# zenon Treiber Handbuch

## OPCUA32

v.8.00

©2018 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. OPCUA32 ..... 5**

**3. OPCUA32 - Datenblatt ..... 6**

**4. Treiber-Historie ..... 7**

**5. Voraussetzungen ..... 8**

5.1 PC ..... 8

5.2 Steuerung ..... 9

**6. Konfiguration ..... 9**

6.1 Anlegen eines Treibers ..... 9

6.2 Einstellungen im Treiberdialog ..... 13

6.2.1 Allgemein ..... 14

6.2.2 Connections ..... 18

**7. Variablen anlegen ..... 32**

7.1 Variablen im Editor anlegen ..... 32

7.2 Adressierung ..... 35

7.3 Treiberobjekte und Datentypen ..... 37

7.3.1 Treiberobjekte ..... 37

7.3.2 Zuordnung der Datentypen ..... 37

7.4 Variablen anlegen durch Import ..... 38

7.4.1 XML Import ..... 39

7.4.2 DBF Import/Export ..... 40

7.4.3 Online-Import ..... 46

7.5 Kommunikationsdetails (Treibervariablen) ..... 47

**8. Treiberspezifische Funktionen ..... 52**

8.1 DataChangeFilter ..... 53

8.2 Konfigurierbare DataChangeTrigger ..... 53

8.3 Mehrere Subscriptions ..... 54

8.4 Schreiben von Arrays ..... 54

**9. Funktion Treiberkommandos ..... 55**

**10. Fehleranalyse ..... 58**

10.1 Analysetool ..... 58

10.2 Checkliste ..... 59

# 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. OPCUA32

Der **OPCUA-Treiber** dient zur Kommunikation mit OPC UA Servern und basiert auf dem offiziellen Stack der OPC Foundation. OPC UA ist die Abkürzung für **OPC Unified Architecture**.

Hauptmerkmale des Treibers:

- Die Kommunikation erfolgt spontan, d.h. geänderte Variablen werden automatisch vom Server gemeldet.

- Der Treiber unterstützt mehrere Server
- Die Variablen können direkt aus dem Server gelesen werden.
- Die Adressierung der Variablen geschieht durch den im Standard vorgesehenen **Browse Name**

> **Info**
>
> *Der OPCU UA Treiber für zenon 800 verwendet die Version 1.02-336-1 des Ansi C Stacks.*

# 3. OPCUA32 - Datenblatt

| **Allgemein:** | |
|---|---|
| Treiberdateiname | OPCUA32.exe |
| Treiberbezeichnung | OPC UA Client Treiber |
| Steuerungs-Typen | Alle OPC-UA Server mit Data Access Kommunikation |
| Steuerungs-Hersteller | OPC; straton; COPA-DATA; |

| **Treiber unterstützt:** | |
|---|---|
| Protokoll | OPC-UA; |
| Adressierung: Adress-basiert | -- |
| Adressierung: Namens-basiert | X |
| Kommunikation spontan | X |
| Kommunikation pollend | -- |
| Online Browsing | X |
| Offline Browsing | -- |
| Echtzeitfähig | X |
| Blockwrite | -- |
| Modemfähig | -- |

| | |
|---|---|
| Serielles Logging | -- |
| RDA numerisch | -- |
| RDA String | -- |
| Hysterese | X |
| erweiterte API | X |
| Unterstützung von Statusbit **WR-SUC** | X |
| alternative IP-Adresse | -- |

| **Voraussetzungen:** | |
|---|---|
| Hardware PC | -- |
| Software PC | -- |
| Hardware Steuerung | -- |
| Software Steuerung | -- |
| Benötigt v-dll | X |

| **Plattformen:** | |
|---|---|
| Betriebssysteme | Windows CE 6.0, Embedded Compact 7; Windows 7, 8, 8.1, 10, Server 2008R2, Server 2012, Server 2012R2, Server 2016; |
| CE Plattformen | x86; ARM; |

# 4. Treiber-Historie

| Datum | Treiberversion | Änderung |
|---|---|---|
| 07.07.08 | 100 | 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*

# 5. Voraussetzungen

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

## 5.1 PC

Der Treiber unterstützt Verbindungen über die Standard-Netzwerkkarte des PCs. Damit PC und Steuerung kommunizieren können:

- müssen sich Steuerung und PC im selben Netzwerkbereich befinden
- müssen die Subnet-Masken auf beiden Geräten entsprechend konfiguriert sein
- muss sich die Treiberdatei **OPCUA32.exe** im aktuellen zenon Installationsordner befinden

## 5.2 Steuerung

Die Steuerung muss das OPC Unified Architekture Protokoll mit dem OPC Binary Transport unterstützen.

**Hinweis:** OPC UA Webservices werden nicht unterstützt.

# 6. Konfiguration

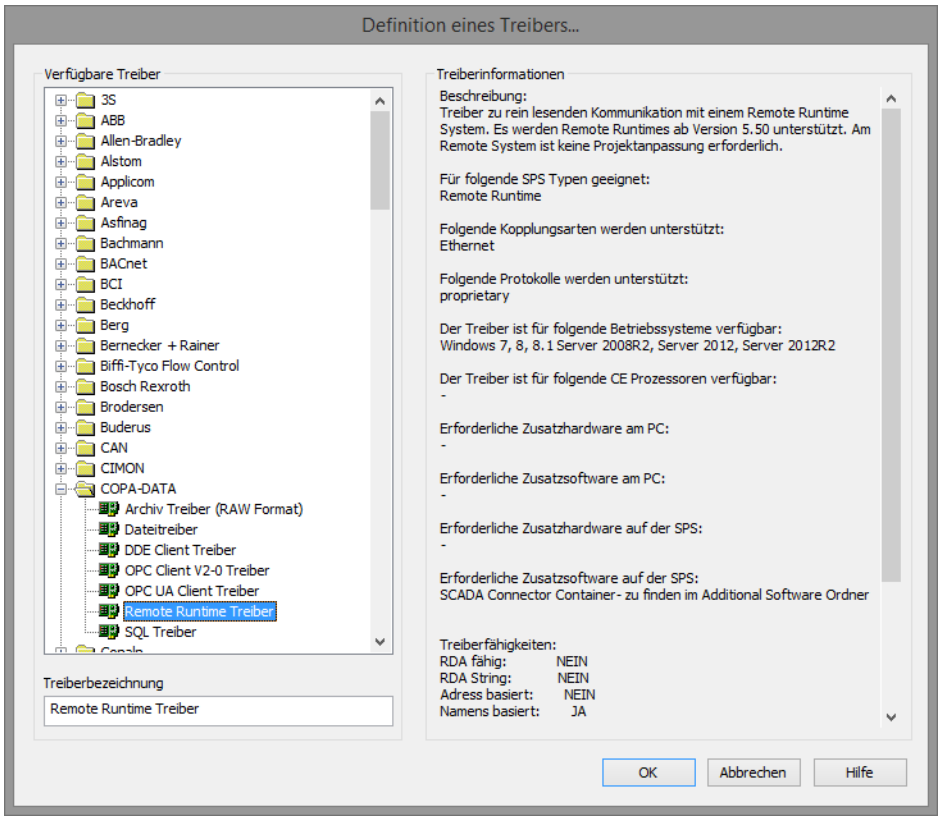
In diesem Kapitel lesen Sie, wie Sie den Treiber im Projekt anlegen und welche Einstellungen beim Treiber möglich sind.

> **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

Beachten Sie, dass der **ADP Treiber** nur ein Mal pro Projekt angelegt werden kann. Der Treiber findet sich in der Treibergruppe COPA-DATA.

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



| Parameter | Beschreibung |
|---|---|
| **Verfügbare Treiber** | Liste aller verfügbaren Treiber.<br>Die Darstellung erfolgt in einer Baumstruktur:<br>`[+]` erweitert die Ordnerstruktur und zeigt die darin enthaltenen Treiber.<br>[-] reduziert die Ordnerstruktur<br>Default: `keine Auswahl` |
| **Treiberbezeichnung** | Eindeutige **Bezeichnung** des Treibers.<br>Default: `leer`<br>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.<br>Default: `leer`<br>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].`

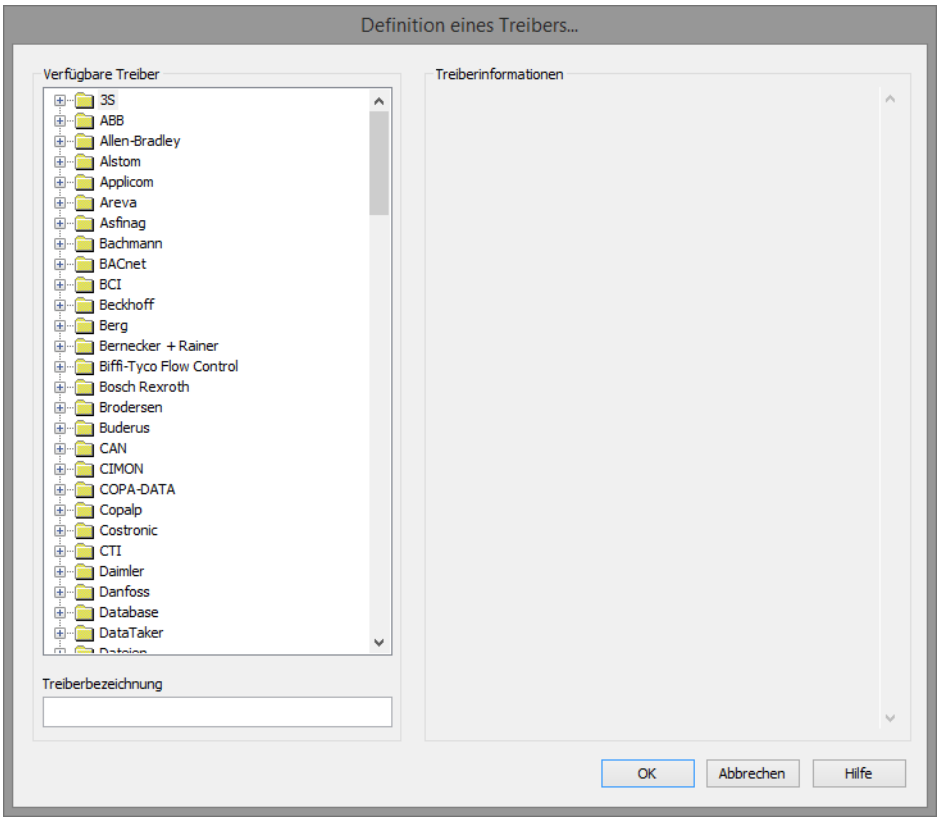
### TREIBER NEU ANLEGEN

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:

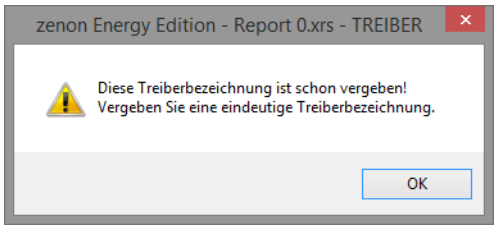
- Die **Treiberbezeichnung** muss eindeutig sein.
  Wird ein Treiber mehrmals im Projekt verwendet, so muss jeweils eine neue Bezeichnung vergeben werden.
  Dies wird durch Klick auf die Schaltfläche **OK** evaluiert. Ist die Treiber im Projekt bereits vorhanden wird dies mit einem Warndialog angezeigt.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

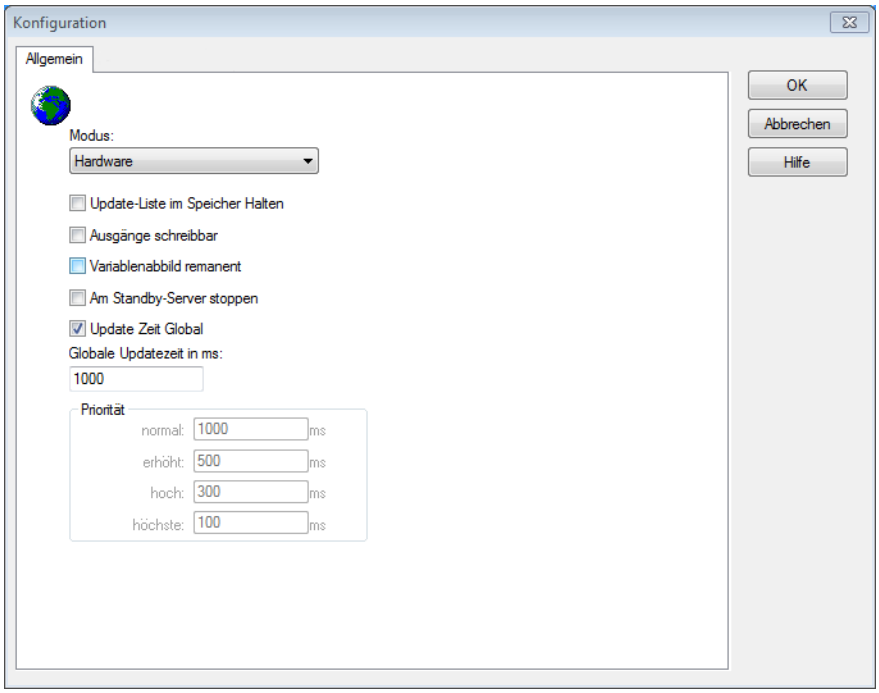
> **Info**
>
> In einem zenon Projekt müssen nur die benötigten Treiber vorhanden sein. Treiber können bei Bedarf zu einem späteren Zeitpunkt hinzugefügt werden.

## 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<br>▸ `Hardware`:<br>Die Verbindung zur Steuerung wird hergestellt.<br>▸ Simulation - statisch:<br>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.<br>▸ `Simulation - zählend`:<br>Es wird keine Kommunikation zur Steuerung aufgebaut, die Werte werden vom Treiber simuliert. In diesem Modus zählt der Treiber die Werte innerhalb ihres Wertebereichs automatisch hoch.<br>▸ `Simulation - programmiert`:<br>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.<br>Details siehe Kapitel Treibersimulation.<br>(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.<br>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`:<br>Ausgänge können beschrieben werden.<br>▸ `Inaktiv`:<br>Das Beschreiben der Ausgänge wird unterbunden.<br>**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.<br>Grundvoraussetzung: Die Variable muss einen gültigen Wert und |

| | |
|---|---|
| | Zeitstempel besitzen.<br>Das Variablenabbild wird im Modus Hardware gespeichert wenn:<br>▸ einer der Status S_MERKER_1(0) bis S_MERKER8(7), REVISION(9), AUS(20) oder ERSATZWERT(27) aktiv ist<br>Das Variablenabbild wird immer gespeichert wenn:<br>▸ die Variable vom Objekttyp **Treibervariable** ist<br>▸ der Treiber im Simulationsmodus läuft. (nicht programmierte Simulation)<br>Folgende Status werden beim Start der Runtime nicht restauriert:<br>▸ `SELECT(8)`<br>▸ `WR-ACK(40)`<br>▸ `WR-SUC(41)`<br>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.<br>**Achtung:** Ist diese Option aktiv, ist die lückenlose Archivierung nicht mehr gewährleistet.<br>▸ `Aktiv`:<br>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.<br>Default: `inaktiv`<br>**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:<br>▸ `Aktiv`:<br>Die eingestellte **Globale Update Zeit** wird für alle Variablen im Projekt verwendet. Die bei den Variablen eingestellte Priorität wird nicht verwendet. |

| | |
|---|---|
| | ▸ Inaktiv:<br>Die eingestellten Prioritäten werden für die einzelnen Variablen verwendet.<br><br>**Ausnahmen:** Spontane Treiber ignorieren diese Option. Sie nutzen in der Regel die kürzest mögliche Update Zeit.<br>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.<br><br>Die Zuordnung der Variablen erfolgt separat bei jeder Variablen über die Einstellungen in den Variableneigenschaften.<br>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.<br><br>**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. |

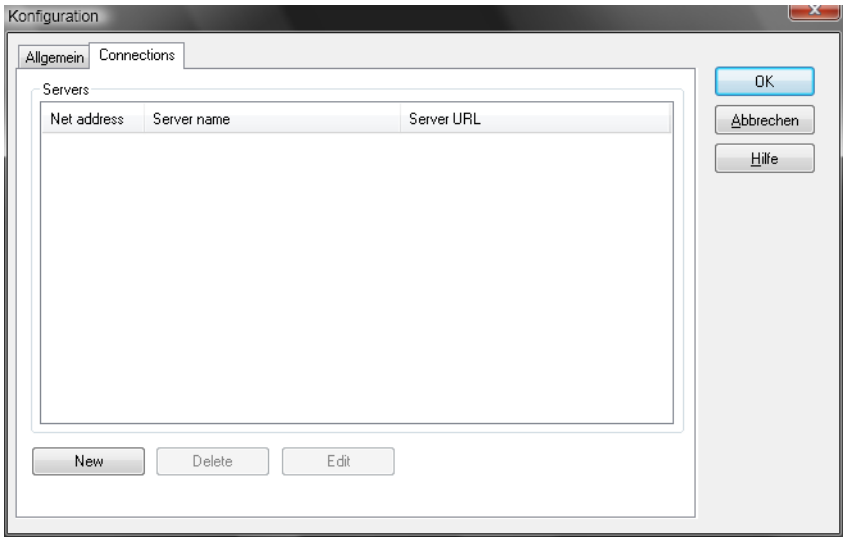
### UPDATE ZEIT SPONTANE TREIBER

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 Connections

In der Registerkarte **Connections** konfigurieren Sie die OPC UA Verbindungen zu einer oder mehrerer Steuerungen.




**SERVERS**

Liste der konfigurierten Verbindungen. Durch Auswählen einer Verbindung kann diese gelöscht oder bearbeitet werden.

| Parameter | Beschreibung |
|---|---|
| Net adress | Die Netzadresse identifiziert die Verbindung. Jede Verbindung muss daher eine eindeutige Netzadresse haben, welche automatisch vergeben wird. Variablen werden einer Verbindung über die Netzadresse zugeordnet. |
| Server name | Frei wählbarer Name zur leichteren Unterscheidung der Verbindungen. |
| Server URL | Die Netzwerkadresse unter der der Verbindungsendpunkt des Servers zu erreichen ist . Z. B. `opc.tcp://server:4840` |
| **New** | Öffnet Dialog zum Anlegen einer neuen Verbindung. |
| **Delete** | Löscht markierten Eintrag aus der Liste. |
| **Edit** | Öffnet markierten Eintrag zum Bearbeiten. |

### DIALOG BEENDEN

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

### NEUE VERBINDUNG ANLEGEN

Klicken sie auf die Schaltfläche **New.**Im folgenden Dialog:

- definieren Sie in der Registerkarte **Communication Settings** (auf Seite 20) die Verbindungsdetails
- setzen Sie unter **Avanced settings** (auf Seite 23) erweiterte Optionen
- konfigurieren Sie unter **Certificates** (auf Seite 28) die Zertifikate

### VERBINDUNG BEARBEITEN

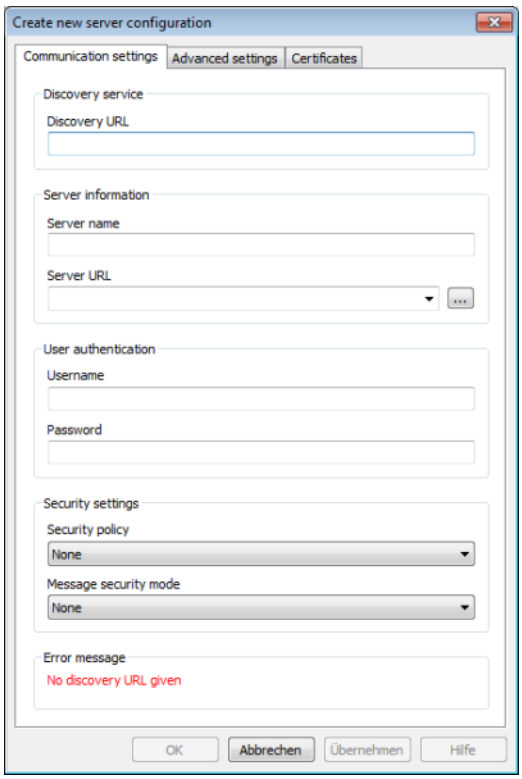
Wird im Verbindungsdialog eine bestehende Verbindung ausgewählt, kann diese über die Schaltfläche **Edit** nachträglich verändert werden. Die Eigenschaften dafür sind identisch mit den Feldern beim Neuanlegen einer Verbindung.

### VERBINDUNG LÖSCHEN

Zum Löschen einer Verbindung:

- wählen Sie die Verbindung in der Verbindungsliste aus
- klicken Sie auf die Schaltfläche **Delete**
- die Verbindung wird entfernt

## Communication settings




| Parameter | Beschreibung |
|---|---|
| **Discovery URL** | Eingabefeld für Adresse des `Discovery service`.<br>Mit diesem `Discovery service` können vorhandene OPC UA Server im Netzwerk identifiziert und abgefragt werden.<br><br>Default: `leer`<br><br>▶ Beispiel: **opc.tcp://192.168.0.1:4840** |
| **Server name** | Frei wählbarer Name der konfigurierten Verbindung. |
| **Server URL** | Die Netzwerkadresse unter der ein OPC UA Server im Netzwerk erreichbar ist.<br><br>Konfiguriert werden können IP-Adressen und DNS-Namen:<br>`opc.tcp://IP-Adresse oder DNS-Name:Port/Servername`<br><br>▶ `Eingabe der Adresse in Eingabefeld`<br><br>▶ `Auswahl aus Dropdownliste`<br>Klick auf Schaltfläche **...** startet die Erkennung der verfügbaren Server auf Basis der konfigurierten **Discovery URL**. |

| | |
|---|---|
| | Beispiel: **opc.tcp://192.168.0.1:4841**<br><br>Beispiel**:opc.tcp://PC1:4841/SimulationServer** |
| **Username** | Optionaler Benutzername bei aktivierter Authentifizierung am Server. |
| **Password** | Optionales Passwort bei aktivierter Authentifizierung am Server. |
| **Security Policy** | Angabe des Algorithmus bei verschlüsselter Kommunikaiton.<br><br>Auswahl aus Dropdownliste:<br>▶ `None`<br>▶ `Basic128`<br>▶ `Basic128RSA15`<br>▶ `Basic256`<br><br>Default: `None`<br><br>Die Werte der Auswahlliste entsprechen der OPC UA Spezifikation, Teil 7.<br><br>Hinweis: Beim Auslesen werden alle unterstützten Werte durch den Discovery service zur Verfügung gestellt. |
| **Message Security Mode** | **Nachrichtensicherheit**, definiert Sicherheitsstufe bei der Übertragung von Nachrichten.<br><br>Auswahl aus Dropdownliste:<br>▶ `None`<br>Alle Nachrichten werden in Klartext (nicht signiert und nicht verschlüsselt) übertragen.<br>▶ `Sign`<br>Alle Nachrichten werden signiert, aber nicht verschlüsselt übertragen.<br>▶ `Sign & Encrypt`<br>Alle Nachrichten werden signiert und verschlüsselt übertragen.<br><br>Default: none |

**Error Message**

Anzeige von falschen oder fehlenden Konfigurationen. Dieses Ausgabefeld gilt für alle **Advanced settings** und ist nicht auf die aktuelle Registerkarte beschränkt.

**Beispiel:** `Incomplete configuration on another tab.`

### 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. |

### KONFIGURATION

Für eine vollständige und gültige Konfiguration ist es ausreichen, in das Eingabefeld **Discovery URL** die Adresse eines korrekt konfigurierten `Discovery Server`s einzugeben.

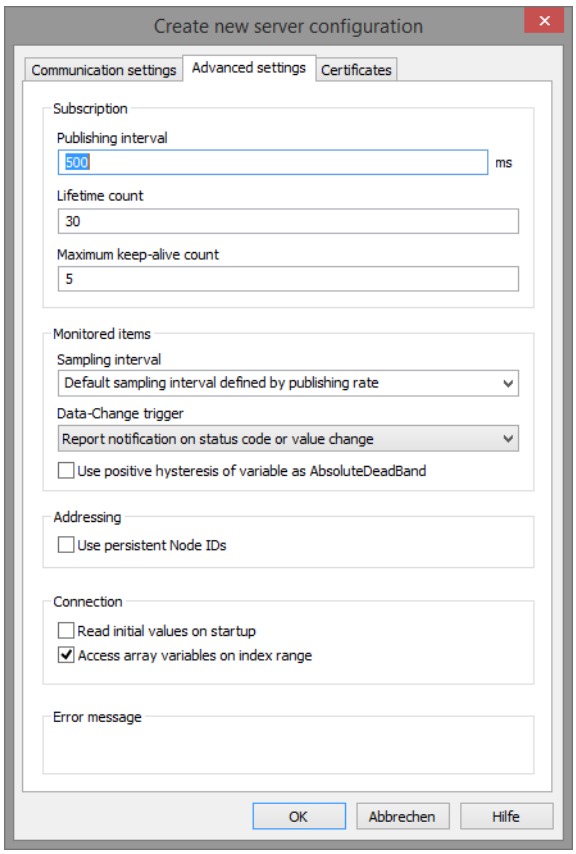
Ist dieser `Discovery Server` korrekt konfiguriert werden mit mit einem Klick auf die Schaltfläche **...** (neben dem Feld **Server URL**) die notwendigen Konfigurationsdaten von diesem `Discovery Server` ausgelesen und übernommen.

**Achtung:** Wenn **Sign** oder **Sign & Encrypt** verwendet wird, muss das Serverzertifikat in der Registerkarte **Certificates** importiert werden. Details zur Authentifizierung finden Sie in der Norm **Part 4 - 5.6.3 ActivateSession**.

Fehleingaben der Discovery URL werden bei Klick auf die Schaltfläche **...** in einem Dialog angezeigt:

- `A value had an invalid Syntax (0x80b60000)`
  falscher Syntax für Discovery URL
- `The requested operation is not supported (0x803d0000)"`
  opc.tcp fehlt bei der Eingabe für die Discovery URL

## Avanced settings




**Subscription**

Optionen für die Verbindung.

| Parameter | Beschreibung |
|---|---|
| **Publishing interval [ms]** | Definiert das Intervall in Millisekunden, innerhalb dessen der Server für die `Subscription` eine Benachrichtigung senden muss.<br>Der Server kann dieses Intervall nach oben anpassen, um technische Begrenzungen einzuhalten.<br>Default: `500` |
| **Lifetime count** | Gibt an, wie oft das **Publishing interval** überschritten werden darf, bevor die `Subscription` vom Server gelöscht wird.<br>Werte: `>=1 und >=3*`**Maximum keep-alive count**<br>Default: `30` |
| **Maximum keep-alive count** | Gibt an, wie oft das **Publishing interval** ablaufen darf, bevor eine `keep-alive` Nachricht gesendet wird.<br>Werte: `>= 1`<br>Default: `5` |

### MONITORED ITEMS

| Parameter | Beschreibung |
|---|---|
| **Sampling interval** | Abtastintervall in Millisekunden Sekunden, in dem der Server die **MonitoredItems** abfragen soll.<br>Der Server kann dieses Intervall nach oben anpassen, um technische Begrenzungen einzuhalten. Auswahl aus Dropdownliste oder direkter Eingabe in Feld. Mögliche Werte:<br>▸ `Fastest practical rate (= 0)`<br>▸ `Default: sampling interval defined by publishing rate (=-1)`<br>▸ `Manuelle Werteingabe in Millisekunden` |
| **Data-Change trigger** | Dropdownliste zur Auswahl des Auslösers, welcher eine Benachrichtigung schickt.<br>▸ `Report notification on status code or value change`<br>Bei Änderung des Status oder bei Wertänderung wird eine Benachrichtigung (`notification`) gesendet.<br>▸ `Report notification on timestamp change also`<br>Diese Option beinhaltet die Benachrichtigungen, wie sie auch für `Report notification on status code or value change` übertragen werden.<br>Zusätzlich wird auch bei Änderungen des Zeitstempels (Wert oder Status) eine Benachrichtigung gesendet. |

| | |
|---|---|
| | Default: `Report notification on status code or value change` |
| **Use positive hysteresis of variables as AbsoluteDeadBand** | Checkbox für Unterstützung der Variablen-Eigenschaft **Positiv für Signal**.<br>▶ `nicht aktiviert`:<br>Projektierte Dead bands werden nicht berücksichtigt.<br>▶ `aktiviert`:<br>nur Werte der projektierten Dead bands werden angezeigt.<br>Ist diese Checkbox aktiviert wird die Projektierung der Eigenschaft **Positiv für Signal** berücksichtigt.<br>Sie finden diese Eigenschaft bei der Variableneigenschaft **Wertberechnung**, in der Eigenschaftengruppe **Hysterese**.<br>Default: `inaktiv`<br>**Hinweis:** Weiter Informationen dazu finden Sie im Handbuch Variablen im Kapitel Hysterese. |

### ADRESSING

| Parameters | Beschreibung |
|---|---|
| **Use persistent Node IDs** | Verhalten des Treibers beim Anmelden von Variablen in der Runtime.<br>▸ `aktiviert`:<br>für die nachfolgenden Zugriffe (Erzeugung der `monitored Items`, Lese- und Schreibzugriffe) wird die gespeicherte `NodeID` und `NamespaceIndex` aus der Eigenschaft **Symbolische Adresse** verwendet. Beim Anlegen von Variablen durch den Online-Import wird die Eigenschaft entsprechend gesetzt. Vorausgesetzt wird, dass die `NodeID` sich im Server nicht ändert.<br>▸ `nicht aktiviert`:<br>Führt einmalig die Funktion `TranslateBrowsePathToNodeId` aus. Im Anschluss verwendet der Treiber die `NodeID`<br>**Hinweis:** Diese Option kann bei manchen (speziell kleineren) OPCA UA Servern zu Verbindungsprobleme führen, wenn die Anfrage `TranslateBrowsePathToNodeId` zu einer erhöhten Serverauslastung führt.<br>Default: `inaktiv` |

### CONNECTION

| Parameters | Beschreibung |
|---|---|

| | |
|---|---|
| **Read initial values on startup** | ▸ `Aktiv`:<br>Zusätzlich zum Anmelden der Variablen (Advise) wird auch eine Leseanforderung übermittelt.<br>▸ `Inaktiv`:<br>Es wird keine zusätzliche Leseanforderung zum Server geschickt. Abhängig vom Server können die Werte verzögert eintreffen.<br>Default: `aktiv`<br>**Empfehlung:** Wählen Sie diese Option, wenn sehr viele Variablen mit dem Server kommunizieren sollen. |
| **Access array variables on index range** | Umgang mit OPC UA Array-Variablen. Zusätzlich zur **Netzadresse** und der Adressierung über die Eigenschaft **Browse name**, ist auch der Offset der Variable relevant. Der Offset ist immer um eins höher als der Array Index (in OPC UA beginnt der Offset eines Arrays bei 0)<br>▸ `Aktiv`:<br>Jedes Array-Element wird als eigene Variable für eine Wertänderung angemeldet. Bei `CreateMonitoredItems` wird der `IndexRange` verwendet.<br>`Inaktiv`:<br>Das gesamte Array wird für Wertänderungen angemeldet. Dadurch lassen sich große Arrays, bei denen die meisten oder alle Array-Elemente benötigt werden, schneller am Server anmelden. Dabei wird immer das gesamte Array übertragen , auch bei nur wenigen Wertänderungen. Dadurch kann sich die Netwerklast erhöhen.<br>Default: `aktiv`<br>**Empfehlung:** Wählen Sie diese Option, wenn nur wenige Elemente aus großen Arrays benötigt werden. |

**Error Message**

Anzeige von falschen oder fehlenden Konfigurationen. Dieses Ausgabefeld gilt für alle **Advanced settings** und ist nicht auf die aktuelle Registerkarte beschränkt.

**Beispiel:** `Incomplete configuration on another tab.`

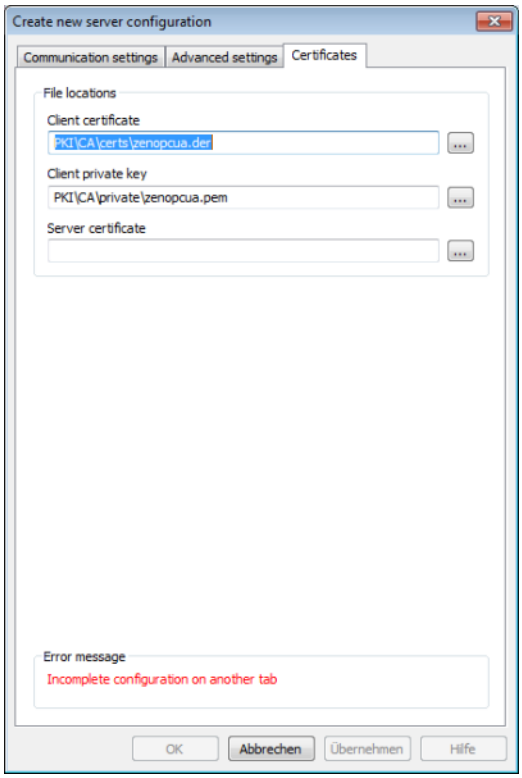
### DIALOG BEENDEN

| 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. |

**Tipp:** Die Erkennung des Verbindungsausfalls zum OPCUA Server macht der OPCUA Client Treiber mittels der im Norm empfohlene Methode: Es wird erwartet, dass der Server spätestens nach `RevisedMaxKeepAliveCount * RevisedPublishingInterval` eine `PublishResponse` sendet, nach dem Zeitpunkt wo die letzte `PublishResponse` gesendet wurde. Wenn sich keinen Wert der `MonitoredItems` geändert hat, wird ein leeres `KeepAlive PublishResponse` erwartet. Der OPCUA Client Treiber wartet nach verstreichen des Zeitpunkts noch 5 Sekunden auf die Antwort vom Server und stellt dann einen Verbindungsausfall fest. Die Verbindung wird Clientseitig getrennt, es wird keine `DeleteSubscriptionRequest` gesendet da nicht garantiert werden kann, dass der Server noch reagiert.

## Certificates

In der Registerkarte **Certificates** konfigurieren Sie Zertifikate für die verschlüsselte Kommunikation:




### FILE LOCATIONS

| Parameter | Beschreibung |
|---|---|
| **Client Certificate** | `Application Instance Certificate` des Clients. Dieses Zertifikat wird vom Server zum Verschlüsseln der Nachrichten verwendet.<br><br>Pfad im Projektordner: `"..\Custom\Drivers\PKI\CA\certs\".` |
| **Client Private Key** | Privater Schlüssel des Clients. Dieser wird zum Entschlüsseln von Server-Nachrichten verwendet.<br><br>Pfad im Projektordner: "..\Custom\Drivers\PKI\CA\certs\" |
| **Server Certificate** | `Application Instance Certificate` mit öffentlicher Schlüssel des Servers. Dieser Schlüssel wird zum Verschlüsseln von Nachrichten an den Server verwendet. |

**Error Message**

Anzeige von falschen oder fehlenden Konfigurationen. Dieses Ausgabefeld gilt für alle **Advanced settings** und ist nicht auf die aktuelle Registerkarte beschränkt.

**Beispiel:** `Incomplete configuration on another tab.`

### 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. |

## ZERTIFIKATE FÜR KOMMUNIKATION

### INITIALE ERSTELLUNG

Sobald eine Verbindung in der Treiberkonfiguration erstellt wird, erzeugt der Treiber automatisch ein `Application Instance Certificate` mit öffentlicher Schlüssel und den passenden privaten Schlüssel, sofern diese noch nicht existieren.

### GÜLTIGKEIT

Dieses selbst erzeugte Zertifikat hat eine Gültigkeitsdauer von einem Jahr. Der OPCUA Client Treiber prüft die Gültigkeit des eigenen Zertifikates nicht. Ein toleranter Server akzeptiert unter Umstände auch ein bereits abgelaufenes Zertifikat. Für einen Server mit strikter Überprüfung ist es daher ratsam, eigene `Application Instance` Zertifikate mit einer längeren Gültigkeitsdauer zu erstellen.

### UNTERSCHIEDLICHE ZERITIFIKATE

Per Default verwenden jede Verbindung und jeder Treiber innerhalb eines Projektes dasselbe `Application Instance Certificate` . Per Definition soll aber jeder OPCUA Client ein einzigartiges `Application Instance Certificate` verwenden. Für diesen Fall, oder wenn ein OPCUA Server dies voraussetzt, kann pro Verbindung im Treiber oder in den jeweiligen Treibern ein selbst erstelltes Zertifikat konfiguriert werden.

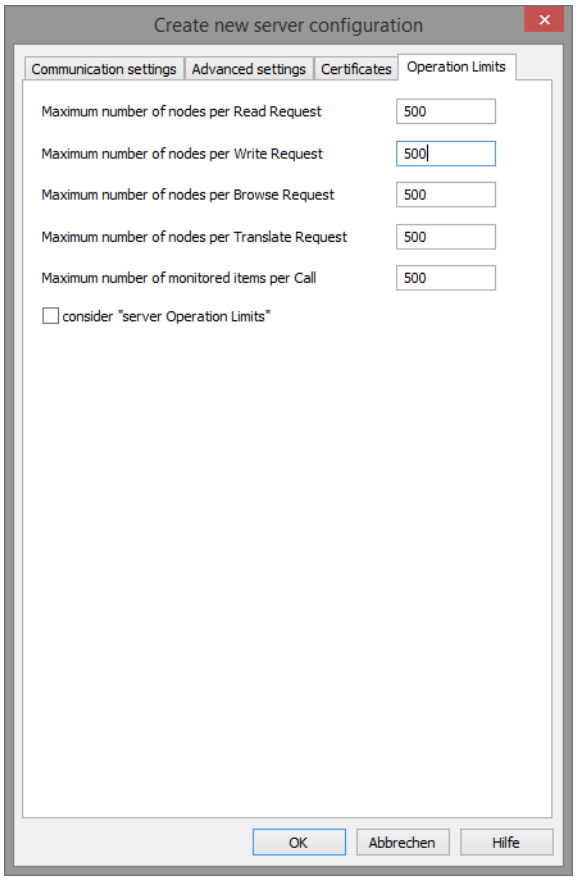
**KONFIGURATION**

Ein selbst erzeugtes `X509 OPC UA Application Instance Certificate`, im DER-Format, muss im Editor unter Dateien -> Treiber -> PKI -> CA -> Certs eingefügt werden. Der passende private Schlüssel, im PEM Format, muss im Editor unter Dateien -> Treiber -> PKI -> CA -> Private eingefügt werden. Anschliessend muss in der Treiberkonfiguration für jede Verbindung die jeweils richtige Datei ausgewählt werden.

**Beispiel**

`X509 OPCUA Application Certificates` können mit dem **OPCUA Configuration Tool** der **OPC Foundation** oder mit Software **XCA** erstellt werden.

## Operation Limits




Diese Registerkarte ist für jede projektierte Verbindung verfügbar.

| Parameter | Beschreibung |
|---|---|
| **Maximum number of nodes per Read Request** | Maximale Anzahl der `Nodes`, die in einem `ReadRequest` berücksichtigt werden.<br>Default: `500`<br>Minimum: `1`<br>Maximum: `4294967295` |
| **Maximum number of nodes per Write Request** | Maximale Anzahl der `Nodes`, die in einem `WriteRequest` berücksichtigt werden.<br>Default: `500`<br>Minimum: `1`<br>Maximum: `4294967295` |
| **Maximum number of nodes per Browse Request** | Maximale Anzahl der `Nodes`, die in einem `BrowseRequest` berücksichtigt werden.<br>Default: `500`<br>Minimum: `1`<br>Maximum: `4294967295` |
| **Maximum number of nodes per Translate Request** | Maximale Anzahl der `Nodes`, die in einem `TranslateBrowsePathRequest` berücksichtigt werden.<br>Default: `500`<br>Minimum: `1`<br>Maximum: `4294967295` |
| **Maxmimum number of monitored items per Call** | Maximale Anzahl der `Nodes`, die bei der Erstellung oder beim Löschen von **Monitored Items** in einem entsprechenden `Request` enthalten sind.<br>Default: `500`<br>Minimum: `1`<br>Maximum: `4294967295` |
| **consider "Server Operation Limits"** | Quelle der Konfiguration der **Operation Limits**.<br>▸ aktiviert:<br>beim Verbindungsaufbau werden die Limits vom Server gelesen. Die in diesem Dialog konfigurierten **Operation Limits** werden nicht mehr berücksichtigt.<br>▸ inaktiv:<br>Die Limits werden, wie in diesem Dialog konfiguriert, verwendet.<br>Default: `inaktiv`<br>Dabei gilt:<br>▸ Beim Verbindungsaufbau zum Server werden die |

| | |
|---|---|
| | Limits vom Server gelesen.<br>▸ Stellt der Server diese Limits bereit und können diese Limits erfolgreich ausgelesen werden, ersetzen diese die konfigurierten **Operation Limits**.<br>▸ Dieses Verhalten gilt sowohl für den Variablenimport als auch für die Kommunikation zur Runtime. |

#### DIALOG BEENDEN

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

# 7. Variablen anlegen

So werden Variablen im zenon Editor angelegt:

## 7.1 Variablen im Editor anlegen

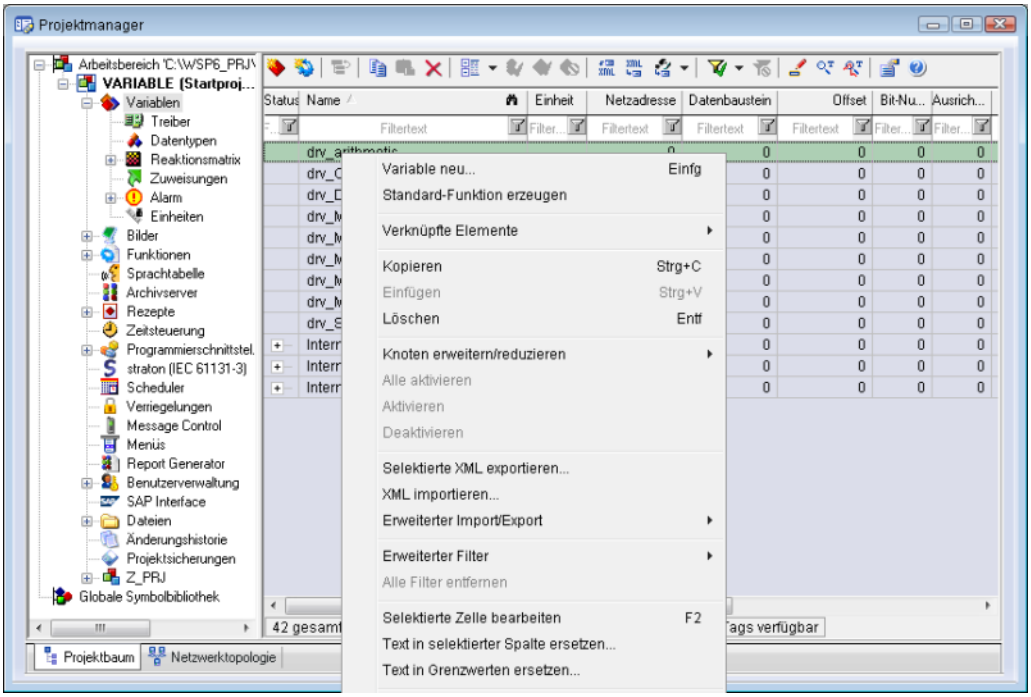
Variablen können angelegt werden:

- 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:

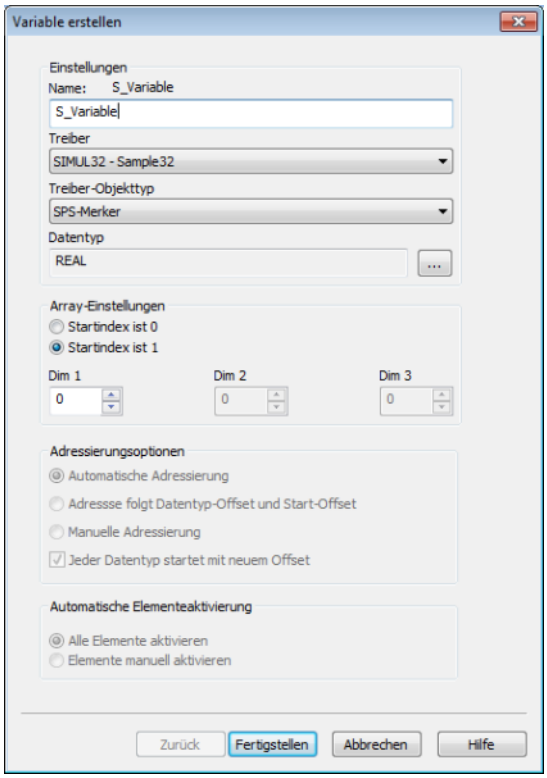
1. Wählen Sie im Knoten **Variablen** im Kontextmenü den Befehl **Variable neu**.




Der Dialog zur Konfiguration der Variable wird geöffnet.

2. Konfigurieren Sie die Variable.

3. Welche Einstellungen möglich sind, hängt ab vom Typ der Variablen.




| 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.<br><br>Maximale Länge: 128 Zeichen<br><br>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.<br>Hinweis: Manche Treiber erlauben die Adressierung auch über die Eigenschaft **Symbolische Adresse**. |
| **Treiber** | Wählen Sie aus der Dropdownliste den gewünschten Treiber.<br><br>Hinweis: Sollte im Projekt noch kein Treiber angelegt sein, wird automatisch der Treiber für interne Variable (**Intern.exe** (**Main.chm::/Intern.chm::/Intern.htm**)) geladen. |
| **Treiberobjekttyp (cti.chm::/28685.htm)** | Wählen Sie aus der Dropdownliste den passenden Treiberobjekttyp aus. |

| | |
|---|---|
| **Datentyp** | Wählen Sie den gewünschten Datentyp. Klick auf die Schaltfläche ... öffnet den Auswahl-Dialog. |
| **Array-Einstellungen** | Erweiterte Einstellungen für Array-Variablen. Details dazu lesen Sie im Abschnitt Arrays. |
| **Adressierungsoptionen** | Erweiterte Einstellungen für Arrays und Struktur-Variablen. Details dazu lesen Sie im jeweiligen Abschnitt. |
| **Automatische Elementeaktivierung** | Erweiterte Einstellungen für Arrays und Struktur-Variablen. Details dazu lesen Sie im jeweiligen Abschnitt. |

### SYMBOLISCHE ADRESSE

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

Maximale Länge: `1024` Zeichen.

### ABLEITUNG VOM DATENTYP

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

- vom Datentyp abgeleitet
- beim Ändern des Datentyps automatisch angepasst

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

## 7.2 Adressierung

Die Adressierung der Variablen definieren Sie im Eigenschaftenfenster:

| Gruppe/Eigenschaft | Beschreibung |
|---|---|
| Allgemein | Gruppe mit allgemeinen Eigenschaften. |
| **Name** | Frei vergebbarer Name.<br>Achtung: Je zenon Projekt muss der Name eindeutig sein. |
| **Kennung** | Frei vergebbare Kennung.<br>Z. B. für **Betriebsmittelkennung**, Kommentar usw. |
| **Adressierung** | |
| **Netzadresse** | Netzadresse der Variable.<br>Diese Adresse bezieht sich auf die Netzadresse der Verbindungsprojektierung im Treiber. Damit wird ausgewählt, auf welchem OPC UA Server sich die Variable befindet. |
| **Datenbaustein** | Wird für diesen Treiber nicht verwendet. |
| **Offset** | Offset der Variablen. Entspricht der Speicheradresse der Variablen in der Steuerung. Einstellbar von `0` bis `4294967295`. |
| **Ausrichtung** | Ausrichtung für Variablen mit Bytelänge 1. Es kann zwischen niederwertigem und höherwertigem Byte gewählt werden. |
| **Bitnummer** | Wird für diesen Treiber nicht verwendet. |
| **Stringlänge** | Nur verfügbar bei String-Variablen.<br>Maximale Anzahl von Zeichen, die die Variable aufnehmen kann. |
| **Treiber Anbindung/Treiberobjekttyp** | Objekttyp der Variablen. Wird abhängig vom verwendeten Treiber beim Erstellen der Variablen ausgewählt und kann hier geändert werden. |
| **Treiber Anbindung/Datentyp** | Datentyp der Variablen. Wird beim Erstellen der Variablen ausgewählt und kann hier geändert werden.<br>ACHTUNG: Wenn der Datentyp nachträglich geändert wird, müssen alle anderen Eigenschaften der Variablen überprüft bzw. angepasst werden. |
| **Browse name** | Entspricht dem **Browse name** der OPC UA Spezifikation. Es sind nur hierarchische Verweise (vorwärts) ausgehend vom Objekt-Ordner erlaubt. Zum Beispiel:<br>`9:Data/9:Dynamic/9:Scalar/9:UInt32Value`<br>oder `Server/ServerStatus/StartTime`.<br>Die vorangestellte Zahl im ersten Beispiel gibt den verwendeten Namespace-Index der Variablen an. |
| Browse name | Wird beim online Import automatisch gesetzt. |

## 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 | Beschreibung |
|---|---|---|---|---|---|
| **SPS-Merker** | 8 | X | X | BOOL, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, 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).<br>**Hinweis**: Die Adressierung und das Verhalten ist bei den meisten zenon Treibern gleich.<br>Weitere Informationen dazu finden Sie im Kapitel Kommunikationsdetails (Treibervariablen) (auf Seite 47). |

**Legende:**

**X**: wird unterstützt

**--**: wird nicht unterstützt

### 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 |
|---|---|---|
| OpcUa_Boolean | BOOL | 8 |
| OpcUa_Byte | USINT | 9 |
| OpcUa_SByte | SINT | 10 |
| OpcUa_UInt16 | UINT | 2 |
| OpcUa_Int16 | INT | 1 |
| OpcUa_UInt32 | UDINT | 4 |
| OpcUa_Int32 | DINT | 3 |
| OpcUa_UInt64 | ULINT | 27 |
| OpcUa_Int64 | LINT | 26 |
| OpcUa_Float | REAL | 5 |
| OpcUa_Double | LREAL | 6 |
| OpcUa_String | STRING | 12 |
| - | WSTRING | 21 |
| - | DATE | 18 |
| - | TIME | 17 |
| OpcUa_DateTime | DATE_AND_TIME | 20 |
| - | TOD (Time of Day) | 19 |

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

Achtung: OpcUa_DateTime basiert auf 100ns Anteile seit 01.01.1601 00:00:00. Der zenon Datentyp DATE_AND_TIME basiert auf den Unix Zeit (ms seit 01.01.1970 00:00:00). OPC UA Zeitstempel vor 01.01.1970 00:00:00 werden in zenon mit dem Wert "0" dargestellt.

## 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.

#### 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.<br>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).<br>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.<br>Länge kann über den Eintrag **MAX_LAENGE** in der **project.ini** eingeschränkt werden. |
| **EINHEIT** | C | 11 | Technische Maßeinheit |
| **DATENART** | C | 3 | Datentyp (z. B. Bit, Byte, Wort, ...) entspricht dem Datentyp. |
| **KANALTYP** | C | 3 | Speicherbereich in der SPS (z. B. Merkerbereich, Datenbereich, ...) entspricht Treiberobjekttyp. |
| **HWKANAL** | Num | 3 | Netzadresse |
| **BAUSTEIN** | N | 3 | Datenbaustein-Adresse (nur bei Variablen aus den Datenbereich der SPS) |
| **ADRESSE** | N | 5 | Offset |
| **BITADR** | N | 2 | Für Bit-Variablen: Bitadresse<br>Für Byte-Variablen: 0=niederwertig, 8=höherwertig<br>Für String-Variablen: Stringlänge (max. 63 Zeichen) |
| **ARRAYSIZE** | N | 16 | Anzahl der Variablen im Array für Index-Variablen<br>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<br>0: Sollwert setzen ist nicht erlaubt<br>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<br>setzt sich zusammen aus TREIBER-OBJEKTTYP und DATENTYP |
| **SIGMIN** | Float | 16 | Rohwertsignal minimal (Signalauflösung) |
| **SIGMAX** | F | 16 | Rohwertsignal maximal (Signalauflösung) |
| **ANZMIN** | F | 16 | technischer Wert minimal (Messbereich) |
| **ANZMAX** | F | 16 | technischer Wert maximal (Messbereich) |
| **ANZKOMMA** | N | 1 | Anzahl der Nachkommastellen für die Darstellung der Werte (Messbereich) |
| **UPDATERATE** | F | 19 | Updaterate für Mathematikvariablen (in sec, eine Dezimalstelle möglich)<br>bei allen anderen Variablen nicht verwendet |
| **MEMTIEFE** | N | 7 | Nur aus Kompatibilitätsgründen vorhanden |
| **HDRATE** | F | 19 | HD-Updaterate für hist. Werte (in sec, eine Dezimalstelle möglich) |
| **HDTIEFE** | N | 7 | HD-Eintragtiefe für hist. Werte (Anzahl) |
| **NACHSORT** | L | 1 | HD-Werte als nachsortierte Werte |
| **DRRATE** | F | 19 | Aktualisierung an die Ausgabe (für zenon DDE-Server, in sec, eine Kommastelle möglich) |
| **HYST_PLUS** | F | 16 | Positive Hysterese; ausgehend vom Messbereich |
| **HYST_MINUS** | F | 16 | Negative Hyterese; ausgehend vom Messbereich |
| **PRIOR** | N | 16 | Priorität der Variable |
| **REAMATRIZE** | C | 32 | Name der zugeordnete Reaktionsmatrix |
| **ERSATZWERT** | F | 16 | Ersatzwert; ausgehend vom Messbereich |
| **SOLLMIN** | F | 16 | Sollwertgrenze Minimum; ausgehend vom Messbereich |
| **SOLLMAX** | F | 16 | Sollwertgrenze Maximum; ausgehend vom Messbereich |
| **VOMSTANDBY** | L | 1 | Variable vom Standby Server anfordern; der Wert der Variable wird im redundanten Netzwerkbetrieb nicht vom Server sondern vom Standby Server angefordert |
| **RESOURCE** | C | 128 | Betriebsmittelkennung.<br>Freier String für Export und Anzeige in Listen.<br>Länge kann über den Eintrag MAX_LAENGE in der **project.ini** eingeschränkt werden. |
| **ADJWVBA** | L | 1 | Nichtlineare Wertanpassung:<br>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:

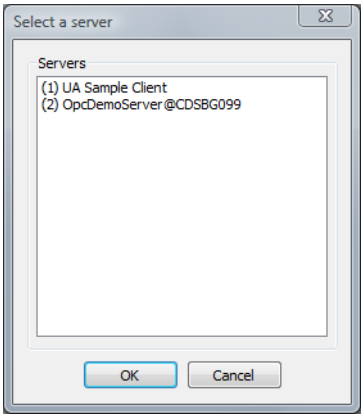
| Bezeichnung | Typ | Feldgröße | Bemerkung |
|---|---|---|---|
| **AKTIV1** | L | 1 | Grenzwert aktiv (pro Grenzwert vorhanden) |
| **GRENZWERT1** | F | 20 | technischer Wert oder ID-Nummer der verknüpften Variable für einen dynamischen Grenzwert (siehe VARIABLEx)<br>(wenn unter VARIABLEx `1` steht 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<br>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<br>(steht hier -1, so wird die bestehende Funktion beim Import nicht überschrieben) |
| **A_GRUPPE1** | N | 10 | Alarm/Ereignis-Gruppe |
| **A_KLASSE1** | N | 10 | Alarm/Ereignis-Klasse |
| **MIN_MAX1** | C | 3 | Minimum, Maximum |
| **FARBE1** | N | 10 | Farbe als Windowskodierung |
| **GRENZTXT1** | C | 66 | Grenzwerttext |
| **A_DELAY1** | N | 10 | Zeitverzögerung |
| **INVISIBLE1** | L | 1 | Unsichtbar |

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

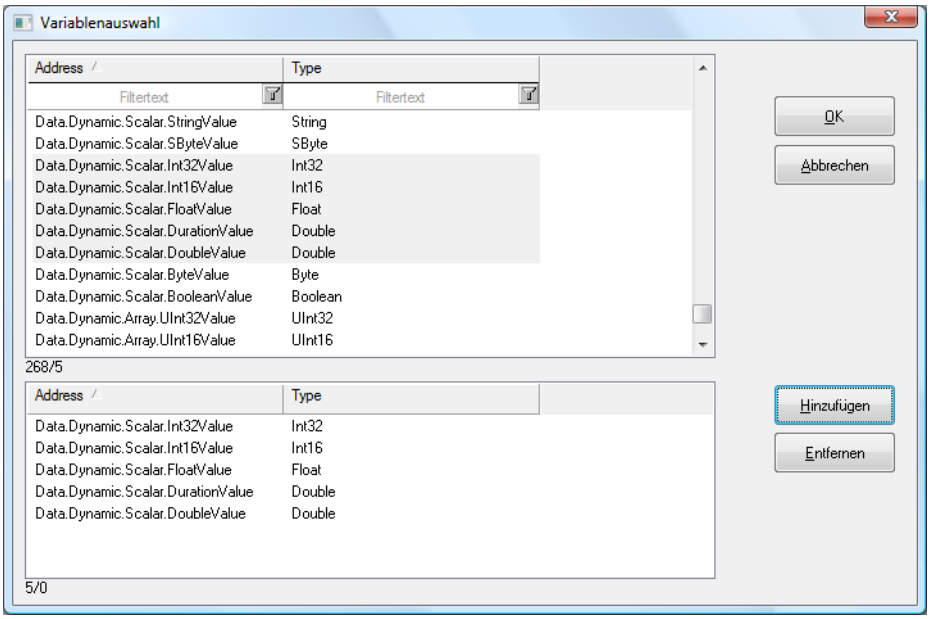
### 7.4.3 Online-Import

Um Variablen über die Funktion Online-Import anzulegen:

1. wählen Sie den Treiber in der Detailansicht des Projektmanagers aus
2. öffnen Sie mit einem Rechtsklick das Kontextmenü
3. klicken Sie auf **Variablen vom Treiber importieren**
4. der Dialog für den Online Import öffnet sich




5. wählen Sie den Server, von dem die Variablen importiert werden sollen
6. nach dem Einlesen werden alle verfügbaren Variablen angezeigt




7. wählen Sie die gewünschten Variablen aus und übernehmen Sie diese über **Hinzufügen** und **OK** in das zenon Projekt

**Info**

*Beim Import sehr vieler Variablen ist es möglich, dass nicht alle Variablen korrekt importiert werden. In diesem Fall werden Sie über eine entsprechende Meldung informiert.*

## 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** (auf der Installationsmedium im Ordner `\Predefined\Variables`) verfügbar und können von dort importiert werden.

**Hinweis:** Variablennamen müssen in zenon einzigartig sein. Soll nach einem Import der 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 werden:*

- Variablen für Modem-Informationen nur von modemfähigen Treibern unterstützt
- Treibervariablen für den Polling-Zyklus nur für rein pollenden Treibern
- verbindungsbezogene Informationen wie ErrorMSG nur von Treibern, die zu einem Zeitpunkt nur eine Verbindung bearbeiten

**INFORMATION**

| Name aus Import | Typ | Offset | Erklärung |
|---|---|---|---|
| MainVersion | UINT | 0 | Haupt-Versionsnummer des Treibers. |
| SubVersion | UINT | 1 | Sub-Versionsnummer des Treibers. |
| BuildVersion | UINT | 29 | Build-Versionsnummer des Treibers. |
| RTMajor | UINT | 49 | zenon Hauptversionsnummer |
| RTMinor | UINT | 50 | zenon Sub-Versionsnummer |
| RTSp | UINT | 51 | zenon Service Pack-Nummer |
| RTBuild | UINT | 52 | zenon Buildnummer |
| LineStateIdle | BOOL | 24.0 | TRUE, wenn die Modemleitung belegt ist. |
| LineStateOffering | BOOL | 24.1 | TRUE, wenn ein Anruf rein kommt. |
| LineStateAccepted | BOOL | 24.2 | Der Anruf wird angenommen. |
| LineStateDialtone | BOOL | 24.3 | Rufton wurde erkannt. |
| LineStateDialing | BOOL | 24.4 | Wahl aktiv. |
| LineStateRingBack | BOOL | 24.5 | Während Verbindungsaufbau. |
| LineStateBusy | BOOL | 24.6 | Zielstation besetzt. |
| LineStateSpecialInfo | BOOL | 24.7 | Spezielle Statusinformation empfangen. |
| LineStateConnected | BOOL | 24.8 | Verbindung hergestellt. |
| LineStateProceeding | BOOL | 24.9 | Wahl ausgeführt. |
| LineStateOnHold | BOOL | 24.10 | Verbindung in Halten. |
| LineStateConferenced | BOOL | 24.11 | Verbindung im Konferenzmodus. |
| LineStateOnHoldPendConf | BOOL | 24.12 | Verbindung in Halten für Konferenz. |
| LineStateOnHoldPendTransfer | BOOL | 24.13 | Verbindung in Halten für Transfer. |
| LineStateDisconnected | BOOL | 24.14 | Verbindung beendet. |
| LineStateUnknow | BOOL | 24.15 | Verbindungszustand nicht bekannt. |
| ModemStatus | UDINT | 24 | Aktueller Modemstatus. |
| TreiberStop | BOOL | 28 | Treiber gestoppt<br>Bei Treiberstop, hat die Variable den Wert TRUE und ein **OFF**-Bit. Nach dem Treiberstart, hat die Variable den Wert FALSE und kein **OFF**-Bit. |
| SimulRTState | UDINT | 60 | Informiert über Status der Runtime bei Treibersimulation. |

| | | | |
|---|---|---|---|
| ConnectionStates | STRING | 61 | Interner Verbindungsstatus des Treibers zur SPS.<br>Verbindungszustände:<br>0: Verbindung OK<br>1: Verbindung gestört<br>2: Verbindung simuliert<br>Formatierung:<br><Netzadresse>:<Verbindungszustand>;…;…;<br>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.<br>Der Zustand einer Verbindung wird nur aktualisiert, wenn eine Variable der Verbindung angemeldet ist. Ansonsten wird nicht mit der entsprechenden Steuerung kommuniziert. |

## KONFIGURATION

| Name aus Import | Typ | Offset | Erklärung |
|---|---|---|---|
| ReconnectInRead | BOOL | 27 | Wenn TRUE, dann wird beim Lesen automatisch ein Neuaufbau der Verbindung durchgeführt. |
| ApplyCom | BOOL | 36 | Änderungen an den Einstellungen der seriellen Schnittstelle zuweisen. Das Schreiben auf diese Variable hat unmittelbar den Aufruf der Methode SrvDrvVarApplyCom zur Folge (aktuell ohne weitere Funktion). |
| ApplyModem | BOOL | 37 | Änderungen an den Modemeinstellungen zuweisen. Das Schreiben auf diese Variable hat unmittelbar den Aufruf der Methode SrvDrvVarApplyModem zur Folge. Diese schließt die aktuelle Verbindung und öffnet eine neue entsprechend den Einstellungen **PhoneNumberSet** und **ModemHwAdrSet**. |

| | | | |
|---|---|---|---|
| PhoneNumberSet | STRING | 38 | Telefonnummer, welche verwendet werden soll. |
| ModemHwAdrSet | DINT | 39 | Hardwareadresse, welche zu der Telefonnummer gehört. |
| GlobalUpdate | UDINT | 3 | Updatezeit in Millisekunden (ms). |
| 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.<br><br>Wert = 0, wenn der Treiber keine serielle Kommunikation herstellen kann. |
| StopBit | USINT | 13 | Anzahl der Stoppbits der seriellen Schnittstelle. |
| Autoconnect | BOOL | 16 | TRUE, wenn die Modemverbindung automatisch beim Lesen/Schreiben aufgebaut werden soll. |
| PhoneNumber | STRING | 17 | Aktuelle Telefonnummer. |
| ModemHwAdr | DINT | 21 | Hardwareadresse zur aktuellen Telefonnummer. |
| RxIdleTime | UINT | 18 | Wenn länger als diese Zeit in Sekunden (s) erfolgreich kein Datenverkehr stattfindet, wird die Modemverbindung beendet. |
| WriteTimeout | UDINT | 19 | Maximale Schreibdauer bei einer Modemverbindung in Millisekunden (ms). |

| | | | |
|---|---|---|---|
| RingCountSet | UDINT | 20 | So oft läutet ein hereinkommender Anruf, bevor dieser angenommen wird. |
| ReCallIdleTime | UINT | 53 | Wartezeit zwischen Anrufen in Sekunden (s). |
| ConnectTimeout | UINT | 54 | Zeit in Sekunden (s) für Verbindungsaufbau. |

**STATISTIK**

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

| | | | |
|---|---|---|---|
| PokeFinish | BOOL | 55 | Geht für eine Abfrage auf 1, wenn alle anstehenden Pokes ausgeführt wurden. |

**FEHLERMELDUNGEN**

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

# 8. Treiberspezifische Funktionen

Dieser Treiber unterstützt folgende Funktionen:

#### MAPPING SONDERSTATUS

Mapping von NAN und NULL auf den zenon Status INVALID. Wird einer der Status aus der Tabelle empfangen, wird in zenon der Status INVALID gesetzt.

| OPCUA | Wert | NT_870 | OV_870 |
|---|---|---|---|
| NAN | 0 | 1 | 1 |
| NULL | 0 | 1 | 0 |
| +INF | +MAXFLOAT | 0 | 1 |
| -INF | -MAXFLOAT | 0 | 1 |

## 8.1 DataChangeFilter

#### DATACHANGEFILTER KONFIGURIERBAR:

- Die Eigenschaft **Data-Change trigger** unterstützt die Status "`STATUS_VALUE_1`" und "`STATUS_VALUE_TIMESTAMP_2`".
  Ist `STATUS_VALUE_TIMESTAMP_2` konfiguriert, inkludiert der Treiber ein `DataChangeFilter` im `CreateMonitoredItemRequest` und fordert den Server auf, entsprechend eine `DataChangeNotification` zu senden, wenn sich auch der Zeitstempel geändert hat. Der Treiber sendet einen aktualisierten Wert zur Runtime, wenn sich der Zeitstempel eines überwachten Elements ändert.
- Die neue Eigenschaft **Use positive hysteresis of variable as AbsoluteDeadband** bewirkt, dass der Treiber ein `DataChangeFilter` im `CreateMonitoredItemRequest` mit entsprechend konfiguriertem `AbsoluteDeadband` inkludiert. In der Projektierung im Editor muss bei der Variable in der Eigenschaft **Positiv für Signal** (Variablen-Eigenschaftengruppe **Wertberechnung**) ein Wert ungleich 0 definiert ist.

## 8.2 Konfigurierbare DataChangeTrigger

Bis zenon 7.60 wurden keine konfigurierbaren `DataChangeTrigger` unterstützt.
Für die Kompatibilität von älteren Version mit zenon 7.60 gilt:

- Sind keine `DataChangeFilter` gesetzt, werden die Statusbits **COT_4** und **COT_5** beim Senden von Daten zur Runtime auf `0` gesetzt.

- Sind bereits Werte vorhanden `DataChangeFilter` gesetzt, werden bei einer Wertänderung sowie bei einer Statusänderung die Statusbits **COT_4** und **COT_5** beim Senden von Daten zur Runtime auf `0` gesetzt.

> **Info**
>
> Änderungen des Zeitstempels werden auch in der CEL angezeigt sowie bei der zenon Archivierung berücksichtigt.

## 8.3 Mehrere Subscriptions

Der **OPCUA32 Treiber** erstellt normal eine einzige **Subscription** für die spontane Kommunikation in der zenon Runtime.

Wenn der OPC UA Server eine begrenzte Anzahl an `MonitoredItems` pro **Subscription** unterszützt, kann es erforderlich sein, die Variablen auf mehrere Subscriptions aufzuteilen.

Dazu kann bei der Projektierung im zenon Editor die Eigenschaft der Variable **Datenbaustein** in der Eigenschaftengruppe **Adressierung** verwendet werden. Per Default hat der Datenbaustein den Wert `0`. Damit werden alle Variablen einer **Subscription** hinzugefügt. Wird für einen Teil der Variablen der **Datenbaustein** mit dem Wert `1` projektiert, erstellt der Treiber zwei **Subscriptions**.

> **Achtung**
>
> Es liegt in der Verantwortung des Projektanten die maximale Anzahl der **Subscriptions**, welche vom Server unterstützt werden, zu berücksichtigen. Werden bei den Variablen mehr **Subscriptions** projektiert, als der Server unterstützt, kann dies zu Beeinträchtigungen bei der Kommunikation führen.

## 8.4 Schreiben von Arrays

#### SCHREIBEN VON ARRAYS ALS KOMPLETTES ARRAY ODER MIT EINZELNEN ARRAY-ELEMENTEN

- Die Treiber-Option **Access array variables on index range** berückischtigt Array-Elemente beim Schreiben.
  Ist die Option inaktiv, wird die **IndexRange** auf "`Null`" gesetzt. Diese Option ist für OPC UA Server von Drittherstellern gedacht, welche das optionale Schreiben von einzelnen

Array-Elementen nicht unterstützen.
Damit ist die Kompatibilität zu OPC UA Server, welche das optionale "**Attribute Write Index**" (in den "**Attribute Services**" des "**Core Server Facet**") nicht unterstützen, gewährleistet.
Voraussetzung ist, dass bei allen Array-Elementen, die Eigenschaft **Variable ständig lesen** aktiv gesetzt ist. Der Treiber stellt damit sicher, dass der aktuelle Wert vom gesamten Array vorhanden ist. Beim Schreiben einer Variable aus einem Array, ändert der Treiber entsprechend dieses Element aus dem Array und sendet ein `Write Request` mit dem gesamten Array an den Server. Ist noch kein aktueller Wert für das gesamte Array vorhanden, schlagt das Schreiben fehl.

- Ist die Option **Access array variables on index range** aktiv, nutzt der Treiber das Feld `IndexRange` und liest und schreibt nach Bedarf einzelne Array-Elemente.

# 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

**Achtung:** Die zenon Funktion **Treiberkommandos** ist nicht ident mit Treiberkommandos, die bei Energy-Treibern zur Runtime ausgeführt werden können!

> **Info**
>
> *Dieses Kapitel beschreibt Standardfunktionalitäten, die für die meisten zenon Treiber gültig sind.*
> *Aber 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.*

### PROJEKTIERUNG DER FUNKTION

Die Projektierung erfolgt über die Funktion **Treiberkommandos**.
Dazu:

1. Legen Sie im zenon Editor eine neue Funktion an.
2. Navigieren Sie zum Knoten **Variable**.

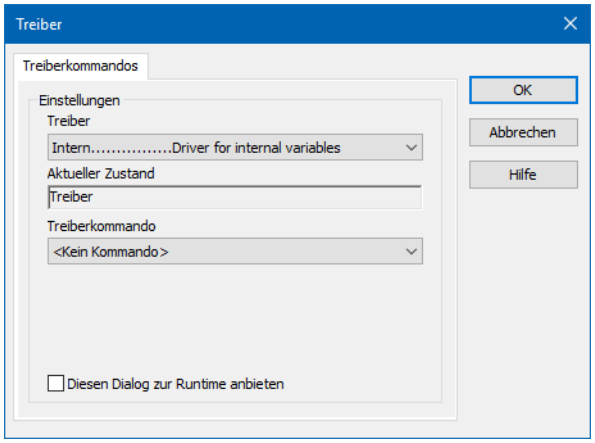
3. Wählen Sie den Eintrag **Treiberkommandos**.

   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 zur Runtime ausgeführt wird.
   Beachten Sie die Hinweise im Abschnitt **Funktion Treiberkommandos im Netzwerk**.

## DIALOG TREIBERKOMMANDOS



| Option | Beschreibung |
|---|---|
| **Treiber** | Auswahl des Treibers aus der Dropdownliste.<br>Diese enthält alle im Projekt geladenen Treibern. |
| **Aktueller Zustand** | Fixer Eintrag, in aktuellen Versionen ohne Funktion. |
| Treiberkommando | Dropdownliste zur Auswahl des Kommandos: |
| <Kein kommando> | Es wird kein Kommando gesendet.<br>Damit kann auch ein bereits bestehendes Kommando aus einer projektierten Funktion entfernt werden. |
| `Treiber starten (Onlinemodus)` | Treiber wird neu initialisiert und gestartet. |
| `Treiber stoppen (Offlinemodus)` | Treiber wird angehalten, es werden keine neuen Daten angenommen.<br>**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.<br>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.<br>Für die Variablen des Treibers werden die Werte von der angeschlossenen Hardware (z. B. SPS, Bussystem, ...) angezeigt. |
| `Treiberspezifisches Kommando` | Eingabe treiberspezifischer Kommandos. Öffnet Eingabefeld für die Eingabe eines Kommandos. |
| `Treiber Sollwertsetzen aktivieren` | Sollwert setzen auf Treiber ist erlaubt. |
| `Treiber Sollwertsetzen deaktivieren` | Sollwert setzen auf Treiber wird verhindert. |
| `Verbindung mit Modem aufbauen` | Verbindung aufbauen (für Modem-Treiber). Öffnet Eingabefelder für Hardware-Adresse und Eingabe der zu wählenden Nummer. |
| `Verbindung mit Modem trennen` | Verbindung beenden (für Modem-Treiber). |
| `Treiber in Simulationsmodus zählend` | Treiber wird in den zählenden Simulationsmodus gesetzt.<br>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` | Treiber wird in den zählenden Simulationsmodus gesetzt.<br>Alle Werte werden mit `0` initialisiert. |
| `Treiber in Simulationsmodus programmiert` | Treiber wird in den zählenden Simulationsmodus gesetzt.<br>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. |
| **Diesen Dialog zur Runtime anbieten** | Dialog wird zur Runtime für Änderungen angeboten. |

#### 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, der dann die gewünschte Aktion auf seinem Treiber durchführt. Zusätzlich sendet der Server das gleiche Treiberkommando zum Standby des Projekts. Der Standby führt die Aktion auch auf seinem Treiber aus.

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

# 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 8.00 -> 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 ZUR FEHLEREINGRENZUNG

- Ist die Steuerung an die Stromversorgung angeschlossen?
- Sind die Teilnehmer im TCP/IP Netz verfügbar?
- Kann die Steuerung über den **Ping** Befehl erreicht werden?
- Kann die Steuerung auf dem entsprechenden Port über **Telnet** erreicht werden?
- Sind Steuerung und PC mit dem passenden Kabel verbunden?
- Wurde die Netzadresse sowohl im Treiberdialog als auch in den Adresseigenschaften der Variablen korrekt eingestellt?
- Wird in der Variablen der richtige Objekttyp verwendet?

- Stimmt die Offset-Adressierung der Variablen mit der Offset-Adressierung in der Steuerung überein?
- Analyse mit Hilfe des Diagnosis Viewers (auf Seite 58): Welche Meldungen werden angezeigt?
- Kann mit einem anderen OPC UA Client kommuniziert werden?