zenon Treiber Handbuch

Treibername

v.7.50

©2016 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.

Inhalt

1. Willkommen bei der COPA-DATA Hilfe 4

2. Treibername 4

2.1 COMLI - Datenblatt 5

2.2 Treiber-Historie 6

2.3 Konfiguration 6

2.3.1 Anlegen eines Treibers 7

2.3.2 Einstellungen im Treiberdialog 10

2.4 Variablen anlegen 18

2.4.1 Variablen im Editor anlegen 18

2.4.2 Adressierung 22

2.4.3 Treiberobjekte und Datentypen 23

2.4.4 Variablen anlegen durch Import 28

2.4.5 Treibervariablen 38

2.5 Treiberkommandos 44

2.6 Fehleranalyse 47

2.6.1 Analysetool 47

2.6.2 Checkliste 48

1. Willkommen bei der COPA-DATA Hilfe

ALLGEMEINE HILFE

Falls Sie in diesem Hilfekapitel Informationen vermissen oder Wьnsche fьr Ergдnzungen haben, wenden Sie sich bitte per E-Mail an documentation@copadata.com (mailto:documentation@copadata.com).

PROJEKTUNTERSTЬTZUNG

Unterstьtzung bei Fragen zu konkreten eigenen Projekten erhalten Sie vom Support-Team, das Sie per E-Mail an support@copadata.com (mailto:support@copadata.com) erreichen.

LIZENZEN UND MODULE

Sollten Sie feststellen, dass Sie weitere Module oder Lizenzen benцtigen, sind unsere Mitarbeiter unter sales@copadata.com (mailto:sales@copadata.com) gerne fьr Sie da.

2. Treibername

Der **COMLI-Treiber** kommuniziert mit dem ABB COMmunication LInk-Protokoll via serieller Schnittstelle. zenon ist bei dieser Kommunikation immer Master.

2.1 COMLI - Datenblatt

Allgemein:	
Treiberdateiname	COMLI.exe
Treiberbezeichnung	ABB-COMLI Treiber
Steuerungs-Typen	Alle Steuerungen, die das ABB COMLI Protokoll unterstützen
Steuerungs-Hersteller	ABB;

Treiber unterstützt:	
Protokoll	ABB COMmunication Link;
Adressierung: Adress-basiert	X
Adressierung: Namens-basiert	--
Kommunikation spontan	X
Kommunikation pollend	X
Online Browsing	--
Offline Browsing	--
Echtzeitfдhig	X
Blockwrite	--
Modemfähig	--
Serielles Logging	X
RDA numerisch	X
RDA String	--

Voraussetzungen:	
Hardware PC	Serielle Schnittstelle RS232
Software PC	--
Hardware Steuerung	--
Software Steuerung	--
Benötigt v-dll	--

Plattformen:	
Betriebssysteme	Windows 7, 8, 8.1, 10, Server 2008R2, Server 2012, Server 2012R2;
CE Plattformen	-;

2.2 Treiber-Historie

Datum	Build Nummer	Änderung
22.06.16	100	Treiberdokumentation wurde neu erstellt

2.3 Konfiguration

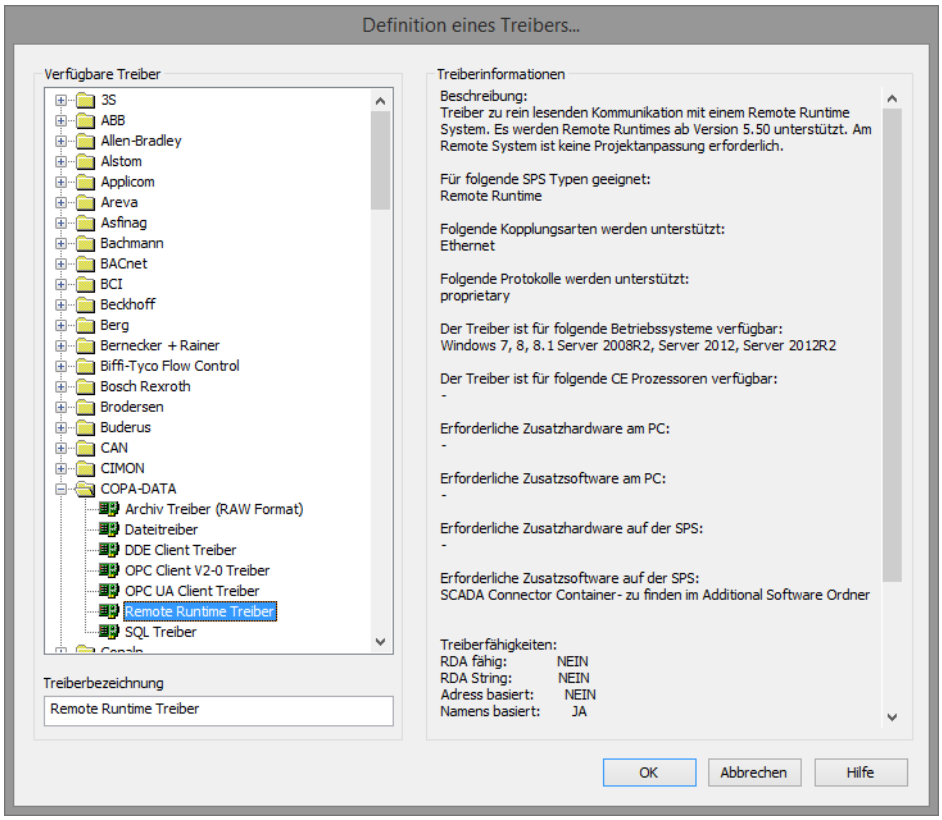
In diesem Kapitel lesen Sie, wie Sie den Treiber im Projekt anlegen und welche Einstellungen beim Treiber möglich sind.

Info

Weitere Einstellungen, die Sie für Variablen in zenon vornehmen können, finden Sie im Kapitel Variablen (main.chm::/15247.htm) der Online-Hilfe.

2.3.1 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 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

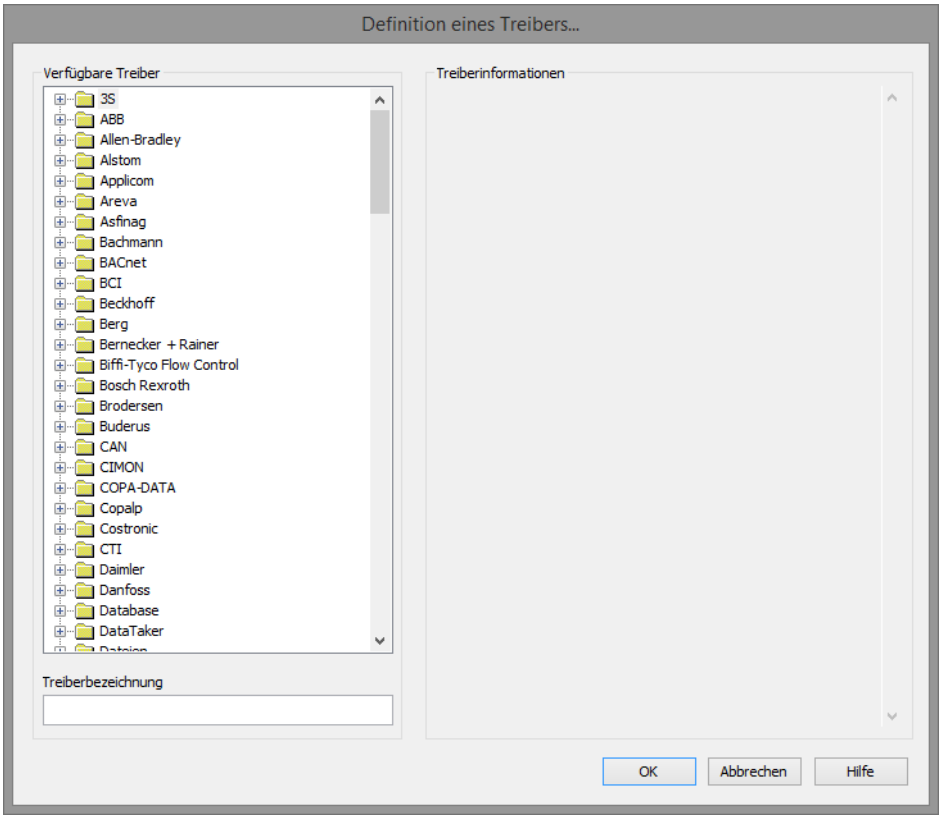
Die Inhalte dieses Dialogs sind in der Datei Treiber_[Sprachkürzel].xml gespeichert. Sie finden diese Datei im Ordner `C:\ProgramData\COPA-DATA\zenon[Versionsnummer].`

TREIBER NEU ANLEGEN

Um einen neuen Treiber anzulegen:

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



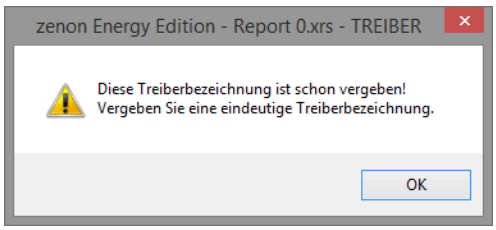
3. Wählen Sie den gewünschten Treiber und benennen Sie diesen im Eingabefeld **Treiberbezeichnung**.
   Dieses Eingabefeld entspricht der Eigenschaft **Bezeichnung**. Per Default wird der Name des ausgewählten Treibers in diesem Eingabefeld automatisch eingefügt.
   Für die Treiberbezeichnung gilt:

   - 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.

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.



<CD_PRODUCNTAME> PROJEKT

Bei neu angelegten Projekten werden die folgenden Treiber automatisch angelegt:

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

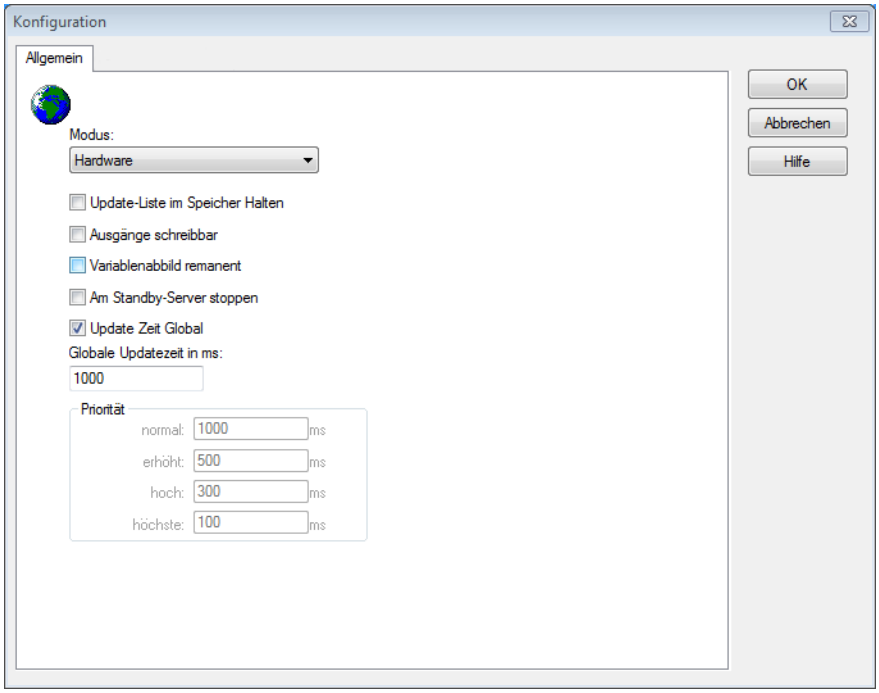
> **Info**
>
> In einem zenon Projekt müssen nur die benötigten Treiber vorhanden sein. Treiber können bei Bedarf zu einem späteren Zeitpunkt hinzugefügt werden.

2.3.2 Einstellungen im Treiberdialog

Folgende Einstellungen können Sie beim Treiber vornehmen:

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**.



Parameter	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. 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. (main.chm::/25206.htm)
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 der Status S_MERKER_1(0) bis S_MERKER8(7), REVISION(9), AUS(20) oder ERSATZWERT(27) aktiv ist Das Variablenabbild wird immer gespeichert wenn: ▸ die Variable vom Objekttyp **Treibervariable** 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** (**statusverarbeitung.chm::/24150.htm**), sondern einen leeren Wert. Damit wird verhindert, dass beim Hochstufen zum Server nicht relevante Werte in AML , CEL und Archiv erzeugt werden. **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	`Aktiv`: Die eingestellte **Globale Update Zeit** in `ms` wird für alle Variablen im Projekt verwendet. Die bei den Variablen eingestellte Priorität wird nicht verwendet. `Inaktiv`: Die eingestellten Prioritäten werden für die einzelnen Variablen verwendet.
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. Zum Beispiel unterstützen spontan kommunizierende Treiber dies nicht.

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.

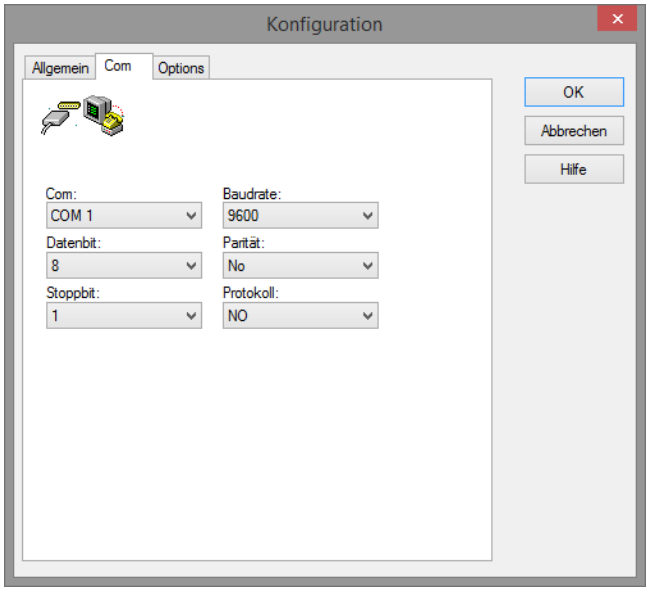
UPDATE ZEIT ZYKLISCHE TREIBER

Für zyklische Treiber gilt:

Beim **Sollwert Setzen**, **Advisen** von Variablen und bei **Requests** wird sofort ein Lesezyklus für alle Treiber ausgelöst - unabhängig von der eingestellten Update Zeit. Damit wird sicher gestellt, dass der Wert nach dem Schreiben in der Visualisierung sofort zur Verfügung steht. Update-Zeiten können damit für zyklische Treiber kürzer ausfallen als eingestellt.

COM

In dieser Registerkarte konfigurieren Sie die Verbindungsparameter für die serielle Kommunikation zur SPS.



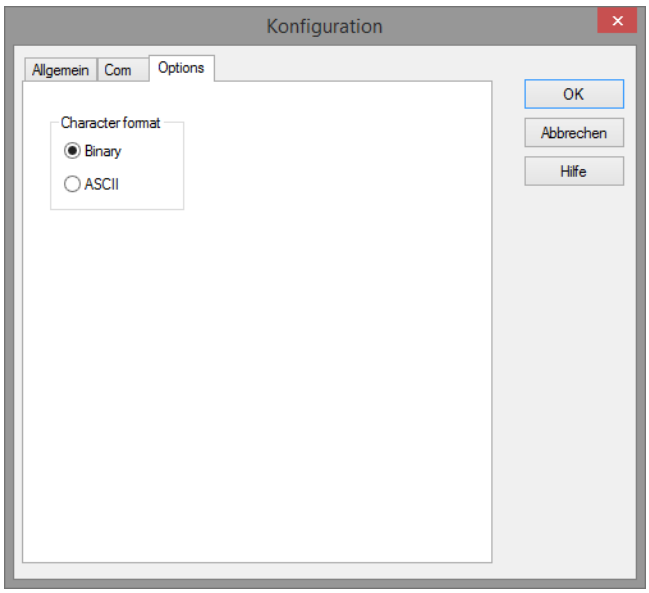
Parameter	Beschreibung
Com	Auswahl der seriellen Schnittstelle (Com-Port) am Rechner. Auswahl aus Dropdownliste: `COM 1` bis `COM 256` Default: `COM 1`
Baudrate	Auswahl der Baudrate für die Kommunikation zur SPS. Die Baudrate muss an die Steuerung angepasst werden. Auswahl aus Dropdownliste: `110` bis `256000` Default: `9600`
Datenbit	Anzahl der Datenbits (Datenwortlänge in Bit) für die Kommunikation zur SPS. Die Datenbitrate muss an die Steuerung angepasst werden. Auswahl aus Dropdownliste: `5, 6, 7, 8` Default: `8`
Stoppbit	Auswahl des Stoppbits für die Kommunikation zur SPS. Das Stoppbit muss an die Steuerung angepasst werden. Auswahl aus Dropdownliste: ▶ `1` ▶ `1.5` ▶ `2` Default: `1`
Parität	Auswahl der Parität für die Kommunikation zur SPS. Das Parität muss an die Steuerung angepasst werden. Anpassen an Steuerung. Auswahl aus Dropdownliste: ▶ `No` ▶ `Odd` ▶ `Even` Default: `No`
Protokoll	Auswahl des Protokolls für die Kommunikation zur SPS. Das Protokoll muss an die Steuerung angepasst werden. 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.

Options

In diesem Dialog konfigurieren Sie die Kommunikationsparameter für die serielle Kommunikation zur SPS.



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

Die Schaltflächen werden in der Systemsprache des Rechners angezeigt.

CHARACTER FORMAT

Auswahl der Enkodierung für die Kommunikation zwischen Treiber und SPS. Die Konfiguration muss mit der Einstellung auf der SPS übereinstimmen. Auswahl mit Optionsfeld.

Parameter	Beschreibung
Binary	Kommunikationsdaten werden im Binär-Format übertragen. Hinweis: nur verfügbar, wenn in der Registerkarte **Com** die Option **Datenbit** mit dem Wert 8 (Datenbit) konfiguriert ist.
ASCII	Kommunikationsdaten werden im ASCII-Format übertragen.

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.

2.4 Variablen anlegen

So werden Variablen im zenon Editor angelegt:

2.4.1 Variablen im Editor anlegen

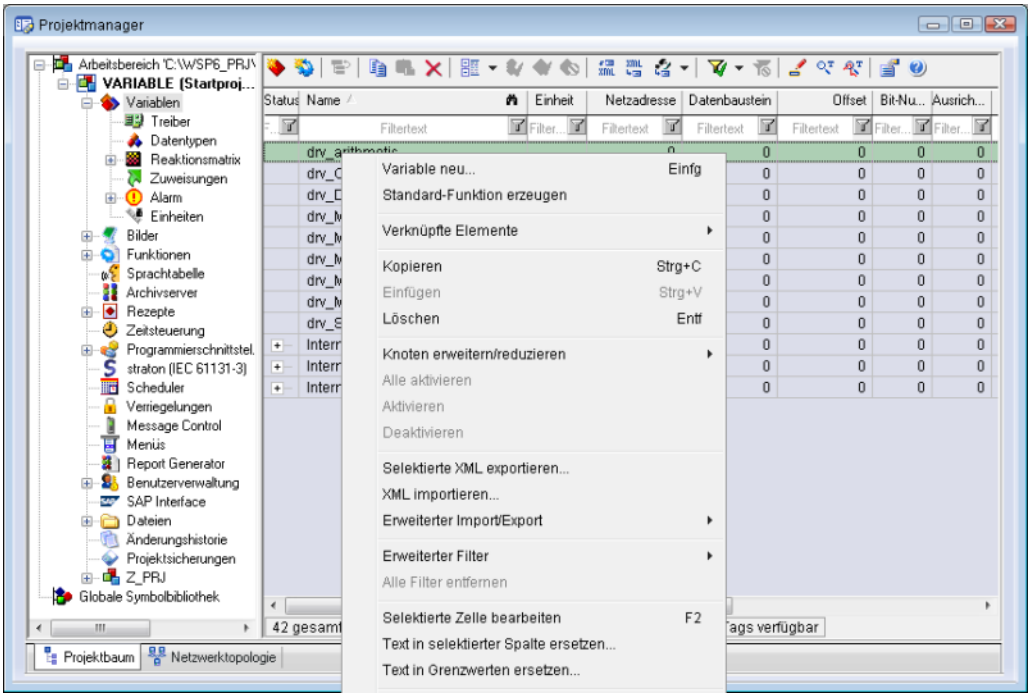
Variablen können angelegt werden:

- als einfache Variable
- in Arrays (main.chm::/15262.htm)
- als Struktur-Variablen (main.chm::/15278.htm)

DIALOG VARIABLE

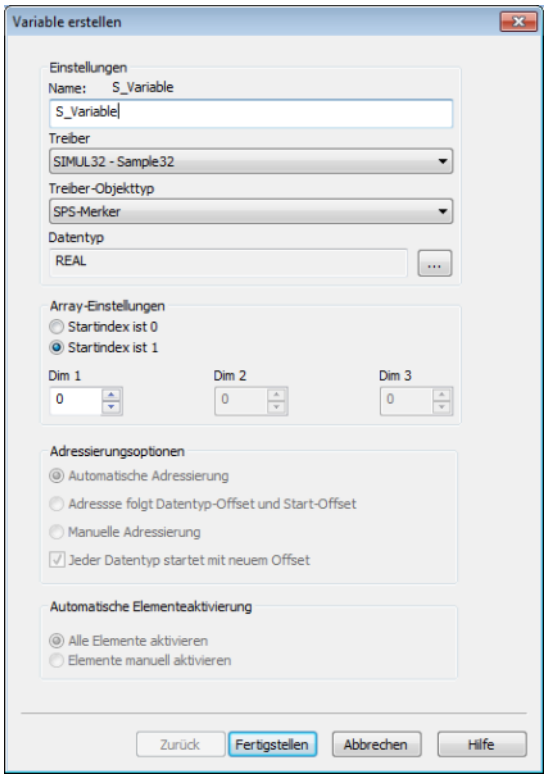
Um eine neue Variable zu erstellen, gleich welchen Typs:

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



2. der Dialog zur Konfiguration der Variable wird geöffnet
3. konfigurieren Sie die Variable

4. welche Einstellungen möglich sind, hängt ab vom Typ der Variablen



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** (**Main.chm::/Intern.chm::/Intern.htm**)) geladen.
Treiberobjekttyp (cti.chm::/28685.htm)	Wählen Sie aus der Dropdownliste den passenden Treiberobjekttyp aus.
Datentyp	Wählen Sie den gewünschten Datentyp. Klick auf die Schaltfläche ... 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.

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.

2.4.2 Adressierung

Gruppe/Eigenschaft	Beschreibung
Allgemein	
Name	Frei vergebbarer Name. **Achtung:** Je zenon Projekt muss der Name eindeutig sein.
Kennung	Frei vergebbare Kennung, z. B. für Betriebsmittelkennung, Kommentar...
Adressierung	
Netzadresse	Diese Adresse bezieht sich auf die Netzadresse der Verbindungsprojektierung im Treiber. Damit wird ausgewählt auf welcher Steuerung sich die Variable befindet. **Achtung:** Für den COMLI-Treiber ist die Netzadresse der projektierte COM-PORT. Diese Projektierung finden Sie im Treiberkonfigurationsdialog, in der Registerkarte **COM** in der Option Com:
Datenbaustein	Wird für diesen Treiber nicht verwendet.
Offset	Offset der Variablen, die Speicheradresse der Variable in der Steuerung. Einstellbar von 0 bis 4294967295 .
Ausrichtung	Wird für diesen Treiber nicht verwendet.
Bitnummer	Nummer des Bits innerhalb des eingestellten Offsets. 0 ... 8
Stringlänge	Nur verfügbar bei String-Variablen: Maximale Anzahl von Zeichen, die die Variable aufnehmen kann.
Treiber Anbindung/Treiberobjekttyp	Objekttyp der Variablen. Wird abhängig vom verwendeten Treiber beim Erstellen der Variablen ausgewählt und kann hier geändert werden.
Treiber Anbindung/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.

2.4.3 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.

Treiberobjekte

Folgende Objekttypen stehen in diesem Treiber zur Verfügung:

Treiberobjekttyp	Kanaltyp	Lesen	Schreiben	Unterstützte Datentypen	Beschreibung
Registers	8	X	-	BOOL, SINT, USINT	Variablen zum Zugriff auf Register (0...3071) **Achtung:** Variablen vom Datentypen BOOL, SINT und USINT können nur gelesen werden.
			X	INT, UINT, DINT, UDINT	
IO Bits	64	X	X	BOOL	Variablen zum Zugriff auf I/O Bits (0...16376)
High Register	65	X	-	BOOL, SINT, USINT	Variablen zum Zugriff auf Register (0...65535) **Achtung:** Variablen vom Datentypen BOOL, SINT und USINT können nur gelesen werden
			X	INT, UINT, DINT, UDINT	
Memory	66	X	-	BOOL	Variablen zum Zugriff auf den SPS Speicher (0...128kB) **Achtung:** Variablen vom Datentyp BOOL können nur gelesen werden
			X	SINT, USINT, INT, UINT, DINT, UDINT	

Time	67	X	X	UDINT	Variable zum Lesen der aktuellen Uhrzeit der Steuerung. Durch Schreiben eines beliebigen Wertes wird die aktuelle Rechnerzeit als Uhrzeit der Steuerung gesetzt.
Time marked events	68	X		USINT	Variablen zum Empfang von zeitgestempelten Ereignissen
Treibervariable	35	X	X	BOOL, SINT, USINT, INT, UINT, DINT, UDINT, REAL, STRING	Variablen für die statische Analyse der Kommunikation; wird zwischen Treiber und Runtime übertragen (nicht zur SPS). **Hinweis**: Die Adressierung und das Verhalten ist bei den meisten zenon Treibern gleich Weitere Infos finden Sie bei den Treibervariablen (auf Seite 38)

REGISTERS

Über diesen Treiberobjekttyp können Register (0...3071) gelesen bzw. geschrieben werden.
Achtung: Variablen vom Datentypen BOOL, SINT und USINT können nur gelesen und nicht geschrieben werden.

Offset = [Registernummer]

IO BITS

Über diesen Treiberobjekttyp können I/O Bits (0...16376) gelesen oder geschrieben werden. Die Adresse eines I/O Bits muss dabei wie folgt konfiguriert werden:

Offset = [IO Bit - Nummer] / 8

Bitnummer = [IO Bit - Nummer] mod 8

HIGH REGISTER

Über diesen Treiberobjekttyp können die Register (0...65535) gelesen und geschrieben werden.

Achtung: Variablen vom Datentypen BOOL, SINT und USINT können nur gelesen und nicht geschrieben werden.

Offset = `[Registernummer]`

MEMORY

Über diesen Treiberobjekttyp kann auf den SPS Speicher (0...128kB) zugegriffen werden.

Achtung: Variablen vom Datentypen `BOOL` können nur gelesen und nicht geschrieben werden.

TIME

Über diesen Treiberobjekttyp kann die aktuelle Uhrzeit der Steuerung gelesen werden.

Durch Schreiben des Wertes 0 auf die SPS, wird eine Zeitsynchronisation durchgeführt. Für diese Synchronisation wird die aktuelle Rechnerzeit als Uhrzeit der Steuerung gesetzt. Die Uhrzeit wird als Unix-Time auf die Variable abgebildet.

> **Achtung**
>
> *Um Werte mit dem Treiberobjekttyp **Time** in zenon anzeigen zu können muss die Uhrzeit in der SPS auf UTC eingestellt sein.*

TIME MARKED EVENTS

Über diesen Treiberobjekttyp können **Time marked events** von der Steuerung gelesen werden.

ZUWEISUNG:

Die Zuordnung eines Ereignisses zu einer Variable erfolgt über die Ereignis-Adresse. Diese muss für die Variable als Offset eingestellt werden.

- Der Ereignistyp wird der Variable als Wert zugewiesen.
- Die Ereigniszeit wird der Variable als Zeitstempel zugewiesen.

TIME MARKED EVENTS TYPEN

Folgende Typen werden vom COMLI Standard definiert:

Event	Hex	Binär
Alarm Ein (1 -> 0)	`Typ 0x30`	`48`
Alarm Aus (0 -> 1)	`Typ 0x31`	`49`
Status Ein (1 -> 0)	`Typ 0x32`	`50`
Status Aus (0 -> 1)	`Typ 0x33`	`51`

Für die Abbildung **Time marked events** auf Variablen gilt:

1. Bei Neuaufschaltung einer Ereignisvariable wird diese mit dem Wert null initialisiert.
2. Der Ereignistyp der Variable wird dann beim Empfang eines Ereignisses zugewiesen.
3. Anschließend wird der Wert der Variable sofort wieder mit dem Wert `Null` überschrieben.

Projektieren Sie für jede Ereignisvariable, die sie auswerten wollen eine entsprechende Reaktionsmatrix.

> **Achtung**
>
> Ist die zenon Redundanz aktiviert, erfolgt das Lesen der **Time marked events** nur am Prozessführenden Server.
>
> Vorrausetzung: Die Uhrzeit in der Steuerung muss auf UTC eingestellt sein.

PROJEKTIEREN IM EDITOR

Führen Sie folgende Projektierungsschritte durch:

1. Erstellen Sie eine neue Variable.
   a) Wählen Sie in der Symbolleiste oder im Kontextmenü des Knotens **Variable** den Befehl **Variable neu...**
   Der Dialog zum Anlegen einer Variable wird geöffnet.
2. Projektieren Sie die Variable:
   a) Stellen Sie sicher, dass in der Option **Treiber** der Treiber `COMLI - [Treibernahme]` ausgewählt ist.
   b) Wählen Sie in der Dropdownliste der Option **Treiberobjekttyp** den Eintrag `Time marked events`.
   c) Bestätigen Sie Ihre Projektierung mit Klick auf die Schaltfläche **OK**.
3. Ordnen Sie der Variable mit der Eigenschaft Netzadresse eine Verbindung zu.
4. Projektieren Sie eine Reaktionsmatrix:
   a) Wählen Sie im Kontextmenü des Knotens **Variable** den Unterknoten **Reaktionsmatrix**.
   b) Wählen Sie in der Symbolleiste oder im Kontextmenü den Befehl **Reaktionsmatrix neu...**
   Der Dialog zum Anlegen einer Reaktionsmatrix wird geöffnet.

c) Wählen Sie die gewünschte Art der Matrix via Optionsfeld bei **Typ der Reaktionsmatrix** und benennen Sie die Reaktionsmatrix.

d) Bestätigen Sie Ihre Eingabe mit Klick auf die Schaltfläche OK.
Der Dialog zur Konfiguration der Reaktionsmatrix wird geöffnet.

5. Konfigurieren sie Zustände für den gewünschten Event.
Hinweis: Bei einer numerischen REMA verwenden Sie die binären Werte der Tabelle **Time marked events Typen** in diesem Abschnitt.

a) Aktivieren Sie die Option **Jede Wertänderung als neue Verletzung behandeln**.

b) Aktivieren Sie die Option **In Chronologische Ereignisliste**.

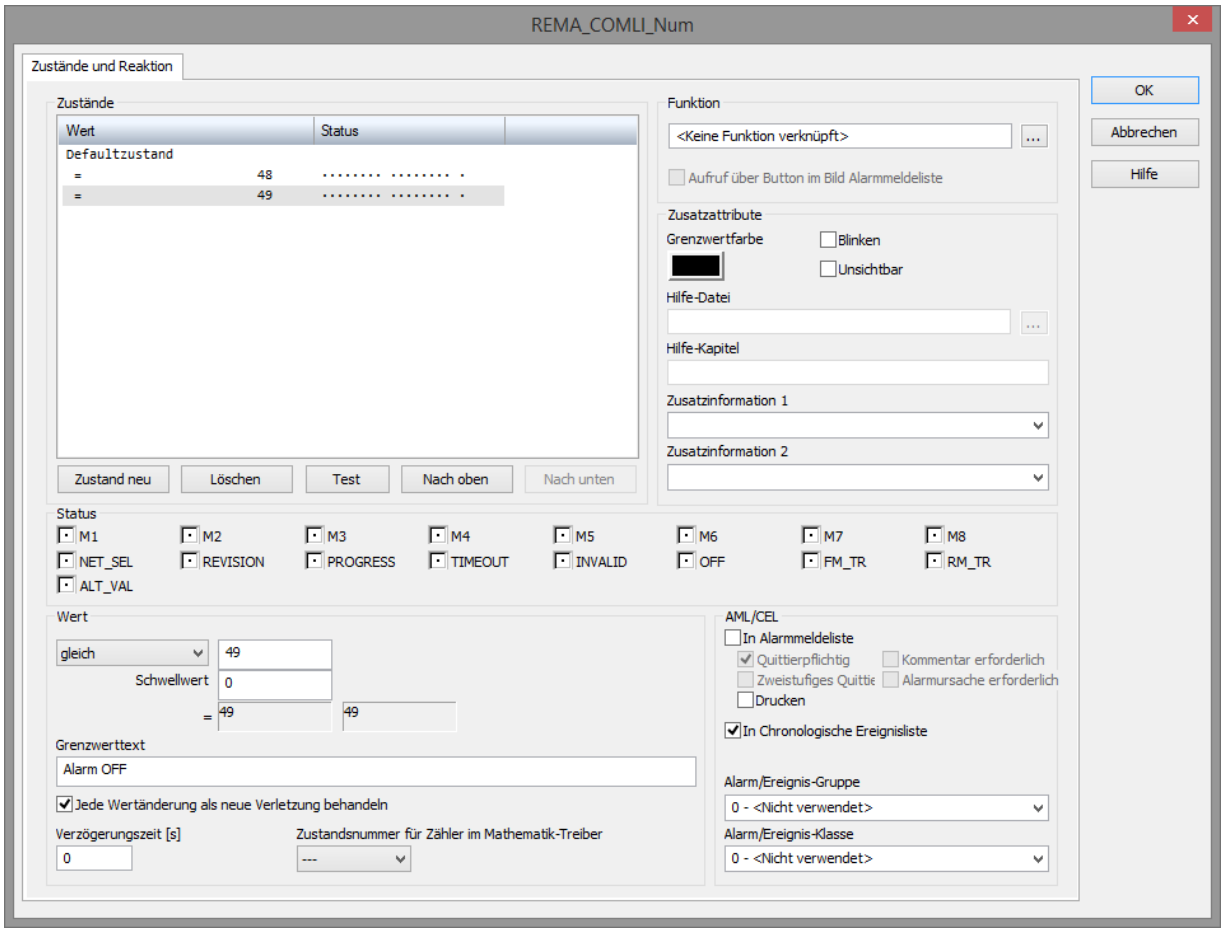
c) Optional: Aktivieren Sie **In Alarmmeldeliste**, um Inhalte auch in die AML aufzunehmen.

6. Verknüpfen Sie die Reaktionsmatrix mit einer Variable.

a) Wählen Sie dazu die entsprechende Variable in der Liste der Variablen aus.

b) Klicken Sie in der Variableneigenschaft **Reaktionsmatrix** auf die Schaltfläche **...**:
Der Dialog zur Auswahl einer Reaktionsmatrix wird geöffnet.

c) Wählen Sie die gewünschte Reaktionsmatrix und bestätigen Sie Ihre Auswahl mit Klick auf die Schaltfläche **OK**.



Zuordnung der Datentypen

Alle Variablen in zenon werden von IEC-Datentypen abgeleitet. In folgender Tabelle werden zur besseren Übersicht die IEC-Datentypen den Datentypen der Steuerung gegenübergestellt.

Steuerung	zenon	Datenart
I/O Bits	BOOL	8
Register	USINT	9
Register	SINT	10
Register	UINT	2
Register	INT	1
Register	UDINT	4
Register	DINT	3
-	ULINT	27
-	LINT	26
-	REAL	5
-	LREAL	6
-	STRING	12
-	WSTRING	21
-	DATE	18
-	TIME	17
-	DATE_AND_TIME	20
-	TOD (Time of Day)	19

Datenart: Die Eigenschaft **Datenart** ist die interne numerische Bezeichnung des Datentyps. Diese wird auch für den erweiterten DBF Import/Export der Variablen verwendet.

2.4.4 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 (main.chm::/13028.htm) im Abschnitt Variablen (main.chm::/13045.htm).

XML Import

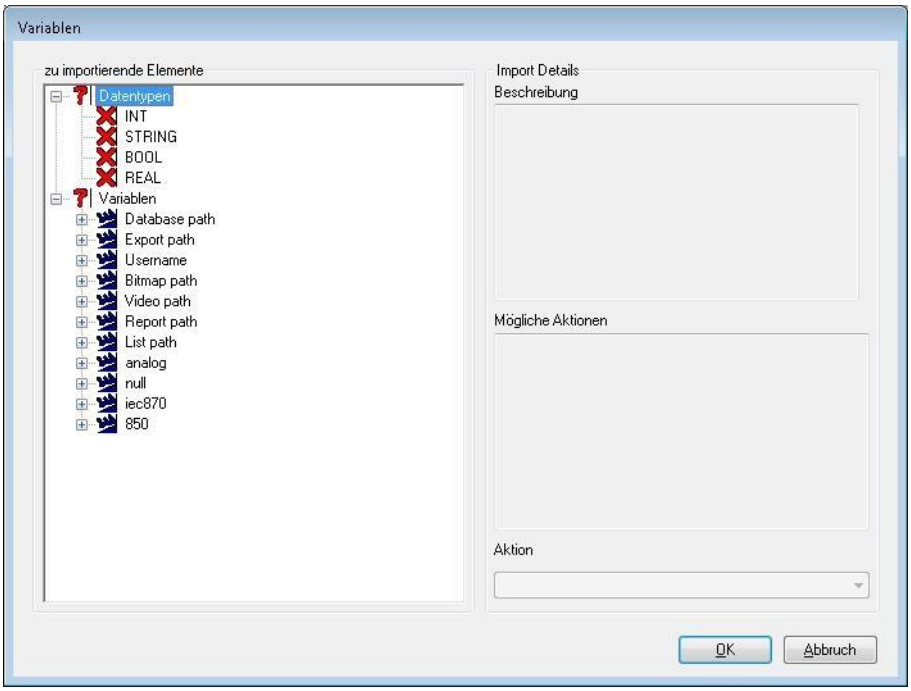
Beim XML- Import von Variablen oder Datentypen werden diese nach der Treiberzuordnung von zenon analysiert. Der Benutzer hat anschließend die Möglichkeit zu bestimmen, was mit den jeweiligen Variablen/Datentypen geschehen soll:

- importieren
- überschreiben
- nicht importieren.

Angeboten werden jeweils nur sinnvolle Aktionen.

Folgende Konflikte verhindern den Import:

- Der Datentyp existiert bereits und die zugehörige Variable kann nicht auf den zu importierenden Datentyp konvertiert werden.
- Der Datentyp ist komplex und die Elemente des existierenden unterscheiden sich von dem zu importierenden durch Anzahl oder Name.
- Der Datentyp ist komplex, die Elemente sind zwar gleich, aber unterscheiden sich in ihren Datentypen wobei eine Konvertierung zugehöriger Variablen nicht möglich ist.
- Die Variable ist von einem Typ, der nicht importiert werden kann (s.o.).
- Die Variable ist von einem anderen Typ als die bestehende (einfach/komplex).
- Die Treiberzuordnung für die Variable ist nicht korrekt (der zugeordnete Treiber unterstützt die entsprechende Datenart und/oder den entsprechenden Bereich nicht).


Variablen
zu importierende Elemente
Datentypen
INT
STRING
BOOL
REAL
Variablen
Database path
Export path
Username
Bitmap path
Video path
Report path
List path
analog
null
iec870
850
Import Details
Beschreibung
Mögliche Aktionen
Aktion
OK
Abbruch

Parameter	Beschreibung
Zu importierende Elemente	Listet alle Variablen und Datentypen, die importiert werden sollen, auf. Bei komplexen Variablen werden die zugehörigen einfachen Variablen angezeigt. Strukturierte Typen können aufgeklappt werden und zur Information werden die Elemente des Strukturdatentyps angezeigt. Symbole zeigen an, ob das Element importiert werden kann (grüner Haken(, nicht importiert werden kann (rotes X) oder eine Konflikt gelöst werden muss (blaues Symbol).
Import-Details	Beschreibt Importstatus und bietet Lösungsmöglichkeiten bei Konflikten.
Beschreibung	Beschreibt den Status des Elements.
Mögliche Aktionen	Erklärt die möglichen Aktionen.
Aktion	Dropdownliste mit den möglichen Aktionen und der aktuell ausgewählten Aktion. Gibt es nur eine mögliche Aktion, dann ist diese vorausgewählt und lässt sich auch nicht ändern.
`Datentyp/Variable neu importieren`	Die Variable/der Datentyp existiert noch nicht und wird durch den Import neu angelegt. (Default) Nicht vom Benutzer änderbar.
`Datentyp/Variable nicht importieren`	Die Variable/der Datentyp existiert schon und es gibt Konflikte. Nicht vom Benutzer änderbar.
`Datentyp/Variable importieren`	Die Variable/der Datentyp existiert schon, es gibt keine Konflikte. Der Benutzer kann entscheiden, ob die Variable/der Datentyp überschrieben oder nicht importiert werden soll. Per Default wird die Variable/der Datentyp überschrieben.

EINSCHRÄNKUNGEN

- 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, bis hin zu Abstürzen.

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

- 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.

- Struktur-Datentypen

  Struktur-Datentypen müssen über die gleiche Anzahl von Strukturelementen verfügen. Hat zum Beispiel ein Strukturdatentyp im Projekt drei Strukturelemente, ein gleichnamiger Datentyp in der XML-Datei aber vier Strukturelemente, wird keiner der auf diesem Datentyp basierenden Variablen der Export-Datei in das Projekt importiert.

> **Tipp**
>
> *Weitere Informationen zum XML-Import finden Sie im Handbuch **Import - Export**, im Kapitel **XML-Import** (**main.chm::/13046.htm**).*

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 dem 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 dem 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.
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)
OBJEKT	N	2	Treiberspezifische ID-Nummer des Primitivobjekts setzt sich zusammen aus TREIBER-OBJEKTTYP und DATENTYP
SIGMIN	Float	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
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
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.

2.4.5 Treibervariablen

Das Treiberkit implementiert eine Reihe von Treibervariablen. Diese sind unterteilt in:

- Information
- Konfiguration
- Statistik und
- Fehlermeldungen

Die Definitionen der im Treiberkit implementierten Variablen sind in der Importdatei **drvvar.dbf** (auf der Installationsmedium im Ordner `\Predefined\Variables`) verfügbar und können von dort importiert werden.

Hinweis: Variablennamen müssen in zenon einzigartig sein. Soll nach einem Import der Treibervariablen aus **drvvar.dbf** ein erneuter Import durchgeführt werden, müssen die zuvor importierten Variablen umbenannt werden.

Info

Nicht jeder Treiber unterstützt alle Treibervariablen.

Zum Beispiel werden:

- Variablen für Modem-Informationen nur von modemfähigen Treibern unterstützt
- Treibervariablen für den Polling-Zyklus nur für rein pollenden Treibern
- verbindungsbezogene Informationen wie ErrorMSG nur von Treibern, 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.
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.

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).
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).
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.

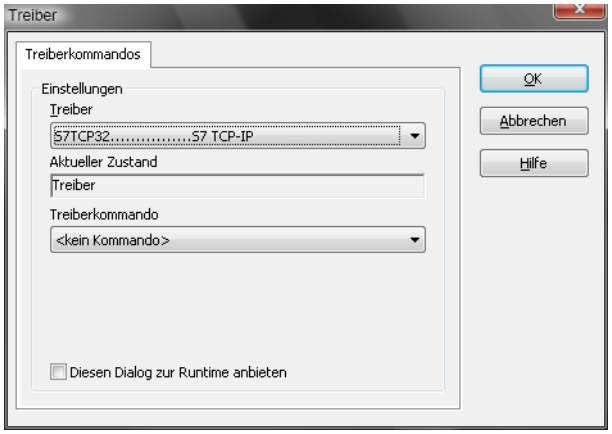
2.5 Treiberkommandos

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.

Treiberkommandos dienen dazu, Treiber über zenon zu beeinflussen, z. B. starten und stoppen.
Die Projektierung erfolgt über die Funktion **Treiber Kommandos**. Dazu:

- legen Sie eine neue Funktion an
- wählen Sie *Variablen -> Treiberkommandos*

- der Dialog zur Konfiguration wird geöffnet



Parameter	Beschreibung
Treiber	Dropdownliste mit allen im Projekt geladenen Treibern.
Aktueller Zustand	Fixer Eintrag, in aktuellen Versionen ohne Funktion.
Treiberkommando	Dropdownliste zur Auswahl des Kommandos.
▸ `Treiber starten (Online-Modus)`	Treiber wird neu initialisiert und gestartet.
▸ `Treiber stoppen (Offline-Modus)`	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 treiberspezifischer Kommandos. Öffnet Eingabefeld für die Eingabe eines Kommandos.
▸ `Treiber Sollwertsetzen aktivieren`	Sollwert setzen auf Treiber ist erlaubt.
▸ `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).
Diesen Dialog zur Runtime anbieten	Dialog wird zur Runtime für Änderungen angeboten.

TREIBERKOMMANDOS IM NETZWERK

Wenn sich der Rechner, auf dem die Funktion **Treiberkommandos** ausgeführt wird, im zenon Netzwerk befindet, werden zusätzliche Aktionen ausgeführt. Ein spezielles Netzwerkkommando wird vom Rechner zum Server des Projekts gesendet, der dann die gewünschte Aktion auf seinem Treiber durchführt. 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.

2.6 Fehleranalyse

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

2.6.1 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 (main.chm::/12464.htm), das mit zenon mitinstalliert wird. Sie finden es unter *Start/Alle Programme/zenon/Tools 7.50 -> Diagviewer.*

zenon Treiber protokollieren alle Fehler in LOG-Dateien. Der Standardordner für die LOG-Dateien ist der Ordner **LOG** unterhalb des Ordners `ProgramData`, zum Beispiel:

`%ProgramData%\COPA-DATA\LOG`. LOG-Dateien sind Textdateien mit einer speziellen Struktur.

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 (main.chm::/12464.htm).

2.6.2 Checkliste

Fragen und Hinweise zur Fehlereingrenzung:

ALLGEMEINE FEHLERSUCHE

- Ist die Steuerung an die Stromversorgung angeschlossen?
- Analyse mit Hilfe des Diagnosis Viewers (auf Seite 47):
  -> 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) -> Taste Eingabe 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. für Modbus: telnet 192.168.0.100 502) -> Taste Eingabe drücken .*

  Wird der Bildschirm schwarz, konnte eine Verbindung aufgebaut werden.
- 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 38).)

- 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

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?

TREIBER FÄLLT SPORADISCH AUS

Analyse mit Hilfe des Diagnosis Viewers (auf Seite 47):
-> Welche Meldungen werden angezeigt?