zenon
by COPA-DATA

zenon Treiber Handbuch

DNP3_NG

v.8.10

© 2019 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 DNP3_NG 5

3 Treiber-Historie 9

4 DNP3_NG - Datenblatt 10

5 Voraussetzungen 11

5.1 PC 11

6 Konfiguration 12

6.1 Anlegen eines Treibers 12

6.2 Einstellungen im Treiberdialog 15

6.2.1 Allgemein 16

6.2.2 Com 20

6.2.3 Verbindungen 21

7 Variablen anlegen 28

7.1 Variablen im Editor anlegen 28

7.2 Adressierung 32

7.3 Treiberobjekte und Datentypen 34

7.3.1 Treiberobjekte 34

7.3.2 Zuordnung der Datentypen 38

7.4 Variablen anlegen durch Import 44

7.4.1 XML Import 44

7.4.2 DBF Import/Export 45

7.4.3 Online-Import 51

7.4.4 Offline-Import 53

7.5 Kommunikationsdetails (Treibervariablen) 56

8 Treiberspezifische Funktionen 62

9 Funktion Treiberkommandos 66

10 Fehleranalyse 70

10.1 Analysetool 71

10.2 Checkliste 72

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 Support-Team, das 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 DNP3_NG

Treiber für das Protokoll nach IEEE1815 -Distributed Network Protocol (DNP3).

Der Treiber ist Master auf Protokollebene und unterstützt serielle Kommunikation mit mehrere Outstations, sowie IP-Kommunikation über TCP.

BEGRIFFSDEFINITIONEN

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

Begriff	Definition
Event Class	Eine Art der Gruppierung in der Outstation. Es gibt die **Event Classes** *1*,*2*, *3* und die **Static Class** *0*. Eine **Event Class** ist in der Regel einem Puffer zugeordnet, in dem Wertänderungen für konfigurierte Objektgruppen mit der konfigurierte Variation in der **Outstation** gespeichert werden. Ein Master kann mittels einer Lese-Anfrage für z.B. **Class** *1* alle Wertänderungen dieser **Event Class** von der **Outstation** anfordern. Die **Static Class** *0* hat eine eigene Aufgabe. Sie liefert bei einer Leseanfrage vom **Master** jeweils den letzten Wert für beinahe alle **Points** zurück. In der **Outstation** kann in der Regel konfiguriert werden, welche **Points** bzw. welche **Object Group** zu welcher **Event Class** gehören. Es gibt keine exakte Definition, allerdings wird in der Regel **Class** *1* für wichtige Meldungen verwendet, z. B. *Binary Inputs* für Alarme.
Eventpoll	Lese-Anfrage vom **Master** an die **Outstation**, wobei der **Master** die **Outstation** auffordert, nur die geänderte Werte für eine bestimmte **Event Class** oder für alle **Event Classes** (*1*, *2* und *3*) zu senden.
Integrity Poll	Lese Anfrage vom **Master** an die **Outstation** für die S**tatic Class** *0*. Der **Master** fordert mit dieser Anfrage ein Initialabbild aller **Points** an.
Master	Kontrollierende Station. Ein **Master** sendet Leseanfragen und Steueranfragen an die **Outstation**.
Objekt Group	Datentyp oder Datenart in der **Outstation** mit einer definierten Funktionalität. **Object Group** *30* steht z.B. für einen statischen Analogeingang. Eine Wertänderung für einen Analogeingang ist **Object Group** *32* zugeordnet. **Object Group** *40* dient z.B. zum Lesen eines Analogausgangs, **Object Group** *41* zum Schreiben auf einen Analogausgang.
Outstation	SPS oder RTU in DNP3. 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 **Object 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-

Begriff	Definition
	oder Statusänderung, ist **DNP Object**.
Unsolicited Response	Meldung von Wertänderungen aus einer **Event Class**, die spontan von der **Outstation** an den **Master** gesendet wird. Dazu muss der **Master** jedoch zuerst in der **Outstation Unsolicited Responses** aktivieren. Die **Outstation** muss **Unsolicited Responses** unterstützen und diese müssen für sie konfiguriert sein.
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. Oder, ob ein Zeitstempel gespeichert wird oder nicht, oder, ob Object Flags gespeichert werden oder nicht.

Weitere Informationen finden Sie in der Dokumentation zum **DNP3/IEEE1815-2012 Standard**. Diese Dokumentation können Sie von der IEEE erwerben. Sie bekommen auch Zugang, wenn Sie der **DNP3 users group** beitreten: http://www.dnp.org/ (http://www.dnp.org/).

DEVICE PROFILE

Ein Device Profile ist ein standard Dokument, das beschreibt, welche Funktionalität aus dem DNP3 Standard unterstützt wird. Für den DNP3_NG Treiber finden Sie das XML Device Profile nach Installation von zenon in den Ordner C:\ProgramData\COPA-DATA*zenonxxx**CommunicationProfiles**Dnp3**Driver**DNP3_NG.xml* (xxx entspricht der aktuellen zenon Versionsnummer).

INFORMATION ZU DNP332, DNP3_NG UND DNP3_TG TREIBER

Der DNP332 Treiber und der DNP3_NG Treiber werden ab der Version 7.20 durch den DNP3_TG Treiber ersetzt. Aus Kompatibilitätsgründen sind der DNP3_NG Treiber und der ältere DNP332 Treiber nach wie vor im Setup enthalten, werden jedoch per Default nicht länger in der Treiberauswahlliste angezeigt. Bestehende Projekte, die konvertiert werden, verwenden nach wie vor den ursprünglich projektierten Treiber. Der alte Treiber kann in konvertierten Projekte normal weiter verwendet werden. Ein Wechsel auf den neuen DNP3_TG Treiber ist jedoch auch möglich.

DNP332/DNP3_NG IN TREIBERLISTE ANZEIGEN

Wenn Sie den DNP332/DNP3_NG Treiber in einem neuen Projekt in 7.20 verwenden möchten, muss der Treiber wieder zu der Treiberliste hinzugefügt werden:

1. Starten Sie das Programm **Driverinfo.exe** vom zenon Installationsmedium, Unterordner *\AdditionalSoftware\COPA-DATA DriverXML Editor*.

2. Öffnen Sie mit dem Programm **Driverinfo** die Treiber XML-Datei.
   Beispiel: **TREIBER_DE.XML** aus dem Ordner *%CD_PROGRAMDATA7200%*
   Dabei steht **DE** für das Kürzel der Sprache im Editor und **7200** für die installierte Version 7.20.
3. Navigieren Sie zum Ordner *DNP3* und wählen Sie im Kontextmenü **New Driver**.
4. Geben Sie **DNP332** bzw. **DNP3_NG** in alle drei Felder des Dialogs ein und bestätigen Sie mit Klick auf **OK**.
5. Speichern Sie die Änderungen mit Klick auf das Symbol **Speichern** in der Symbolleiste.
   Ab diesem Zeitpunkt steht der Treiber im Editor wieder zur Auswahl.
6. Wiederholen Sie diese Schritte für jede Sprache, in der Sie den Editor verwenden.

AUF DEN NEUEN DNP3_TG TREIBER WECHSELN

Der DNP3_TG Treiber ist im Prinzip kompatibel mit dem alten DNP3_NG Treiber und den DNP332 Treiber. Mittels der Funktion **Treiber Austauschen** im Editor kann vom DNP332 Treiber oder vom DNP3_NG Treiber auf den DNP3_TG Treiber gewechselt werden. Nach dem Treiber Austausch müssen einige Einstellungen manuell neu gemacht werden. Einige Funktionen sind im DNP3_TG Treiber anders implementiert und erfordern eine Änderung der Projektierung.

Beachten Sie bei einem geplanten Treibertausch:

Allgemein:

- Machen Sie als erstes eine Projektsicherung
- Notieren Sie die Treiberkonfiguration des alten Treibers. Nach dem Treibertausch muss diese im neuen Treiber erneut eingegeben werden.
- Das Schreiben auf eine Variable vom Typ Frozen Counter führt kein Immediate Freeze No Ack mehr aus. Diese Funktionalität kann über eine Variable vom Typ Commando erreicht werden.

DNP332 auf DNP3_TG:

- Wenn Sie beim DNP332 Treiber Select Before Operate bei der Befehlsgabe verwenden, deaktivieren Sie die Eigenschaft **Select Before Operate** bei der Variable. Wählen Sie stattdessen für die Variablen vom Typ *Binary Ouput* und *Analog Output* bei der Eigenschaft **Command Mode** den Eintrag *Auto-SBO* aus.
- Falls Sie beim DNP332 Treiber Datentypen für **Analog Inputs** oder **Counter** verwendet haben, die es im DNP3_NG Treiber nicht mehr gibt, dann müssen Sie vor dem Wechsel die Datentypen der Variablen anpassen. Hintergrund: Mit dem DNP332 Treiber war es möglich, Datentypen zu wählen, die vom DNP3 Standard nicht vorgesehen waren. Im DNP3_NG können diese Datentypen nicht länger verwendet werden.

DNP3_NG auf DNP3_TG:

- Das Property **Command Mode** für Variablen vom Typ "*Analog Output*" und "*Binary Output*" muss manuell richtig gesetzt werden. Alternativ ist es möglich die Variablen vor dem Treiber

Austausch in XML zu exportieren und nach dem Austausch wieder zu importieren. Der DNP3_TG Treiber unterstützt neben Direct Operate und Auto SBO auch Direct Operate No Ack.

- Das Property Variation bei der Variable wurde entfernt. Anstelle der Variation bei der Variable zu konfigurieren kann die Variable vom Class Poll ausgenommen werden über den Checkbox **Classless Read** und mittels Benutzerdefiniertem Kommando mit der gewünschte Variation explizit gelesen werden.
- Der Variablentyp "Class Scan" existiert bei dem DNP3_TG nicht mehr. Löschen Sie am besten Variablen dieses Typs vor dem Treiber Austausch. Um einen Class Poll zu triggern, einen explizites Lesen zu triggern oder auch ein Cold Restart zu triggern, erstellen Sie vom DNP3_TG Treiber variablen vom Variablentyp "Command".
-

3 Treiber-Historie

Datum	Treiberversion	Änderung
09.11.12	3754	Treiberdokumentation wurde neu erstellt

TREIBERVERSIONIERUNG

Mit zenon 7.10 wurde die Versionierung der Treiber verändert. Ab dieser Version gibt es eine versionsübergreifende Build-Nummer. Das ist die Zahl an der 4. Stelle der Dateiversion.
Zum Beispiel: **7.10.0.4228** bedeutet: Der Treiber ist für Version **7.10**, Service Pack **0** und hat die Build-Nummer **4228**.

Erweiterungen oder Fehlerbehebungen werden zukünftig in einem Build eingebaut und sind dann ab der nächsthöheren Build-Nummer verfügbar.

Beispiel

Eine Treibererweiterung wurde in Build **4228** implementiert. Der Treiber, den Sie im Einsatz haben, verfügt über die Build-Nummer **8322**. Da die Build-Nummer Ihres Treibers höher ist als die Build-Nummer der Erweiterung, ist die Erweiterung enthalten. Die Versionsnummer des Treiber (die ersten drei Stellen der Dateiversion) spielen dabei keine Rolle. Die Treiber sind versionsunabsabhängig

4 DNP3_NG - Datenblatt

Allgemein:	
Treiberdateiname	DNP3_NG.exe
Treiberbezeichnung	DNP3 Treiber Neue Generation
Steuerungs-Typen	DNP3 / IEEE 1815 Outstations
Steuerungs-Hersteller	DNP3; GE Harris

Treiber unterstützt:	
Protokoll	TCP/IP; DNP3; IEEE Std 1815
Adressierung: Adress-basiert	Address based
Adressierung: Namens-basiert	--
Kommunikation spontan	X
Kommunikation pollend	X
Online Browsing	X
Offline Browsing	X
Echtzeitfähig	X
Blockwrite	--
Modemfähig	--
RDA numerisch	--
RDA String	--
Hysterese	X
erweiterte API	--
Unterstützung von Statusbit **WR-SUC**	X
alternative IP-Adresse	X

Voraussetzungen:	
Hardware PC	Serielle Schnittstelle; 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

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/IP

Für Dual Endpoint muss der konfigurierte Listening Socket entsprechend in der Firewall konfiguriert sein.

SOFTWARE

Wenn nicht bereits vorhanden, Treiber **DNP3_NG.exe** in das zenon Programmverzeichnis kopieren und sicherstellen, dass auch die **DNP3_NGV.dll** vorhanden ist.

CE

Treiber **DNP3_NG.dll** in das zenon CE-Programmverzeichnis kopieren. (Für die Runtime wird die DN3_NGV.dll nicht benötigt.)

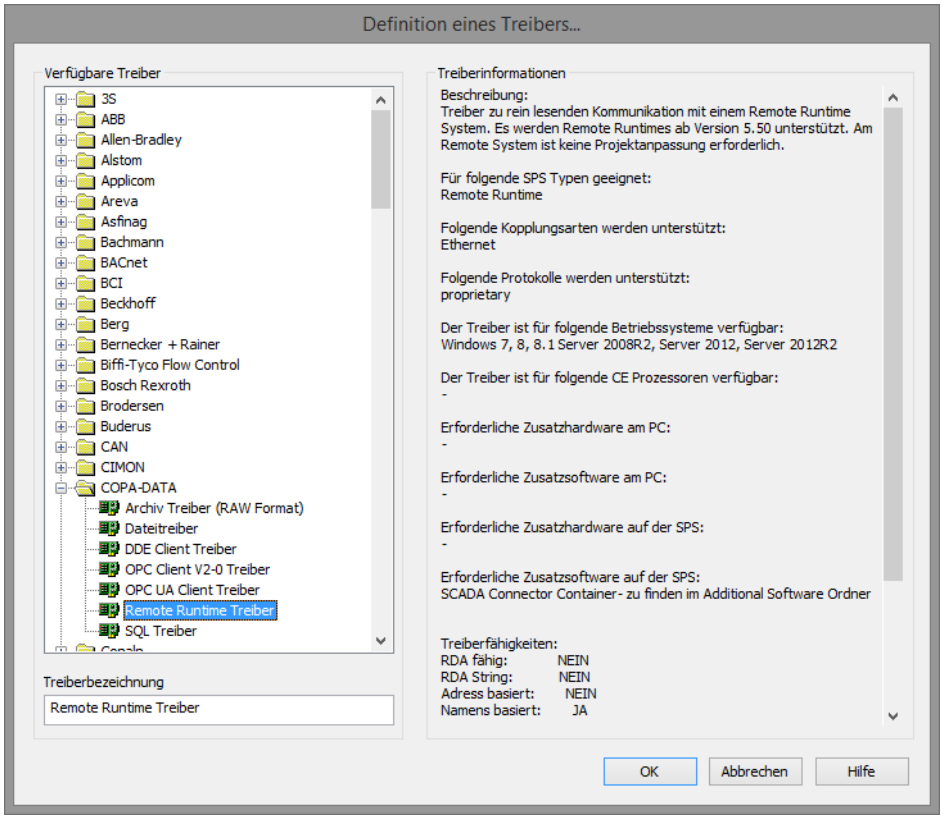
6 Konfiguration

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 (main.chm::/15247.htm) 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 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.

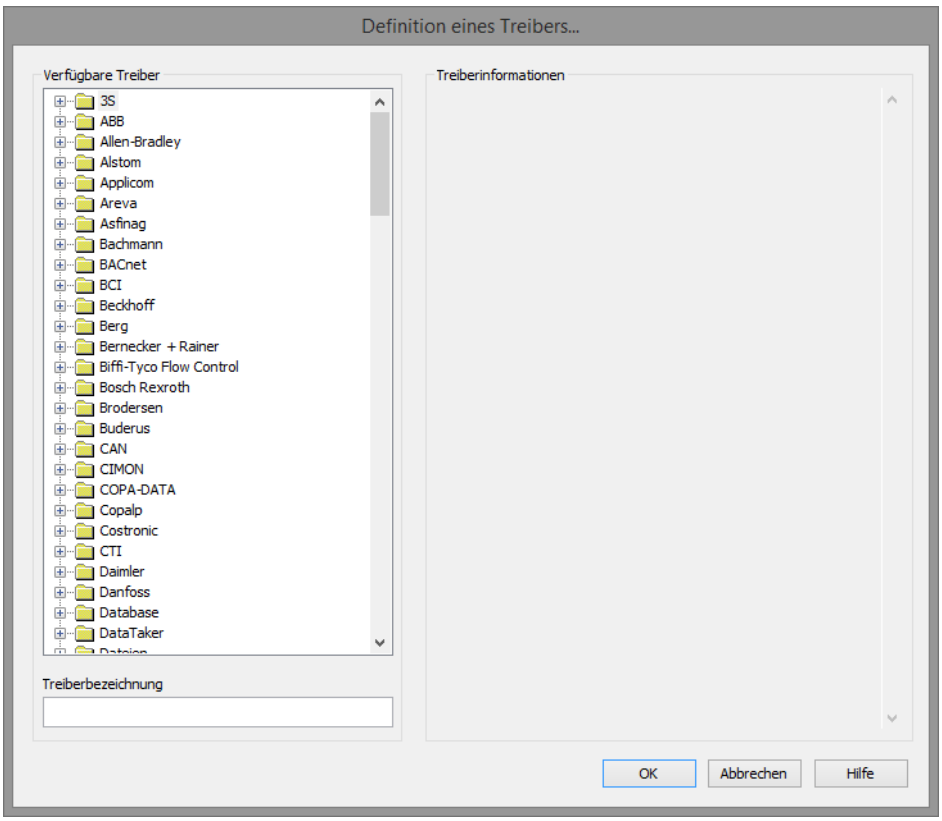
Info

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:

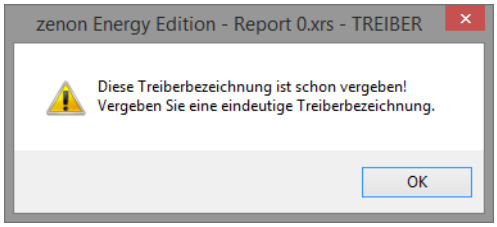
   - Die **Treiberbezeichnung** muss eindeutig sein.
     Wird ein Treiber mehrmals im Projekt verwendet, so muss jeweils eine neue Bezeichnung vergeben werden.
     Dies wird durch Klick auf die Schaltfläche **OK** evaluiert. Ist die Treiber im Projekt bereits vorhanden wird dies mit einem Warndialog angezeigt.
   - 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**

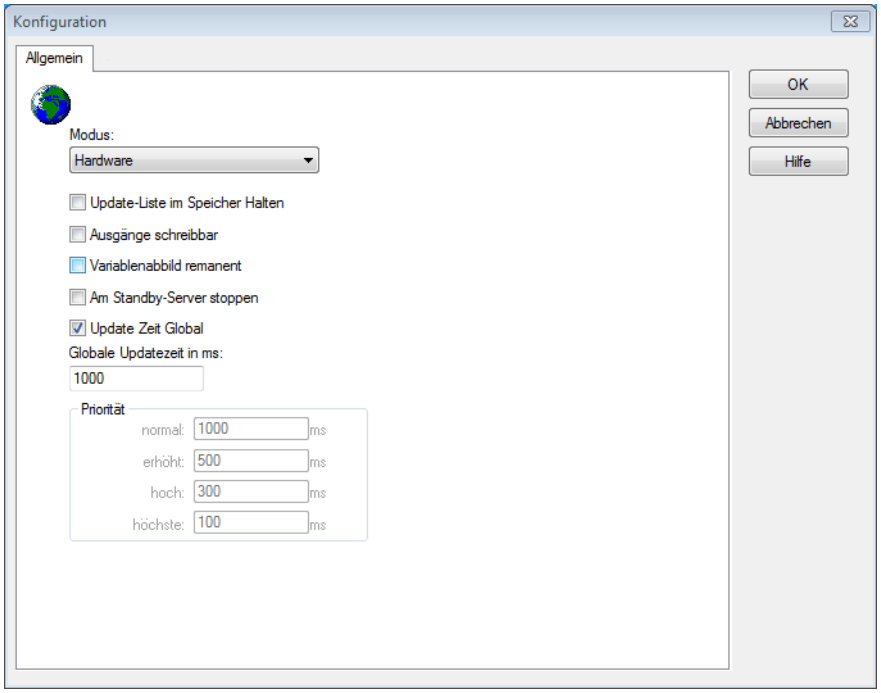
> **Info**
>
> In einem zenon Projekt müssen nur die benötigten Treiber vorhanden sein. Treiber können bei Bedarf zu einem späteren Zeitpunkt hinzugefügt werden.

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. (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 dieser Status aktiv ist: ▸ Benutzerstatus *M1* (*0*) bis *M8* (*7*) ▸ *REVISION(9)* ▸ *AUS(20)* ▸ *ERSATZWERT(27)* Das Variablenabbild wird immer gespeichert wenn:

Option	Beschreibung
	▸ 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. 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

Option	Beschreibung
	Variablen eingestellte Priorität wird nicht 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

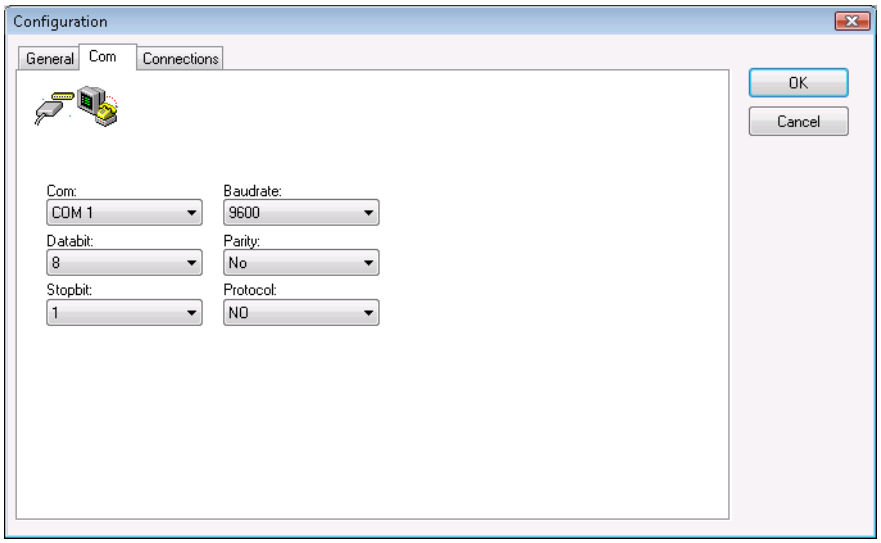
Option	Beschreibung
OK	**Übernimmt alle Änderungen in allen Registerkarten und schließt den Dialog.**
Abbrechen	Verwirft alle Änderungen in allen Registerkarten und schließt den Dialog.
Hilfe	Öffnet die Online-Hilfe.

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**.

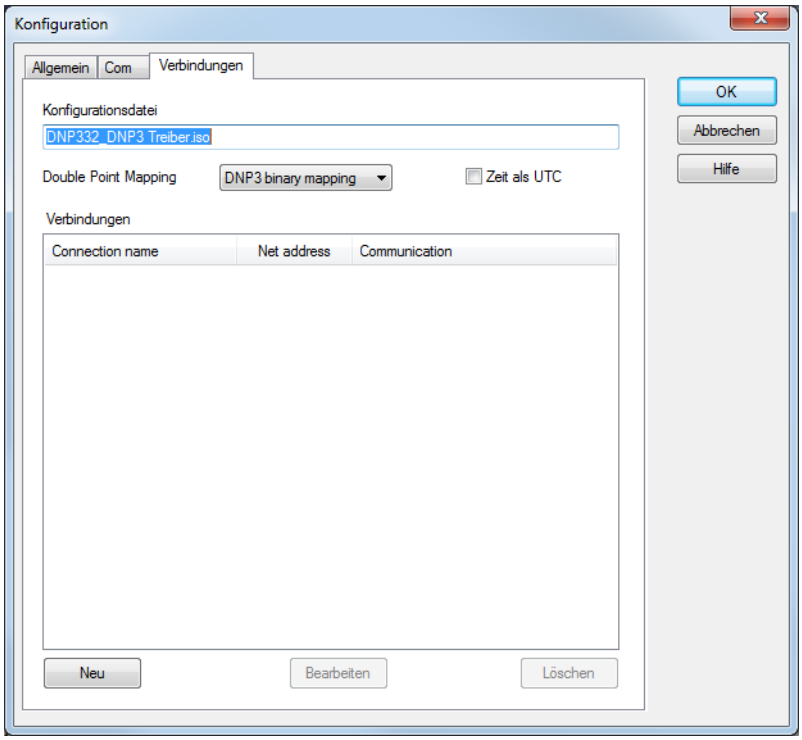
6.2.2 Com



Parameter	Beschreibung
Com	Auswahl Com-Port. Default: *1*
Baudrate	Auswahl Baudrate. Anpassen an Steuerung. Default: *9600*
Datenbits	Anzahl Datenbits. Anpassen an Steuerung. Default: *8*
Stopbit	Auswahl Stopbit. Anpassen an Steuerung. Default: *1*
Parity	Auswahl Parity. Anpassen an Steuerung. Default: *No*
Protocol	Auswahl Protokoll. Anpassen an Steuerung. Default: *No*

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

6.2.3 Verbindungen



Parameter	Beschreibung
Konfigurationsdatei	Name der Datei, in der die Verbindungsdaten gespeichert werden. Nur zur Information, nicht änderbar.
Double Point Mapping	Art des Double Point Mappings (auf Seite 28) auf einen Integerwert. Auswahl aus Dropdownliste: ▸ *SCADA default mapping* ▸ *DNP3 binary mapping* ▸ *Custom legacy mapping* Default: *DNP3 binary mapping*
Zeit als UTC	Alle Zeiten, von und zur Steuerung, werden als UTC und nicht als

Parameter	Beschreibung
	lokale Zeit behandelt.
Verbindungen	Zeigt die konfigurierten Verbindungen an.
Neu	Öffnet Dialog zum Anlegen einer neuen Verbindung (auf Seite 23).
Bearbeiten	Öffnet Dialog zum Bearbeiten der ausgewählten Verbindung.
Löschen	Löscht ausgewählte Verbindung.
OK	Übernimmt Änderungen an allen Registerkarten und schließt Dialog.
Abbrechen	Verwirft alle Änderungen und schließt Dialog.
Hilfe	Öffnet die Online-Hilfe.

Info

Maximale Anzahl der Verbindungen: 256 (0-255).

NEUE VERBINDUNG ANLEGEN

1. klicken Sie auf die Schaltfläche **Neu**
2. Tragen Sie die Verbindungsdetails ein
3. klicken Sie auf **Speichern**

VERBINDUNG BEARBEITEN

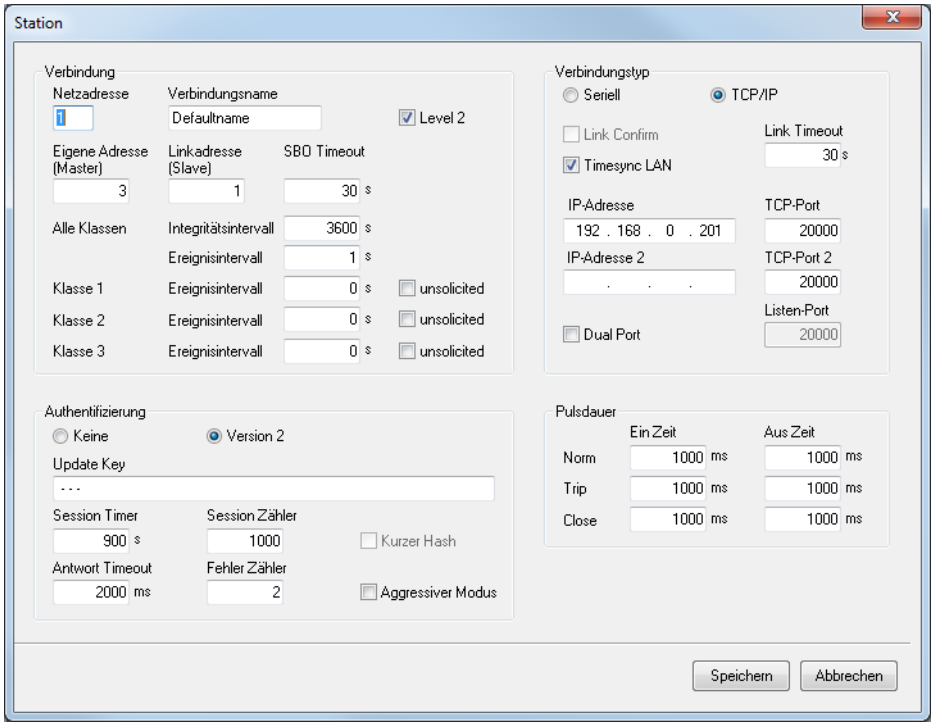
1. wählen Sie in der Verbindungsliste die gewünschte Verbindung
2. klicken Sie auf die Schaltfläche **Edit**
3. ändern Sie die Verbindungsparameter
4. schließen Sie mit **Speichern** ab

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

6.2.3.1 Konfiguration Stationen

Klick auf die Schaltfläche **Neu** im Konfigurationsdialog öffnet den Dialog zur Konfiguration der Station.



Parameter	Beschreibung
Verbindung	Verbindungseinstellungen
Netzadresse	Netzadresse der Verbindung. Wert zwischen *0* und *255*.
Verbindungsname	Name der Verbindung. Frei wählbar.
Level 2	**Aktiv: Befehle und Funktionen von Level 2** und **Level 1** können verwendet werden.
Eigene Adresse (Master)	Eigene Adresse (DNP3 Master-Treiber).
Linkadresse (Slave)	Linkadresse der Steuerung (des DNP3 Slaves).
SBO Timeout	Zeit in Sekunden, die auf eine Antwort der Steuerung auf das *Select* gewartet wird. Ist die Zeit abgelaufen, wird das *Select* als abgelehnt gewertet. Default: *30 s*
Alle Klassen	Integritätsintervall und Ereignisintervall für alle Klassen.

Parameter	Beschreibung
Integritätsintervall	Integrity Poll des Treibers in Sekunden. Default: *3600*
Ereignisintervall	Abfrage des Treibers auf neue Events in Sekunden. ▸ Wert: *0 bis 99999 s* ▸ 0: *kein Polling* Default: *1*
kein Auto-IIN	Zur Kompatibilität: Treiber ignoriert die Internal Indication Flags IIN1.1, IIN1.2 und IIN1.3. Stellen Sie sicher, regelmässig für Events zu pollen, um eventuelle Pufferüberläufe in der Outstation zu vermeiden.
Klasse 1	
Ereignisintervall	Für diese Klasse spezifischer Intervall für das Polling von Events in Sekunden. ▸ Wert: *0 bis 99999 s* ▸ 0: *kein Polling* Default: *0*
unsolicited	*Aktiv*: Für diese Klasse werden unsolicited Events akzeptiert. Siehe auch Kapitel Adressierung (auf Seite 32).
Klasse 2	
Ereignisintervall	Für diese Klasse spezifischer Intervall für das Polling von Events in Sekunden. ▸ Wert: *0 bis 99999 s* ▸ 0: *kein Polling* Default: *0*
unsolicited	*Aktiv*: Für diese Klasse werden unsolicited Events akzeptiert. Siehe auch Kapitel Adressierung (auf Seite 32).
Klasse 3	
Ereignisintervall	Für diese Klasse spezifischer Intervall für das Polling von Events in Sekunden. ▸ Wert: *0 bis 99999 s*

Parameter	Beschreibung
	▸ 0: *kein Polling* Default: *0*
unsolicited	*Aktiv*: Für diese Klasse werden unsolicited Events akzeptiert. Siehe auch Kapitel Adressierung (auf Seite 32).
Verbindungstyp	Art der Verbindung. Mögliche Auswahl über Radiobuttons: ▸ seriell ▸ TCP/IP
Seriell	*Aktiv*: serielle Verbindung wird verwendet
TCP/IP	*Aktiv*: Verbindung über TCP/IP wird verwendet
Link Confirm	*Aktiv*: Link Layer Confirmation ist aktiv. Nur für serielle Kommunikation verfügbar.
Link Timeout	Zeit in Sekunden für Timneout der Verbindung. Default: *30s*
Timesync LAN	*Aktiv*: Passende Zeitsynchronisation verwenden. Nur für TCP-Verbindung verfügbar. Verwendet die Variation 3 des Zeitobjekts. Diese wird nicht von allen Stationen akzeptiert (z. B. Brodersen RTU) und kann abgewählt werden.
IP-Adresse	IP- Adresse der Steuerung.
TCP-Port	Port über den kommuniziert wird. Default: *20000*
IP-Adresse 2	Backupadresse bei redundanter Anbindung dieser Steuerung
TCO-Port 2	TCP Port auf der Backupadresse
Dual Port	Aktiv: Dual Endpoint Kommunikation ist erlaubt.
Listen-Port	TCP Port, der als lokaler Dual-Endpoint verwendet wird. Nur aktiv, wenn Eigenschaft **Dual Port** aktiv ist.

Parameter	Beschreibung
Authentifizierung	Auswahl der Authentifizierung: ▸ keine ▸ Version 2
Keine	*Aktiv*: Es wird keine Authentifizierung verwendet.
Version 2	*Aktiv*: Authentifizierung Version 2.
Update Key	Der Authentifizierungs-Schlüssel für die gesicherte Kommunikation über Version 2. Erwartet 32 hexadezimale Digits. **Erlaubte Zeichen:** ▸ Ziffern: *0* bis *9* ▸ Buchstaben: *a* bis *f* und *A* bis *F* ▸ Sonderzeichen: Leerzeichen, Punkte und Doppelpunkte ▸ am Beginn: Header *0x* oder *0X* erlaubt ▸ alle übrigen Zeichen werden als *0* interpretiert und ergeben eventuell einen Fehler beim Erstellen des Sitzungsschlüssels.
Session Timer	Gültigkeitsdauer des Sitzungsschlüssels in Sekunden. Wert: *<700000 s* Default: *900 s*
Session Zähler	Vorgabe, wie oft ein Sitzungsschlüssel verwendet werden darf. Wert: *<10000 s* Default: *1000*
Antwort Timeout	Maximale Zeit bis zur Beantwortung einer Authentifizierungsanfrage in Millisekunden. Wert: *100 bis 120000 ms* Default: *2000 ms*
Fehler Zähler	Vorgabe, wie viele Fehler bei der Authentifizierung gemeldet werden.

Parameter	Beschreibung
	Wert: *0 bis 10* Default: *2*
Kurzer Hash	*Aktiv*: Bei serieller Kommunikation wird bei der Authentifizierung ein verkürzter Hash-Wert verwendet. **Hinweis:** Nur für serielle Verbindung und nur mit **Session Timer** bis *1800 s* (*30 Minuten*) zulässig.
Aggressiver Modus	*Aktiv*: Authentifizierung wird im aggressiven Modus durchgeführt.
Pulsdauer	Definiert Puls-Dauer für **Norm**, **Trip** und **Close** je Verbindung. **Norm**, **Trip** und **Close** legen fest, welches Relais geschaltet wird.
Norm	Entspricht NUL. Pulsdauer Norm: ▸ **Ein Zeit:** Zeitraum, in dem der Output auf EIN ist, in Millisekunden Default: *1000* ▸ **Aus Zeit** Zeitraum, in dem der Output auf AUS ist, in Millisekunden Default: *1000*
Trip	Pulsdauer Trip: ▸ **Ein Zeit:** Zeitraum, in dem der Output auf EIN ist, in Millisekunden Default: *1000* ▸ **Aus Zeit** Zeitraum, in dem der Output auf AUS ist, in Millisekunden Default: *1000*
Close	Pulsdauer Close: ▸ **Ein Zeit:** Zeitraum, in dem der Output auf EIN ist, in Millisekunden Default: *1000* ▸ **Aus Zeit** Zeitraum, in dem der Output auf AUS ist, in Millisekunden Default: *1000*

Parameter	Beschreibung
Speichern	Speichert Parameter für Verbindung und deaktiviert Editiermodus.
Abbrechen	Verwirft Änderungen und deaktiviert Editiermodus, ohne zu speichern.

6.2.3.2 Double Point Mapping

Double Point Mapping, abhängig von der Auswahl im Konfigurationsdialog (auf Seite 21):

Offset 15	Offset 14	DNP3 binary	Custom Legacy	SCADA default
0	*0*	*0*	*0*	*2*
0	*1*	*1*	*2*	*0*
1	*0*	*2*	*1*	*1*
1	*1*	*3*	*3*	*3*

7 Variablen anlegen

So werden Variablen im zenon Editor angelegt:

7.1 Variablen im Editor anlegen

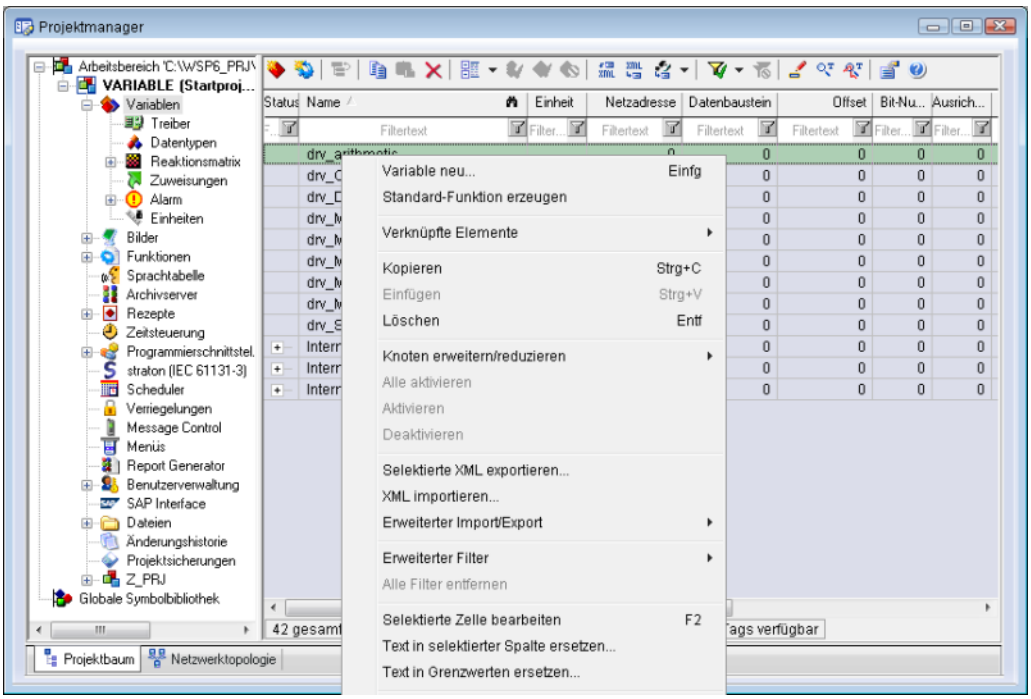
Variablen können angelegt werden:

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

DIALOG VARIABLE

Um eine neue Variable zu erstellen, gleich welchen Typs:

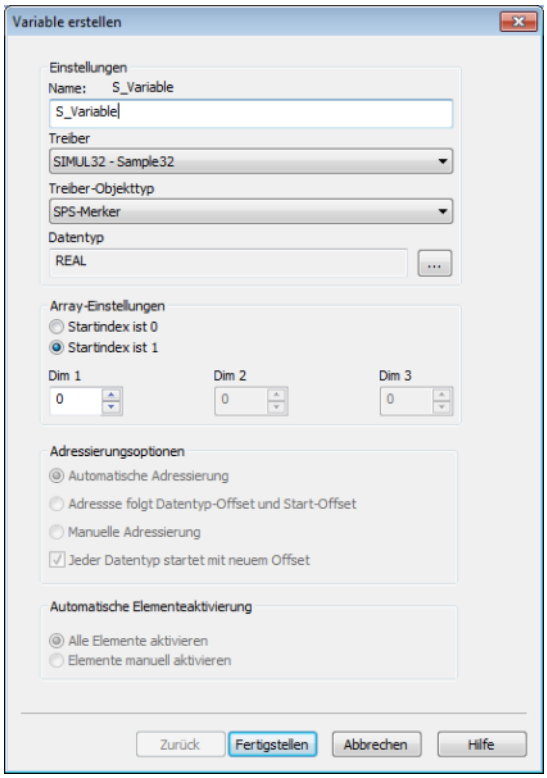
1. Wählen Sie im Knoten **Variablen** im Kontextmenü den Befehl **Variable neu**.



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** (**Main.chm::/Intern.chm::/Intern.htm**)) geladen.
Treiberobjekttyp (**cti.chm::/28685.htm**)	Wählen Sie aus der Dropdownliste den passenden Treiberobjekttyp aus.

Eigenschaft	Beschreibung
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.

Folgende Treiber unterstützen die **Symbolische Adresse**:

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

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 DNP Point Nummer. In Kombination mit dem Treiberobjekttyp wird die DNP Objekt Gruppe bestimmt.

Gruppe/Eigenschaft	Beschreibung
Allgemein	Gruppe mit allgemeinen Eigenschaften.
Name	Frei vergebbarer Name. **Achtung:** Je zenon Projekt muss der Name eindeutig sein.
Kennung	Frei vergebbare Kennung. Z. B. für **Betriebsmittelkennung**, Kommentar usw.
Adressierung	
Netzadresse	Netzadresse der Variablen. Diese Adresse bezieht sich auf die Netzadresse 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	DNP Point in der Outstation. Immer in Kombination mit dem Treiberobjekttyp zu sehen. Zum Beispiel: **Analog Input**, **Group** *20*, **Point** *0* entspricht hier **Offset** *0* mit einer Variable vom Typ *Analog Input*.
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/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.
Command Mode	Nur für *Binary Output* und *Analog Output*. Bestimmt, ob der Treiber ein *Direct operate* oder ein *Select Before Operate* durchführt. Default: *direct operate*
Variation	Nur für Variablen in Melde-Richtung, die entweder nicht im Klassenpoll vorhanden sind oder für die eine spezielle Variation gelesen werden sollen. Wird eine andere Variation, als die von der Outstation gemeldete Variation, gewünscht, muss bei der Variable die Eigenschaft **Variable ständig lesen** aktiviert werden. Variablen werden explizit, nicht optimiert, als klassenlose Variablen gelesen: ▸ nach jedem Integrity Poll oder ▸ manuell angestoßen über Offset *9* der Class Poll Variablen

KOMMUNIKATION

Die Kommunikation erfolgt vor allem pollend. Dabei fragt der Treiber die Outstation in konfigurierbaren Abständen nach Events (Wertänderungen und Statusänderungen) ab. Diesen sind in der Outstation Klassen (*1*, *2* oder *3*) zugeordnet.

Je nach Konfiguration in der Outstation kann als Antwort auch mehr als ein Wert für ein einzelnes DNP-Objekt gesendet werden. (**Sequence Of Events Buffer** im Gegensatz zu **Latest Value-in** der Oustation).

Der Treiber verarbeitet die empfangenen Werte und sendet deise weiter zu der Runtime.

Sofern die Outstation es unterstützt, können im Treiber auch pro Eventklasse **unsolicited** (auf Seite 23) responses aktiviert werden. Der Treiber sendet in diesem Fall am Ende der Startup-Routine, nachdem der Integrity Poll abgeschlossen worden ist, die Anforderung **unsolicited** responses für die jeweilige

Klasse zu aktivieren. Die Outstation kann anschließend ohne Aufforderung des Masters (Polling von Eventklassen) Wertänderungen an den Master senden. In diesem Fall ist es nicht unbedingt erforderlich, regelmäßig nach Events zu pollen. Der Polling-Zyklus kann dann höher oder überhaupt auf *0* (Polling für Eventklassen deaktiviert) eingestellt werden. Das Pollen für Events kann auch manuell angestossen werden von der Runtime aus (Class Poll Variablen).

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:

Treiberobjekttyp	Kanaltyp	Lesen	Schreiben	Unterstützte Datentypen	Kommentar
Analog Inputs	68	X	--	*REAL*	
Analog Output Status	69	X	X	*UDINT, DINT, REAL, UINT, INT*	
Binary Inputs	64	X	--	*BOOL*	
Binary Inputs Double	71	X	--	*USINT*	
Binary Output Statuses	65	X	X	*BOOL, USINT*	
Frozen Counters	67	X	X	*BOOL, UDINT, DINT, USINT, UINT, INT, SINT*	
Running Counters	66	X	X	*BOOL, UDINT, DINT, USINT, UINT, INT, SINT*	

Treiberobjekttyp	Kanaltyp	Lesen	Schreiben	Unterstützte Datentypen	Kommentar
String Data	70	X	X	*STRING*	
Kommunikationsdetails	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 Informationen dazu finden Sie im Kapitel Kommunikationsdetails (Treibervariablen) (auf Seite 56).

Legende:

X: wird unterstützt

--: wird nicht unterstützt

KOMMANDOS

BINARY OUTPUTS

Befehlsgabe mit *Binary Output* Variablen:

- Es dürfen ausschließlich Doppelkommandos verwendet werden.
- Die Eigenschaft **Select Before Operate** bei der Variable muss deaktiviert sein; der DNP3_NG 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

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*.

VERHALTEN VON COMMAND MODE

Für die Eigenschaft **Command Mode** bei einer Variablen gilt:

- *direct Operate:* Der Wert wird direkt geschrieben. (Default)
- *automatic SBO:* Beim Schreiben vom Stack wird zunächst ein Select gesendet, und nach positiver Antwort mit sofortigem *Operate* bestätigt.

Das gilt für binäre und analoge Outputs.

ZUORDNUNG VON SOLLWERSETZEN AUF BINARYOUTPUT OHNE BEFEHLSGABE

- Bei einer Binary Output Variable vom Datentyp BOOL wird *LatchON* für *High* oder *LatchOFF* für *Low* gesendet.
- Bei einer Binary Output Variable vom Datentyp USINT wird das Sollwert setzen gemäß untenstehende Tabelle behandelt

Wert USINT	Aktion	Kommentar
0	keine	
1	Pulse On	

Wert USINT	Aktion	Kommentar
2	Pulse Off	Nicht vollständig kompatibel. Wird nicht zwingend von der Outstation unterstützt.
3	Latch On	
4	Latch Off	
65	Close	
129	Trip	

Auch für das direkte Sollwertsetzen ohne Befehlsgabe wird die Eigenschaft **Command Mode** berücksichtigt.

REGELN FÜR DAS SETZEN DER STATUSBITS BEI EINSATZ DER BEFEHLSGABE

Statusbits werden entsprechend den Regeln in der Tabelle gesetzt:

- Sobald ein *Select* gesendet wurde, werden *SE* und *CoT_act* gesetzt.
- Trat beim Senden ein Fehler auf, oder wurde eine negative Antwort erhalten, werden *SE*, *P/N* und *CoT_actcon* gesetzt (*4*).
- Folgt statt einer Antwort ein Timeout, werden *SE*, *P/N* und *CoT_actterm* gesetzt (*5*).
- Nach einer positiven Antwort ist der Status *SE* und *CoT_actcon* (*1*, *2*, *3*). Nun wird automatisch das *Operate* gesendet und der Status auf *CoT_act* gesetzt. Tritt jetzt beim Senden ein Fehler auf, oder wurde eine negative Antwort erhalten, werden *P/N* und *CoT_actcon* gesetzt (*2*).
- Folgt statt einer Antwort ein Timeout oder ist die Antwort ein Timeout vom *Select*, werden *P/N* und *CoT_actterm* gesetzt (*3*).
- Ist die Antwort auf das *Operate* positiv, wird der Status erst auf *CoT_actcon* und dann auf *CoT_actterm* gesetzt (*1*).
- Der Status nach einem *DirectOperate* wird ebenso behandelt (*1*, *2*, *3*).
- Bei einem Fehler im Senden wird auch das Invalid-Bit gesetzt (*2*, *4*).

Fall	Aktion	Status	Erfolg	Status	Aktion	Status	Erfolg	Status	Status
1	*select*	*act SE*	*ack*	*actcon SE*	*operate*	*act*	*ack*	*actcon*	*actterm*
2	*select*	*act SE*	*ack*	*actcon SE*	*operate*	*act*	*nack*	*actcon P/N*	
3	*select*	*act SE*	*ack*	*actcon SE*	*operate*	*act*	*t/o*	*actterm P/N*	

Fall	Aktion	Status	Erfolg	Status	Aktion	Status	Erfolg	Status	Status
4	*select*	*act SE*	*nack*	*actcon SE P/N*					
5	*select*	*act SE*	*t/o*	*actterm SE P/N*					

Bedeutung der Begriffe in der Spalte **Erfolg**:

- ack = positiv
- nack = negativ
- t/o = Timeout

7.3.2 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
	BOOL	8
	USINT	9
	SINT	10
	UINT	2
	INT	1
	UDINT	4
	DINT	3
	ULINT	27
	LINT	26
	REAL	5
	LREAL	6
	STRING	12
	WSTRING	21
	DATE	18

Steuerung	zenon	Datenart
	TIME	17
	DATE_AND_TIME	20
	TOD (Time of Day)	19

TREIBER OBJEKTTYPEN UND UNTERSTÜTZTE IEC DATENTYPEN FÜR VARIABLEN IN ZENON

Treiber Objekttypen	Kanaltyp	Unterstützte Datentypen (Datenart)	Lesen	Schreiben	Kommentar
Binary Input *Group 1* *Event Group 2*	*64*	*BOOL*	X	--	
Binary Inputs Double	*71*	*USINT*	X	--	USINT-Mapping von zwei binären Status mit aufeinanderfolgenden Offsets. (kein Double-Bit Binary Input! Group 3 / 4)
Binary Output status *Group 10* *Event Group 11* *Command Group 12*	*65*	*BOOL*	--	X	Wert 1 : LATCH_ON Wert 0 : LATCH_OFF
Binary Output status *Group 10* *Event Group 11* *Comamnd 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

Treiber Objekttypen	Kanaltyp	Unterstützte Datentypen (Datenart)	Lesen	Schreiben	Kommentar
Running Counter *Group 20* *Event Group 22*	*66*	*UINT, UDINT*	X	--	
Frozen Counter *Group 21* *Event Group 23*	*67*	*UINT, UDINT*	X	X	Schreiben friert den Counter ein
Analog Input *Group 30* *Event Group 32*	*68*	*INT, DINT, REAL, LREAL*	X	--	
Analog Output status *Group 40* *Event Group 41* *Command Group 42*	*69*	*INT, DINT, REAL, LREAL*	--	X	geschriebener Wert wird nach erfolgreichem Schreiben als Rückmeldung gespiegelt
String Data *Group 110* *Event Group 111*	*70*	*STRING*	X	X	geschriebenen Wert wird nicht gespiegelt, somit erst nach einem Update aktualisiert.
Class Scans	73	*BOOL*	--	X	status INVALID bis Integrity Poll nach Treiberstart abgeschlossen ist. Klassenpoll Wert "1" wenn erfolgreich. Cold Restart: Wert "1" während Cold Restart ausgeführt wird.

Treiber Objekttypen	Kanaltyp	Unterstützte Datentypen (Datenart)	Lesen	Schreiben	Kommentar
Device Attributes Group 0	72	*UINT, STRING*	X	--	

DATENART

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

BEISPIEL BINARY INPUTS DOUBLE

Mit dem Treiberobjekttyp *Binary Inputs Double* führt der Treiber ein Mapping durch, abhängig von der Einstellung in der Treiberkonfiguration.

Zum Beispiel: Bei einem USINT Offset 14 werden die binären Status mit den Offsets 14 und 15 kombiniert.

Das Mapping zweier *Binary Inputs* auf ein *Binary Input Double* ist eine treiberinterne Funktion und darf nicht verwechselt werden mit der DNP3 *Gruppe 3/4*, *Double-Bit Binary Inputs*, die aktuell durch den Treiber nicht unterstützt werden.

ERGEBNISSE

Binäre Status-Bits SCADA default mapping	Binäre Status-Bits DNP3 binary logic mapping	Binäre Status-Bits Custom legacy mapping	Status in zenon (USINT mit *Binary Inputs Double)*
▸ beide Bits: *off (false)*	▸ erstes Bit (Offset 14): *off (false)* ▸ zweites Bit (Offset 15): *on (true)*	▸ erstes Bit (Offset 14): *on (true)* ▸ zweites Bit (Offset 15): *off (false)*	*2 - intermediate*
▸ beide Bits: *on (true)*	▸ beide Bits: *on (true)*	▸ beide Bits: *on (true)*	*3 - faulty*
▸ erstes Bit (Offset 14): *on (true)*	▸ beide Bits: *off (false)*	▸ beide Bits: *off (false)*	*0 - off*

Binäre Status-Bits SCADA default mapping	Binäre Status-Bits DNP3 binary logic mapping	Binäre Status-Bits Custom legacy mapping	Status in zenon (USINT mit *Binary Inputs Double)*
▸ zweites Bit (Offset 15): *off (false)*			
▸ erstes Bit (Offset 14): *off (false)* ▸ zweites Bit (Offset 15): *on (true)*	▸ erstes Bit (Offset 14): *on (true)* ▸ zweites Bit (Offset 15): *off (false)*	▸ erstes Bit (Offset 14): *off (false)* ▸ zweites Bit (Offset 15): *on (true)*	*1 - on*

DEVICE ATTRIBUTES

Device Attributes werden aktuell vom Treiber nur gelesen. Es wird immer der Index *1* verwendet (Standard-Satz an Device Attributes). Die **Variation** entspricht dabei dem Offset der Variable. Der Benutzer muss selbst den richtigen Datentyp wählen. Beim manuell Anlegen muss die Netzadresse entsprechend der Stationsnummer eingestellt werden.

Variation *254* (alle Device Attributes) kann gelesen werden, jedoch wird dieser Wert nicht direkt an der Variable in der Runtime gesendet. Stattdessen werden vorhandene **Device Attributes** automatisch mit der Antwort auf der Leseanforderung von **Variation** *254* aktualisiert.

BINARY OUTPUTS

Das Schreiben von *Binary Outputs* erfolgt immer über ein **CROB** (Gruppe 12), wahlweise mit *Direct Operate* oder *Select Before Operate*. 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.

Binary Output Command Events (Group 13) werden derzeit vom Treiber geparsed, allerdings nicht an die Runtime gesendet.

ANALOG OUTPUTS

Das Schreiben von Analog Outputs erfolgt immer über ein Objekt der Gruppe 41, wahlweise mit *Direct Operate* oder *Select Before Operate*. 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.

Analog Output Command Events (Group 43) werden derzeit vom Treiber geparsed, aber nicht an die Runtime gesendet.

CLASS SCANS

Variablen vom Treiberojbekttyp *Class Scan* sind Steuervariablen zum Beeinflussen des Treiberverhaltens, und werden nicht von der Outstation gelesen. *Class Scan* Variablen müssen manuell erstellt werden. Dabei muss die Netzadresse der Stationsadresse in der Treiberkonfiguration entsprechen. Der Offset der Variable bestimmt die Funktion die vom Treiber ausgeführt wird.

Class Scan Variablen haben nach dem Starten des Treibers in der Runtime den Status *INVALID* so lange der Integrity Poll vom Treiber noch nicht erfolgreich abgeschlossen ist. Sobald der *Integrity Poll* abgeschlossen ist, bekommen die Variable den Status *SPONT*.

Aktuell werden folgenden Offsets unterstützt:

- **Offset 0:**
  Beim Schreiben auf dieser Variable wird ein Integrity Poll ausgelöst (Read Request Group 60, Variation 2,3,4,1). Wenn das Lesen erfolgreich ist, erhält die Variable den Rückgabewert *1*. Im Anschluss werden eventuellen klassenlosen Variablen gelesen.
- **Offset 1:**
  Beim Schreiben auf dieser Variable wird ein Klasse 1 Poll ausgelöst (Read Request Group 60, Variation 2). Wenn das Lesen erfolgreich ist, erhält die Variable den Rückgabewert *1*.
- **Offset 2:**
  Beim Schreiben auf dieser Variable wird ein Klasse 2 Poll ausgelöst (Read Request Group 60, Variation 3). Wenn das Lesen erfolgreich ist, erhält die Variable den Rückgabewert *1*.
- **Offset 3:**
  Beim Schreiben auf dieser Variable wird ein Klasse 3 Poll ausgelöst (Read Request Group 60, Variation 4). Wenn das Lesen erfolgreich ist, erhält die Variable den Rückgabewert *1*.
- **Offset 9:**
  Beim Schreiben auf dieser Variable wird ein Lesen aller Klassenlosen Variablen ausgelöst. Wenn das Lesen erfolgreich ist, erhält die Variable den Rückgabewert *1*.
- **Offset 13:**
  Beim Schreiben auf dieser Variable wird vom Treiber ein Cold Restart Befehl zu der Outstation

gesendet (Function Code 13). Das in der Antwort von der Outstation enthaltene Objekt der Gruppe 52 (time delay) wird vom Treiber ausgewertet, und der Treiber sendet so lange keine neue Anfragen an der Oustation, wie in das Time Delay Objekt spezifiziert wurde. In der Runtime ist der Wert dieser Variable "0" so lange der Cold Restart aktiv ist, danach wird der Wert *1*.

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

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** (**main.chm::/13046.htm**).

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 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)

Bezeichnung	Typ	Feldgröße	Bemerkung
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	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,

Bezeichnung	Typ	Feldgröße	Bemerkung
			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

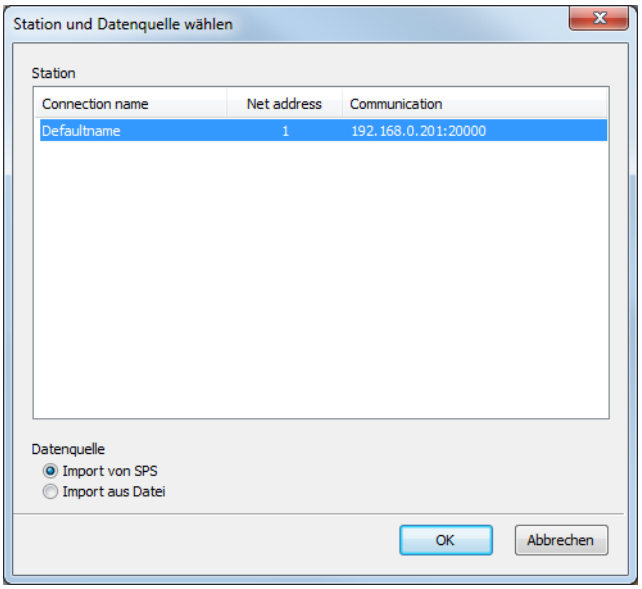
Bezeichnung	Typ	Feldgröße	Bemerkung
GRENZTXT1	C	66	Grenzwerttext
A_DELAY1	N	10	Zeitverzögerung
INVISIBLE1	L	1	Unsichtbar

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

7.4.3 Online-Import

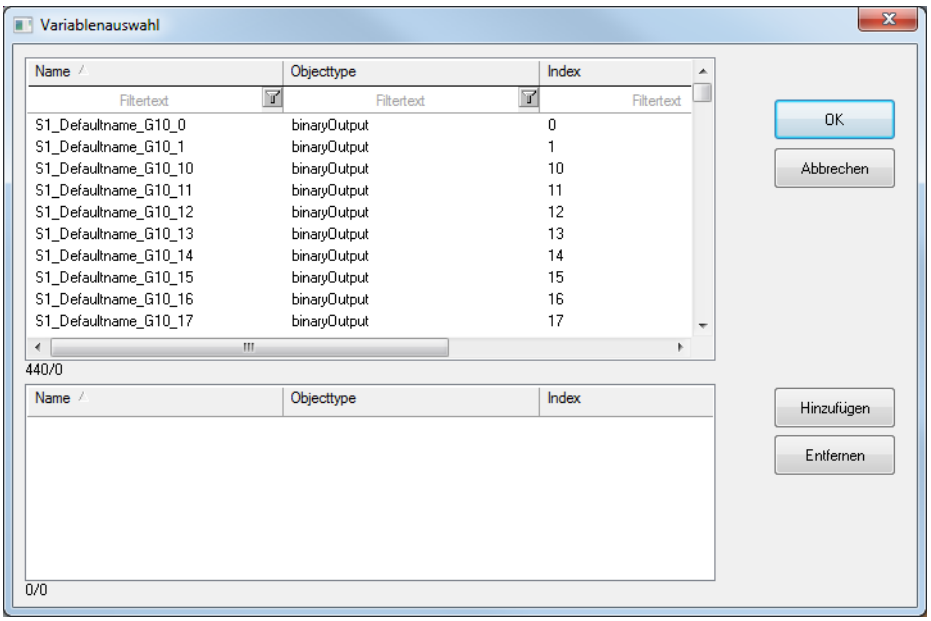
Um Variablen online von der Steuerung zu importieren:

1. wählen Sie den Treiber aus
2. wählen Sie in der Symbolleiste oder im Kontextmenü **Variablen vom Treiber importieren**
3. der Dialog für den Import wird geöffnet

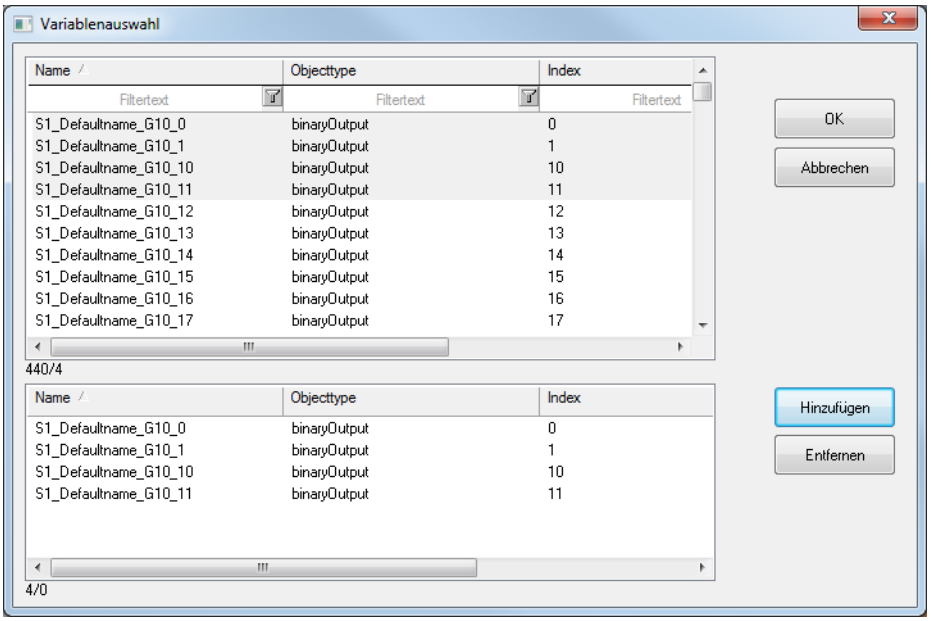


4. wählen Sie die gewünschte Verbindung
5. wählen Sie **Auswahl von SPS**
6. bestätigen Sie die Auswahl mit Klick auf **OK**

7. der Dialog zur Variablenauswahl wird geöffnet



8. wählen Sie die gewünschten Variablen (Mehrfachauswahl ist möglich)
9. fügen Sie die Variablen durch Klick auf die Schaltfläche **Hinzufügen** zur Liste der zu importierenden Variablen hinzu



10. mit Klick auf **Entfernen** können Sie Variablen auch wieder abwählen
11. starten Sie den Import durch Klick auf die Schaltfläche **OK**

Die ausgewählten Variablen werden beim Import automatisch im zenon Projekt erstellt und dem ausgewählten Treiber zugeordnet. Dabei wird die **Netzadresse** der Variablen entsprechend der ausgewählte Station aus der Treiberkonfiguration (auf Seite 23) konfiguriert.

REGELN FÜR DEN ONLINE-IMPORT

Beim Online-Import gilt:

- Die Antwort auf einen Integritypoll wird ausgewertet.
- Der Variablenname wird aus **Netzadresse**, Verbindungsname, Gruppennummer und Index gebildet.
- Die Kennung enthält **Netzadresse**, Index und Beschreibung des Objekttyps.
- Achten Sie darauf, dass die Runtime nicht aktiv ist wenn sie einen Online-Import starten, die Outstation unterstützt unter Umständen nur einen Master oder nur eine Verbindung vom gleichen Rechner.
- Achten Sie darauf die Antwort-Timeout in der Treiberkonfiguration entsprechend höher einzustellen, wenn Sie bei langsamer (serieller) Verbindung eine Outstation mit großer Point Database verwenden .

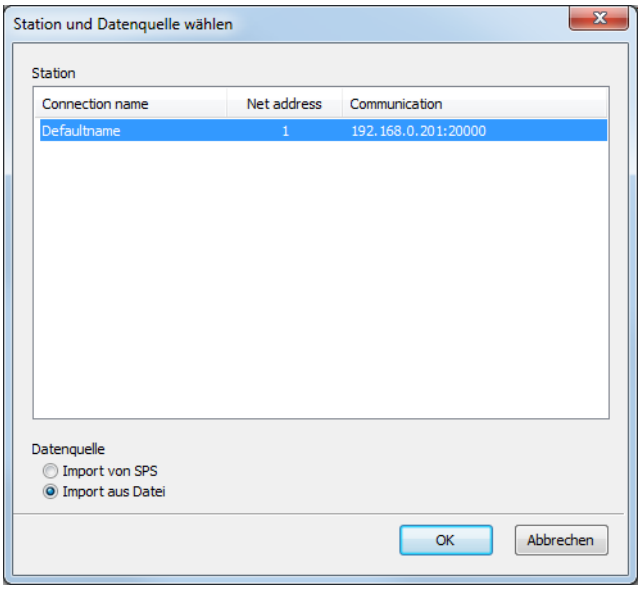
7.4.4 Offline-Import

Der Treiber unterstützt den Offline-Import von Variablen aus einer **DNP3 XML Device Profile** Datei für die Versionen:

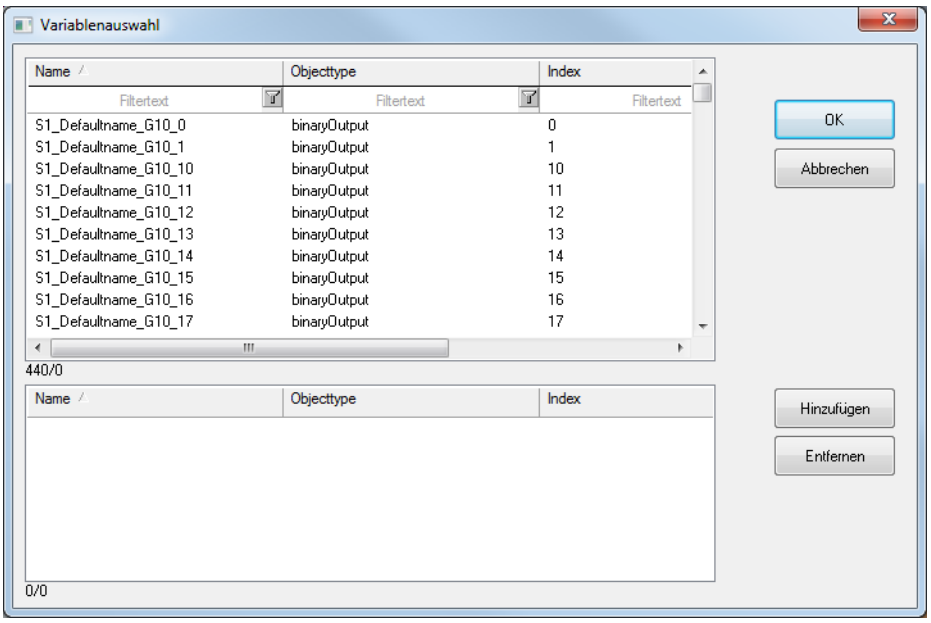
- 2.07 (Januar 2012)
- 2.08 (Juli 2012)

Um Variablen aus einer **DNP3 XML Device Profile** Datei zu importieren:

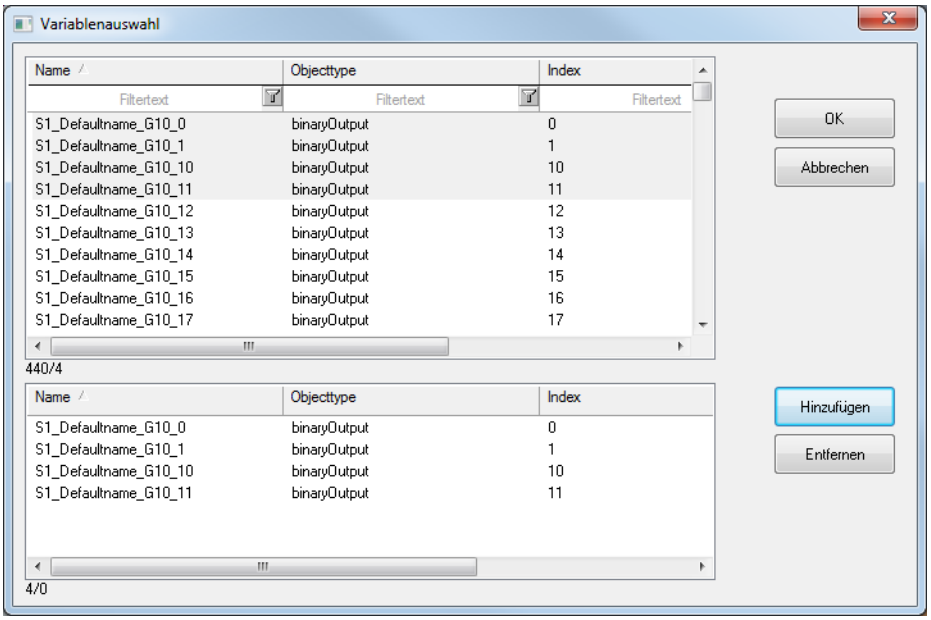
1. wählen Sie den Treiber aus
2. wählen Sie in der Symbolleiste oder im Kontextmenü **Variablen vom Treiber importieren**
3. der Dialog für den Import wird geöffnet



4. wählen Sie die gewünschte Verbindung
5. wählen Sie **Import aus Datei**
6. der Dialog zur Dateiauswahl wird geöffnet
7. wählen Sie die gewünschte Datei und bestätigen Sie die Auswahl mit Klick auf **OK**
8. der Dialog zur Variablenauswahl wird geöffnet



9. wählen Sie die gewünschten Variablen (Mehrfachauswahl ist möglich)
10. fügen Sie die Variablen durch Klick auf die Schaltfläche **Hinzufügen** zur Liste der zu importierenden Variablen hinzu



11. mit Klick auf **Entfernen** können Sie Variablen auch wieder abwählen

12. starten Sie den Import durch Klick auf die Schaltfläche **OK**

Die ausgewählten Variablen werden beim Import automatisch im zenon Projekt erstellt und dem ausgewählten Treiber zugeordnet. Dabei wird die **Netzadresse** der Variablen entsprechend der ausgewählte Station aus der Treiberkonfiguration (auf Seite 23) konfiguriert.

REGELN FÜR DEN OFFLINE-IMPORT

Beim Offline-Import gilt:

- Die Variablendefinition muss einen Namen enthalten.
- Der Variablenname wird aus einem zusammengesetzten Namen aus dem XML Device Profile Dokument nach folgendem Schema gebildet:

  Gerätename_Variablenname

  Es muss also sichergestellt sein, dass alle DNP Variablen, auch gruppenübergreifend, im Dokument einen eindeutigen Namen besitzen.
- Falls die Variablendefinition im Dokument ein Feld für **Description** enthält, wird diese Information beim Import in der **Kennung** der Variablen gespeichert. Der Variablenname und die Variablenkennung können nach dem Import geändert werden.
- Ist im Projekt bereits eine Variable mit dem gleichen Namen vorhanden, erhalten Sie beim erneuten Import eine Fehlermeldung. Die Variable wird nicht überschrieben oder gemerged. Diese Fehlermeldung kann auch angezeigt werden, wenn:
  - im **XML Device Profile** Dokument keine eindeutigen Namen verwendet wurden
  - wenn der Gerätename im Dokument identisch mit bereits importierten Variablen ist
- Es werden nur Variablen aus dem **XML Device Profile** Dokument zum Import angeboten, die vom Treiber unterstützt werden.
- **Frozen Counters** sind im **XML Device Profile** Dokument nicht explizit vorhanden. Wenn jedoch beim Counter der Wert für **frozenCounterExists** auf *True* (bzw. *1*) gesetzt ist, wird beim Import auch die Möglichkeit angeboten, Variablen für **Frozen Counter** zu importieren.

NICHT IMPORTIERTE VARIABLEN

Folgende Variablen werden nicht importiert und müssen manuell erstellt werden:

- Device attributes
- Steuervariablen für Klassenpolls, klassenloses Lesen und Cold Restart.
- Binary inputs double

Achten Sie beim manuellen Erstellen dieser Variablen auf die korrekte **Netzadresse**.

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

Name aus Import	Typ	Offset	Erklärung
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.
ConnectionStates	*STRING*	61	Interner Verbindungsstatus des Treibers zur

Name aus Import	Typ	Offset	Erklärung
			SPS. Verbindungszustände: *0:* Verbindung OK *1:* Verbindung gestört *2:* Verbindung simuliert Formatierung: *<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

Name aus Import	Typ	Offset	Erklärung
			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,

Name aus Import	Typ	Offset	Erklärung
			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).

Name aus Import	Typ	Offset	Erklärung
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.

Name aus Import	Typ	Offset	Erklärung
ErrorFile	*STRING*	15	Name der Fehlerprotokolldatei.

8 Treiberspezifische Funktionen

Dieser Treiber unterstützt folgende Funktionen:

Parameter	Beschreibung
Blockwrite	Nicht unterstützt.
Redundanz	Wenn die Outstation mehrere Master unterstützt
RDA	Nicht unterstützt. **Sequence Of Events** wird unterstützt, wenn in der Outstation aktiviert.
Echtzeitstempelung	Ja, bei entsprechend konfigurierter Variation in der Outstation.
Browsen	Online und Offline.
Polling	Polling für Eventklassen, in konfigurierbaren Zyklen oder manuell gesteuert.
Spontan	Ja. Nur Wertänderungen werden übertragen.
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 Diagnosis Server.

KOMMUNIKATIONSDETAILS

Treiberstatistikvariablen werden von dem DNP3_NG Treiber nicht gesetzt. Kommunikationsüberwachung ist über das INVALID statusbit möglich.

ERWEITERTE FEHLERDATEI

Der Treiber unterstützt erweitertes Logging am Diagnosis Server. Die Konfiguration erfolgt über den Diagnosis Viewer.

INTEGRITY POLL

Nach dem Start des Treibers wird automatisch als erstes ein Integrity Poll vom Treiber an die Outstation gesendet (**Read Request Group** *60*, **Variation** *2,3,4* und *0*, **Range** *all*). Die empfangenen Werte stehen entsprechend in der Runtime zur Verfügung. Achten Sie darauf, dass die Outstation unter Umstände eine andere Variation als Antwort auf den Integrity Poll sendet als bei einer normale Wertänderung. (Mit/ohne Zeitstempel, mit/ ohne Flags.)

Über die Steuervariable vom Typ *Klassenpoll* kann in der Runtime auch explizit ein Integrity Poll angestoßen werden. Wird der Treiber mittels Treiberkommando-Funktionen gestoppt und gestartet, löst das ebenfalls einen Integrity Poll aus.

INTERNAL INDICATIONS (IIN)

Die Outstation kann ihren Status über **Internal Indication** Statusbits dem Master mitteilen.

Der DNP3_NG Treiber wertet **Internal Indication** Bits wie folgt aus:

- *IIN 1.1 CLASS_1_EVENTS:*
  Die Outstation setzt dieses Bit, wenn der Eventbuffer für Klasse 1 Events weitere DNP Objekte beinhaltet, die nicht in der aktuelle Antwort enthalten sind. Der DNP3_NG Master reagiert auf dieses Bit, indem sofort eine Leseanfrage für Gruppe 60, Variation 2,3,4 an die Outstation gesendet wird. Wenn die Outstation dieses Statusbit häufig setzt, kann es zu einem höheren Lesezyklus, als in der Treiberkonfiguration definiert, führen.
- *IIN 1.2 CLASS_2_EVENTS:*
  Die Outstation setzt dieses Bit, wenn der Eventbuffer für Klasse 2 Events weitere DNP Objekte beinhaltet, die nicht in der aktuellen Antwort enthalten sind. Der DNP3_NG Master reagiert auf dieses Bit, indem sofort eine Leseanfrage für Gruppe 60, Variation 2,3,4 an die Outstation gesendet wird. Wenn die Outstation dieses Statusbit häufig setzt, kann es zu einem höheren Lesezyklus, als in der Treiberkonfiguration definiert, führen.
- *IIN 1.3 CLASS_3_EVENTS:*
  Die Outstation setzt dieses Bit, wenn der Eventbuffer für Klasse 3 Events weitere DNP Objekte beinhaltet, die nicht in der aktuelle Antwort enthalten sind. Der DNP3_NG Master reagiert auf dieses Bit, indem sofort eine Leseanfrage für Gruppe 60, Variation 2,3,4 an die Outstation gesendet wird. Wenn die Outstation dieses Statusbit häufig setzt, kann es zu einem höheren Lesezyklus, als in der Treiberkonfiguration definiert, führen.
- *IIN 1.4 NEED_TIME:*
  Die Outstation setzt dieses Bit, wenn eine Zeitsynchronisierung vom Master verlangt wird. Der

DNP3_NG Master reagiert sofort und antwortet der Outstation abhängig von der gewählte Zeitsynchronisierungsoption in der Treiberkonfiguration.

- *IIN 1.7 DEVICE_RESTART:*
  Wird von der Outstation bei einem Neustart gesetzt. Wenn ein Cold Restart vom DNP3_NG Master an die Outstation gesendet wird, setzt die Outstation dieses Bit. Es wird vom DNP3_NG Master in diesem Prozess zurück gesetzt.
- *IIN 2.2 PARAMETER_ERROR:*
  Wird von der Outstation gesetzt, wenn z.B. explizit ein Objekt gelesen wird, das in der Outstation mit dieser DNP3-Index nicht vorhanden ist. Der DNP3_NG Treiber setzt in diesem Fall bei der Variable in der Runtime das INVALID Statusbit.

Wenn die Option **NO_AUTO_IIN** in der Treiberkonfiguration bei der Station aktiv ist, ignoriert der Treiber die **Internal Indication** Flags *IIN1.1*, *IIN1.2* und *IIN1.3*. Der Treiber sendet keine automatische Lese-Anfrage für Gruppe 60, Variation 2, 3, 4. Diese Option kann aktiviert werden, um Kompatibilitätsprobleme zu umgehen, wenn die Outstation eines dieser Flags nicht zeitgerecht zurücksetzt, was dazu führt, dass der Treiber ausschließlich Event Polls sendet.

In der Regel setzt die Outstation diesen Flags um den Master nochmals zum Lesen aufzufordern, weil weiteren Daten verfügbar sind und um somit eventuell einen Pufferüberlauf in der Outstation vermeiden zu können. Achten Sie darauf, wenn Sie diese Option aus Kompatibilitätsgründen aktivieren, und stellen Sie sicher, dass der Treiber in regelmäßigen Intervallen die Outstation für Events pollt.

DNP3 OBJECT FLAGS MAPPING

Derzeit werden die DNP Objects Flags **ONLINE** und **COMM_LOST** vom DNP3_NG 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.

ZEITSYNCHRONISIERUNG

Der DNP3_NG 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 DNP3_NG Treiber die aktuelle Systemzeit entsprechend den Optionen für UTC/lokale Zeit und LAN Zeitsynchronisierung.

Es ist derzeit nicht möglich, an die Outstation zyklisch eine Zeitsynchronisierung ohne Aufforderung zu senden.

Achten Sie darauf, dass 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 DNP3_NG Master Station mit der gleiche Zeitquelle synchronisiert wird.

Das DNP3 Protokoll sieht nicht vor, dass der Master seine eigene Zeit mit der Zeit der Outstation synchronisiert.

SELECT UND CANCEL

Der DNP3_NG Treiber beantwortet ein Select und Cancel sofort positiv mit passender COT. Das Execute wird nach einem Select mit angepasster COT durchgeführt. Automatisches Select und Execute werden dabei bevorzugt behandelt.
Desweiteren haben Select, Cancel und Execute ein zusätzliches Statusbit. Dadurch ist eine saubere Laufzeitüberwachung bei projektiertem Routing möglich.

DNP3 SECURE AUTHENTICATION V2

SELECT BEFORE OPERATE - DIRECT OPERATE

Der DNP3_NG Treiber verwendet wahlweise **Direct operate** (Default) oder **Select Before Operate** für das Schreiben von Variablen vom Typ *Binary Output Status* oder *Analog Output*. Die Konfiguration erfolgt über die treiberspezifische Variableneigenschaft . **Command Mode**.

Achten Sie darauf, dass Sie bei der Variable die Eigenschaft **Select Before Operate** inaktiv bleibt! Diese Eigenschaft ändert das Verhalten der zweistufigen Befehlsgabe und ist nicht kompatibel mit dem DNP3 Standard. Wenn diese Eigenschaft doch aktiv ist:

- blockieren Sie die Befehlsgabe für weitere Befehle, wenn *Direct operate* bei der Variable gesetzt ist
- führen Sie den Befehl bei der erste Stufe und auch bei der zweite Stufe durch, wenn *Auto-SBO* bei der Variable gesetzt ist

Hinweis: Beim DNP332 Treiber kann nur über diese Option ein **Select Before Operate** durchgeführt werden. Dies ist allerdings nicht konform mit den im DNP3 Standard definierten Regeln für **Select Before Operate**.

DNP3 SEQUENTIAL FILE TRANSFER

DNP3 sequential file transfer wird aktuell vom DNP3_NG driver nicht unterstützt.

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.

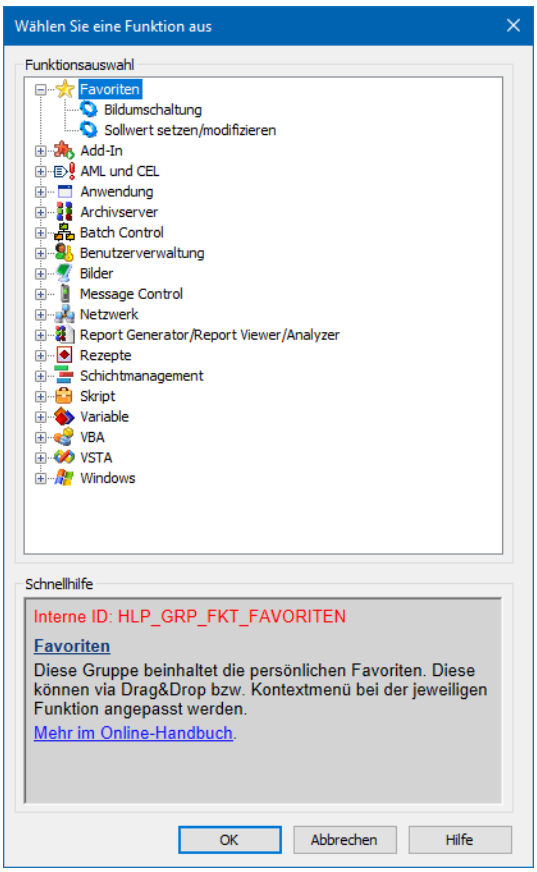
> **Achtung**
>
> Die zenon Funktion **Treiberkommandos** ist nicht ident mit den Treiberkommandos, die bei Energy-Treibern in der Runtime ausgeführt werden können!

PROJEKTIERUNG DER FUNKTION

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

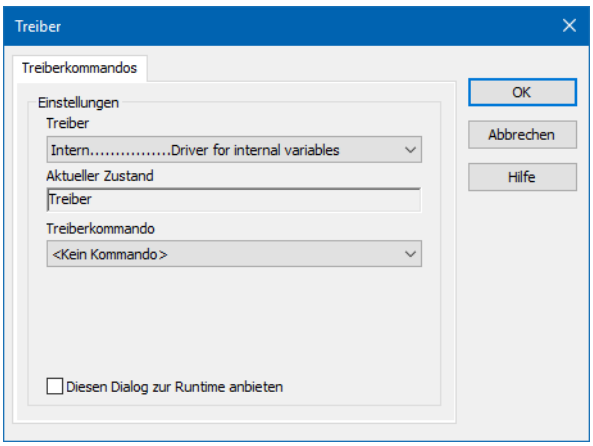
1. Legen Sie im zenon Editor eine neue Funktion an.

Der Dialog zur Auswahl einer Funktion wird geöffnet.



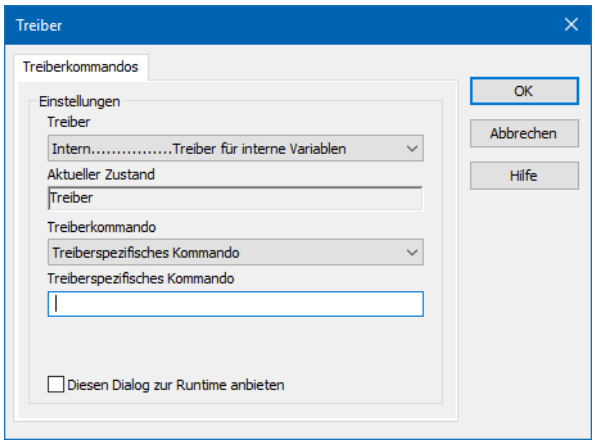
2. Navigieren Sie zum Knoten **Variable.**
3. Wählen Sie den Eintrag **Treiberkommandos**.

   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.

Treiberkommando	Beschreibung
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 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 (main.chm::/12464.htm), das mit zenon mitinstalliert wird. Sie finden es unter **Start/Alle Programme/zenon/Tools 8.10 -> 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 (main.chm::/12464.htm).

10.2 Checkliste

Fragen und Hinweise zur Fehlereingrenzung:

ALLGEMEINE FEHLERSUCHE

- Ist die Steuerung an die Stromversorgung angeschlossen?
- Analyse mit Hilfe des Diagnosis Viewers (auf Seite 71):
  -> 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: Variablen vom Treiberobjekttyp **Kommunikationsdetails** sind reine Statistikvariablen und kommunizieren nicht mit der Steuerung.
  Weitere Informationen dazu finden Sie im Kapitel Kommunikationsdetails (Treibervariablen) (auf Seite 56).
- 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.
- Class Poll Variablen sind nach dem Treiberstart INVALID so lange der Integritypoll noch nicht abgeschlossen ist.

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 DNP3-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 Diagnosis Viewers (auf Seite 71):
-> 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.