: Gruppe
- Parameter 3: Anzahl der zu lesenden Points

Bestimmte Points einer Gruppe lesen: (Qualifier Code 00, 01 - Sie können diese Parameteranzahl für statische Object Groups, zB. Group 1 - Binary Input verwenden)

- Parameter 1: **Netzadresse**
- Parameter 2: Gruppe
- Parameter 3: Start-Index

- Parameter 4: Stop-Index

EXPLIZITES LESEN

Kommando: READ_VAR

Alle Points einer Gruppe lesen: (Qualifier Code 06 - Sie können diese Parameteranzahl für statische Object Groups und Event Object Groups verwenden)

- Parameter 1: **Netzadresse**
- Parameter 2: Gruppe
- Parameter 3: Variation

N-Points einer Gruppe lesen: (Qualifier Code 07, 08, 09 - Sie können diese Parameteranzahl für Event Object Groups verwenden. Die Nutzung einer Anzahl grösser als 65535 (Qualifier Code 09) wird nicht empfohlen))

- Parameter 1: **Netzadresse**
- Parameter 2: Gruppe
- Parameter 3: Variation
- Parameter 4: Anzahl der zu lesenden Points

Bestimmte Points einer Gruppe lesen: (Qualifier Code 00, 01 - Sie können diese Parameteranzahl für statische Object Groups, zB. Group 1 - Binary Input verwenden)

- Parameter 1: **Netzadresse**
- Parameter 2: Gruppe
- Parameter 3: Variation
- Parameter 4: Start-Index
- Parameter 5: Stop-Index

OBJEKT POLL AKTIVIEREN/DEAKTIVIEREN

Kommando: CYCLIC_POLL

- Parameter 1: **Netzadresse**
  Wurde eine ungültige Netzadresse angegeben, wird der Befehl abgelehnt.
- Parameter 2: Polling:
  - 0 - deaktiviert
  - 1 - aktiviert

KOMMANDO STATUSCODES

Statuscode	Beschreibung
16	Syntaxfehler.
17	Unbekanntes Kommando.
18	**Netzadresse** nicht vorhanden.
3	Kommando wurde versandt.
4	Kommando erfolgreich abgeschlossen.
5	Kommando konnte nicht verarbeitet werden.
6	Kommando wurde negativ beantwortet.
7	Fehler beim Ausführen des Kommandos (Timeout).
8	Ausführung des Kommandos wurde abgebrochen.

7.4 Variablen anlegen durch Import

Variablen können auch mittels Variablenimport angelegt werden. Für jeden Treiber stehen XML- und DBF-Import zur Verfügung.

Info

Details zu Import und Export von Variablen finden Sie im Handbuch Import-Export im Abschnitt Variablen.

7.4.1 XML Import

Beim XML- Import von Variablen oder Datentypen werden diese erst einem Treiber zugeordnet und dann analysiert. Vor dem Import entscheidet der Benutzer, ob und wie das jeweilige Element (Variable oder Datentyp) importiert werden soll:

- *Importieren*:
  Das Element wird neu importiert.
- *Überschreiben*:
  Das Element wird importiert und überschreibt ein bereits vorhandenes Element.
- *Nicht importieren*:
  Das Element wird nicht importiert.

Hinweis: Beim Import werden die Aktionen und deren Dauer in einem Fortschrittsbalken angezeigt. In der folgenden Dokumentation wird der Import von Variablen beschrieben. Datentypen werden analog dazu importiert.

VORAUSSETZUNGEN

Beim Import gelten folgende Bedingungen:

- **Abwärtskompatibilität**

  Beim XML Import/Export ist keine Abwärtskompatibilität gegeben. Daten aus älteren zenon Versionen können übernommen werden. Die Übergabe von Daten aus neueren Versionen an ältere wird nicht unterstützt.

- **Konsistenz**

  Die zu importierende XML-Datei muss konsistent sein. Beim Import der Datei erfolgt keine Plausibilitätsprüfung. Weisen die importierten Daten Fehler auf, kann es zu unerwünschten Effekten im Projekt kommen.

  Dies muss vor allem auch beachtet werden, wenn in einer XML-Datei nicht alle Eigenschaften vorhanden sind und diese dann durch Default-Werte ersetzt werden. Z. B.: Eine binäre Variable hat einen Grenzwert von *300*.

- **Struktur-Datentypen**

  Struktur-Datentypen müssen über die gleiche Anzahl von Strukturelementen verfügen. Beispiel: Ein Strukturdatentyp im Projekt hat 3 Strukturelemente. Ein gleichnamiger Datentyp in der XML-Datei hat 4 Strukturelemente. Dann wird keine der auf diesem Datentyp basierenden Variablen der Datei in das Projekt importiert.

Tipp

Weitere Informationen zum XML-Import finden Sie im Handbuch **Import - Export**, im Kapitel **XML-Import**.

7.4.2 DBF Import/Export

Daten können nach dBase exportiert und aus dBase importiert werden.

Info

Import und Export über CSV oder dBase unterstützt keine treiberspezifischen Variableneinstellungen wie z. B. Formeln. Nutzen Sie dafür den Export/Import über XML.

IMPORT DBF-DATEI

Um den Import zu starten:

1. Führen Sie einen Rechtsklick auf die Variablenliste aus.
2. Wählen Sie in der Dropdownliste von **Erweiterter Export/Import** ... den Befehl **dBase importieren**.
3. Folgen Sie den Anweisungen des Importassistenten.

Das Format der Datei ist im Kapitel Dateiaufbau beschrieben.

Info

Beachten Sie:

- Treiberobjekttyp und Datentyp müssen in der DBF-Datei an den Zieltreiber angepasst werden, damit Variablen importiert werden.
- dBase unterstützt beim Import keine Strukturen oder Arrays (komplexe Variablen).

EXPORT DBF-DATEI

Um den Export zu starten:

1. Führen Sie einen Rechtsklick auf die Variablenliste aus.
2. Wählen Sie im Dropdownliste von **Erweiterter Export/Import** ... den Befehl **dBase exportieren...** .
3. Folgen Sie den Anweisungen des Exportassistenten.

> **Achtung**
>
> DBF-Dateien:
>
> - müssen in der Benennung dem 8.3 DOS Format für Dateinamen entsprechen (8 alphanumerische Zeichen für Name, 3 Zeichen Erweiterung, keine Leerzeichen)
> - dürfen im Pfadnamen keinen Punkt (.) enthalten.
> Z. B. ist der Pfad *C:\users\Max.Mustermann\test.dbf* ungültig.
> Gültig wäre: *C:\users\MaxMustermann\test.dbf*
> - müssen nahe am Stammverzeichnis (Root) abgelegt werden, um die eventuelle Beschränkungen für Dateinamenlänge inklusive Pfad zu erfüllen: maximal 255 Zeichen

Das Format der Datei ist im Kapitel Dateiaufbau beschrieben.

> **Info**
>
> dBase unterstützt beim Export keine Strukturen oder Arrays (komplexe Variablen).

DATEIAUFBAU DER DBASE EXPORTDATEI

Für den Variablenimport und -export muss die dBaseIV-Datei folgende Struktur und Inhalte besitzen.

> **Achtung**
>
> dBase unterstützt keine Strukturen oder Arrays (komplexe Variablen).
>
> DBF-Dateien müssen:
>
> - in der Benennung dem 8.3 DOS Format für Dateinamen entsprechen (8 alphanumerische Zeichen für Name, 3 Zeichen Erweiterung, keine Leerzeichen)
> - nahe am Stammverzeichnis (Root) abgelegt werden

STRUKTUR

Bezeichnung	Typ	Feldgröße	Bemerkung
KANALNAME	Char	128	Variablenname. Länge kann über den Eintrag **MAX_LAENGE** in der **project.ini** eingeschränkt werden.

Bezeichnung	Typ	Feldgröße	Bemerkung
KANAL_R	C	128	Ursprünglicher Name einer Variablen, der durch den Eintrag unter VARIABLENNAME ersetzt werden soll (Feld/Spalte muss manuell angelegt werden). Länge kann über den Eintrag **MAX_LAENGE** in der **project.ini** eingeschränkt werden.
KANAL_D	Log	1	Variable wird bei Eintrag *1* gelöscht (Feld/Spalte muss manuell angelegt werden).
TAGNR	C	128	Kennung. Länge kann über den Eintrag **MAX_LAENGE** in der **project.ini** eingeschränkt werden.
EINHEIT	C	11	Technische Maßeinheit
DATENART	C	3	Datentyp (z. B. Bit, Byte, Wort, ...) entspricht dem Datentyp.
KANALTYP	C	3	Speicherbereich in der SPS (z. B. Merkerbereich, Datenbereich, ...) entspricht Treiberobjekttyp.
HWKANAL	Num	3	Netzadresse
BAUSTEIN	N	3	Datenbaustein-Adresse (nur bei Variablen aus den Datenbereich der SPS)
ADRESSE	N	5	Offset
BITADR	N	2	Für Bit-Variablen: Bitadresse Für Byte-Variablen: 0=niederwertig, 8=höherwertig Für String-Variablen: Stringlänge (max. 63 Zeichen)
ARRAYSIZE	N	16	Anzahl der Variablen im Array für Index-Variablen ACHTUNG: Nur die erste Variable steht voll zur Verfügung. Alle folgenden sind nur über VBA oder den Rezeptgruppen Manager zugänglich
LES_SCHR	L	1	Lese-Schreib-Berechtigung 0: Sollwert setzen ist nicht erlaubt 1: Sollwert setzen ist erlaubt
MIT_ZEIT	L	1	Zeitstempelung in zenon (nur wenn vom Treiber unterstützt)

Bezeichnung	Typ	Feldgröße	Bemerkung
OBJEKT	N	2	Treiberspezifische ID-Nummer des Primitivobjekts setzt sich zusammen aus TREIBER-OBJEKTTYP und DATENTYP
SIGMIN	Floa t	16	Rohwertsignal minimal (Signalauflösung)
SIGMAX	F	16	Rohwertsignal maximal (Signalauflösung)
ANZMIN	F	16	technischer Wert minimal (Messbereich)
ANZMAX	F	16	technischer Wert maximal (Messbereich)
ANZKOMMA	N	1	Anzahl der Nachkommastellen für die Darstellung der Werte (Messbereich)
UPDATERATE	F	19	Updaterate für Mathematikvariablen (in sec, eine Dezimalstelle möglich) bei allen anderen Variablen nicht verwendet
MEMTIEFE	N	7	Nur aus Kompatibilitätsgründen vorhanden
HDRATE	F	19	HD-Updaterate für hist. Werte (in sec, eine Dezimalstelle möglich)
HDTIEFE	N	7	HD-Eintragtiefe für hist. Werte (Anzahl)
NACHSORT	L	1	HD-Werte als nachsortierte Werte
DRRATE	F	19	Aktualisierung an die Ausgabe (für zenon DDE-Server, in sec, eine Kommastelle möglich)
HYST_PLUS	F	16	Positive Hysterese; ausgehend vom Messbereich
HYST_MINUS	F	16	Negative Hyterese; ausgehend vom Messbereich
PRIOR	N	16	Priorität der Variable
REAMATRIZE	C	32	Name der zugeordnete Reaktionsmatrix
ERSATZWERT	F	16	Ersatzwert; ausgehend vom Messbereich
SOLLMIN	F	16	Sollwertgrenze Minimum; ausgehend vom Messbereich
SOLLMAX	F	16	Sollwertgrenze Maximum; ausgehend vom Messbereich
VOMSTANDBY	L	1	Variable vom Standby Server anfordern; der Wert der Variable wird im redundanten Netzwerkbetrieb nicht vom Server sondern vom Standby Server angefordert

Bezeichnung	Typ	Feldgröße	Bemerkung
RESOURCE	C	128	Betriebsmittelkennung. Freier String für Export und Anzeige in Listen. Länge kann über den Eintrag MAX_LAENGE in der **project.ini** eingeschränkt werden.
ADJWVBA	L	1	Nichtlineare Wertanpassung: *0*: Nichtlineare Wertanpassung wird verwendet *1*: Nichtlineare Wertanpassung wird nicht verwendet
ADJZENON	C	128	Verknüpftes VBA-Makro zum Lesen der Variablenwerte für die nichtlineare Wertanpassung.
ADJWVBA	C	128	Verknüpftes VBA-Makro zum Schreiben der Variablenwerte für die nichtlineare Wertanpassung.
ZWREMA	N	16	Verknüpfte Zählwert-Rema.
MAXGRAD	N	16	Maximaler Gradient für die Zählwert-Rema.

Achtung

Beim Import müssen Treiberobjekttyp und Datentyp in der DBF-Datei an den Zieltreiber angepasst werden, damit Variablen importiert werden.

GRENZWERTDEFINITION

Grenzwertdefinition für Grenzwert *1* bis *4*, oder Zustand *1* bis *4*:

Bezeichnung	Typ	Feldgröße	Bemerkung
AKTIV1	L	1	Grenzwert aktiv (pro Grenzwert vorhanden)
GRENZWERT1	F	20	technischer Wert oder ID-Nummer der verknüpften Variable für einen dynamischen Grenzwert (siehe VARIABLEx) (wenn unter VARIABLEx *1* steht und hier *-1*, wird die bestehende Variablenzuordnung nicht überschrieben)
SCHWWERT1	F	16	Schwellwert für den Grenzwert
HYSTERESE1	F	14	wird nicht verwendet
BLINKEN1	L	1	Blinkattribut setzen

Bezeichnung	Typ	Feldgröße	Bemerkung
BTB1	L	1	Protokollierung in CEL
ALARM1	L	1	Alarm
DRUCKEN1	L	1	Druckerausgabe (bei CEL oder Alarm)
QUITTIER1	L	1	quittierpflichtig
LOESCHE1	L	1	löschpflichtig
VARIABLE1	L	1	dyn. Grenzwertverknüpfung der Grenzwert wird nicht durch einen absoluten Wert (siehe Feld GRENZWERTx) festgelegt.
FUNC1	L	1	Funktionsverknüpfung
ASK_FUNC1	L	1	Ausführung über die Alarmmeldeliste
FUNC_NR1	N	10	ID-Nummer der verknüpften Funktion (steht hier -1, so wird die bestehende Funktion beim Import nicht überschrieben)
A_GRUPPE1	N	10	Alarm/Ereignis-Gruppe
A_KLASSE1	N	10	Alarm/Ereignis-Klasse
MIN_MAX1	C	3	Minimum, Maximum
FARBE1	N	10	Farbe als Windowskodierung
GRENZTXT1	C	66	Grenzwerttext
A_DELAY1	N	10	Zeitverzögerung
INVISIBLE1	L	1	Unsichtbar

Bezeichnungen in der Spalte Bemerkung beziehen sich auf die in den Dialogboxen zur Definition von Variablen verwendeten Begriffe. Bei Unklarheiten, siehe Kapitel Variablendefinition.

7.5 Kommunikationsdetails (Treibervariablen)

Das Treiberkit implementiert eine Reihe von Treibervariablen, welche in dem Treiberobjekttyp *Kommunikationsdetails* zusammengefasst sind. Diese sind unterteilt in:

- Information
- Konfiguration
- Statistik und

- Fehlermeldungen

Die Definitionen der im Treiberkit implementierten Variablen sind in der Importdatei **DRVVAR.DBF** verfügbar und können von dort importiert werden.
Pfad zur Datei: *%ProgramData%\COPA-DATA\zenon<Versionsnummer>\PredefinedVariables*

Hinweis: Variablennamen müssen in zenon einzigartig sein. Soll nach einem Import der Variablen vom Treiberobjekttyp *Kommunikationsdetails* aus **DRVVAR.DBF** ein erneuter Import durchgeführt werden, müssen die zuvor importierten Variablen umbenannt werden.

Info

Nicht jeder Treiber unterstützt alle Treibervariablen des Treiberobjekttyps *Kommunikationsdetails*.

Zum Beispiel:

- Variablen für Modem-Informationen werden nur von modemfähigen Treibern unterstützt.
- Treibervariablen für den Polling-Zyklus stehen nur für rein pollende Treiber zur Verfügung.
- Verbindungsbezogene Informationen wie **ErrorMSG** werden nur von Treibern unterstützt, die zu einem Zeitpunkt nur eine Verbindung bearbeiten.

INFORMATION

Name aus Import	Typ	Offset	Erklärung
MainVersion	*UINT*	0	Haupt-Versionsnummer des Treibers.
SubVersion	*UINT*	1	Sub-Versionsnummer des Treibers.
BuildVersion	*UINT*	29	Build-Versionsnummer des Treibers.
RTMajor	*UINT*	49	zenon Hauptversionsnummer
RTMinor	*UINT*	50	zenon Sub-Versionsnummer
RTSp	*UINT*	51	zenon Service Pack-Nummer
RTBuild	*UINT*	52	zenon Buildnummer
LineStateIdle	*BOOL*	24.0	TRUE, wenn die Modemleitung belegt ist.
LineStateOffering	*BOOL*	24.1	TRUE, wenn ein Anruf rein kommt.
LineStateAccepted	*BOOL*	24.2	Der Anruf wird angenommen.

Name aus Import	Typ	Offset	Erklärung
LineStateDialtone	*BOOL*	24.3	Rufton wurde erkannt.
LineStateDialing	*BOOL*	24.4	Wahl aktiv.
LineStateRingBack	*BOOL*	24.5	Während Verbindungsaufbau.
LineStateBusy	*BOOL*	24.6	Zielstation besetzt.
LineStateSpecialInfo	*BOOL*	24.7	Spezielle Statusinformation empfangen.
LineStateConnected	*BOOL*	24.8	Verbindung hergestellt.
LineStateProceeding	*BOOL*	24.9	Wahl ausgeführt.
LineStateOnHold	*BOOL*	24.10	Verbindung in Halten.
LineStateConferenced	*BOOL*	24.11	Verbindung im Konferenzmodus.
LineStateOnHoldPendConf	*BOOL*	24.12	Verbindung in Halten für Konferenz.
LineStateOnHoldPendTransfer	*BOOL*	24.13	Verbindung in Halten für Transfer.
LineStateDisconnected	*BOOL*	24.14	Verbindung beendet.
LineStateUnknow	*BOOL*	24.15	Verbindungszustand nicht bekannt.
ModemStatus	*UDINT*	24	Aktueller Modemstatus.
TreiberStop	*BOOL*	28	Treiber gestoppt Bei *Treiberstop*, hat die Variable den Wert *TRUE* und ein **OFF**-Bit. Nach dem Treiberstart, hat die Variable den Wert *FALSE* und kein **OFF**-Bit.
SimulRTState	*UDINT*	60	Informiert über Status der Runtime bei Treibersimulation.
ConnectionStates	*STRING*	61	Interner Verbindungsstatus des Treibers zur SPS. Verbindungszustände: ▸ *0:* Verbindung OK ▸ *1:* Verbindung gestört ▸ *2:* Verbindung simuliert Formatierung:

Name aus Import	Typ	Offset	Erklärung
			<Netzadresse>:<Verbindungszustand>;...;...; Eine Verbindung ist erst nach dem ersten Anmelden einer Variablen bekannt. Damit eine Verbindung im String enthalten ist, muss einmal eine Variable dieser Verbindung angemeldet worden sein. Der Zustand einer Verbindung wird nur aktualisiert, wenn eine Variable der Verbindung angemeldet ist. Ansonsten wird nicht mit der entsprechenden Steuerung kommuniziert.

KONFIGURATION

Name aus Import	Typ	Offset	Erklärung
ReconnectInRead	*BOOL*	27	Wenn TRUE, dann wird beim Lesen automatisch ein Neuaufbau der Verbindung durchgeführt.
ApplyCom	*BOOL*	36	Änderungen an den Einstellungen der seriellen Schnittstelle zuweisen. Das Schreiben auf diese Variable hat unmittelbar den Aufruf der Methode SrvDrvVarApplyCom zur Folge (aktuell ohne weitere Funktion).
ApplyModem	*BOOL*	37	Änderungen an den Modemeinstellungen zuweisen. Das Schreiben auf diese Variable hat unmittelbar den Aufruf der Methode SrvDrvVarApplyModem zur Folge. Diese schließt die aktuelle Verbindung und öffnet eine neue entsprechend den Einstellungen **PhoneNumberSet** und **ModemHwAdrSet**.
PhoneNumberSet	*STRING*	38	Telefonnummer, welche verwendet werden soll.
ModemHwAdrSet	*DINT*	39	Hardwareadresse, welche zu der Telefonnummer gehört.
GlobalUpdate	*UDINT*	3	Updatezeit in Millisekunden (ms).

Name aus Import	Typ	Offset	Erklärung
BGlobalUpdaten	*BOOL*	4	TRUE, wenn die Updatezeit global ist.
TreiberSimul	*BOOL*	5	TRUE, wenn der Treiber in Simulation ist.
TreiberProzab	*BOOL*	6	TRUE, wenn das Prozessabbild gehalten werden soll.
ModemActive	*BOOL*	7	TRUE, wenn das Modem bei diesem Treiber aktiv ist.
Device	*STRING*	8	Name der seriellen Schnittstelle oder Name des Modem.
ComPort	*UINT*	9	Nummer der seriellen Schnittstelle.
Baudrate	*UDINT*	10	Baudrate der seriellen Schnittstelle.
Parity	*SINT*	11	Parität der seriellen Schnittstelle.
ByteSize	*USINT*	14	Bitanzahl pro Zeichen der seriellen Schnittstelle. Wert = *0*, wenn der Treiber keine serielle Kommunikation herstellen kann.
StopBit	*USINT*	13	Anzahl der Stoppbits der seriellen Schnittstelle.
Autoconnect	*BOOL*	16	TRUE, wenn die Modemverbindung automatisch beim Lesen/Schreiben aufgebaut werden soll.
PhoneNumber	*STRING*	17	Aktuelle Telefonnummer.
ModemHwAdr	*DINT*	21	Hardwareadresse zur aktuellen Telefonnummer.
RxIdleTime	*UINT*	18	Wenn länger als diese Zeit in Sekunden (s) erfolgreich kein Datenverkehr stattfindet, wird die Modemverbindung beendet.
WriteTimeout	*UDINT*	19	Maximale Schreibdauer bei einer Modemverbindung in Millisekunden (ms).
RingCountSet	*UDINT*	20	So oft läutet ein hereinkommender Anruf, bevor dieser angenommen wird.
ReCallIdleTime	*UINT*	53	Wartezeit zwischen Anrufen in Sekunden (s).

Name aus Import	Typ	Offset	Erklärung
ConnectTimeout	*UINT*	54	Zeit in Sekunden (s) für Verbindungsaufbau.

STATISTIK

Name aus Import	Typ	Offset	Erklärung
MaxWriteTime	*UDINT*	31	Längste Zeit in Millisekunden (ms), die zum Schreiben benötigt wird.
MinWriteTime	*UDINT*	32	Kürzeste Zeit in Millisekunden (ms), die zum Schreiben benötigt wird.
MaxBlkReadTime	*UDINT*	40	Längste Zeit in Millisekunden (ms), die zum Lesen eines Datenblocks benötigt wird.
MinBlkReadTime	*UDINT*	41	Kürzeste Zeit in Millisekunden (ms), die zum Lesen eines Datenblocks benötigt wird.
WriteErrorCount	*UDINT*	33	Anzahl der Schreibfehler.
ReadSucceedCount	*UDINT*	35	Anzahl der erfolgreichen Leseversuche.
MaxCycleTime	*UDINT*	22	Längste Zeit in Millisekunden (ms), die zum Lesen aller angeforderten Daten benötigt wurde.
MinCycleTime	*UDINT*	23	Kürzeste Zeit in Millisekunden (ms), die zum Lesen aller angeforderten Daten benötigt wurde.
WriteCount	*UDINT*	26	Anzahl der Schreibversuche.
ReadErrorCount	*UDINT*	34	Anzahl der fehlerhaften Leseversuche.
MaxUpdateTimeNormal	*UDINT*	56	Zeit seit letzter Aktualisierung der Prioritätsgruppe **Normal** in Millisekunden (ms).
MaxUpdateTimeHigher	*UDINT*	57	Zeit seit letzter Aktualisierung der Prioritätsgruppe **Höher** in Millisekunden (ms).
MaxUpdateTimeHigh	*UDINT*	58	Zeit seit letzter Aktualisierung der Prioritätsgruppe **Hoch** in Millisekunden (ms).
MaxUpdateTimeHighest	*UDINT*	59	Zeit seit letzter Aktualisierung der Prioritätsgruppe **Höchste** in Millisekunden (ms).
PokeFinish	*BOOL*	55	Geht für eine Abfrage auf *1*, wenn alle anstehenden Pokes ausgeführt wurden.

FEHLERMELDUNGEN

Name aus Import	Typ	Offset	Erklärung
ErrorTimeDW	*UDINT*	2	Zeit (in Sekunden seit 1.1.1970), wann der letzte Fehler auftrat.
ErrorTimeS	*STRING*	2	Zeit (in Sekunden seit 1.1.1970), wann der letzte Fehler als String auftrat.
RdErrPrimObj	*UDINT*	42	Nummer des PrimObjektes, als der letzte Lesefehler verursacht wurde.
RdErrStationsName	*STRING*	43	Name der Station, als der letzte Lesefehler verursacht wurde.
RdErrBlockCount	*UINT*	44	Anzahl der zu lesenden Blöcke, als der letzte Lesefehler verursacht wurde.
RdErrHwAdresse	*DINT*	45	Hardwareadresse, als der letzte Lesefehler verursacht wurde.
RdErrDatablockNo	*UDINT*	46	Bausteinnummer, als der letzte Lesefehler verursacht wurde.
RdErrMarkerNo	*UDINT*	47	Merkernummer, als der letzte Lesefehler verursacht wurde.
RdErrSize	*UDINT*	48	Blockgröße, als der letzte Lesefehler verursacht wurde.
DrvError	*USINT*	25	Fehlermeldung als Nummer.
DrvErrorMsg	*STRING*	30	Fehlermeldung als Klartext.
ErrorFile	*STRING*	15	Name der Fehlerprotokolldatei.

8 Treiberspezifische Funktionen

Dieser Treiber unterstützt folgende Funktionen:

Funktion	Beschreibung
Blockwrite	Nicht unterstützt.
Redundanz	Wenn die Outstation mehrere Master unterstützt.

Funktion	Beschreibung
RDA	Nicht unterstützt.
Echtzeitstempelung	Bei entsprechend konfigurierter **Variation** in der Outstation.
Browsen	Nicht unterstützt.
	Polling für Objekte, in konfigurierbaren Zyklen oder manuell gesteuert.
Spontan	Nicht unterstützt.
Anzahl der Steuerungen	Es können mit einem Treiber beliebig viele Outstations angesprochen werden. Bei serielle Kommunikation können mehrere Outstations für eine serielle Schnittstelle (Bus) konfiguriert werden. Mit einem Treiber ist ebenfalls ein Mischbetrieb *seriell/TCP* möglich.

FEHLERDATEI

Der Treiber unterstützt das zentrale Logging am Diagnose Server (auf Seite 67).

TREIBERVARIABLEN

Kommunikationsüberwachung ist über das **INVALID** Statusbit möglich.

KDNP1 OBJECT FLAGS MAPPING

Die DNP Object Flags **ONLINE** und **COMM_LOST** werden vom KDNP1-Treiber ausgewertet. Bei **ONLINE** = *false* oder **COMM_LOST** = *true*, ist in der Runtime das *INVALID* Bit bei der Variable gesetzt.

Die konfigurierte Variation in der Outstation bestimmt, ob für eine Objektgruppe Objekte mit oder ohne Flags gesendet werden.

Außerdem werden die nachfolgend angeführten Flags/IIN Bits auf die Variablen Statusbits übertragen:

KDNP1 Flag / IIN	Statusbit
Restart	*NT* Bit
Remote forced	*SB* Bit
Local forced	*SB* Bit
Rollover	*OV* Bit

KDNP1 Flag / IIN	Statusbit
IIN.1.4 Need time	*Time-inval* Bit

ZEITSYNCHRONISIERUNG

Der Treiber unterstützt die Zeitsynchronisierung von Outstations mit der Zeit der Master Station. Meldet die Outstation den Bedarf nach einer Zeitsynchronisierung mittels **Internal Indication** Flag 1.4 an, sendet der Treiber die aktuelle Systemzeit entsprechend den Optionen für UTC/lokale Zeit und LAN Zeitsynchronisierung.

Die Outstation unter Umstände die Zeit von einer andere Quelle bezieht (e.g GPS Emfänger), und dadurch eventuell keinen Bedarf zur Zeitsynchronisierung an den Master sendet. Stellen Sie in diesem Fall sicher , dass der Runtime Rechner mit der KDNP1 Master Station mit der gleiche Zeitquelle synchronisiert wird.

HYSTERESE

Der Treiber unterstützt die Hysterese für Spontanwerte (unsolicited responses). Die Hysterese wird nicht berücksichtigt von:

- Werten, die als Antwort auf eine Leseanforderung für eine Eventklasse empfangen werden
- Variablen, die als klassenlose Variablen explizit gelesen werden

Wird ein identer Wert, aber mit neuerem Zeitstempel, empfangen, dann wird dieser Wert vom Treiber als neu zur Runtime geschickt.

9 Funktion Treiberkommandos

Die zenon Funktion **Treiberkommandos** dient dazu, Treiber über zenon zu beeinflussen.
Mit einem Treiberkommando können Sie einen Treiber:

- starten
- stoppen
- in einen bestimmten Treibermodus versetzen
- zu bestimmten Aktionen veranlassen

Hinweis: Dieses Kapitel beschreibt Standardfunktionalitäten, die für die meisten zenon Treiber gültig sind.
Nicht alle hier beschriebenen Funktionalitäten stehen für jeden Treiber zur Verfügung. Zum Beispiel enthält ein Treiber, der laut Datenblatt keine Modemverbindung unterstützt, auch keine Modem-Funktionalitäten.

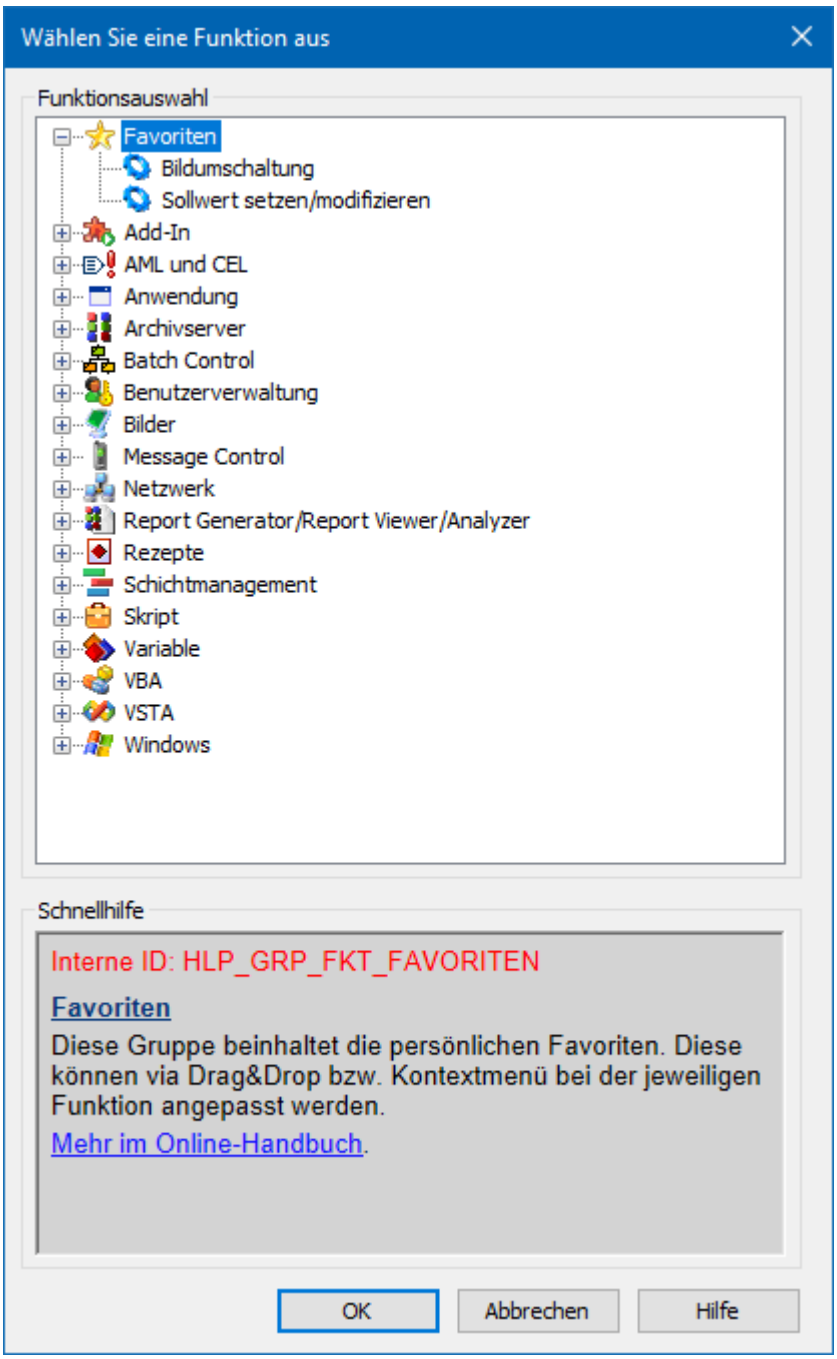
> **Achtung**
>
> Die zenon Funktion **Treiberkommandos** ist nicht ident mit den Treiberkommandos, die bei Energy-Treibern in der Runtime ausgeführt werden können!

PROJEKTIERUNG DER FUNKTION

Die Projektierung erfolgt über die Funktion **Treiberkommandos**.
Um die Funktion zu projektieren:

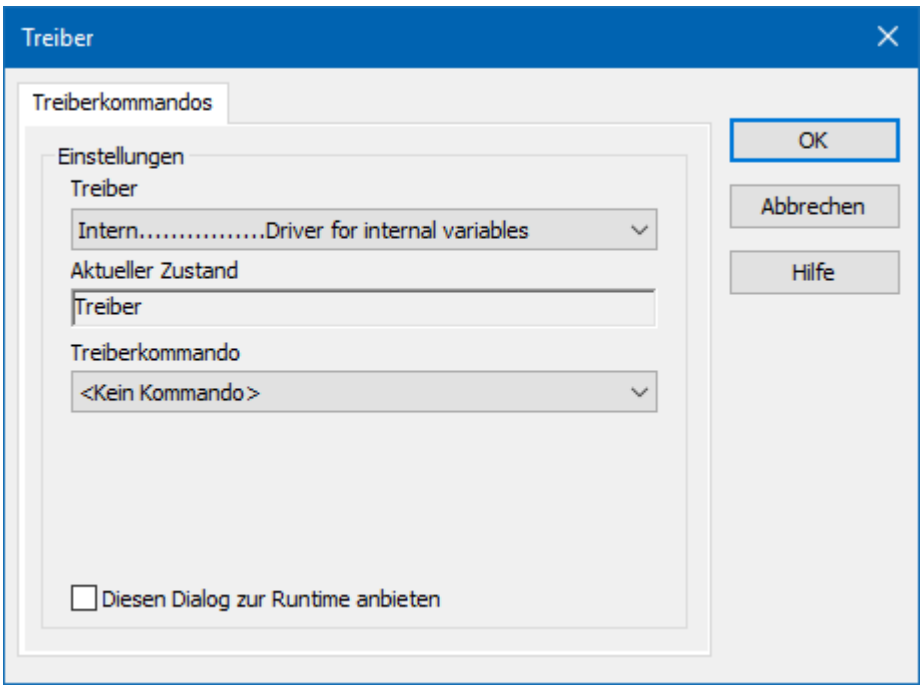
1. Legen Sie im zenon Editor eine neue Funktion an.

Der Dialog zur Auswahl einer Funktion wird geöffnet.



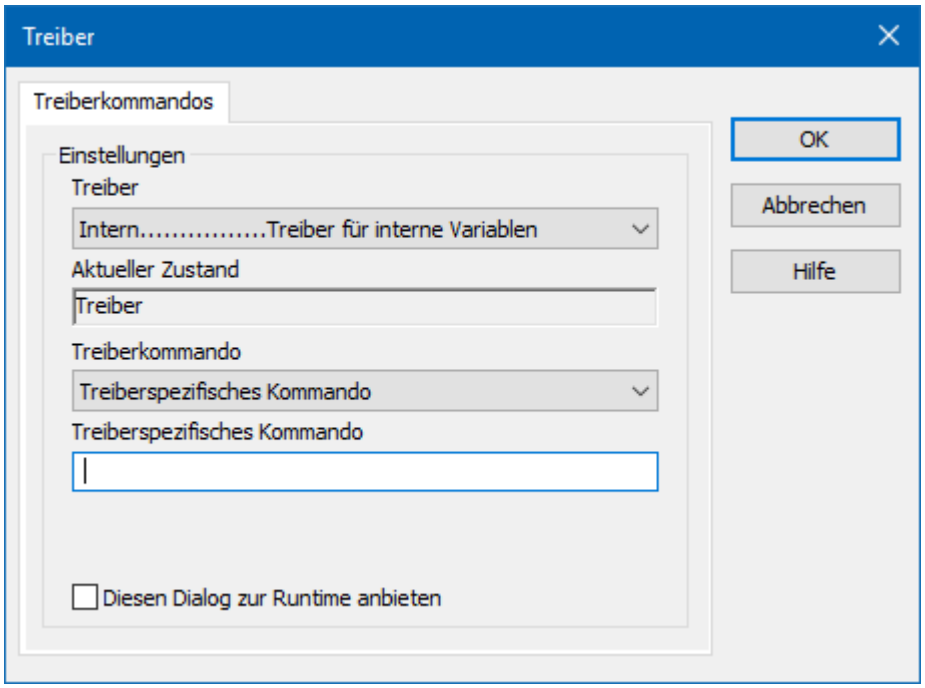
2. Navigieren Sie zum Knoten **Variable.**
3. Wählen Sie den Eintrag **Treiberkommandos**.

Der Dialog zur Konfiguration wird geöffnet.



4. Wählen Sie den gewünschten Treiber und das benötigte Kommando aus.
5. Schließen Sie den Dialog mit Klick auf **OK** und stellen Sie sicher, dass die Funktion in der Runtime ausgeführt wird.
   Beachten Sie die Hinweise im Abschnitt **Funktion Treiberkommandos im Netzwerk**.

DIALOG TREIBERKOMMANDOS



Option	Beschreibung
Treiber	Auswahl des Treibers aus der Dropdownliste. Diese enthält alle im Projekt geladenen Treibern.
Aktueller Zustand	Fixer Eintrag, der vom System gesetzt wird. In aktuellen Versionen ohne Funktion.
Treiberkommando	Auswahl des gewünschten Treiberkommandos aus Dropdownliste. Details zu den konfigurierbaren Treiberkommandos siehe Abschnitt **Verfügbare Treiberkommandos**.
Treiberspezifisches Kommando	Eingabe eines für den gewählten Treiber spezifischen Kommandos. **Hinweis:** Nur verfügbar, wenn für die Option **Treiberkommando** der Eintrag *Treiberspezifisches Kommando* gewählt wurde.
Diesen Dialog zur Runtime anbieten	Konfiguration, ob die Konfiguration in der Runtime geändert werden kann: ▸ *Aktiv*: Dieser Dialog wird in der Runtime vor dem Ausführen der Funktion geöffnet. Die Konfiguration kann damit in der Runtime vor der Ausführung noch geändert werden. ▸ *Inaktiv*: Die Editor-Konfiguration wird in der Runtime beim Ausführen der Funktion angewendet. Default: *inaktiv*

DIALOG BEENDEN

Option	Beschreibung
OK	Übernimmt Einstellungen und schließt den Dialog.
Abbrechen	Verwirft alle Änderungen und schließt den Dialog.
Hilfe	Öffnet die Online-Hilfe.

VERFÜGBARE TREIBERKOMMANDOS

Diese Treiberkommandos stehen - abhängig vom gewählten Treiber - zur Verfügung:

Treiberkommando	Beschreibung
Kein Kommando	Es wird kein Kommando gesendet. Damit kann auch ein bereits bestehendes Kommando aus einer projektierten Funktion entfernt werden.
Treiber starten (Onlinemodus)	Treiber wird neu initialisiert und gestartet. **Hinweis:** Wenn der Treiber bereits gestartet wurde, muss er gestoppt werden. Erst dann kann der Treiber wieder neu initialisiert und gestartet werden.
Treiber stoppen (Offlinemodus)	Treiber wird angehalten, es werden keine neuen Daten angenommen. **Hinweis:** Ist der Treiber im Offline-Modus, erhalten alle Variablen, die für diesem Treiber angelegt wurden, den Status *Abgeschaltet* (*OFF*; Bit *20*).
Treiber in Simulationsmodus	Treiber wird in den Simulationsmodus gesetzt. Die Werte aller Variablen des Treibers werden vom Treiber simuliert. Es werden keine Werte von der angeschlossenen Hardware (z. B. SPS, Bussystem, ...) angezeigt.
Treiber in Hardwaremodus	Treiber wird in den Hardwaremodus gesetzt. Für die Variablen des Treibers werden die Werte von der angeschlossenen Hardware (z. B. SPS, Bussystem, ...) angezeigt.
Treiberspezifisches Kommando	Eingabe eines treiberspezifischen Kommandos. Öffnet Eingabefeld für die Eingabe eines Kommandos.
Treiber Sollwertsetzen aktivieren	Sollwert setzen auf Treiber ist möglich.
Treiber Sollwertsetzen deaktivieren	Sollwert setzen auf Treiber wird verhindert.
Verbindung mit Modem aufbauen	Verbindung aufbauen (für Modem-Treiber). Öffnet Eingabefelder für Hardware-Adresse und Eingabe der zu wählenden Nummer.
Verbindung mit Modem trennen	Verbindung beenden (für Modem-Treiber).
Treiber in Simulationsmodus zählend	Treiber wird in den zählenden Simulationsmodus gesetzt. Alle Werte werden mit *0* initialisiert und in der eingestellten Updatezeit jeweils um *1* bis zum Maximalwert inkrementiert und beginnen dann wieder

Treiberkommando	Beschreibung
	bei *0*.
Treiber in Simulationsmodus statisch	Es wird keine Kommunikation zur Steuerung aufgebaut. Alle Werte werden mit *0* initialisiert.
Treiber in Simulationsmodus programmiert	Die Werte werden von einem frei programmierbaren Simulationsprojekt berechnet. Das Simulationsprojekt wird mit der zenon Logic Workbench erstellt und läuft in der zenon Logic Runtime ab.

FUNKTION TREIBERKOMMANDOS IM NETZWERK

Wenn sich der Rechner, auf dem die Funktion **Treiberkommandos** ausgeführt wird, im zenon Netzwerk befindet, werden zusätzlich weitere Aktionen ausgeführt:

- Ein spezielles Netzwerkkommando wird vom Rechner zum Server des Projekts gesendet. Dieser führt dann die gewünschte Aktion auf seinem Treiber aus.
- Zusätzlich sendet der Server das gleiche Treiberkommando zum Standby des Projekts. Der Standby führt die Aktion auch auf seinem Treiber aus.

Dadurch ist gewährleistet, dass Server und Standby synchronisiert sind. Dies funktioniert nur, wenn Server und Standby jeweils eine funktionierende und unabhängige Verbindung zur Hardware haben.

10 Fehleranalyse

Sollte es zu Kommunikationsproblemen kommen, bietet dieses Kapitel Hilfe, um den Fehler zu finden.

10.1 Analysetool

Alle zenon Module wie z. B. Editor, Runtime, Treiber, usw. schreiben Meldungen in eine gemeinsame LOG-Datei. Um sie korrekt und übersichtlich anzuzeigen, benutzen Sie das Programm Diagnosis Viewer, das mit zenon mitinstalliert wird. Sie finden es unter **Start/Alle Programme/zenon/Tools 8.20 -> Diagviewer.**

zenon Treiber protokollieren alle Fehler in LOG-Dateien. LOG-Dateien sind Textdateien mit einer speziellen Struktur. Der Standardordner für die LOG-Dateien ist der Ordner **LOG** unterhalb des Ordners **ProgramData**, zum Beispiel:

%ProgramData%\COPA-DATA\LOG.

Achtung: Mit den Standardeinstellungen zeichnet ein Treiber nur Fehlerinformationen auf. Mit dem Diagnosis Viewer kann bei den meisten Treibern die Diagnose-Ebene auf „Debug" und „Deep Debug"

erweitert werden. Damit protokolliert der Treiber auch alle anderen wesentlichen Aufgaben und Ereignisse.

Im Diagnosis Viewer kann man auch:

- neu erstellte Einträge in Echtzeit mitverfolgen
- die Aufzeichnungseinstellungen anpassen
- den Ordner, in dem die LOG-Dateien gespeichert werden, ändern

Hinweise:

1. Der Diagnosis Viewer zeigt alle Einträge in UTC (Koordinierter Weltzeit) an und nicht in der lokalen Zeit.
2. Der Diagnosis Viewer zeigt in seiner Standardeinstellung nicht alle Spalten einer LOG-Datei an. Um mehr Spalten anzuzeigen, aktivieren Sie die Eigenschaft **Add all columns with entry** im Kontextmenü der Spaltentitel.
3. Bei Verwendung von reinem **Error-Logging** befindet sich eine Problembeschreibung in der Spalte **Error text**. In anderen Diagnose-Ebenen befindet sich diese Beschreibung in der Spalte **General text**.
4. Viele Treiber zeichnen bei Kommunikationsprobleme auch Fehlernummern auf, die die SPS ihnen zuweist. Diese werden in **Error text** und/oder **Error code** und/oder **Driver error parameter(1** und **2)** angezeigt. Hinweise zur Bedeutung der Fehlercodes erhalten Sie in der Treiberdokumentation und der Protokoll/SPS-Beschreibung.
5. Stellen Sie am Ende Ihrer Tests den Diagnose-Level von **Debug** oder **Deep Debug** wieder zurück. Bei **Debug** und **Deep Debug** fallen beim Protokollieren sehr viele Daten an, die auf der Festplatte gespeichert werden und die Leistung Ihres Systems beeinflussen können. Diese werden auch nach dem Schließen des Diagnosis Viewers weiter aufgezeichnet.

> **Achtung**
>
> Unter Windows CE werden aus Ressourcegründen Fehler standardmäßig nicht protokolliert.

Weitere Informationen zum Diagnosis Viewer finden Sie im Handbuch Diagnosis Viewer.

10.2 Treiberüberwachung

Die Runtime überwacht die Verfügbarkeit des Treibers via Watchdog. Ist ein Treiber nicht mehr verfügbar, wird für alle angemeldeten Variablen des Treibers zusätzlich das Statusbit *INVALID* gesetzt.

Mögliche Ursachen für Auslösen des Watchdogs:

- Der Treiberprozess läuft nicht mehr.

  Überprüfen Sie im Task Manager ob die Treiber-Exe noch läuft.

- Betriebssystem ist mit höher priorisierten Prozessen ausgelastet.

  Überprüfen Sie die Konfiguration Ihres Systems auf zu wenig Arbeitsspeicher und zu geringe CPU-Leistung. In diesem Fall erfolgt das Rücksetzen des *INVALID* Statusbits durch den Treiber nur bei Wertänderung auf der Gegenstelle. Statische Werte behalten das *INVALID* Statusbit bis zum nächsten Start der Runtime oder des Treibers.

WATCHDOG KONFIGURATION

Für die Überwachung der Kommunikation zur Runtime wird die Verbindung zum Treiber in einem fix vorgegebenen Zeitraum von 60 Sekunden überprüft. Dieser Vorgang wird mehrmals wiederholt. Konnte nach 5 Versuchen (= innerhalb von 5 Minuten) keine valide Verbindung zum Treiber erkannt werden, wird das *INVALID*-Bit bei der angemeldeten (*advised*) Variablen gesetzt. Zusätzlich wird das *INVALID*-Bit auch gesetzt, wenn neue Variablen angemeldeten werden. Das *INVALID*-Bit wird nicht mehr zurückgesetzt.

Dafür werden entsprechende LOG-Einträge erstellt.

LOG-EINTRAG

Bei Auslösen des Watchdogs wird eine Fehlermeldung im LOG protokolliert:

Parameter	Beschreibung
Communication with driver:<drvExe>/<drvDesc>(id:<drvId>) timed out. No communication for <time> ms.	Keine Kommunikation mit Treiber innerhalb der angegeben Zeit. - *<time>*: Zeitangabe in Millisekunden - *<drvDesc>*: Treibername - *<drvExe>*: Treiber EXE-Name - *<drvId>*: Treiber-ID im zenon Projekt
Communication with %s timed out. Invalid-Bit will be set.	Die Kommunikation zum Treiber *%s* konnte bei 5 Versuchen nicht innerhalb von 60 Sekunden aufgebaut werden. Bei der Variable wird das *INVALID*-Bit gesetzt.
Communication with %s timed out. Timeout happened %d times	Die Kommunikation zum Treiber *%s* konnte *%d* Mal nicht innerhalb von 60 Sekunden aufgebaut werden.

10.3 Checkliste

Fragen und Hinweise zur Fehlereingrenzung:

ALLGEMEINE FEHLERSUCHE

- Ist die Steuerung an die Stromversorgung angeschlossen?
- Analyse mit Hilfe des **Diagnose Viewers** (auf Seite 67):
  -> Welche Meldungen werden angezeigt?
- Sind die Teilnehmer im **TCP/IP**-Netz verfügbar?
- Kann die Steuerung über den Befehl *Ping* erreicht werden?

  Ping: **Kommandozeile öffnen -> ping <IP-Adresse> (z.B.: ping 192.168.0.100) -> Eingabe-Taste drücken.**

  Kommt eine Antwort mit Zeitangabe oder ein Timeout?
- Kann die Steuerung auf dem entsprechenden Port über *Telnet* erreicht werden?

  Telnet: **Kommandozeile öffnen, telnet <IP-Adresse Port-Nummer> eingeben (z.B. telnet 192.168.0.100 20000) -> Eingabe-Taste drücken .**

  Wird der Bildschirm schwarz, und blinkt der Cursor konnte eine Verbindung aufgebaut werden.
- Analyse mit Hilfe eines Programms zur Analyse des Netzwerkverkehrs (Sniffer, zB. Wireshark, Microsoft Network Monitor / Microsoft Message Analyzer)
- Wird für die Verbindung von Steuerung und PC das korrekte, vom Hersteller empfohlene Kabel verwendet?
- Wurde der richtige COM Port ausgewählt?
- Stimmen die seriellen Kommunikationsparameter (Baudrate, Parität, Start/Stop Bits,...) überein?
- Wird der COM Port durch eine andere Anwendung blockiert?
- Wurde die Netzadresse in den Adresseigenschaften der Variable korrekt eingestellt?
  - Stimmt die Adressierung mit der Konfiguration im Treiberdialog überein?
  - Entspricht die Netzadresse der Adresse der Zielstation?
- Wird in der Variable der richtige Objekttyp verwendet?

  Beispiel: Treibervariablen sind reine Statistikvariablen und kommunizieren nicht mit der Steuerung. (Siehe Kapitel Treibervariablen (auf Seite 53).)
- Stimmt die Offset-Adressierung der Variable mit der in der Steuerung überein?

MANCHE VARIABLEN MELDEN INVALID

- INVALID Bits beziehen sich immer auf eine Netzadresse.
- Mindestens eine Variable der Netzadresse ist gestört.

WERTE WERDEN NICHT ANGEZEIGT, ZAHLENWERTE BLEIBEN LEER

Unter Umstände wurde auf eine Lese-Anfrage an die Steuerung noch keine Antwort Empfangen.

Treiber läuft nicht. Überprüfen Sie die:

- Installation von zenon
- Installation des Treibers
- Installation aller Komponenten
  -> Achten Sie auf Fehlermeldungen beim Start der Runtime.

VARIABLEN WERDEN MIT BLAUEM PUNKT ANGEZEIGT

Die Kommunikation im Netzwerk ist gestört:

- Bei einem Netzwerkprojekt:
  Läuft das Netzwerkprojekt auch auf dem Server?
- Bei einem Standalone-Projekt oder Netzwerkprojekt, das auf dem Server läuft:
  Deaktivieren Sie die Eigenschaft Nur von Standby Server anfordern im Knoten Treiber Anbindung/Adressierung.

WERTE WERDEN FALSCH ANGEZEIGT

Überprüfen Sie die Angaben zur Berechnung im Knoten Wertberechnung der Variablen-Eigenschaften?

Überprüfen Sie die Konfiguration der Outstation,ob tatsächlich die gewünschte Variation gesendet wird (z.B. **Float** mit Nachkommsstellen).

WERTE WERDEN NICHT SCHNELL GENUG ANGEZEIGT

Überprüfen Sie in der Outstation, ob für die gewünschten Werte Events generiert werden und in welche Klasse. Prüfen Sie das Pollen diese Eventklasse in der Treiberkonfiguration.

Überprüfen Sie in der Outstation die Konfiguration von **unsolicited responses**.

ZEITSTEMPEL DER VARIABLE IST FALSCH

Überprüfen Sie in der Konfiguration der Outstation, ob die KDNP1-Objekte tatsächlich mit Zeitstempel gesendet werden. (Entsprechende Variation gewählt mit Zeitstempel.)

Überprüfen Sie, ob die Outstation lokale Zeit verwendet oder UTC und setzen Sie die Option in der Treiberkonfiguration entsprechend,

Überprüfen Sie die Uhrzeit des lokalen Rechners und die Uhrzeit der Outstation, inklusive Einstellungen für die Zeitzone.

TREIBER FÄLLT SPORADISCH AUS

Analyse mit Hilfe des **Diagnose Viewers** (auf Seite 67):
-> Welche Meldungen werden angezeigt?

Überprüfen Sie die Antwortzeit-Timeout in der Treiberkonfiguration, speziell bei Outstations mit vielen Datenpunkte und langsame Verbindung.

Bei *Secure Authentication*, überprüfen Sie ob der **pre-shared Update Key** im Treiber und in der Outstation identisch ist.