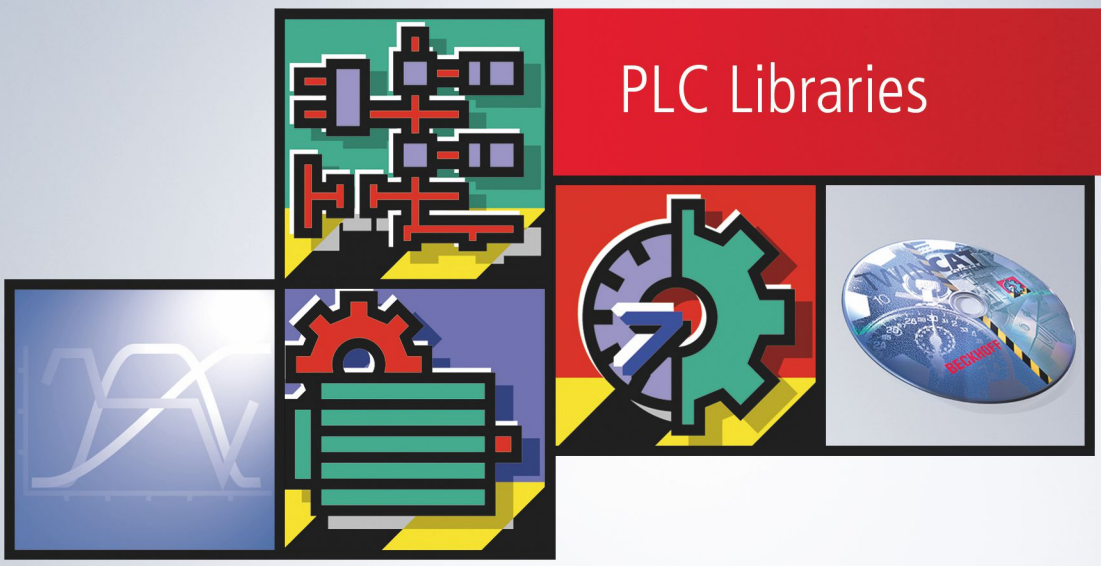
BECKHOFF New Automation Technology

Handbuch | DE

TX1200

TwinCAT 2 | PLC-Bibliothek: TcGENIbus



27.10.2022 | Version: 1.1

Inhaltsverzeichnis

1 Vorwort

1.1 Hinweise zur Dokumentation

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs- und Automatisierungstechnik, das mit den geltenden nationalen Normen vertraut ist.
Zur Installation und Inbetriebnahme der Komponenten ist die Beachtung der Dokumentation und der nachfolgenden Hinweise und Erklärungen unbedingt notwendig.
Das Fachpersonal ist verpflichtet, für jede Installation und Inbetriebnahme die zu dem betreffenden Zeitpunkt veröffentliche Dokumentation zu verwenden.

Das Fachpersonal hat sicherzustellen, dass die Anwendung bzw. der Einsatz der beschriebenen Produkte alle Sicherheitsanforderungen, einschließlich sämtlicher anwendbaren Gesetze, Vorschriften, Bestimmungen und Normen erfüllt.

Disclaimer

Diese Dokumentation wurde sorgfältig erstellt. Die beschriebenen Produkte werden jedoch ständig weiter entwickelt.
Wir behalten uns das Recht vor, die Dokumentation jederzeit und ohne Ankündigung zu überarbeiten und zu ändern.
Aus den Angaben, Abbildungen und Beschreibungen in dieser Dokumentation können keine Ansprüche auf Änderung bereits gelieferter Produkte geltend gemacht werden.

Marken

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® und XPlanar® sind eingetragene und lizenzierte Marken der Beckhoff Automation GmbH.
Die Verwendung anderer in dieser Dokumentation enthaltenen Marken oder Kennzeichen durch Dritte kann zu einer Verletzung von Rechten der Inhaber der entsprechenden Bezeichnungen führen.

Patente

Die EtherCAT-Technologie ist patentrechtlich geschützt, insbesondere durch folgende Anmeldungen und Patente:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
mit den entsprechenden Anmeldungen und Eintragungen in verschiedenen anderen Ländern.

EtherCAT®

EtherCAT® ist eine eingetragene Marke und patentierte Technologie lizenziert durch die Beckhoff Automation GmbH, Deutschland

Copyright

1.2 Sicherheitshinweise

Sicherheitsbestimmungen

Beachten Sie die folgenden Sicherheitshinweise und Erklärungen!
Produktspezifische Sicherheitshinweise finden Sie auf den folgenden Seiten oder in den Bereichen Montage, Verdrahtung, Inbetriebnahme usw.

Haftungsausschluss

Die gesamten Komponenten werden je nach Anwendungsbestimmungen in bestimmten Hard- und Software-Konfigurationen ausgeliefert. Änderungen der Hard- oder Software-Konfiguration, die über die dokumentierten Möglichkeiten hinausgehen, sind unzulässig und bewirken den Haftungsausschluss der Beckhoff Automation GmbH & Co. KG.

Qualifikation des Personals

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs-, Automatisierungs- und Antriebstechnik, das mit den geltenden Normen vertraut ist.

Erklärung der Symbole

In der vorliegenden Dokumentation werden die folgenden Symbole mit einem nebenstehenden Sicherheitshinweis oder Hinweistext verwendet. Die Sicherheitshinweise sind aufmerksam zu lesen und unbedingt zu befolgen!

⚠ GEFAHR

Akute Verletzungsgefahr!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, besteht unmittelbare Gefahr für Leben und Gesundheit von Personen!

⚠ WARNUNG

Verletzungsgefahr!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, besteht Gefahr für Leben und Gesundheit von Personen!

⚠ VORSICHT

Schädigung von Personen!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, können Personen geschädigt werden!

HINWEIS

Schädigung von Umwelt oder Geräten

Wenn der Hinweis neben diesem Symbol nicht beachtet wird, können Umwelt oder Geräte geschädigt werden.

Tipp oder Fingerzeig

Dieses Symbol kennzeichnet Informationen, die zum besseren Verständnis beitragen.

1.3 Hinweise zur Informationssicherheit

Die Produkte der Beckhoff Automation GmbH & Co. KG (Beckhoff) sind, sofern sie online zu erreichen sind, mit Security-Funktionen ausgestattet, die den sicheren Betrieb von Anlagen, Systemen, Maschinen und Netzwerken unterstützen. Trotz der Security-Funktionen sind die Erstellung, Implementierung und ständige Aktualisierung eines ganzheitlichen Security-Konzepts für den Betrieb notwendig, um die jeweilige Anlage, das System, die Maschine und die Netzwerke gegen Cyber-Bedrohungen zu schützen. Die von Beckhoff verkauften Produkte bilden dabei nur einen Teil des gesamtheitlichen Security-Konzepts. Der Kunde ist dafür verantwortlich, dass unbefugte Zugriffe durch Dritte auf seine Anlagen, Systeme, Maschinen und Netzwerke verhindert werden. Letztere sollten nur mit dem Unternehmensnetzwerk oder dem Internet verbunden werden, wenn entsprechende Schutzmaßnahmen eingerichtet wurden.

Zusätzlich sollten die Empfehlungen von Beckhoff zu entsprechenden Schutzmaßnahmen beachtet werden. Weiterführende Informationen über Informationssicherheit und Industrial Security finden Sie in unserem https://www.beckhoff.de/secguide.

Die Produkte und Lösungen von Beckhoff werden ständig weiterentwickelt. Dies betrifft auch die Security-Funktionen. Aufgrund der stetigen Weiterentwicklung empfiehlt Beckhoff ausdrücklich, die Produkte ständig auf dem aktuellen Stand zu halten und nach Bereitstellung von Updates diese auf die Produkte aufzuspielen. Die Verwendung veralteter oder nicht mehr unterstützter Produktversionen kann das Risiko von Cyber-Bedrohungen erhöhen.

Um stets über Hinweise zur Informationssicherheit zu Produkten von Beckhoff informiert zu sein, abonnieren Sie den RSS Feed unter https://www.beckhoff.de/secinfo.

2 Zielgruppen

Für den Nutzer dieser Bibliothek werden folgende Grundkenntnisse vorausgesetzt:

- TwinCAT PLC Control
- TwinCAT System Manager
- PC und Netzwerkkenntnisse
- Aufbau und Eigenschaften der Beckhoff Embedded-PC und deren Busklemmensystem
- Serielle Kommunikation (RS485) und GENIbus-Protokoll
- Einschlägige Sicherheitsvorschriften der technischen Gebäudeausrüstung

Diese Softwarebibliothek ist für Gebäudeautomation-Systempartner der Beckhoff Automation GmbH & Co. KG. Die Systempartner sind tätig in dem Bereich Gebäudeautomation und beschäftigen sich mit Errichtung, Inbetriebsetzung, Erweiterung, Wartung und Service von mess-, steuer- und regelungstechnischen Anlagen der technischen Gebäudeausrüstung.

3 GENIbus

Die TwinCAT SPS Bibliothek **TcGENIbus.Lib** beinhaltet Kommunikationsbausteine für die GENIbus Master-/Slave-Kommunikation aus der TwinCAT SPS. GENIbus (Grundfos Electronic Network Intercommunications bus) ist ein Protokoll, welches von der Firma Grundfos speziell für den Datenaustausch mit deren Geräten entwickelt worden ist. Mehrere Grundfos Geräte können über GENIbus zu einem Netzwerk verbunden und in ein Automatisierungssystem integriert werden.

GENIbus basiert auf der RS485-Hardwareschnittstelle. Der Datenaustausch erfolgt mit 9600 Baud. Ein GENIbus Netzwerk besitzt in den meisten Fällen einen Master und bis zu 200 Slaves.

Weiterführende Dokumentation

- GENIbus Protocol Specification
- Grundfos: Operating the MAGNA3 and MGE model H/I via the GENIpro interface

3.1 Geräte Adressierung

Das GENIbus-Protokoll kennt grundsätzlich nur zwei Adressierungsarten: Einzeladressierung und Broadcast- bzw. Sammelbefehle. Dabei sind die Adressen wie folgt vergeben:

- 0 - 31 : Master Adressen, d.h. die TwinCAT-Steuerungen
- 32 - 231 : Slaveadressen, z. B. Pumpen
- 255 : Broadcast-Adressierung an alle Slaves

Auf der Bausteinebene in der Bibliothek wird der Adressbereich der Slaves auf 1 - 200 gesetzt, also 31 weniger als im seriellen Netzwerk. Der Grund dafür ist, dass die Grundfos-Parametriergeräte ebenfalls mit einem Adressbereich von 1 - 200 arbeiten. Für die serielle Kommunikation wird intern wieder 31 auf die Slave Adresse aufaddiert.

3.2 Verdrahtung

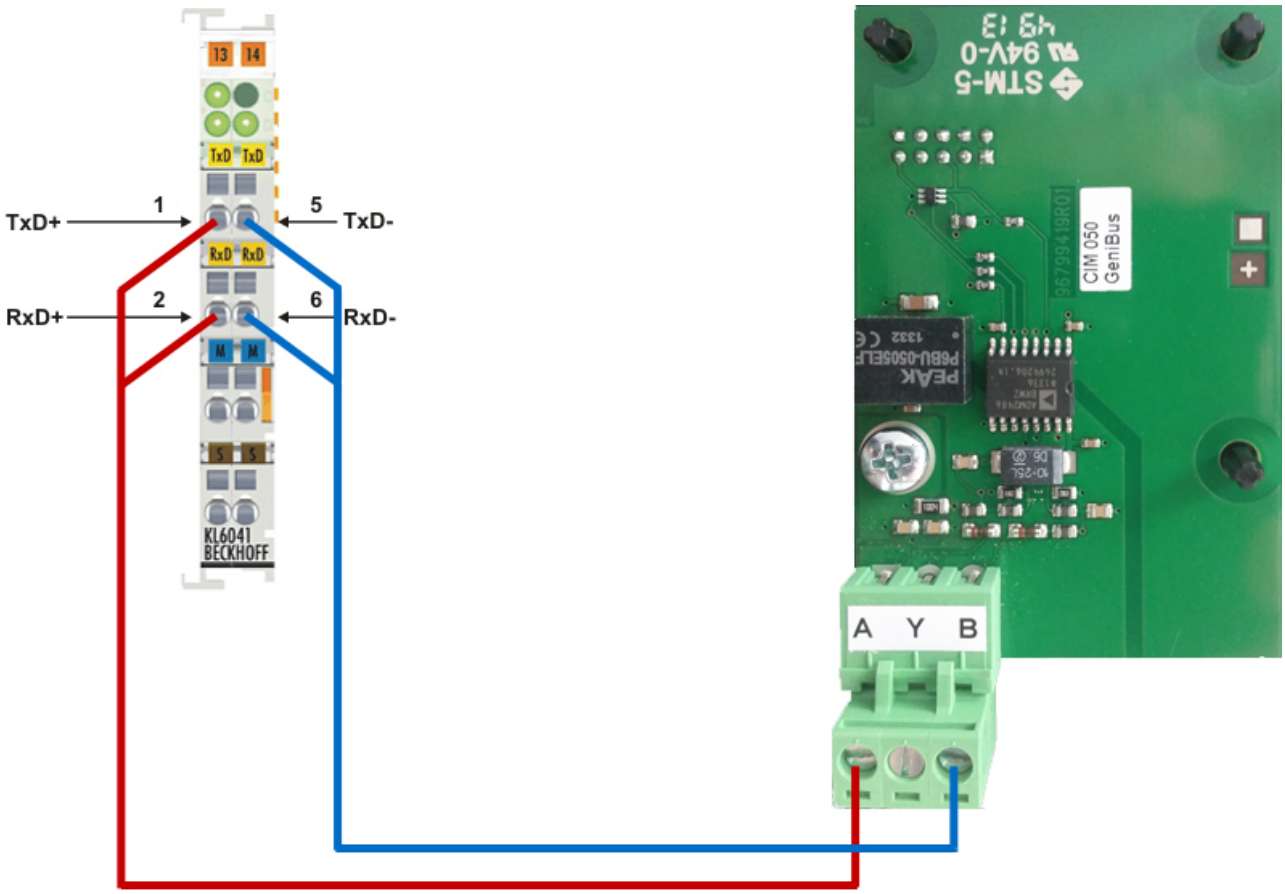
Die GENIbus-Kommunikation erfolgt auf Basis des RS485-Standard im Halbduplex-Verfahren. Dazu ist eine Zweidraht-Kommunikation aufzubauen mit einer plus- bzw. A-Ader und einer minus- bzw. B-Ader. Die seriellen Klemmen KL6021, KL6041 und EL6021, aber auch die serielle Schnittstelle des CX9020 beherrschen sowohl das Halb- als auch das Vollduplex-Verfahren. Hier sind plus und minus noch einmal in Senden (Transmit-Data, TxD) und Empfangen (Receive-Data, RxD) unterteilt. Für die Verwendung des Halbduplex-Verfahrens sind daher jeweils TxD und RxD zu verbinden. Die serielle Schnittstelle des CX8080 beherrscht nur das Halbduplex-Verfahren, so dass hier ein Brücken von Pins nicht notwendig ist.

HINWEIS

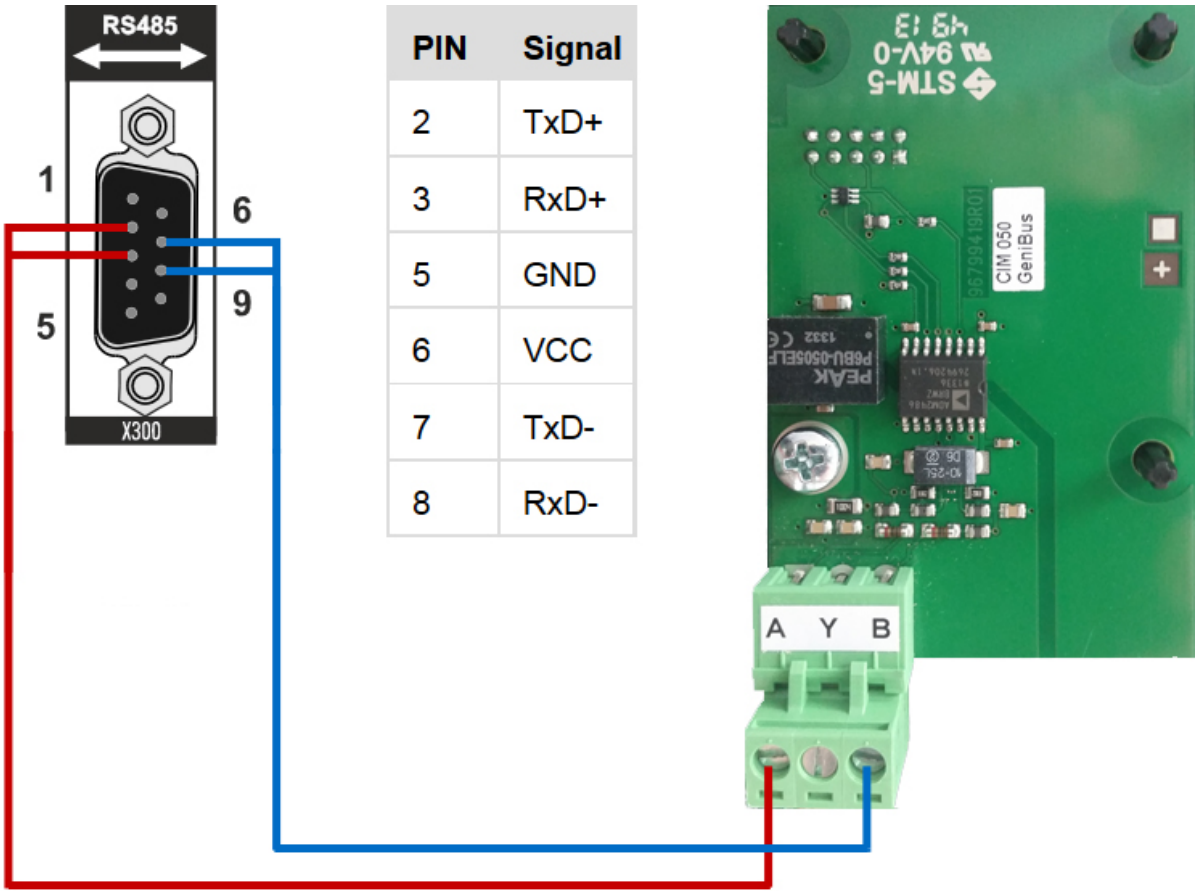
Die erwähnte Belegung der plus- und minus-Klemme am GENIbus-Modul (A für plus, B für minus) gilt für das zum Test verwendete Gerät CIM 050. Bitte lesen Sie sich die Dokumentation Ihres Modules für die spezifische Klemmbelegung genau durch. Des weiteren behandelt dieses Kapitel nur die Verdrahtung der Kommunikationsadern. Für eine längere Verdrahtung ist eine Abschirmung unerlässlich. Lesen Sie hierzu bitte ebenfalls die Dokumentation der verwendeten Komponenten.

Es ergeben sich folgende Anschlussbilder:

KL6021, KL6041 und EL6021

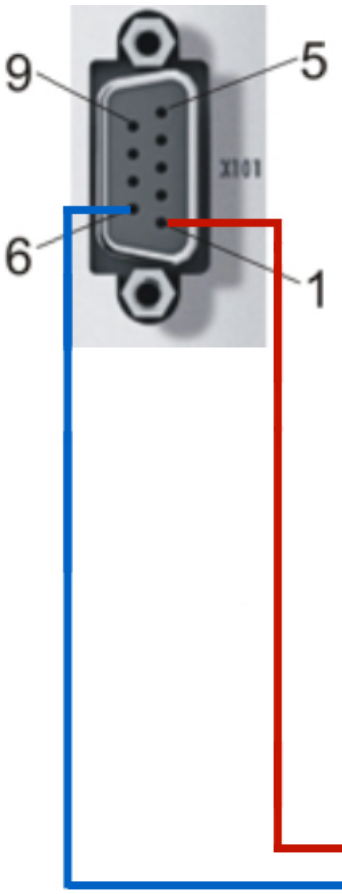


CX9020 (als Programmbeispiel in dieser Form nicht vorhanden)

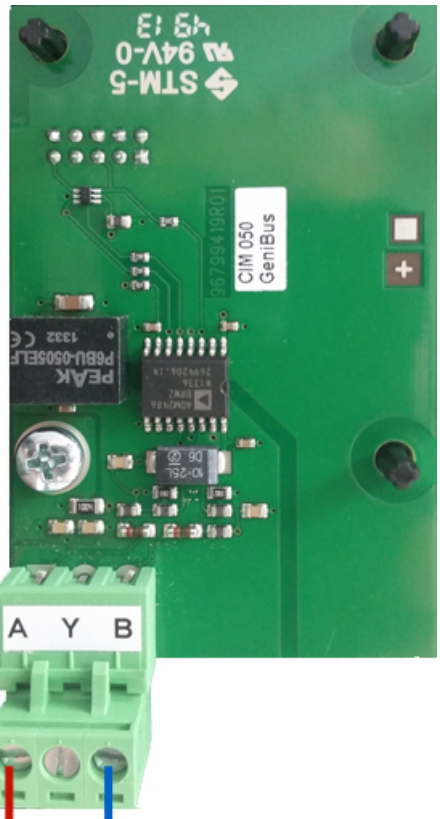


PIN	Signal
2	TxD+
3	RxD+
5	GND
6	VCC
7	TxD-
8	RxD-

CX8080



PIN	Bedeutung	Signal
1	**RS485**	**A (+)**
2	RxD (RS232)	Receive Data
3	TxD (RS232)	Transmit Data
4	+ 5 V	Vcc
5	GND	Ground
6	**RS485**	**B (-)**
7	RTS (RS232)	Request to Send
8	CTS (RS232)	Clear to Send
9	GND	Ground



4 Integration in TwinCAT

4.1 Integration in TwinCAT (CX5010)

Dieses Programm zeigt die Anwendung der einzelnen Funktionsbausteine in https://infosys.beckhoff.com/content/1031/tcplclibgenibus/Resources/12167387787.zip. https://infosys.beckhoff.com/content/1031/tcplclibgenibus/Resources/12167387787.zip

Die Kommunikation läuft über eine EtherCAT-Klemme.

Hardware

Einrichtung der Komponenten

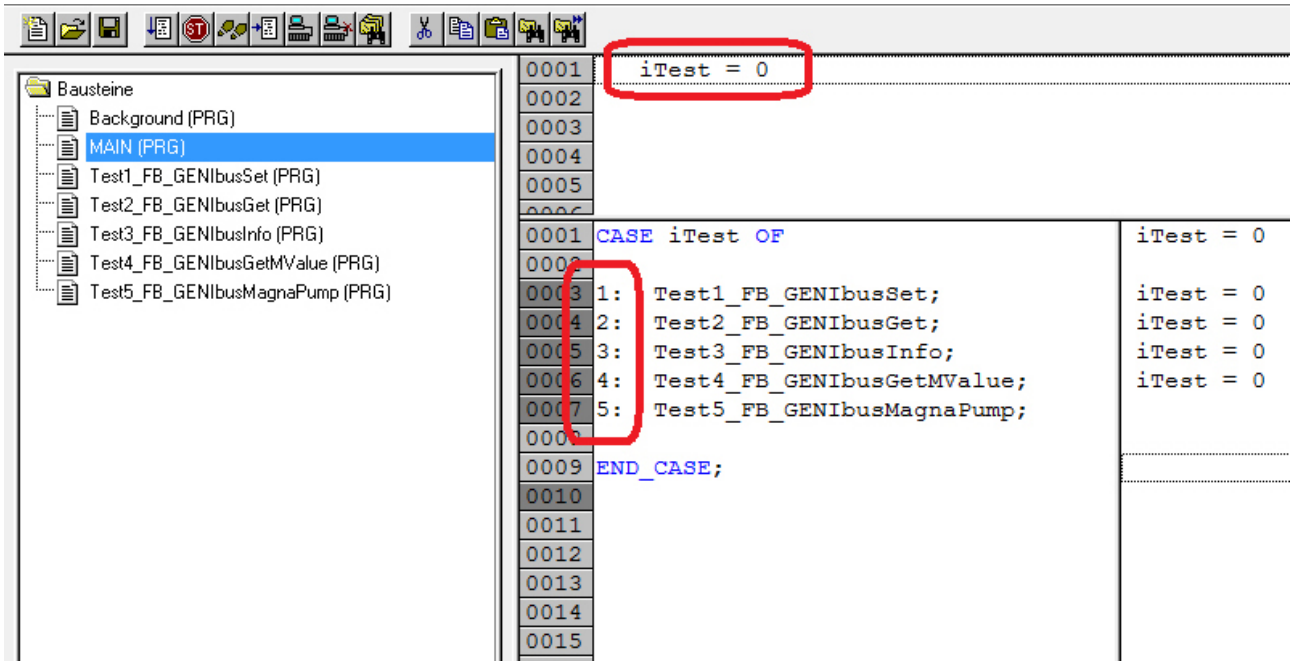
Es wird folgende Hardware benötigt:

- 1x Embedded-PC CX5010
- 1x Digitale 8-Kanal-Eingangsklemme EL1408 für die Ausführung der einzelnen Tests
- 1x serielle RS485-Klemme EL6021

Software

SPS-Programm

Im MAIN-Programm lässt sich durch Setzen der Variable *iTest* auf Werte von 1 bis 5 der jeweilige Test-Programmteil auswählen.



In den jeweiligen Programmteilen sind dann Funktionsbausteine vorbereitet, die Sie durch die Testeingänge *ib1* bis *ib8* bedienen können:

```
VAR_GLOBAL
   ib1      AT %I* : BOOL;
   ib2      AT %I* : BOOL;
   ib3      AT %I* : BOOL;
   ib4      AT %I* : BOOL;
   ib5      AT %I* : BOOL;
   ib6      AT %I* : BOOL;
   ib7      AT %I* : BOOL;
   ib8      AT %I* : BOOL;
```

```
   stInData     AT %I* : ST_GENIbusInData;
   stOutData    AT %Q* : ST_GENIbusOutData;
   stCommandBuffer     : ST_GENIbusCommandBuffer;
END_VAR
```

ib1..ib8: Tast- Schalteingänge für die Tests.

stInData: Struktur mit den Eingangsvariablen für verschiedene Klemmentypen.

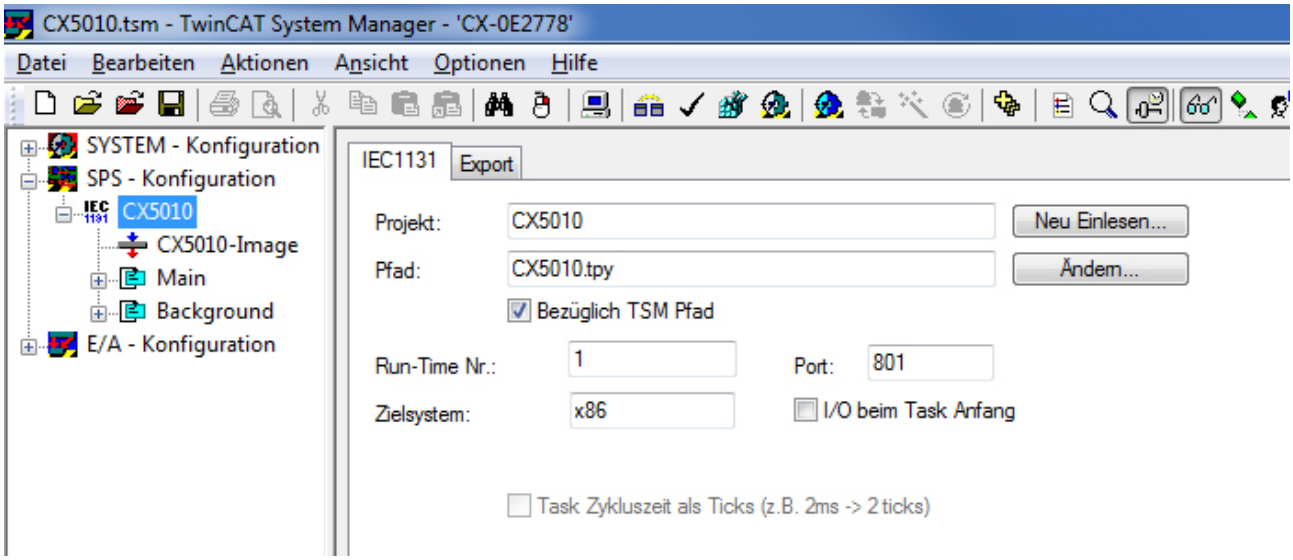
stOutData: Struktur mit den Ausgangsvariablen für verschiedene Klemmentypen.

stCommandBuffer: Verweis auf die Struktur zur Kommunikation (Puffer) mit dem FB_GENIbusCommunication [▶ 45]()-Baustein.

TwinCAT System Manager

Im TwinCAT System Manager sind bereits die Variablen verknüpft und den beiden Tasks (Background: schnell für Kommunikation, Main: langsamer für Applikation) zugeordnet.

Übersetzen Sie das SPS-Programm und lesen Sie es im TwinCAT System Manager ein:



Überprüfen Sie bitte nach dieser Anleitung [▶ 17], ob die Variablenzuordnung korrekt ist und die Variablen verknüpft sind.

Danach können Sie den TwinCAT System Manager aktivieren und das SPS-Programm laden und starten.

Sehen Sie dazu auch

- ST_GENIbusInData [▶ 60]
- ST_GENIbusOutData [▶ 62]
- ST_GENIbusCommandBuffer [▶ 58]

4.2 Integration in TwinCAT (CX8080)

Dieses Programm zeigt die Anwendung der einzelnen Funktionsbausteine in https://infosys.beckhoff.com/content/1031/tcplclibgenibus/Resources/12167389195.zip. https://infosys.beckhoff.com/content/1031/tcplclibgenibus/Resources/12167389195.zip

Die Kommunikation läuft über die onBoard PC-Schnittstelle des CX.

Hardware

Einrichtung der Komponenten

Es wird folgende Hardware benötigt:

- 1x Embedded-PC CX8080
- 1x Digitale 8-Kanal-Eingangsklemme KL1408 für die Ausführung der einzelnen Tests
- 1x Endklemme KL9010

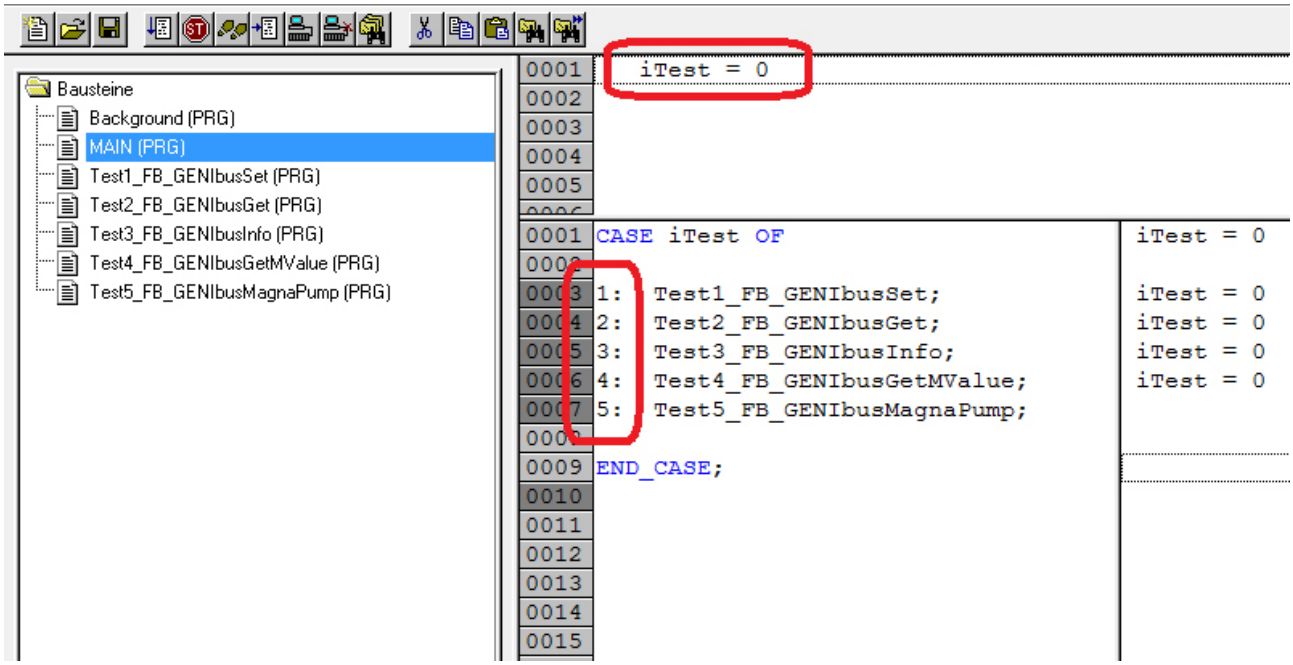
Die RS485 Kommunikationspins der Sub-D-Schnittstelle sind:

- A (+) = Pin1
- B (-) = Pin6

Software

SPS-Programm

Im MAIN-Programm lässt sich durch Setzen der Variable *iTest* auf Werte von 1 bis 5 der jeweilige Test-Programmteil auswählen.



In den jeweiligen Programmteilen sind dann Funktionsbausteine vorbereitet, die Sie durch die Testeingänge *ib1* bis *ib8* bedienen können:

```
VAR_GLOBAL
   ib1      AT %I* : BOOL;
   ib2      AT %I* : BOOL;
   ib3      AT %I* : BOOL;
   ib4      AT %I* : BOOL;
   ib5      AT %I* : BOOL;
   ib6      AT %I* : BOOL;
   ib7      AT %I* : BOOL;
   ib8      AT %I* : BOOL;

   stInData     AT %I* : ST_GENIbusInData;
   stOutData    AT %Q* : ST_GENIbusOutData;
   stCommandBuffer     : ST_GENIbusCommandBuffer;
END_VAR
```

ib1..ib8: Tast- Schalteingänge für die Tests.

stInData: Struktur mit den Eingangsvariablen für verschiedene Klemmentypen.

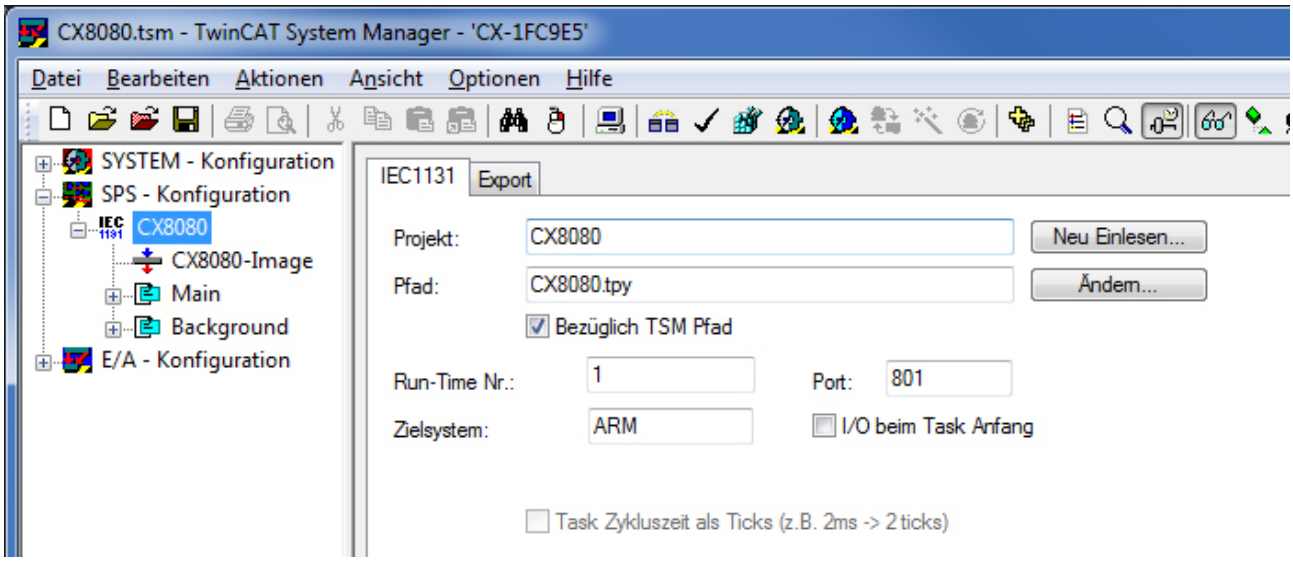
stOutData: Struktur mit den Ausgangsvariablen für verschiedene Klemmentypen.

stCommandBuffer: Verweis auf die Struktur zur Kommunikation (Puffer) mit dem FB_GENIbusCommunication [▶ 45]()-Baustein.

TwinCAT System Manager

Im TwinCAT System Manager sind bereits die Variablen verknüpft und den beiden Tasks (Background: schnell für Kommunikation, Main: langsamer für Applikation) zugeordnet.

Übersetzen Sie das SPS-Programm und lesen Sie es im TwinCAT System Manager ein:



Überprüfen Sie bitte nach dieser Anleitung [▶ 17], ob die Variablenzuordnung korrekt ist und die Variablen verknüpft sind.

Danach können Sie den TwinCAT System Manager aktivieren und das SPS-Programm laden und starten.

Sehen Sie dazu auch

- ST_GENIbusInData [▶ 60]
- ST_GENIbusOutData [▶ 62]
- ST_GENIbusCommandBuffer [▶ 58]

4.3 Integration in TwinCAT (CX9020)

Dieses Programm zeigt die Anwendung der einzelnen Funktionsbausteine in https://infosys.beckhoff.com/content/1031/tcplclibgenibus/Resources/12167390603.ziphttps://infosys.beckhoff.com/content/1031/tcplclibgenibus/Resources/12167390603.zip.

Die Kommunikation läuft über eine K-Bus-Klemme.

Hardware

Einrichtung der Komponenten

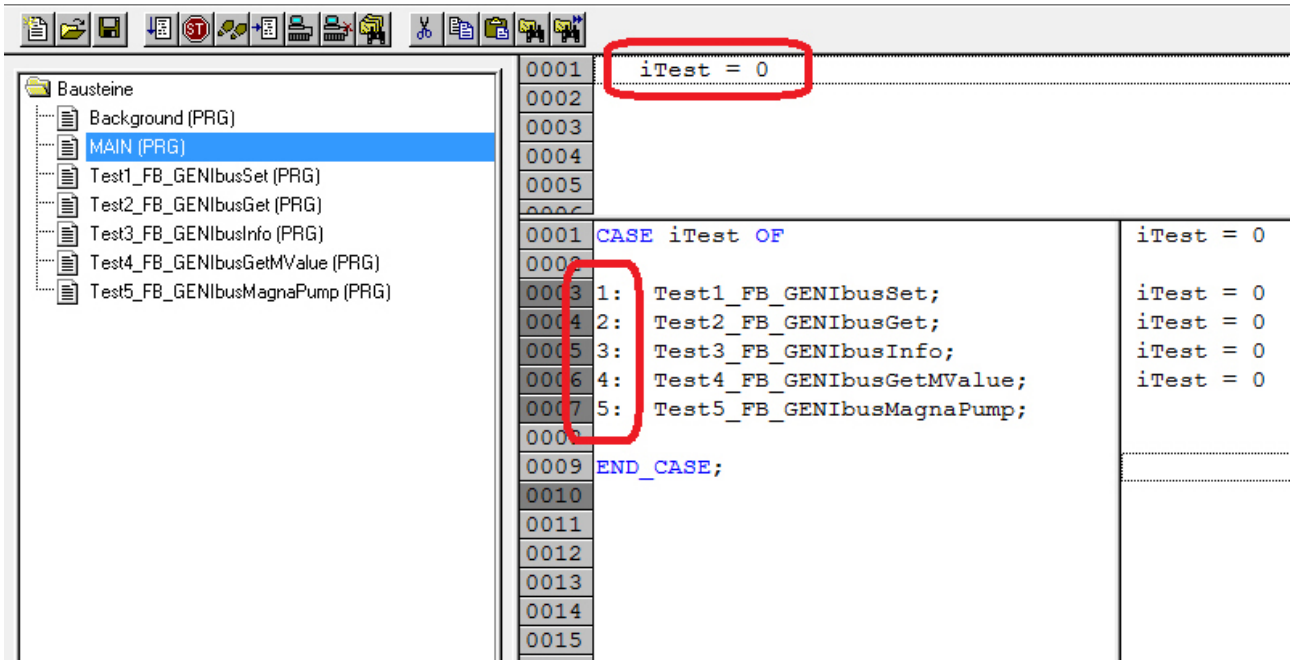
Es wird folgende Hardware benötigt:

- 1x Embedded-PC CX9020
- 1x Digitale 8-Kanal-Eingangsklemme KL1408 für die Ausführung der einzelnen Tests
- 1x serielle RS485-Klemme KL6041
- 1x Endklemme KL9010

Software

SPS-Programm

Im MAIN-Programm lässt sich durch Setzen der Variable *iTest* auf Werte von 1 bis 5 der jeweilige Test-Programmteil auswählen.



In den jeweiligen Programmteilen sind dann Funktionsbausteine vorbereitet, die Sie durch die Testeingänge *ib1* bis *ib8* bedienen können:

```
VAR_GLOBAL
   ib1      AT %I* : BOOL;
   ib2      AT %I* : BOOL;
   ib3      AT %I* : BOOL;
   ib4      AT %I* : BOOL;
   ib5      AT %I* : BOOL;
   ib6      AT %I* : BOOL;
   ib7      AT %I* : BOOL;
   ib8      AT %I* : BOOL;

   stInData     AT %I* : ST_GENIbusInData;
   stOutData    AT %Q* : ST_GENIbusOutData;
   stCommandBuffer     : ST_GENIbusCommandBuffer;
END_VAR
```

ib1..ib8: Tast- Schalteingänge für die Tests.

stInData: Struktur mit den Eingangsvariablen für verschiedene Klemmentypen.

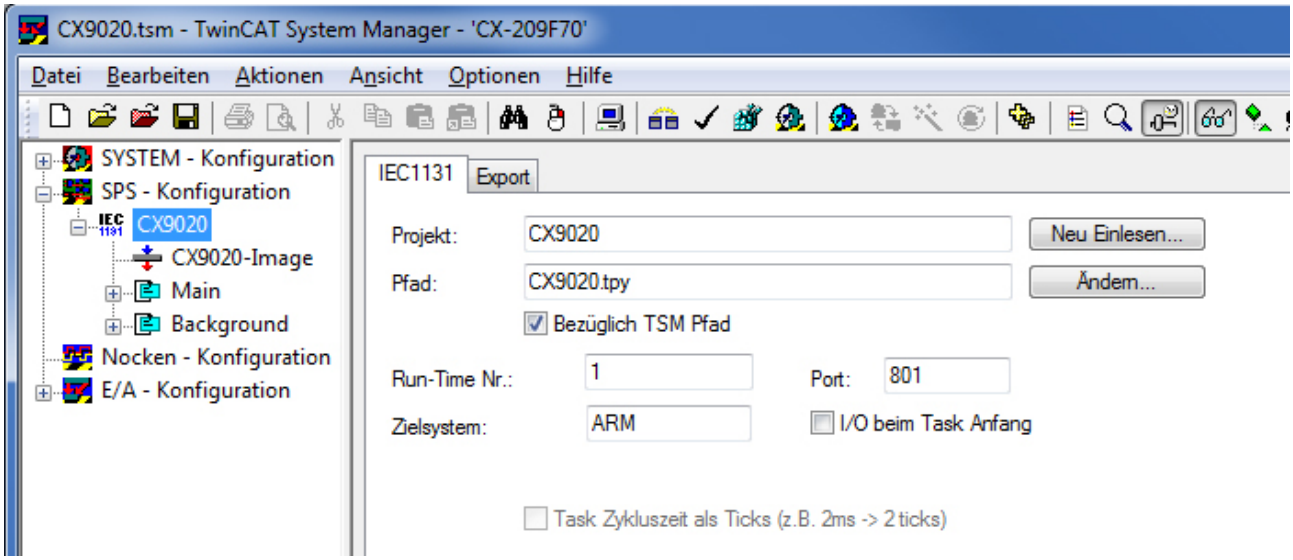
stOutData: Struktur mit den Ausgangsvariablen für verschiedene Klemmentypen.

stCommandBuffer: Verweis auf die Struktur zur Kommunikation (Puffer) mit dem FB_GENIbusCommunication [▶ 45]()-Baustein.

TwinCAT System Manager

Im TwinCAT System Manager sind bereits die Variablen verknüpft und den beiden Tasks (Background: schnell für Kommunikation, Main: langsamer für Applikation) zugeordnet.

Übersetzen Sie das SPS-Programm und lesen Sie es im TwinCAT System Manager ein:



Überprüfen Sie bitte nach dieser Anleitung [▶ 17], ob die Variablenzuordnung korrekt ist und die Variablen verknüpft sind.

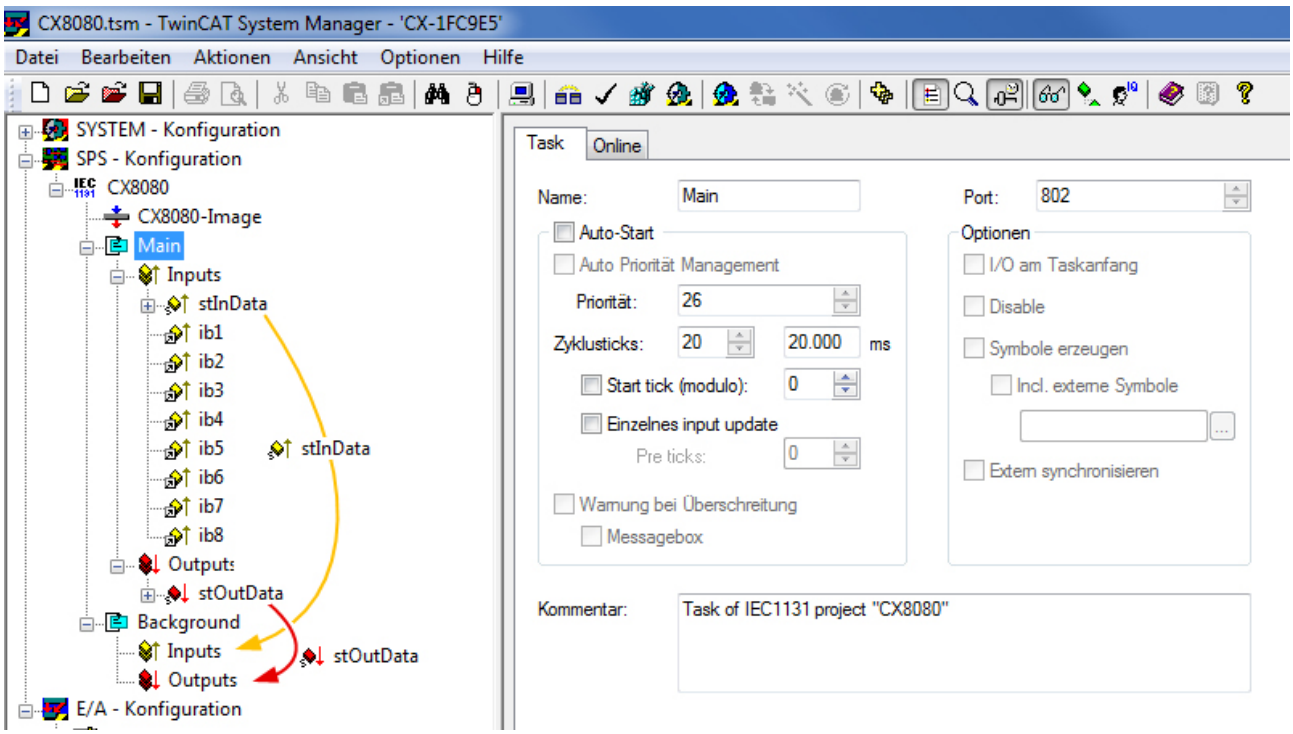
Danach können Sie den TwinCAT System Manager aktivieren und das SPS-Programm laden und starten.

Sehen Sie dazu auch

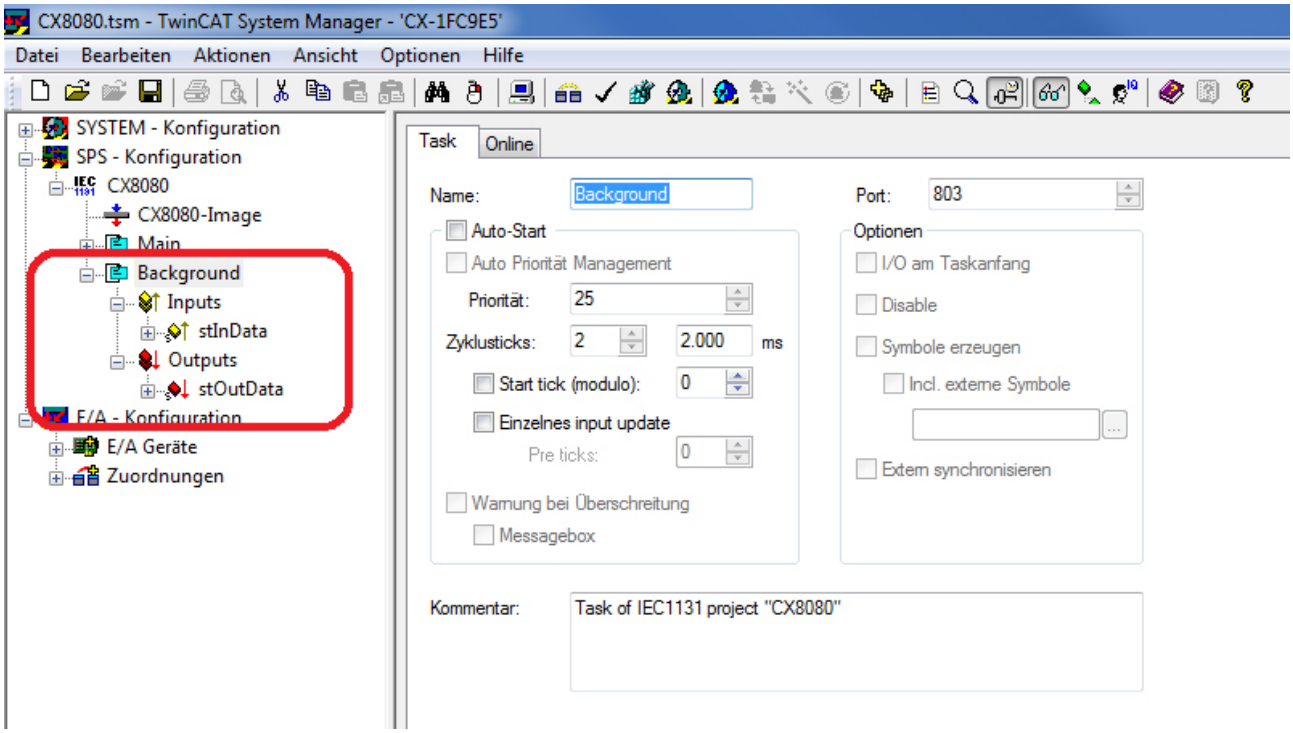
- ST_GENIbusInData [▶ 60]
- ST_GENIbusOutData [▶ 62]
- ST_GENIbusCommandBuffer [▶ 58]

4.4 Konfigurationen im TwinCAT System Manager

Ist das SPS-Programm fehlerfrei übersetzt, ist es wichtig, dass die Kommunikations-Variablen im TwinCAT System Manager der richtigen Task zugeordnet sind. Anderenfalls würden diese nicht mit der gewünschten Zykluszeit abgefragt bzw. beschrieben werden.
Nach dem Einlesen des SPS-Programms ist es in der Regel der Fall, dass die Kommunikationsvariablen *stInData* und *stOutData* der langsameren Task MAIN zugeordnet sind. Per "drag&drop" lassen sich diese in die schnelle Task, hier "Background" ziehen:



Danach sollte sich folgendes Bild ergeben:

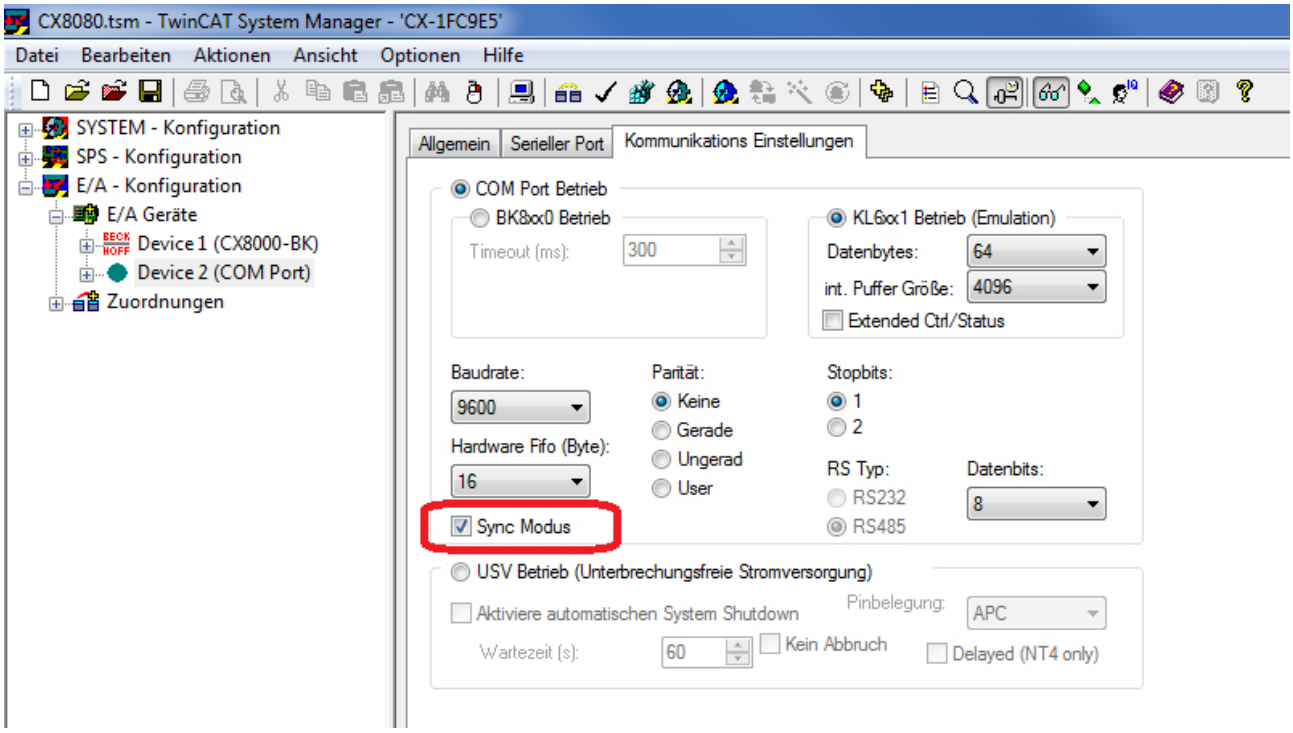


Nun müssen die Kommunikationsvariablen mit der verwendeten Hardware verbunden werden. Dabei stehen diese Möglichkeiten zur Auswahl:

- OnBoard-RS485-Schnittstelle (PcComm) [▶ 19]
- KL6021 (5Byte) [▶ 24]
- KL6041 (22Byte) [▶ 29]
- EL6021 (22Byte) [▶ 33]

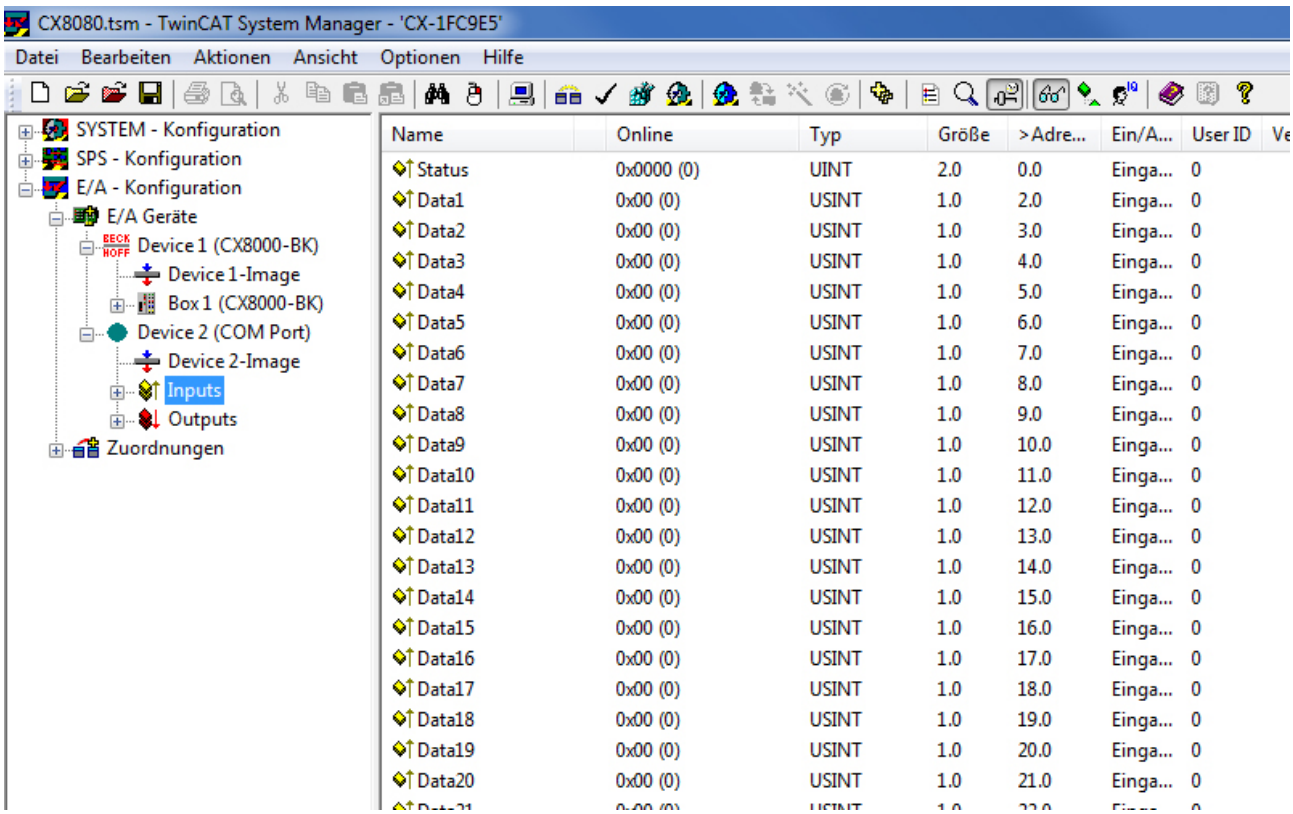
4.5 Einstellungen bei Verwendung der OnBoard-RS485-Schnittstelle

Wird eine OnBoard-Schnittstelle gewählt, so muss diese zunächst im TwinCAT System Manager richtig eingestellt werden, nicht zuletzt, um das richtige Prozessabbild zu erhalten. Anders als bei den Klemmen-basierten seriellen Schnittstellen werden im TwinCAT System Manager auch die Kommunikationsparameter eingestellt. Unter dem Karteireiter "Communication-Properties" des COM-Ports sind exakt die folgenden Einstellungen vorzunehmen:

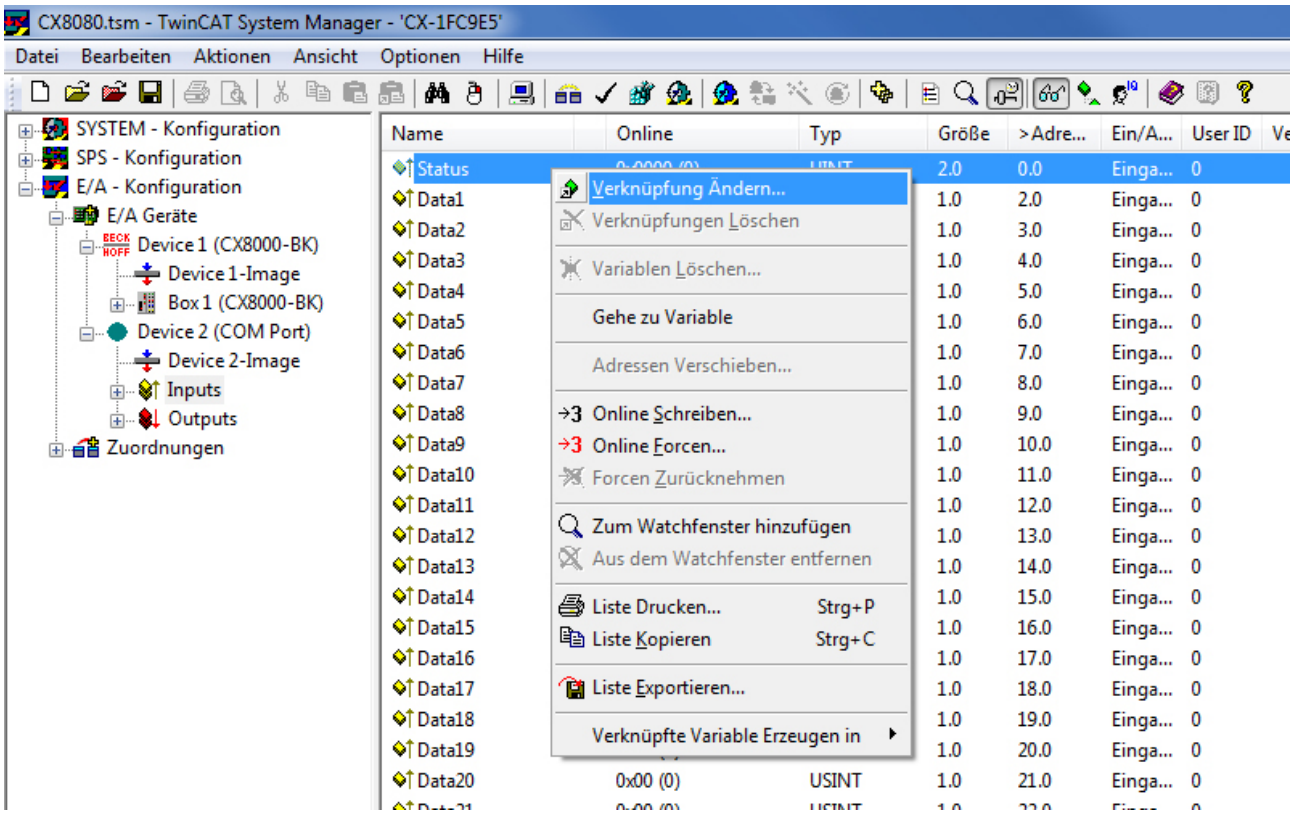


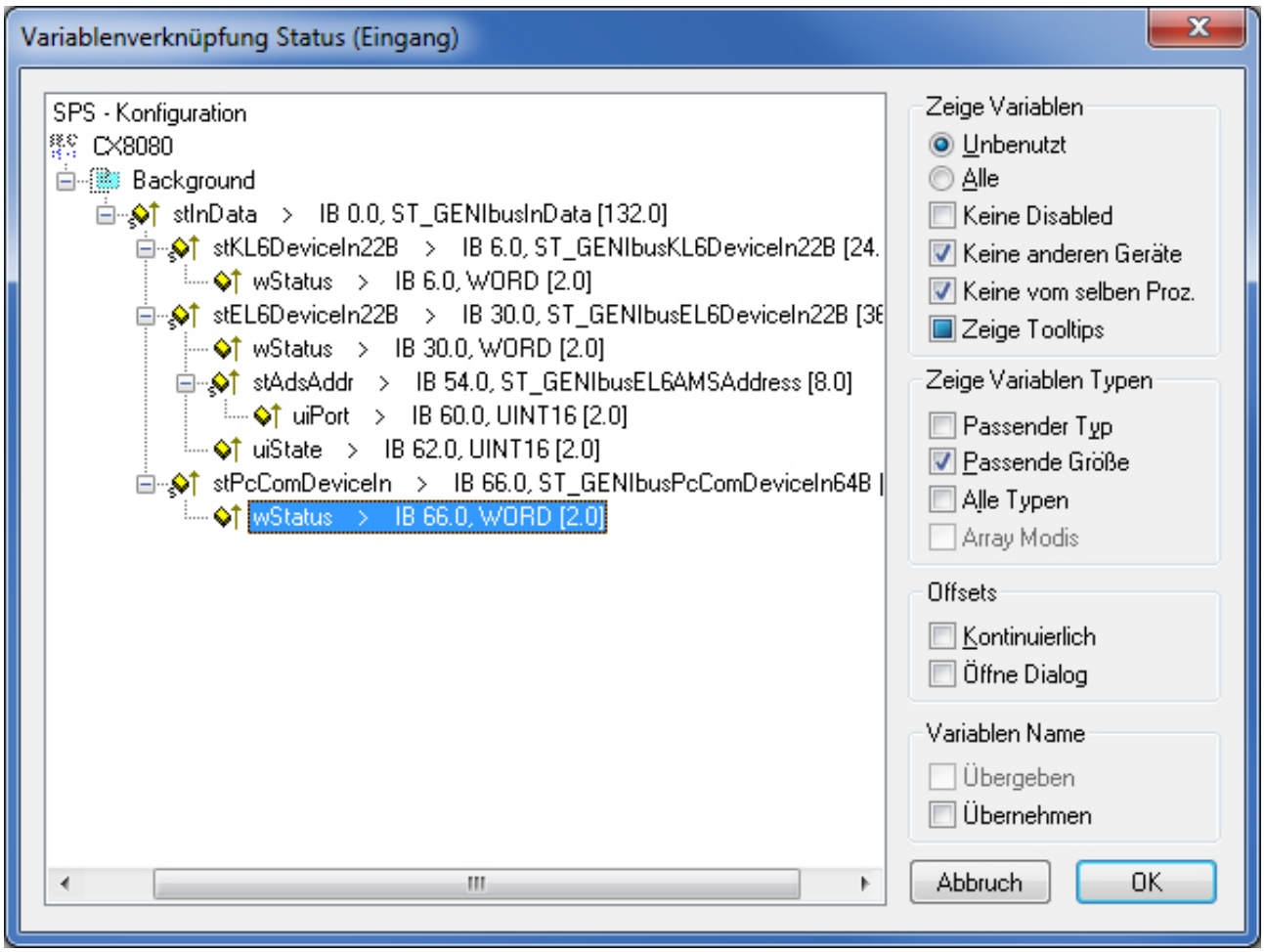
Besonders wichtig ist die Aktivierung des "Sync-Mode". Dieser sorgt dafür, dass die Schnittstelle, welche eine KL6xx1 nur emuliert, auch bei höheren SPS-Auslastungen richtig arbeitet.

Die Verbindung des Prozessabbildes zu den SPS- Ein- und Ausgangsvariablen lässt sich am einfachsten von der Hardware-Seite her realisieren, da von dort aus Multi-Verknüpfungen möglich sind. Die Variablen müssen dazu im rechten Teil des TwinCAT System Managers zu sehen sein:

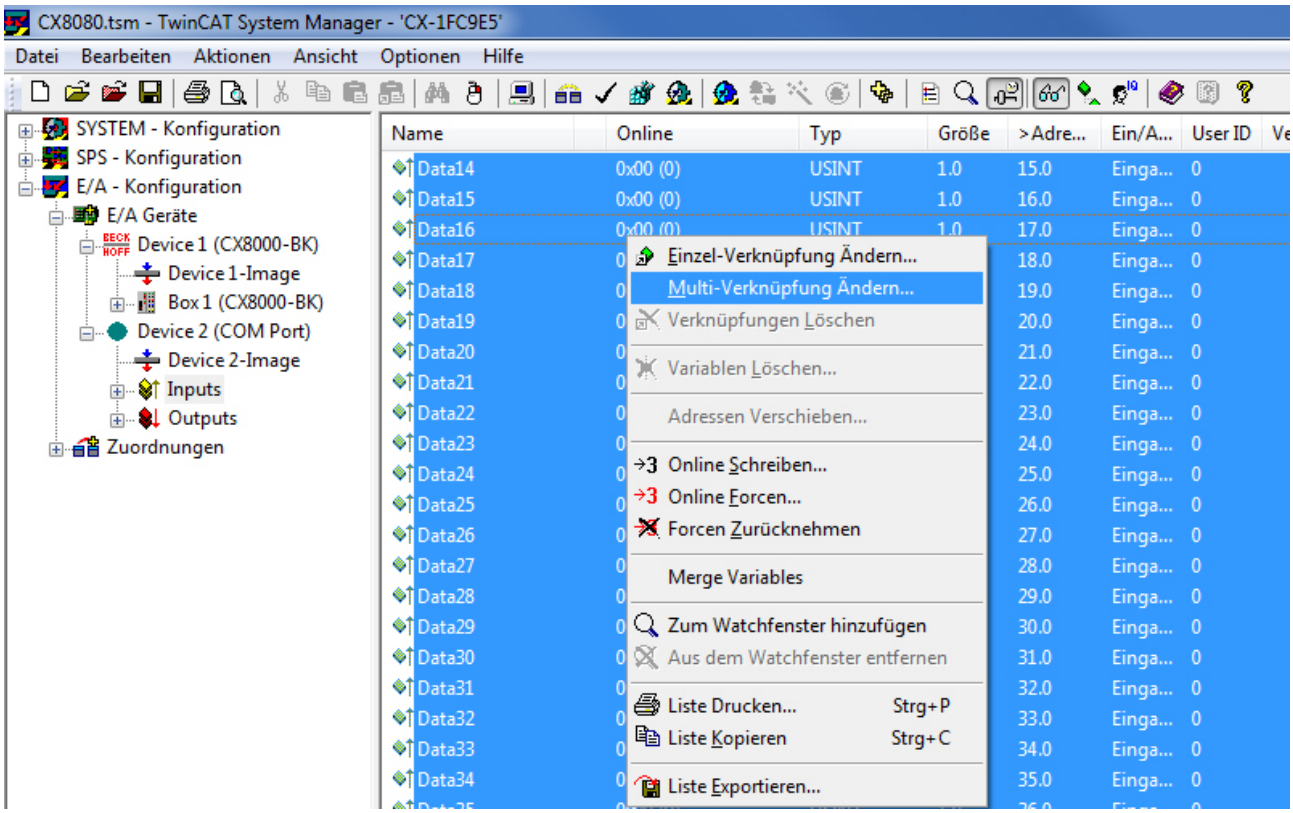


Zunächst wird allein der Status mit der Statusvariable des Kommunikationseingangs verbunden. Es ist dabei zu beachten, die Struktur für PC-Kommunikation zu wählen.

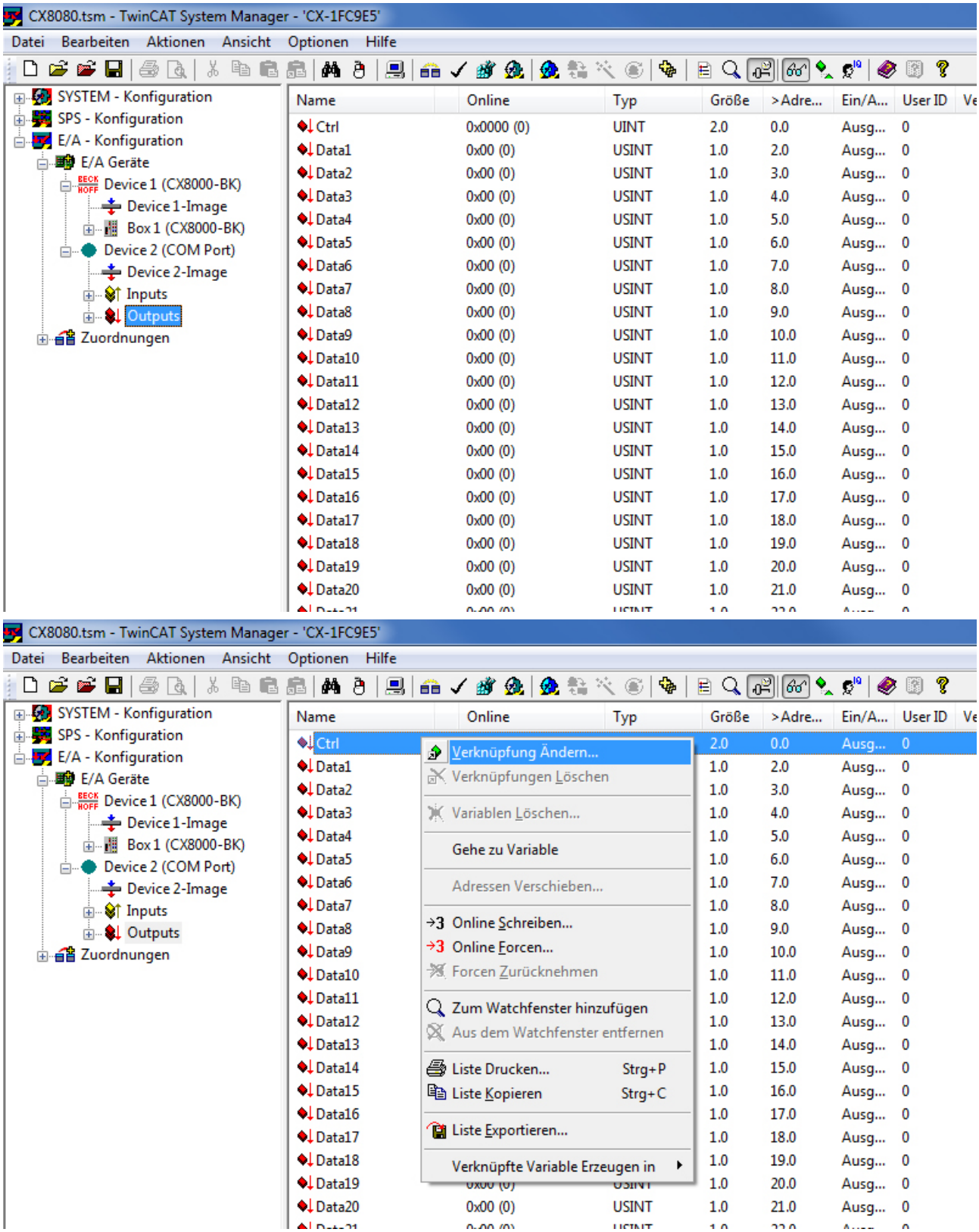


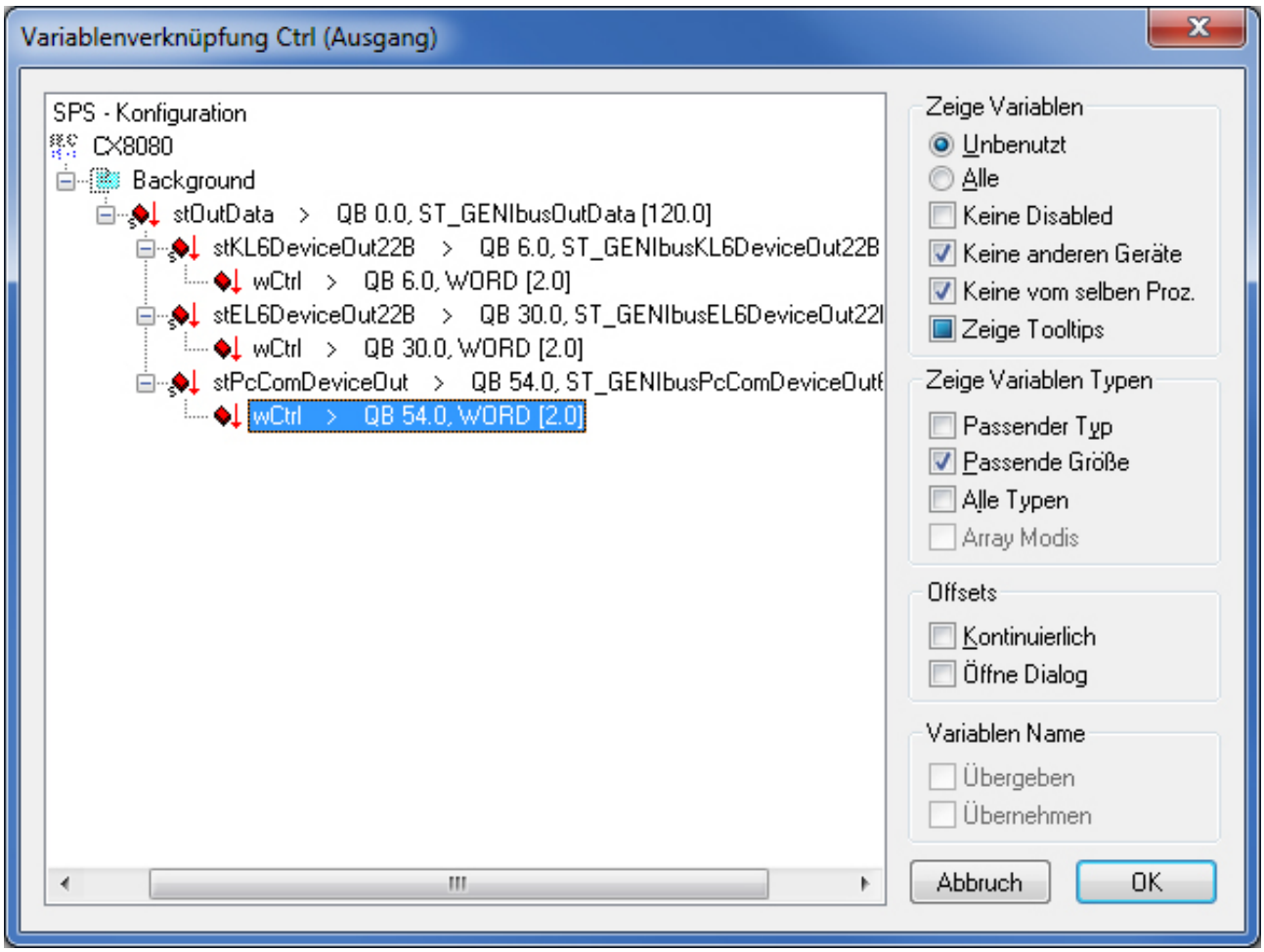


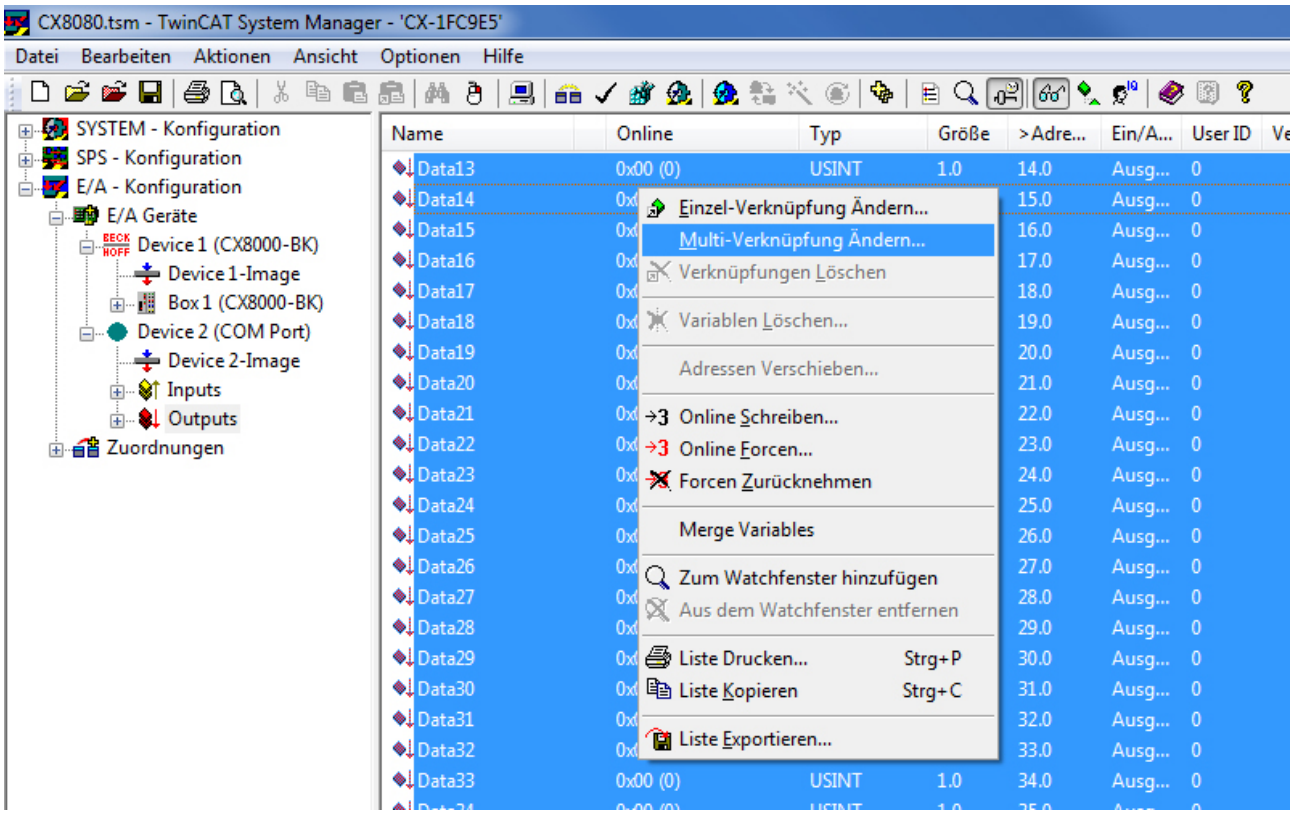
Danach können die Datenbytes bequem per Multi-Link auf die entsprechenden Variablen verknüpft werden. Die Auswahl mehrerer Variablen erreichen Sie dadurch, indem Sie "Data1" anklicken und bei gedrückter Shift-Taste die ↓-Taste betätigen

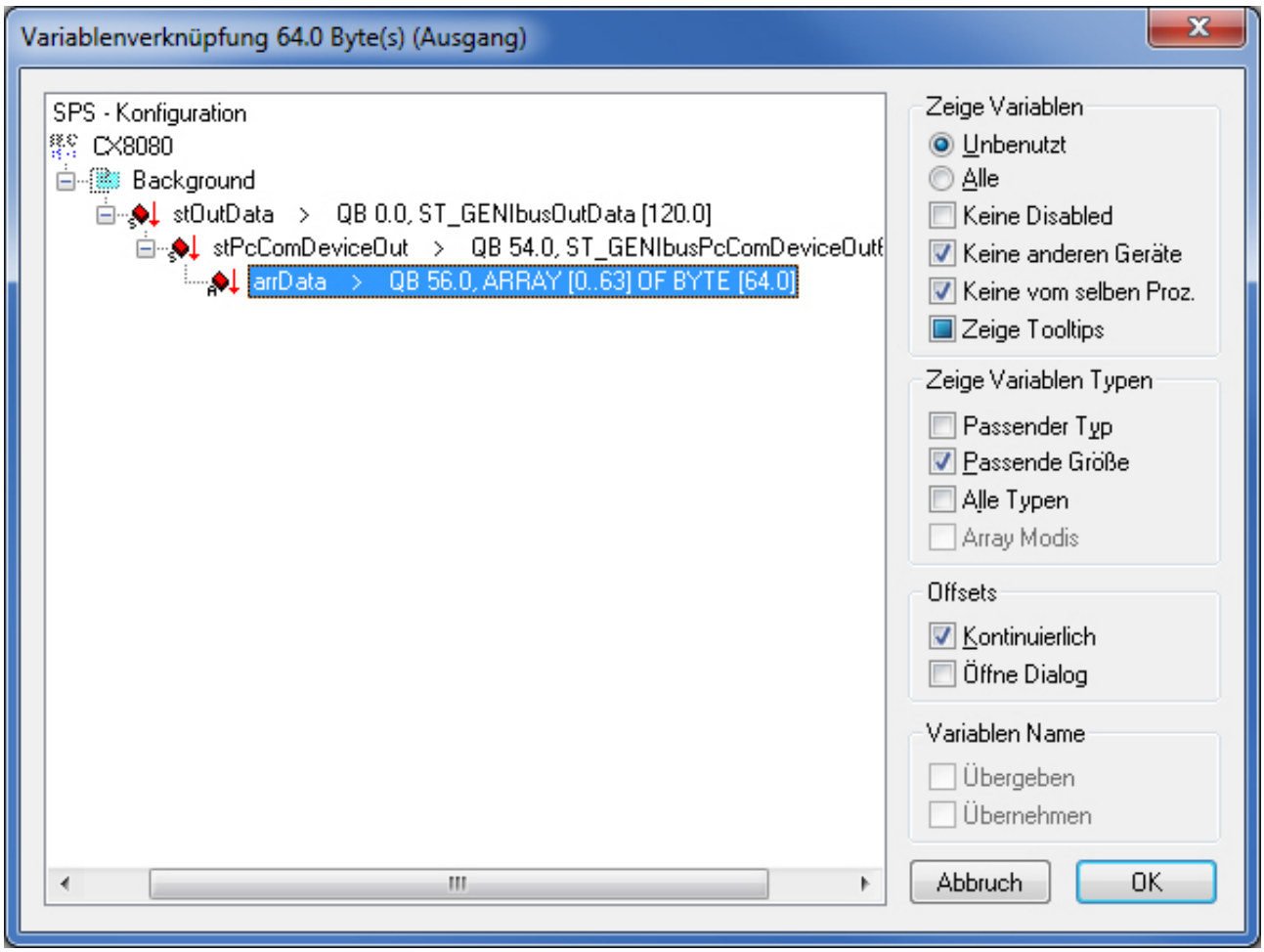


Die Ausgangsvariablen sind analog zu verknüpfen:




Variablenverknüpfung Ctrl (Ausgang)
SPS - Konfiguration
CX8080
Background
stOutData > QB 0.0, ST_GENIbusOutData [120.0]
stKL6DeviceOut22B > QB 6.0, ST_GENIbusKL6DeviceOut22B
wCtrl > QB 6.0, WORD [2.0]
stEL6DeviceOut22B > QB 30.0, ST_GENIbusEL6DeviceOut22I
wCtrl > QB 30.0, WORD [2.0]
stPcComDeviceOut > QB 54.0, ST_GENIbusPcComDeviceOutI
wCtrl > QB 54.0, WORD [2.0]
Zeige Variablen
Unbenutzt
Alle
Keine Disabled
Keine anderen Geräte
Keine vom selben Proz.
Zeige Tooltips
Zeige Variablen Typen
Passender Typ
Passende Größe
Alle Typen
Array Modis
Offsets
Kontinuierlich
Öffne Dialog
Variablen Name
Übergeben
Übernehmen
Abbruch
OK


CX8080.tsm - TwinCAT System Manager - 'CX-1FC9E5'
Datei Bearbeiten Aktionen Ansicht Optionen Hilfe
SYSTEM - Konfiguration
SPS - Konfiguration
E/A - Konfiguration
E/A Geräte
Device 1 (CX8000-BK)
Device 1-Image
Box 1 (CX8000-BK)
Device 2 (COM Port)
Device 2-Image
Inputs
Outputs
Zuordnungen
Name Online Typ Größe >Adre... Ein/A... User ID
Data13 0x00 (0) USINT 1.0 14.0 Ausg... 0
Einzel-Verknüpfung Ändern...
Multi-Verknüpfung Ändern...
Verknüpfungen Löschen
Variablen Löschen...
Adressen Verschieben...
Online Schreiben...
Online Forcen...
Forcen Zurücknehmen
Merge Variables
Zum Watchfenster hinzufügen
Aus dem Watchfenster entfernen
Liste Drucken... Strg+P
Liste Kopieren Strg+C
Liste Exportieren...
Data33 0x00 (0) USINT 1.0 34.0 Ausg... 0



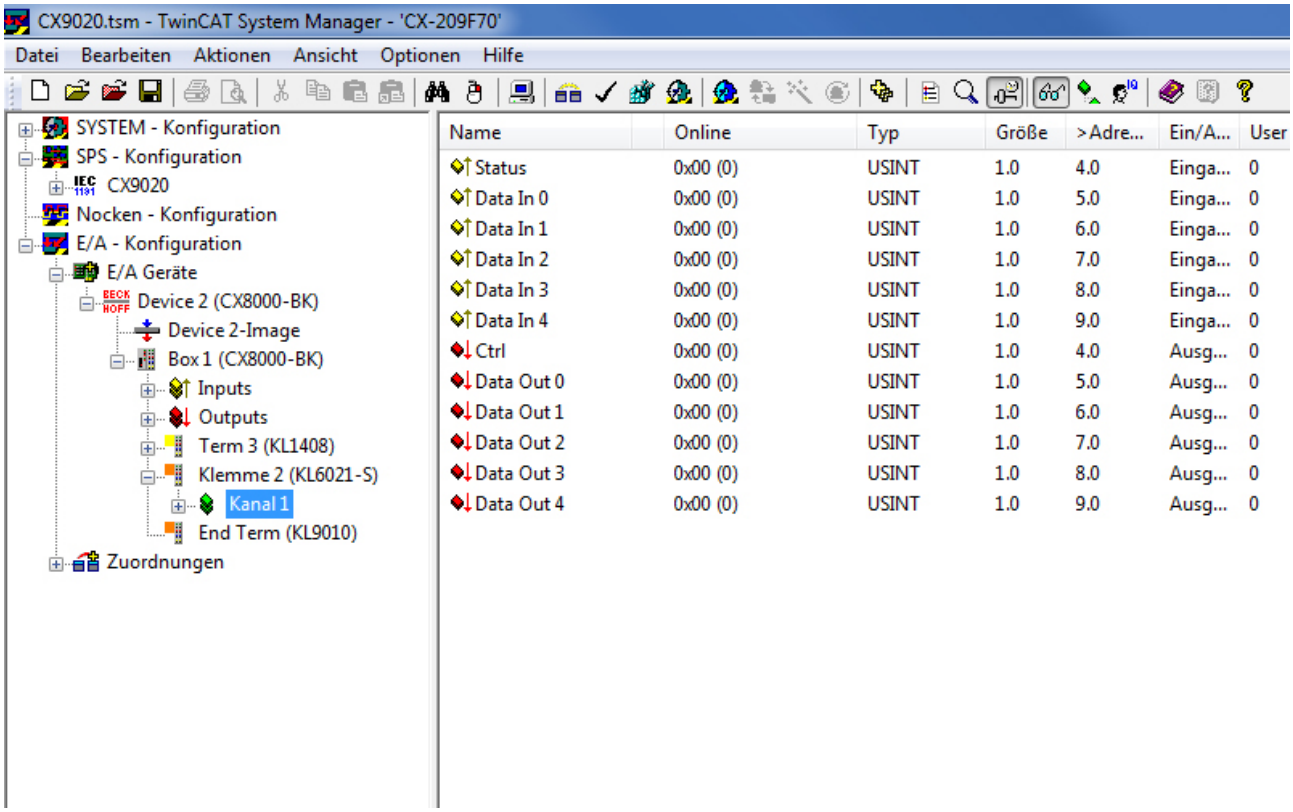
4.6 Verbindung der Kommunikationsvariablen bei Verwendung einer KL6021

Voraussetzung für das Verknüpfen des Prozessabbildes ist, dass die Klemme von vornherein auf 5Byte eingestellt ist. Anders als bei der PC-Schnittstelle lässt sich dieses nicht im TwinCAT System Manager konfigurieren, sondern allein über die KS2000-Software. Die Kommunikationsparameter

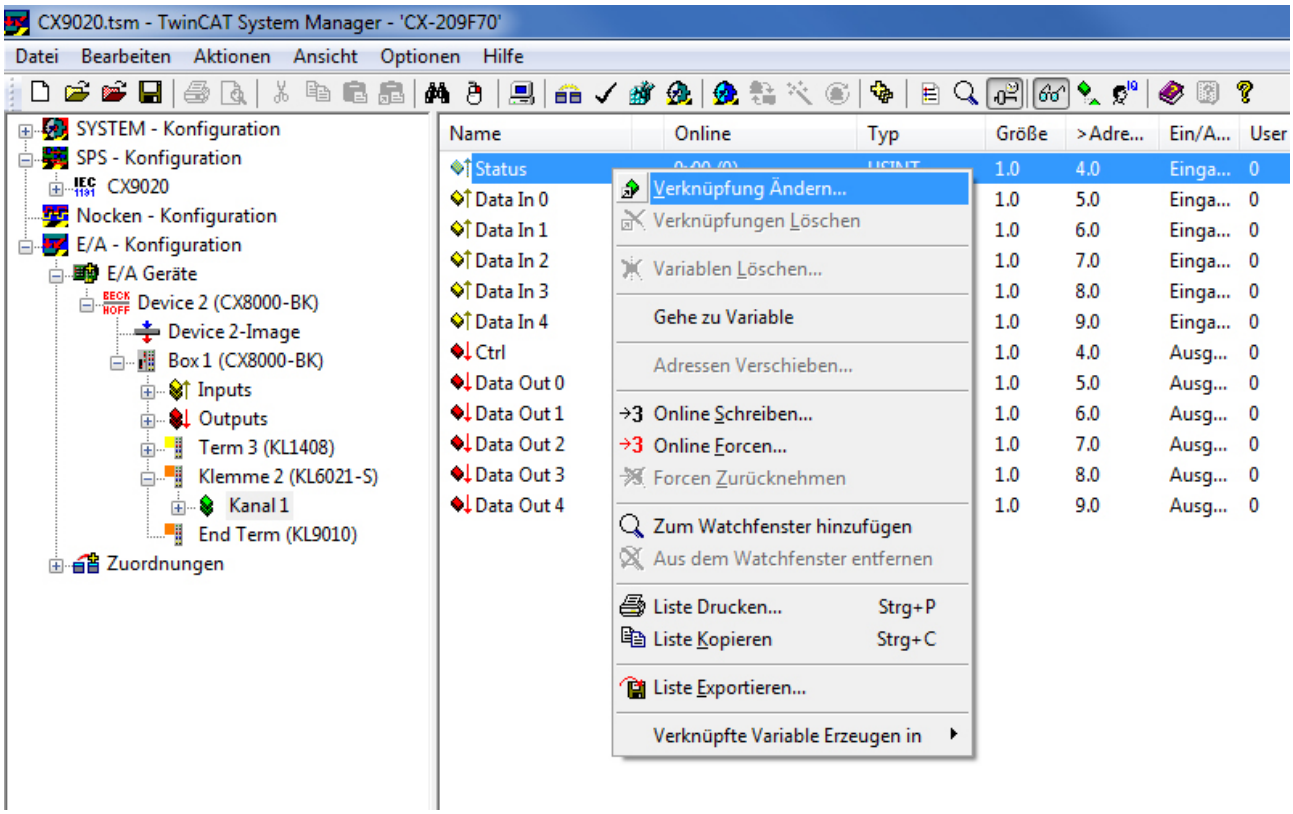
- Baud-Rate: 9600 bits/s
- Data Bits: 8
- Parity: None
- Stop-Bits: 1

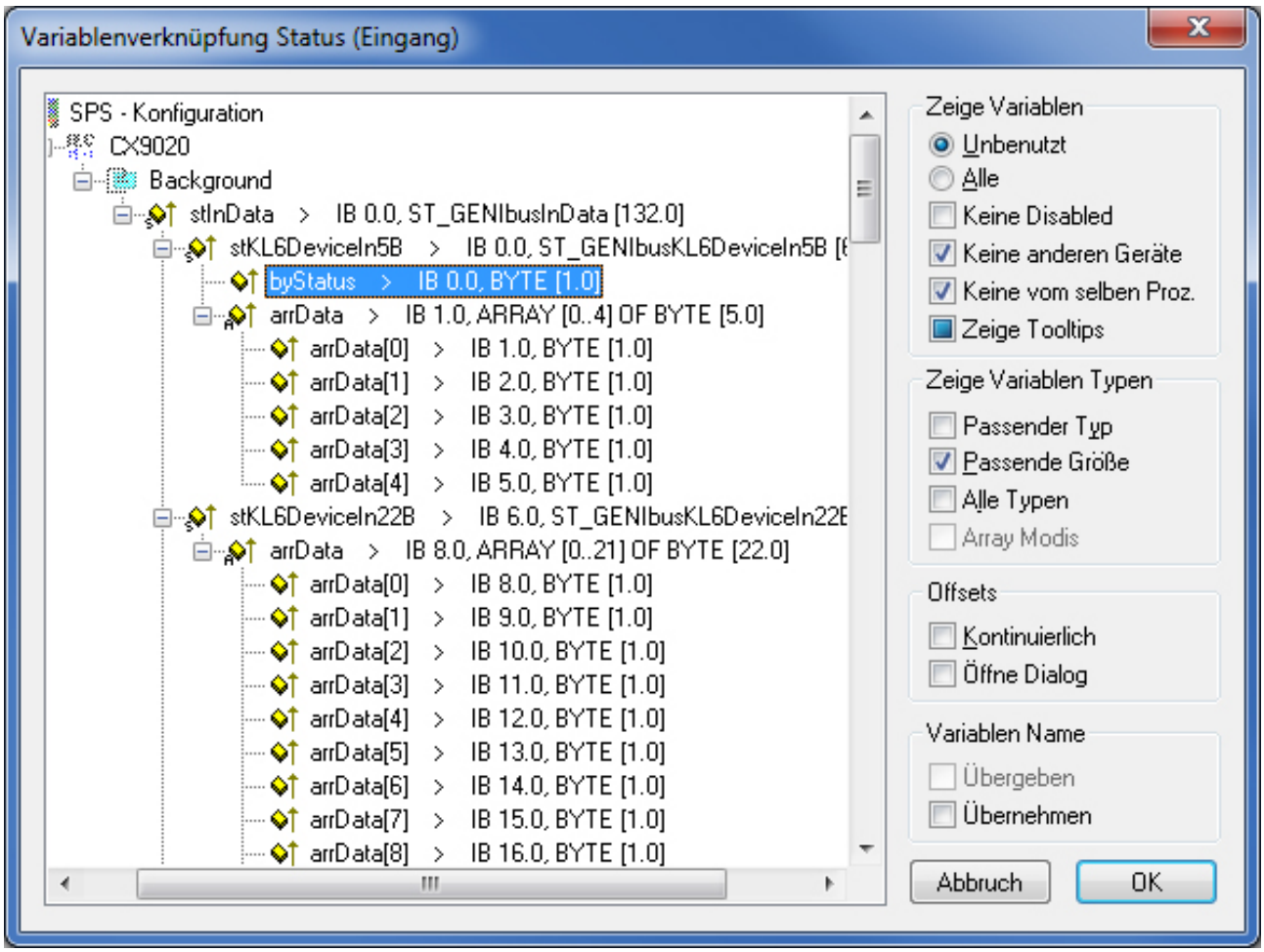
werden durch die SPS-Applikation automatisch eingestellt, so dass beim Wechsel der Klemme und anschließendem Neustart diese richtig eingestellt ist.

Die Verbindung des Prozessabbildes zu den SPS- Ein- und Ausgangsvariablen lässt sich am einfachsten von der Hardware-Seite her realisieren, da von dort aus Multi-Verknüpfungen möglich sind. Die Variablen müssen dazu im rechten Teil des TwinCAT System Managers zu sehen sein:

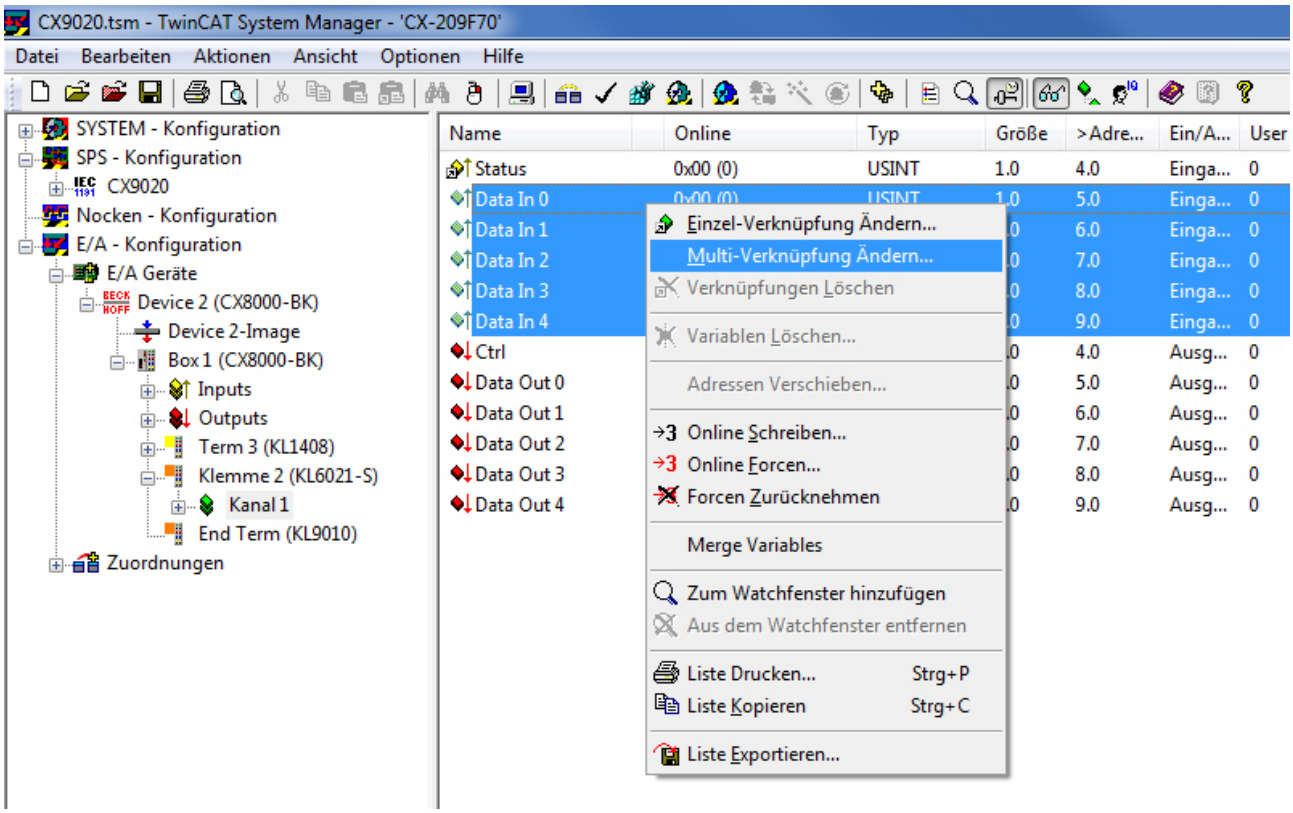


Zunächst wird allein der Status mit der Statusvariable des Kommunikationseingangs verbunden. Es ist dabei zu beachten, die Struktur für PC-Kommunikation zu wählen.

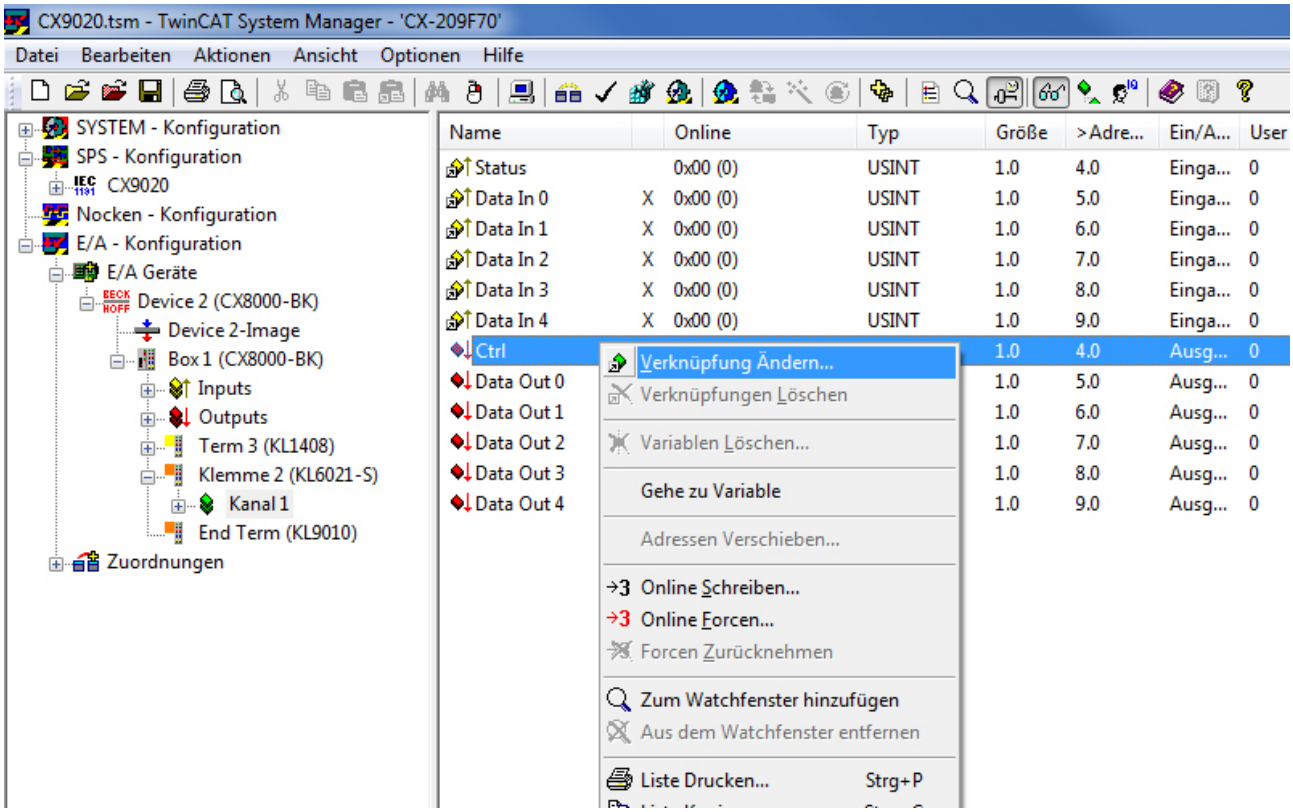


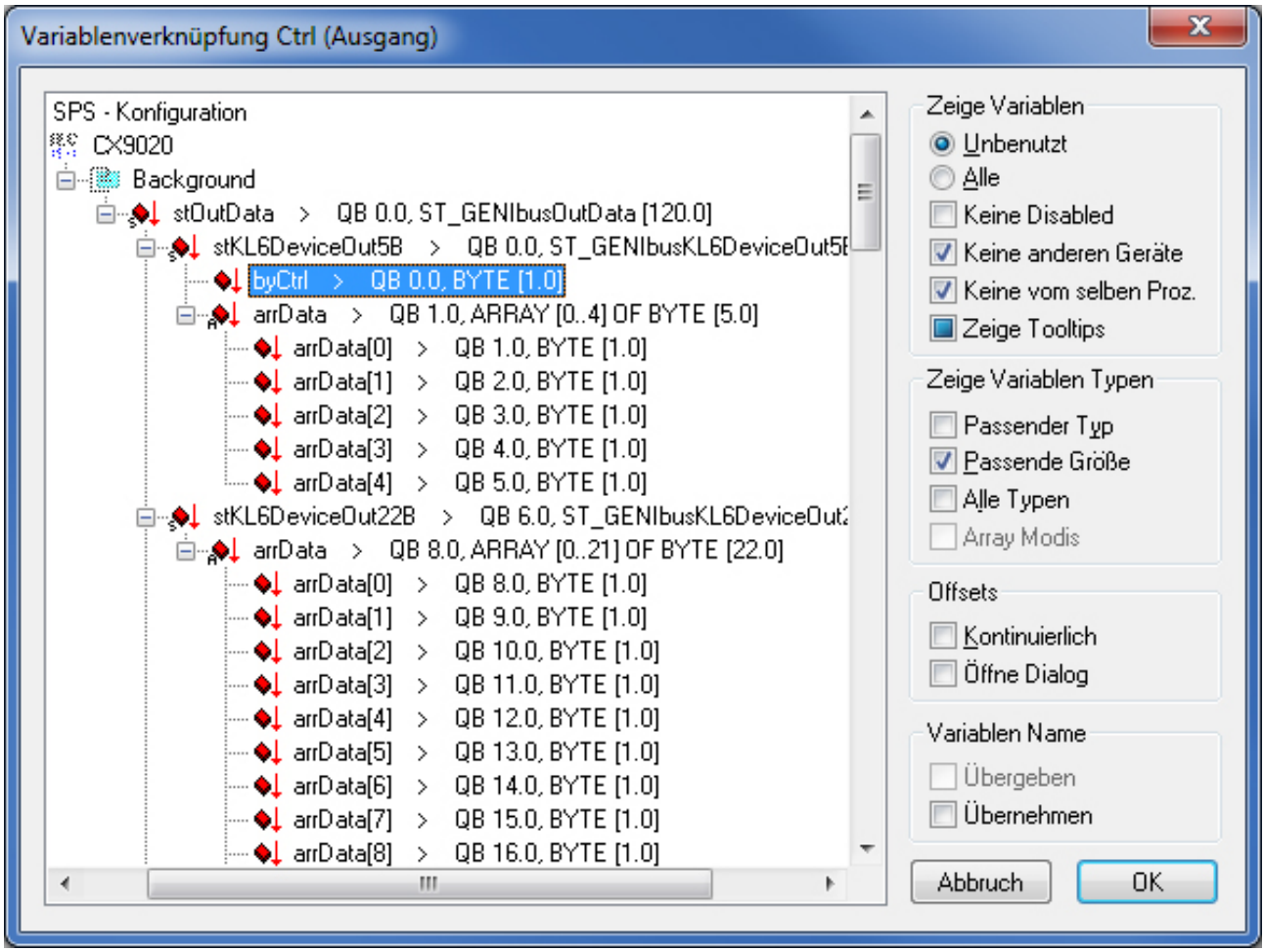


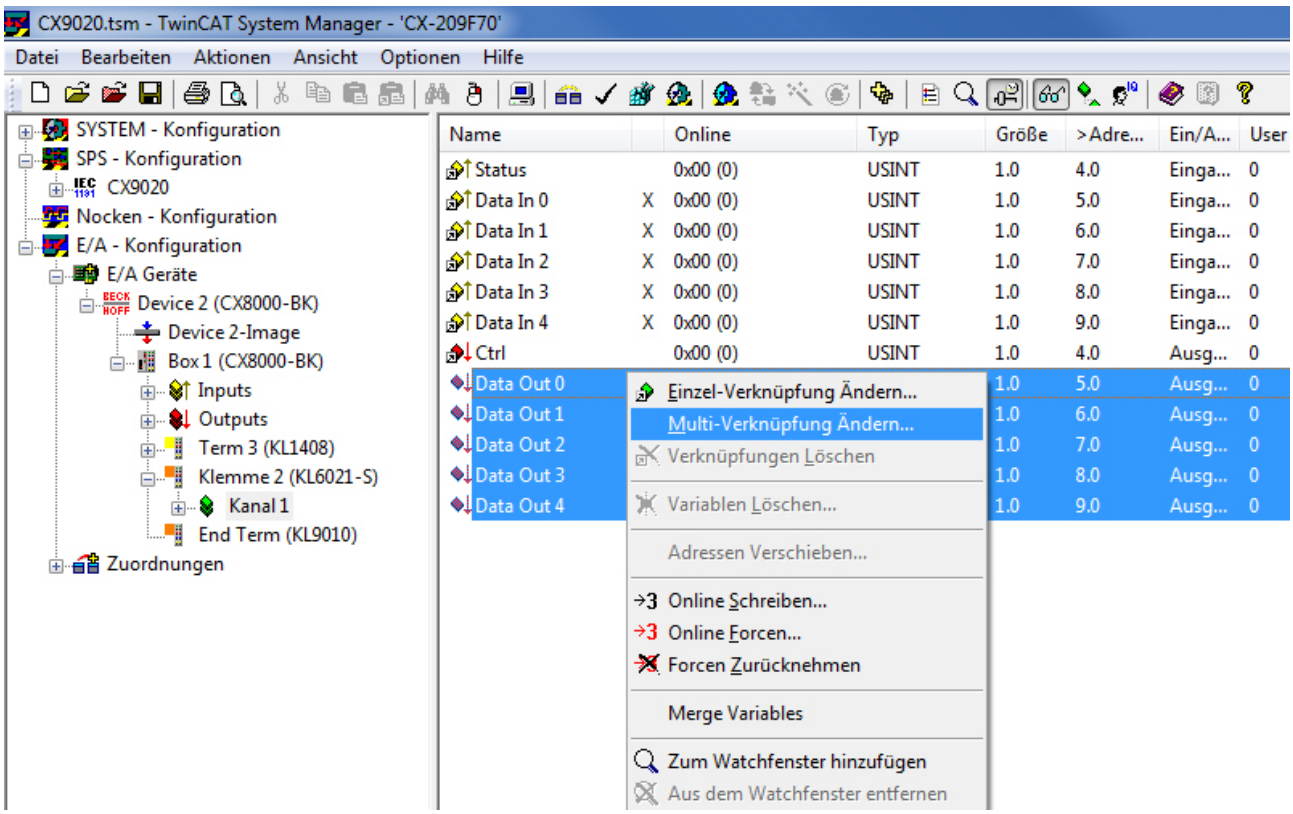
Danach können die Datenbytes bequem per Multi-Link auf die entsprechenden Variablen verknüpft werden. Die Auswahl mehrerer Variablen erreichen Sie dadurch, indem Sie "Data1" anklicken und bei gedrückter Shift-Taste die ↓-Taste betätigen.

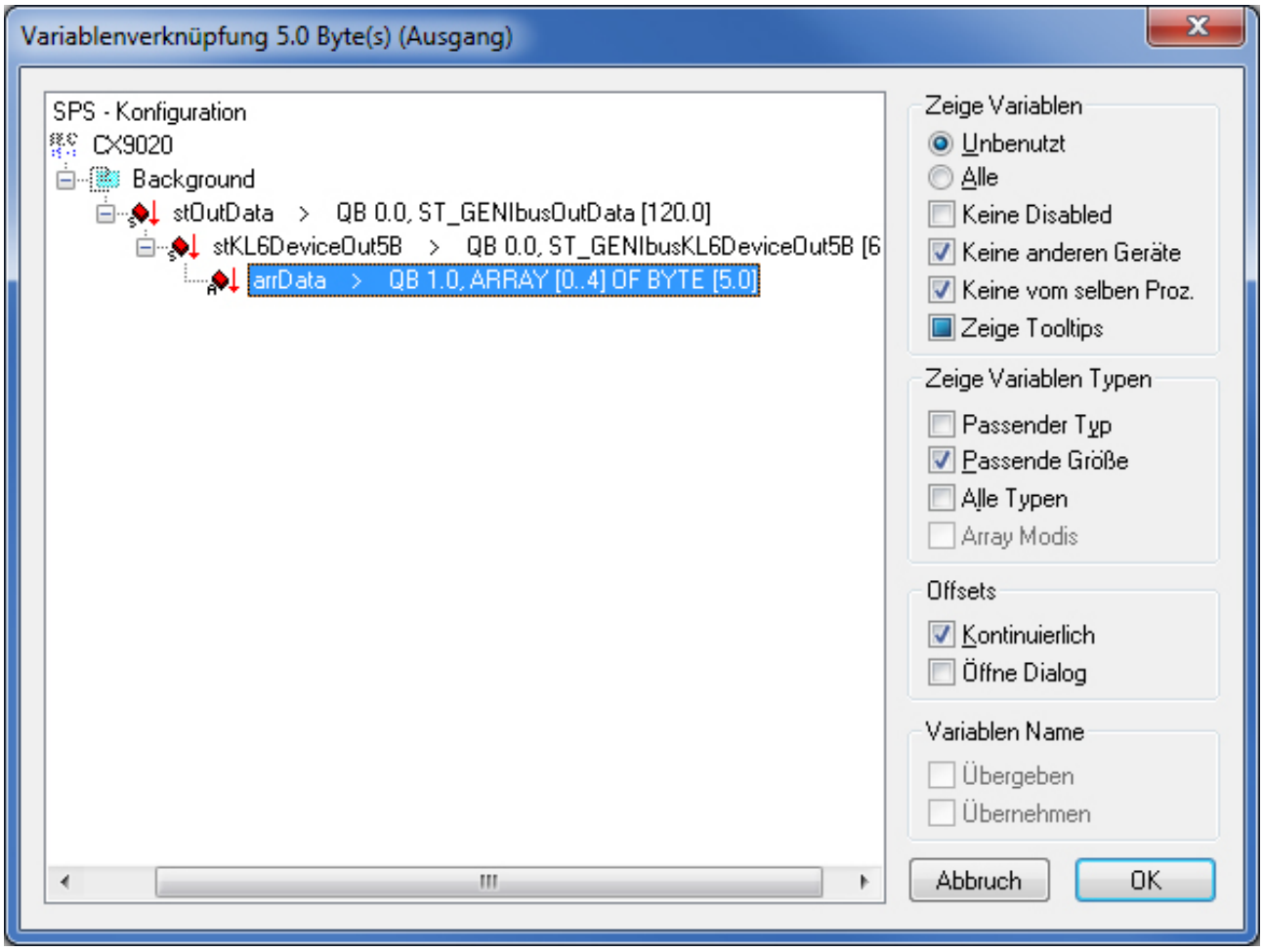


Die Ausgangsvariablen sind Analog zu verknüpfen:






CX9020.tsm - TwinCAT System Manager - 'CX-209F70'
Datei Bearbeiten Aktionen Ansicht Optionen Hilfe
SYSTEM - Konfiguration
SPS - Konfiguration
CX9020
Nocken - Konfiguration
E/A - Konfiguration
E/A Geräte
Device 2 (CX8000-BK)
Device 2-Image
Box 1 (CX8000-BK)
Inputs
Outputs
Term 3 (KL1408)
Klemme 2 (KL6021-S)
Kanal 1
End Term (KL9010)
Zuordnungen
Name Online Typ Größe >Adre... Ein/A... User
Status 0x00 (0) USINT 1.0 4.0 Einga... 0
Data In 0 X 0x00 (0) USINT 1.0 5.0 Einga... 0
Data In 1 X 0x00 (0) USINT 1.0 6.0 Einga... 0
Data In 2 X 0x00 (0) USINT 1.0 7.0 Einga... 0
Data In 3 X 0x00 (0) USINT 1.0 8.0 Einga... 0
Data In 4 X 0x00 (0) USINT 1.0 9.0 Einga... 0
Ctrl 0x00 (0) USINT 1.0 4.0 Ausg... 0
Data Out 0 1.0 5.0 Ausg... 0
Data Out 1 1.0 6.0 Ausg... 0
Data Out 2 1.0 7.0 Ausg... 0
Data Out 3 1.0 8.0 Ausg... 0
Data Out 4 1.0 9.0 Ausg... 0
Einzel-Verknüpfung Ändern...
Multi-Verknüpfung Ändern...
Verknüpfungen Löschen
Variablen Löschen...
Adressen Verschieben...
Online Schreiben...
Online Forcen...
Forcen Zurücknehmen
Merge Variables
Zum Watchfenster hinzufügen
Aus dem Watchfenster entfernen


Variablenverknüpfung 5.0 Byte(s) (Ausgang)
SPS - Konfiguration
CX9020
Background
stOutData > QB 0.0, ST_GENIbusOutData [120.0]
stKL6DeviceOut5B > QB 0.0, ST_GENIbusKL6DeviceOut5B [6
arrData > QB 1.0, ARRAY [0..4] OF BYTE [5.0]
Zeige Variablen
Unbenutzt
Alle
Keine Disabled
Keine anderen Geräte
Keine vom selben Proz.
Zeige Tooltips
Zeige Variablen Typen
Passender Typ
Passende Größe
Alle Typen
Array Modis
Offsets
Kontinuierlich
Öffne Dialog
Variablen Name
Übergeben
Übernehmen
Abbruch
OK

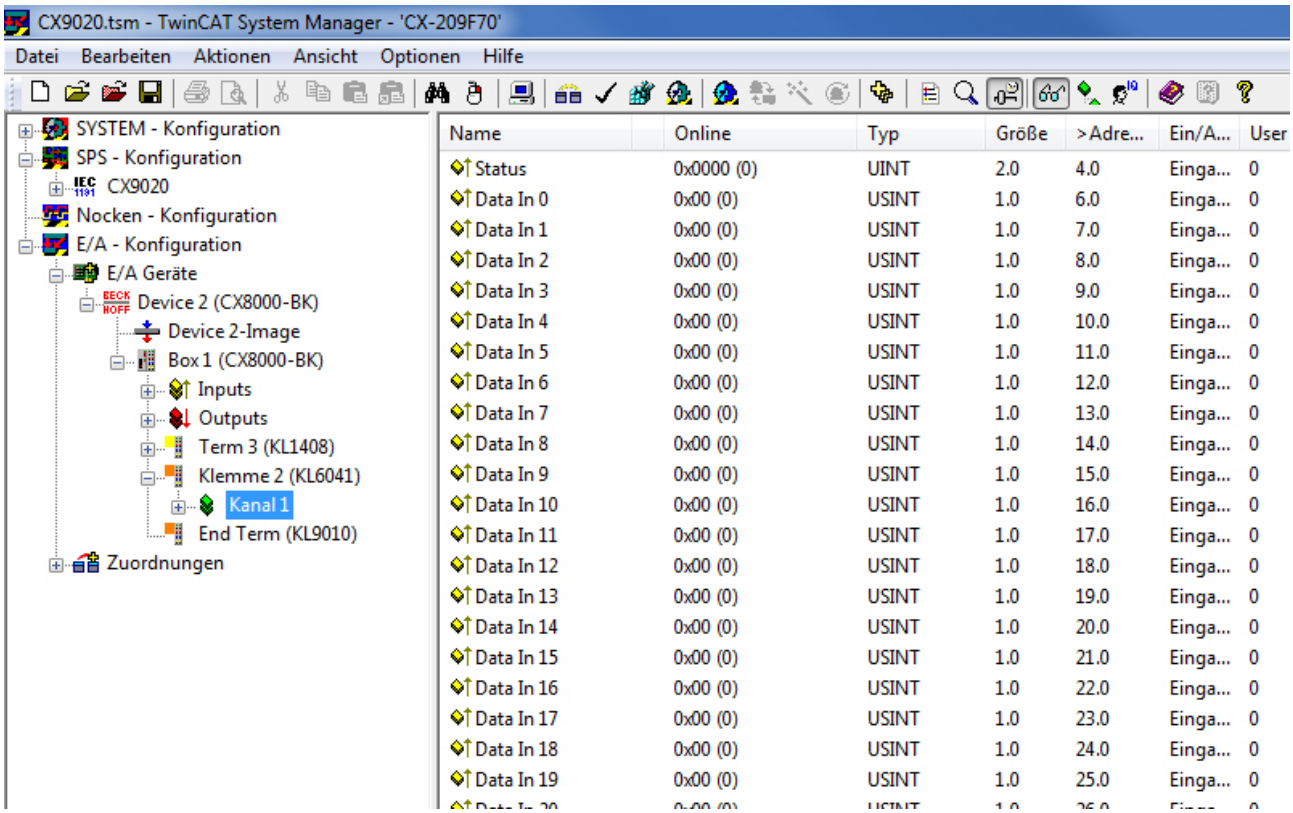
4.7 Verbindung der Kommunikationsvariablen bei Verwendung einer KL6041

Voraussetzung für das Verknüpfen des Prozessabbildes ist, dass die Klemme von vornherein auf 22Byte eingestellt ist. Anders als bei der PC-Schnittstelle lässt sich dieses nicht im TwinCAT System Manager konfigurieren, sondern allein über die KS2000-Software. Die Kommunikationsparameter

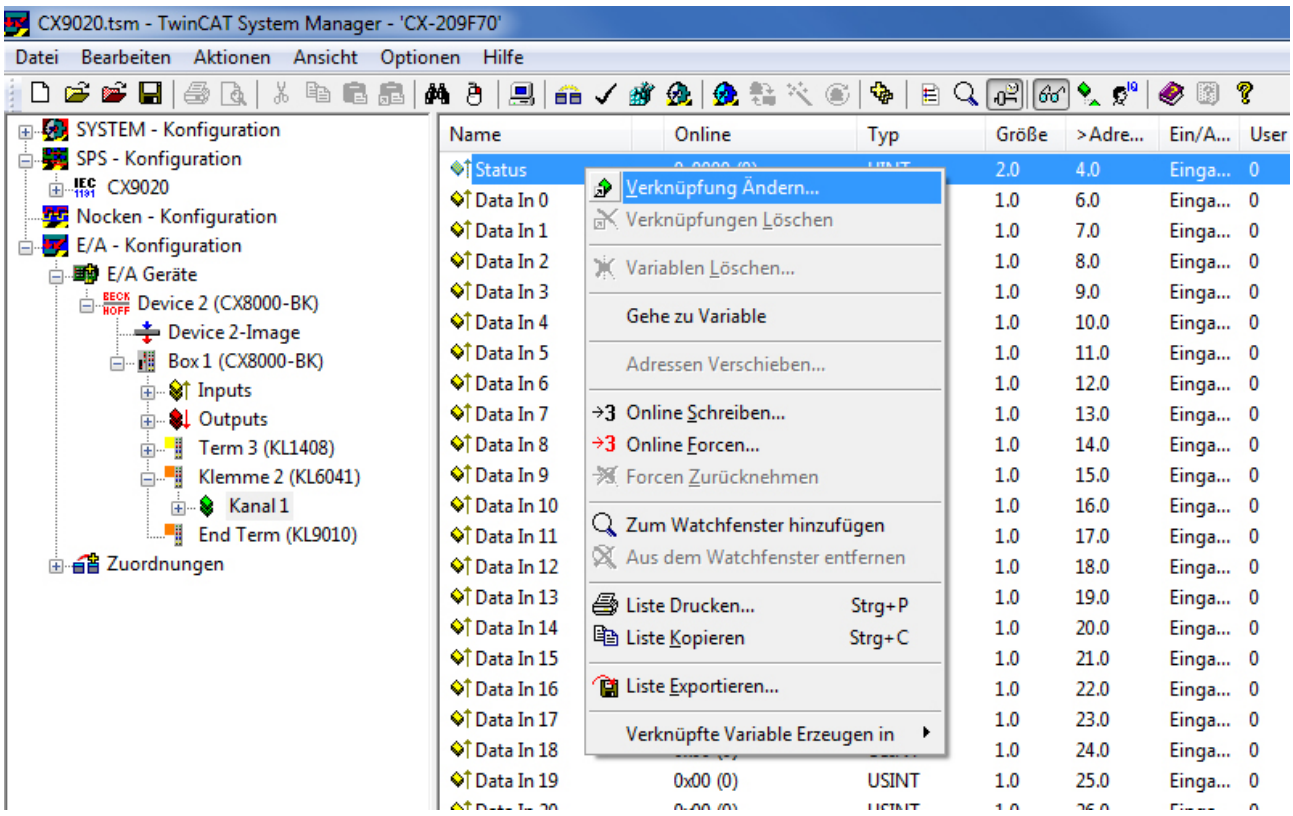
- Baud-Rate: 9600 bits/s
- Data Bits: 8
- Parity: None
- Stop-Bits: 1

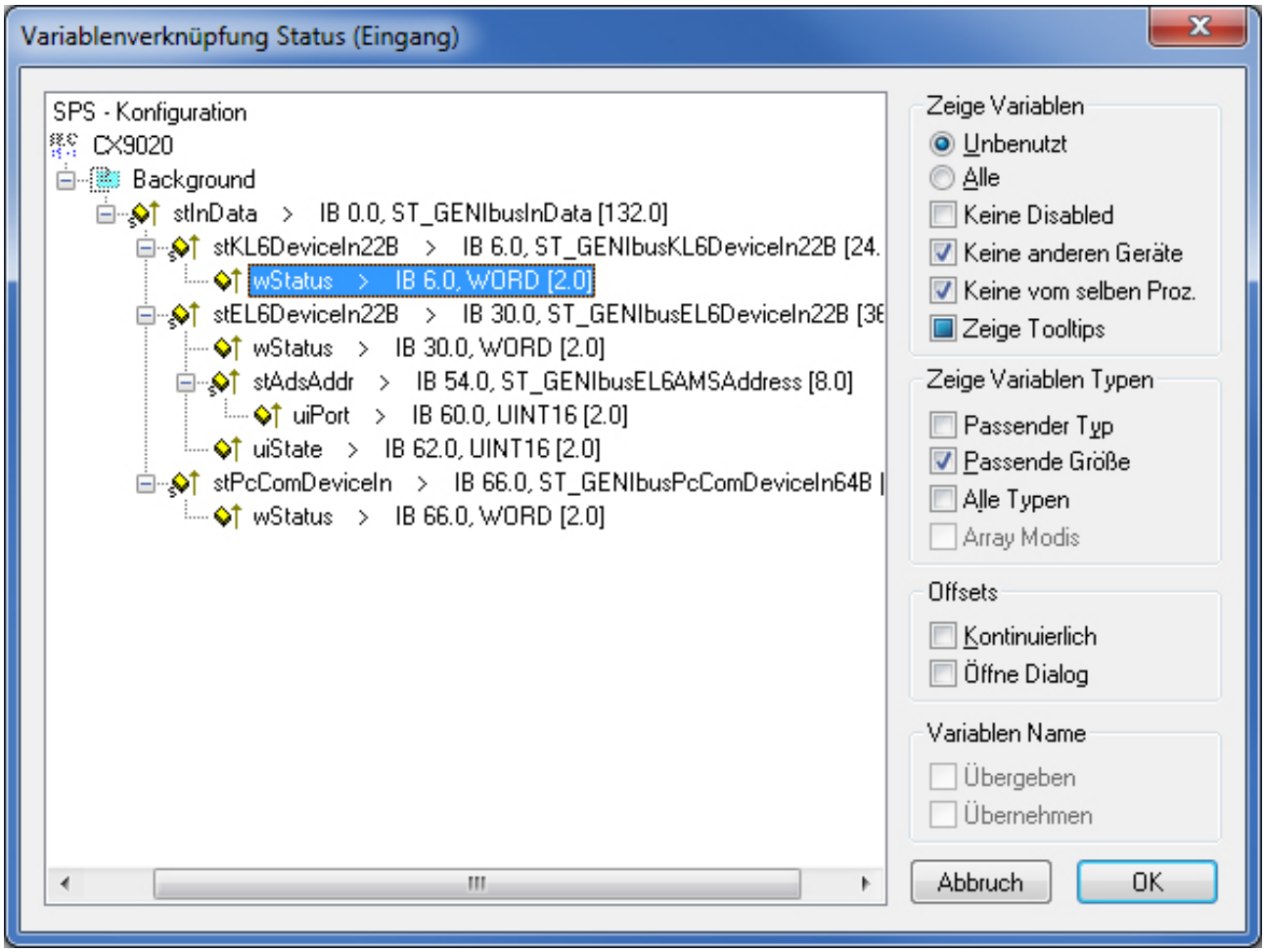
werden durch die SPS-Applikation automatisch eingestellt, so dass beim Wechsel der Klemme und anschließendem Neustart diese richtig eingestellt ist.

Die Verbindung des Prozessabbildes zu den SPS- Ein- und Ausgangsvariablen lässt sich am einfachsten von der Hardware-Seite her realisieren, da von dort aus Multi-Verknüpfungen möglich sind. Die Variablen müssen dazu im rechten Teil des TwinCAT System Managers zu sehen sein:

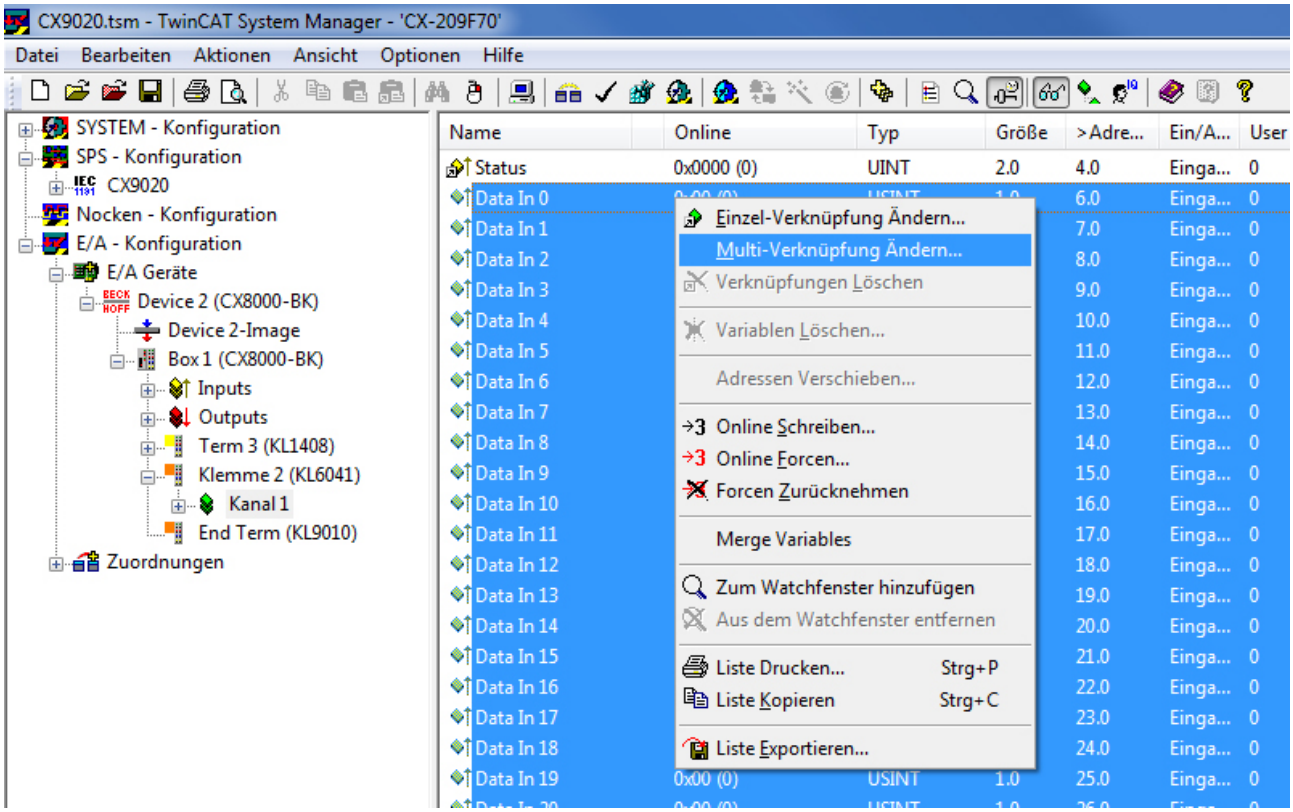


Zunächst wird allein der Status mit der Statusvariable des Kommunikationseingangs verbunden. Es ist dabei zu beachten, die Struktur für PC-Kommunikation zu wählen.

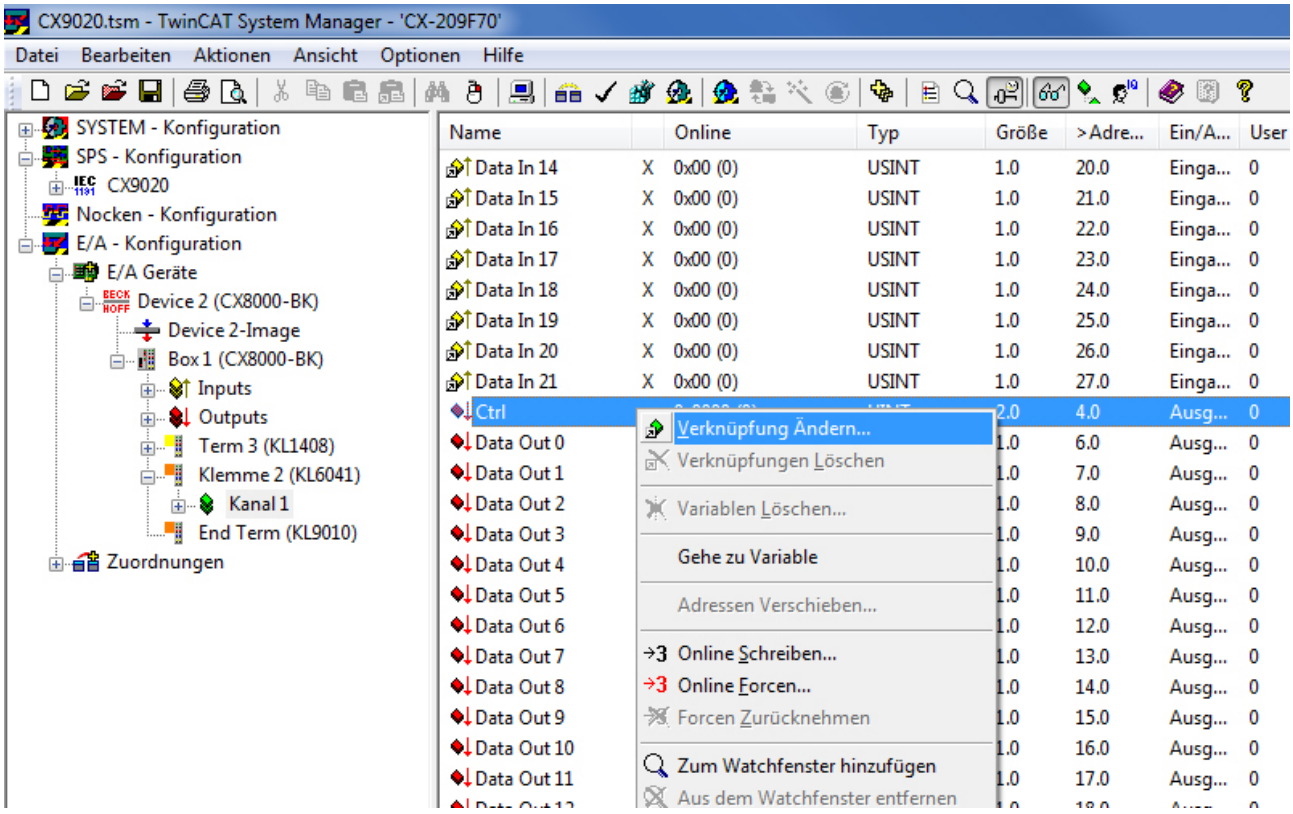


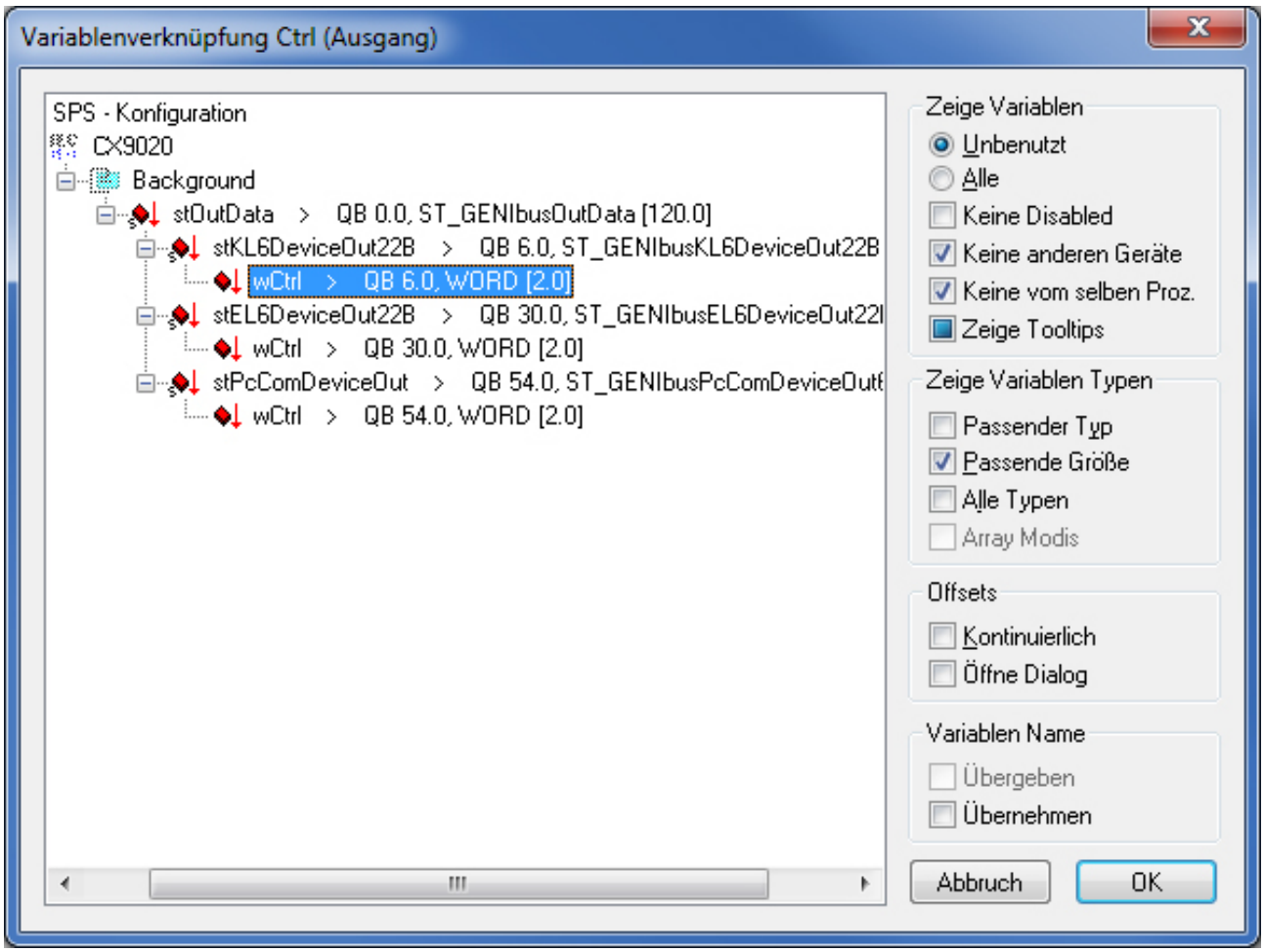


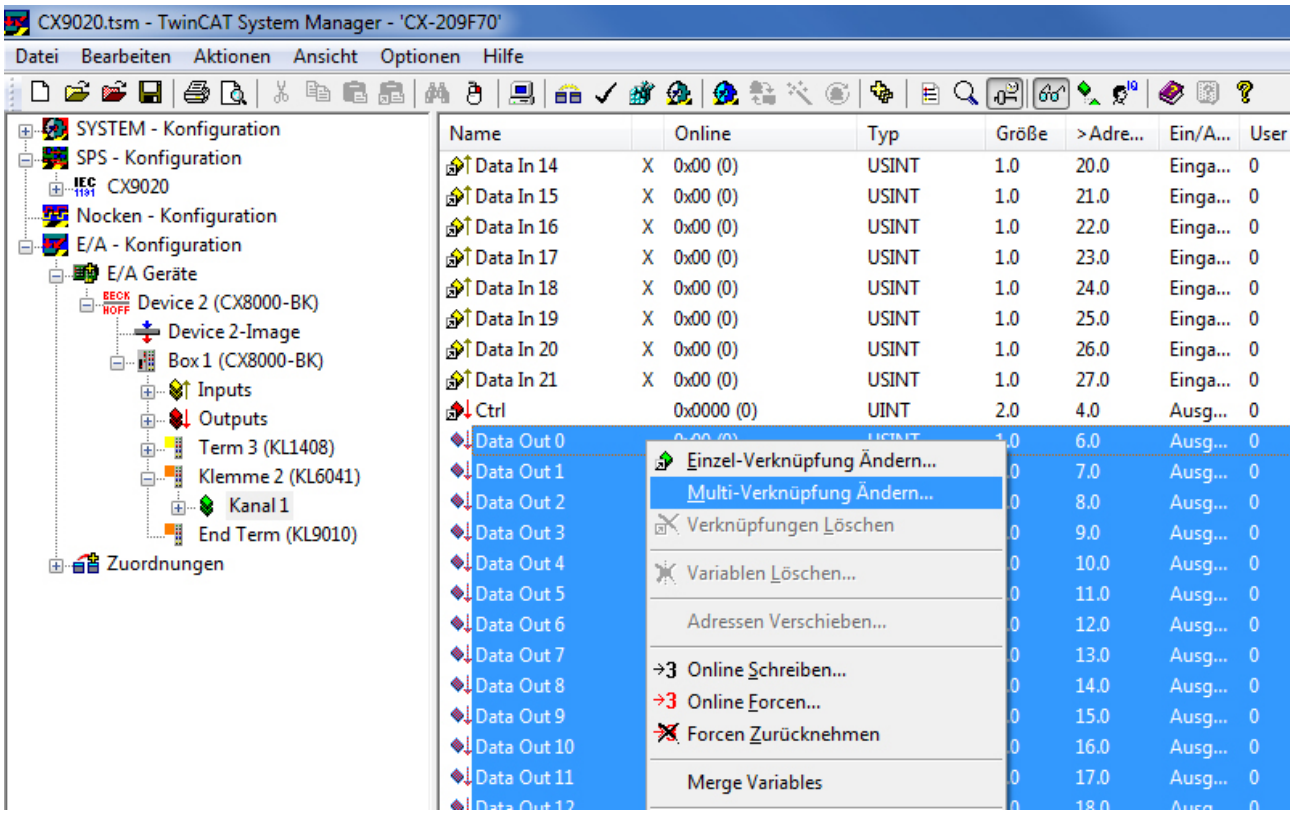
Danach können die Datenbytes bequem per Multi-Link auf die entsprechenden Variablen verknüpft werden. Die Auswahl mehrerer Variablen erreichen Sie dadurch, indem Sie "Data1" anklicken und bei gedrückter Shift-Taste die ↓-Taste betätigen

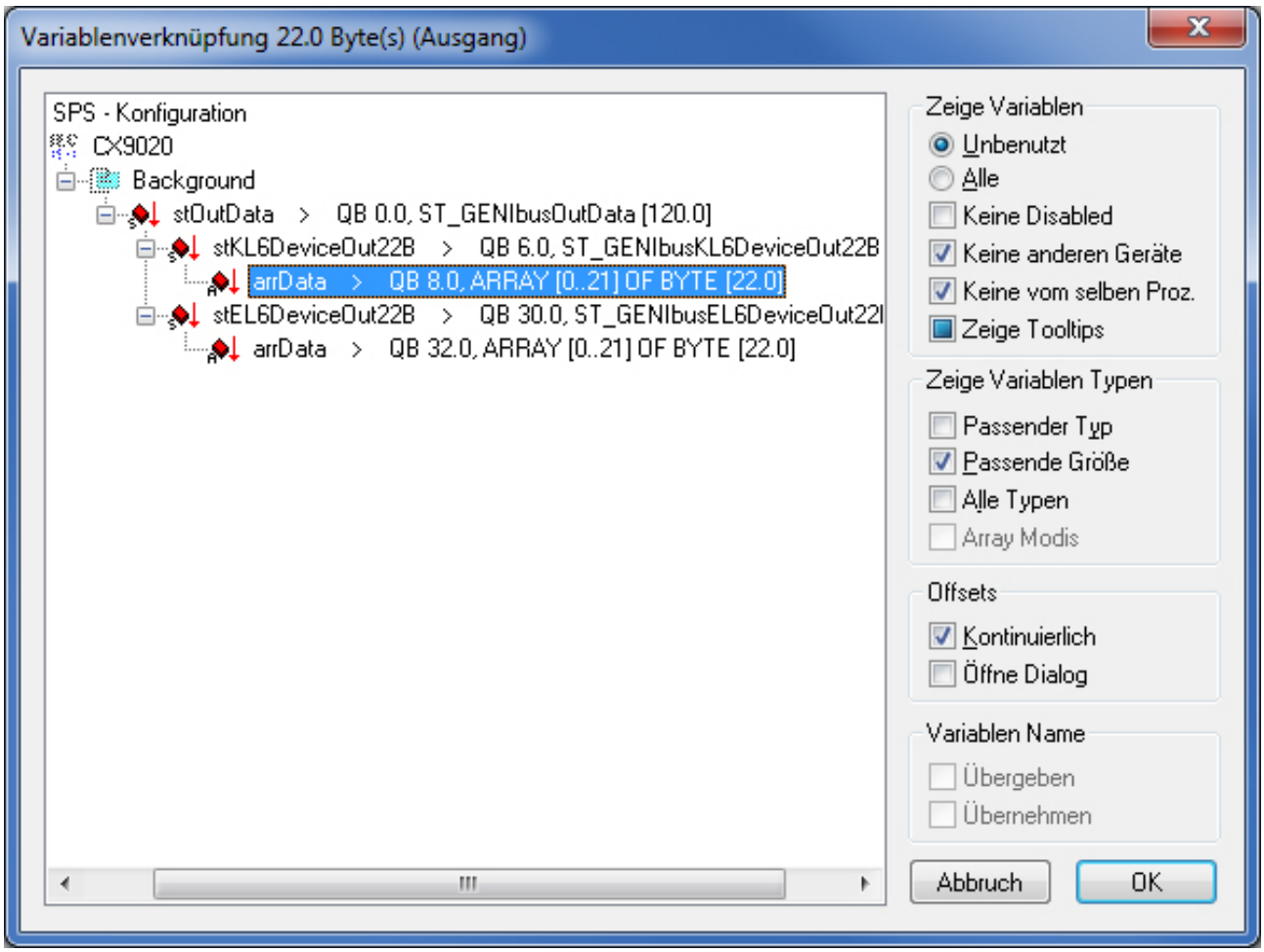


Die Ausgangsvariablen sind Analog zu verknüpfen:




Variablenverknüpfung Ctrl (Ausgang)
SPS - Konfiguration
CX9020
Background
stOutData > QB 0.0, ST_GENIbusOutData [120.0]
stKL6DeviceOut22B > QB 6.0, ST_GENIbusKL6DeviceOut22B
wCtrl > QB 6.0, WORD [2.0]
stEL6DeviceOut22B > QB 30.0, ST_GENIbusEL6DeviceOut22I
wCtrl > QB 30.0, WORD [2.0]
stPcComDeviceOut > QB 54.0, ST_GENIbusPcComDeviceOutI
wCtrl > QB 54.0, WORD [2.0]
Zeige Variablen
Unbenutzt
Alle
Keine Disabled
Keine anderen Geräte
Keine vom selben Proz.
Zeige Tooltips
Zeige Variablen Typen
Passender Typ
Passende Größe
Alle Typen
Array Modis
Offsets
Kontinuierlich
Öffne Dialog
Variablen Name
Übergeben
Übernehmen
Abbruch
OK


CX9020.tsm - TwinCAT System Manager - 'CX-209F70'
Datei Bearbeiten Aktionen Ansicht Optionen Hilfe
SYSTEM - Konfiguration
SPS - Konfiguration
CX9020
Nocken - Konfiguration
E/A - Konfiguration
E/A Geräte
Device 2 (CX8000-BK)
Device 2-Image
Box 1 (CX8000-BK)
Inputs
Outputs
Term 3 (KL1408)
Klemme 2 (KL6041)
Kanal 1
End Term (KL9010)
Zuordnungen
Name Online Typ Größe >Adre... Ein/A... User
Data In 14 X 0x00 (0) USINT 1.0 20.0 Einga... 0
Data In 15 X 0x00 (0) USINT 1.0 21.0 Einga... 0
Data In 16 X 0x00 (0) USINT 1.0 22.0 Einga... 0
Data In 17 X 0x00 (0) USINT 1.0 23.0 Einga... 0
Data In 18 X 0x00 (0) USINT 1.0 24.0 Einga... 0
Data In 19 X 0x00 (0) USINT 1.0 25.0 Einga... 0
Data In 20 X 0x00 (0) USINT 1.0 26.0 Einga... 0
Data In 21 X 0x00 (0) USINT 1.0 27.0 Einga... 0
Ctrl 0x0000 (0) UINT 2.0 4.0 Ausg... 0
Data Out 0 6.0 Ausg... 0
Data Out 1 7.0 Ausg... 0
Data Out 2 8.0 Ausg... 0
Data Out 3 9.0 Ausg... 0
Data Out 4 10.0 Ausg... 0
Data Out 5 11.0 Ausg... 0
Data Out 6 12.0 Ausg... 0
Data Out 7 13.0 Ausg... 0
Data Out 8 14.0 Ausg... 0
Data Out 9 15.0 Ausg... 0
Data Out 10 16.0 Ausg... 0
Data Out 11 17.0 Ausg... 0
Einzel-Verknüpfung Ändern...
Multi-Verknüpfung Ändern...
Verknüpfungen Löschen
Variablen Löschen...
Adressen Verschieben...
Online Schreiben...
Online Forcen...
Forcen Zurücknehmen
Merge Variables



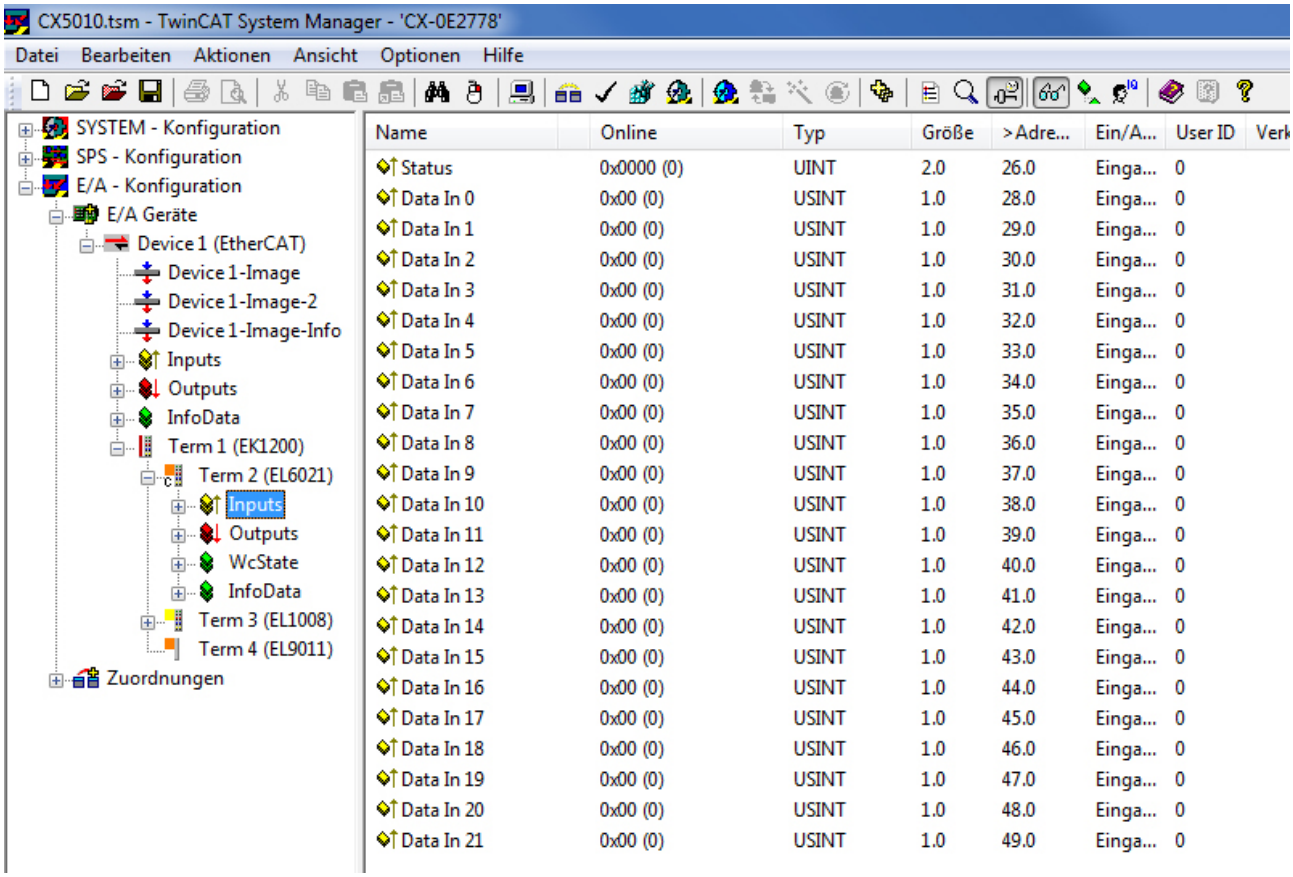
4.8 Konfiguration im TwinCAT System Manager

Anders als bei der PC-Schnittstelle werden die Kommunikationsparameter

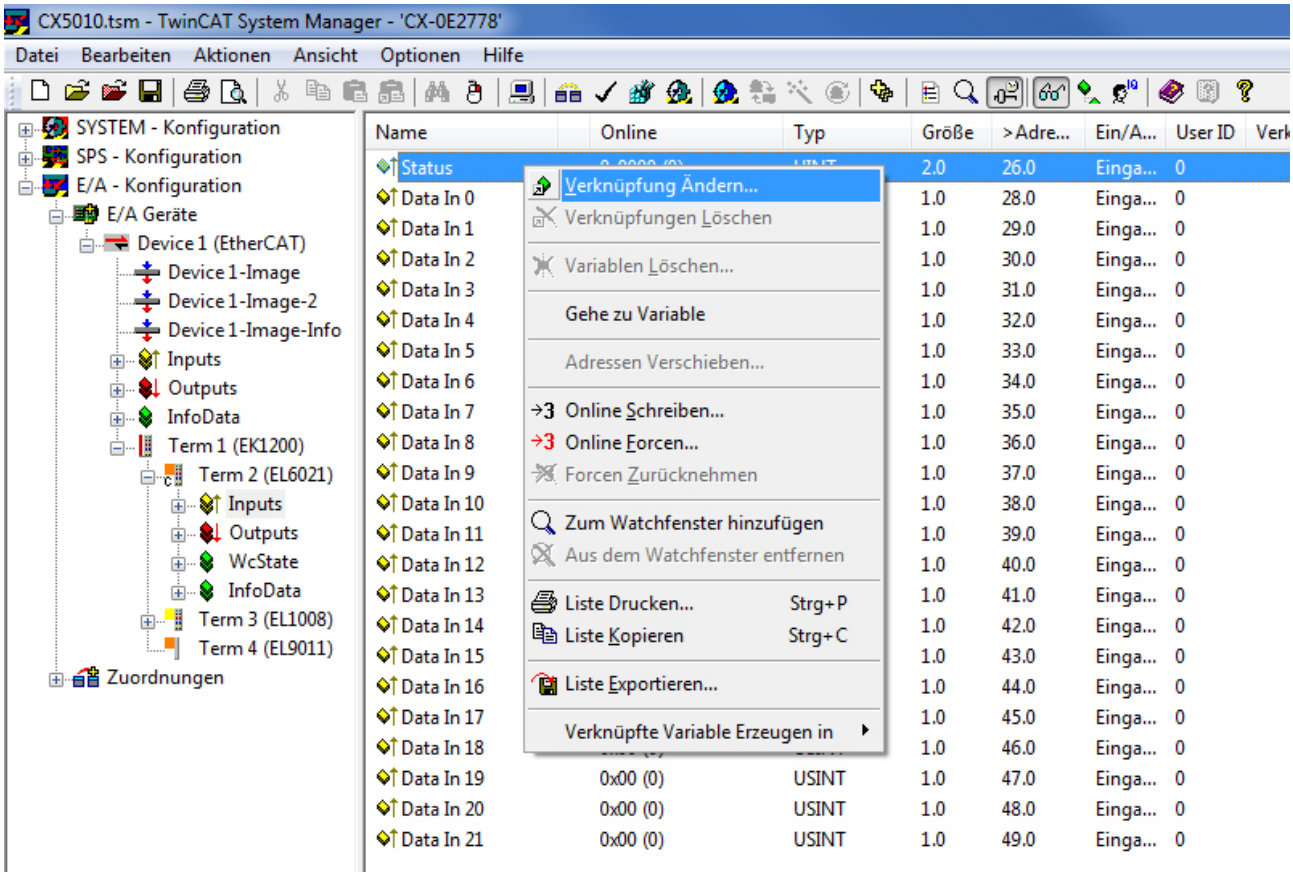
- Baud-Rate: 9600 bits/s
- Data Bits: 8
- Parity: None
- Stop-Bits: 1

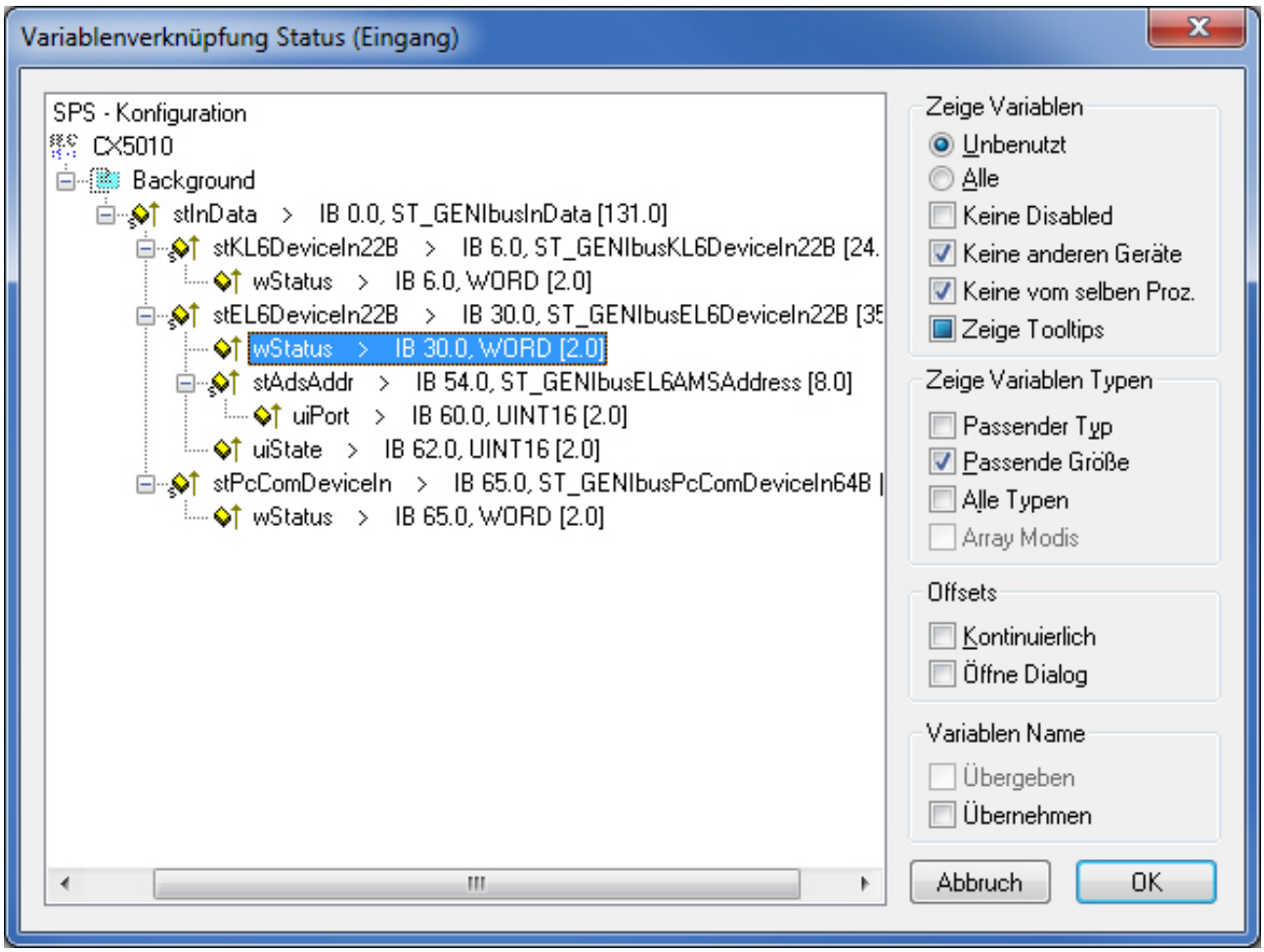
durch die SPS-Applikation automatisch eingestellt, so dass beim Wechsel der Klemme und anschließendem Neustart diese richtig eingestellt ist.

Die Verbindung des Prozessabbildes zu den SPS- Ein- und Ausgangsvariablen lässt sich am einfachsten von der Hardware-Seite her realisieren, da von dort aus Multi-Verknüpfungen möglich sind. Die Variablen müssen dazu im rechten Teil des TwinCAT System Managers zu sehen sein:

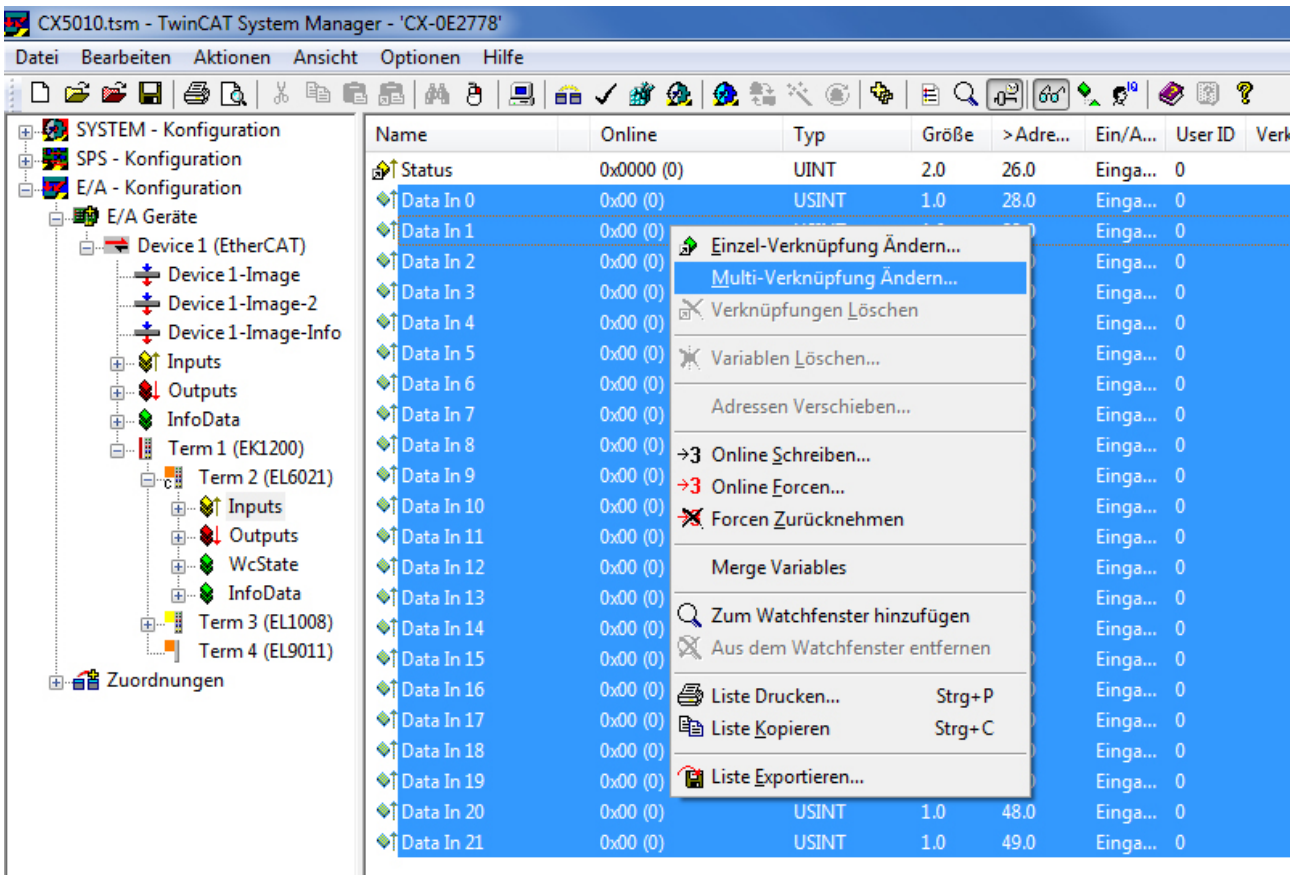


Zunächst wird allein der Status mit der Statusvariable des Kommunikationseingangs verbunden. Es ist dabei zu beachten, die Struktur für PC-Kommunikation zu wählen.

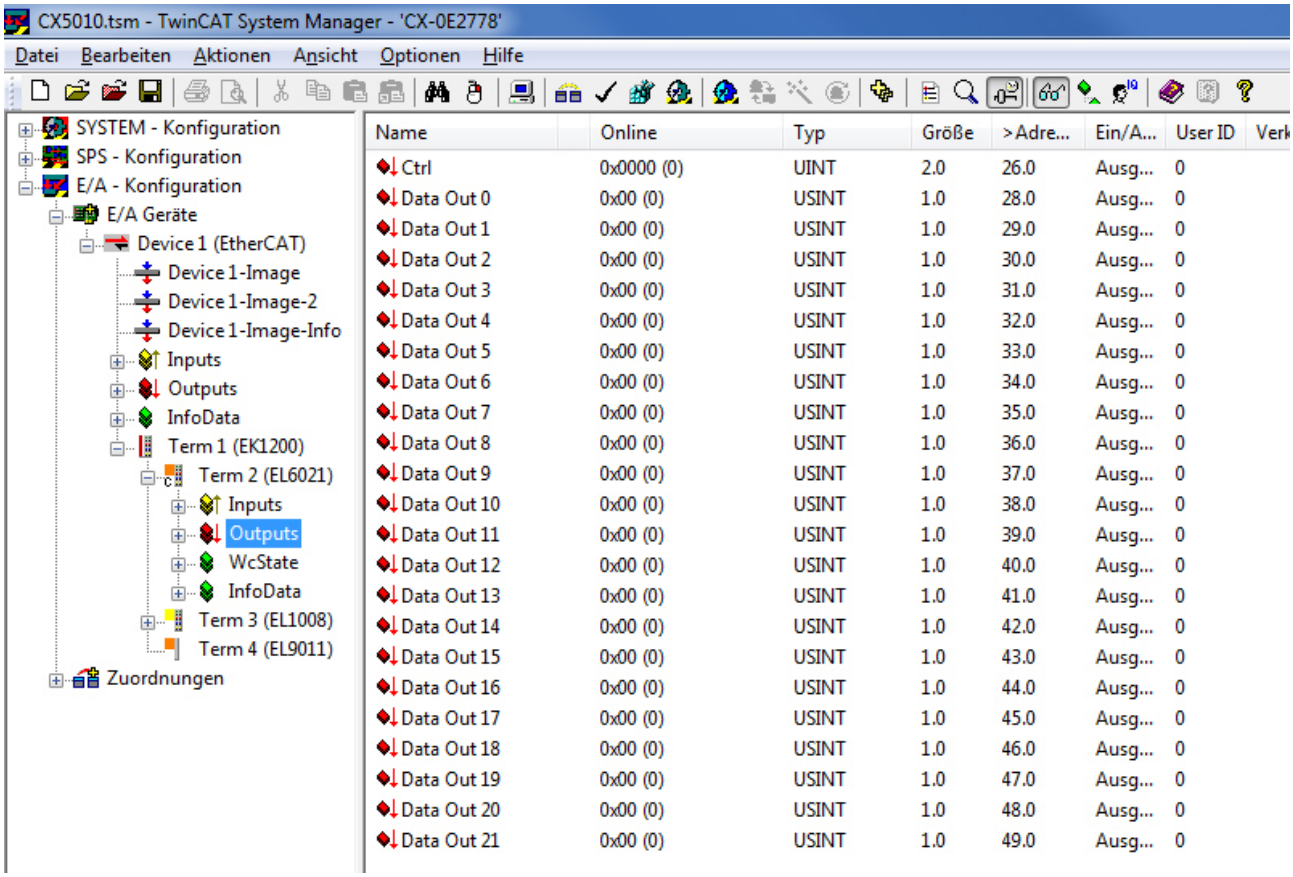


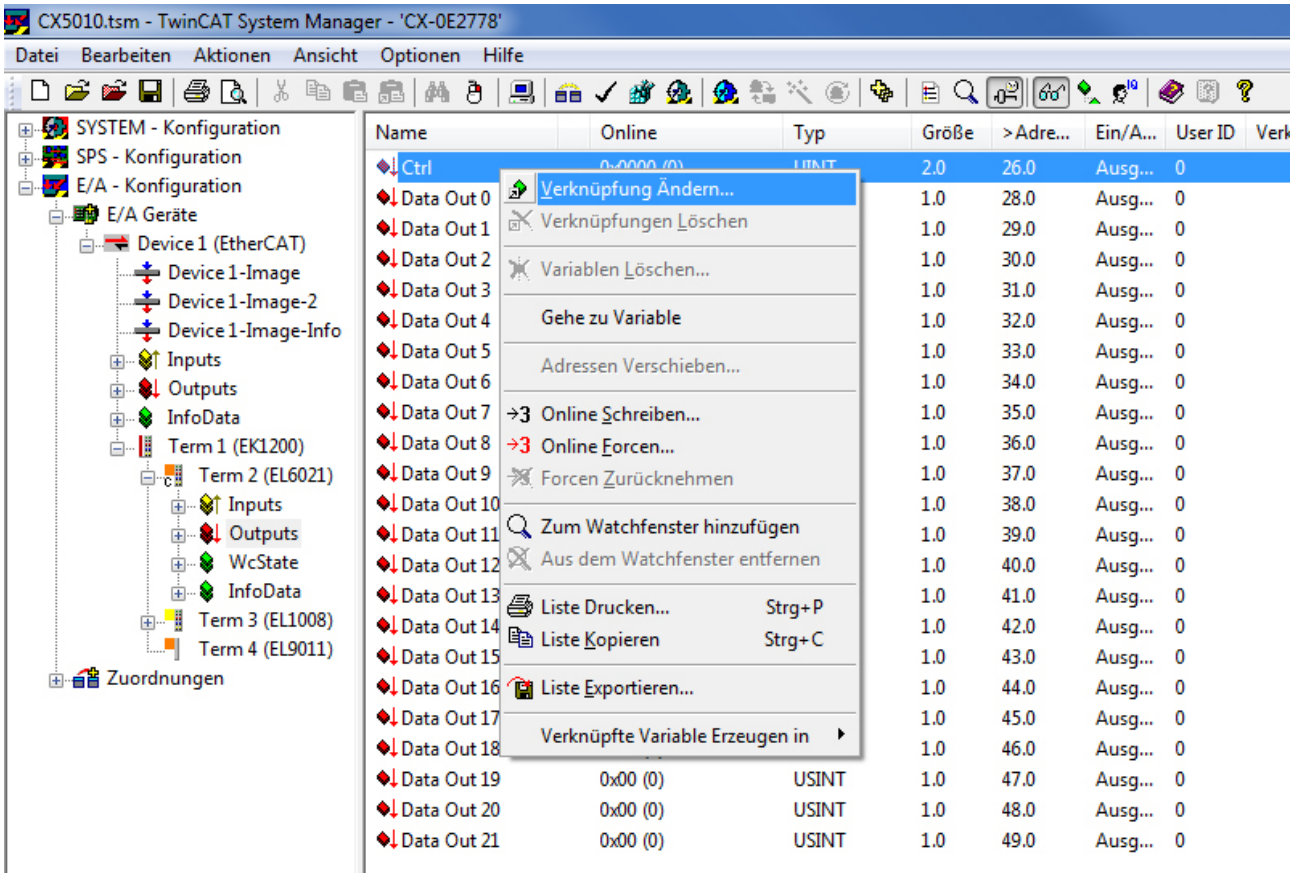


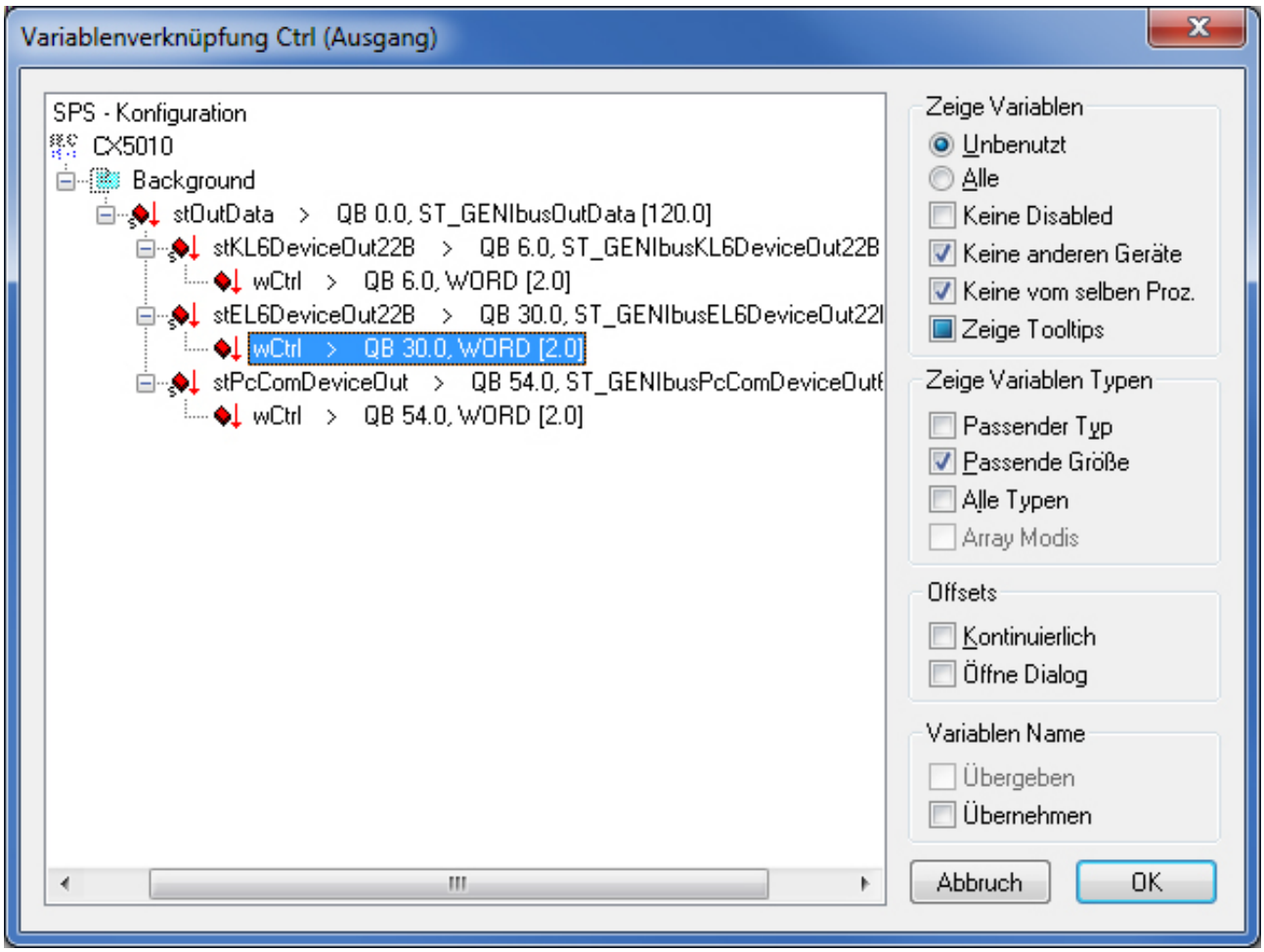
Danach können die Datenbytes bequem per Multi-Link auf die entsprechenden Variablen verknüpft werden. Die Auswahl mehrerer Variablen erreichen Sie dadurch, indem Sie "Data1" anklicken und bei gedrückter Shift-Taste die ↓-Taste betätigen.

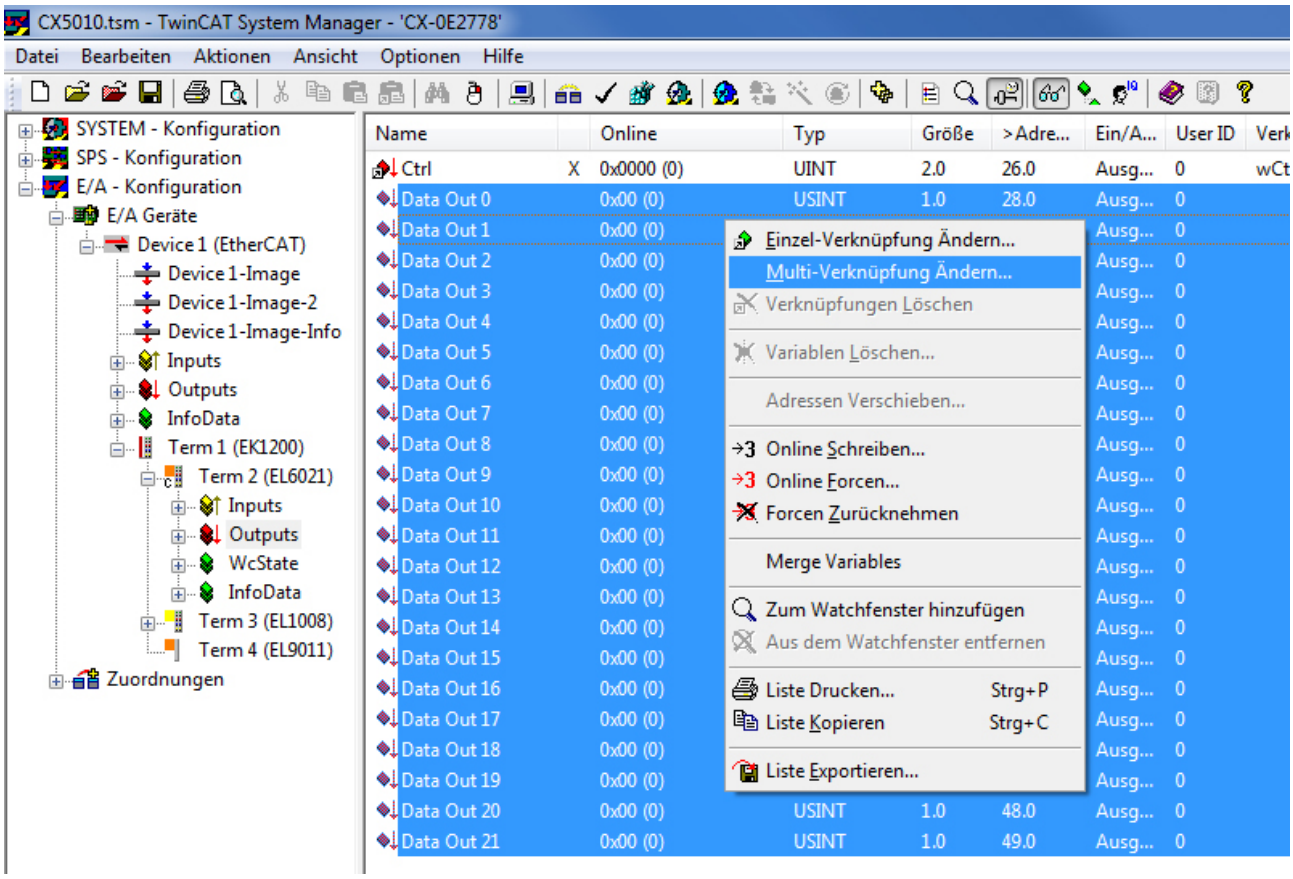


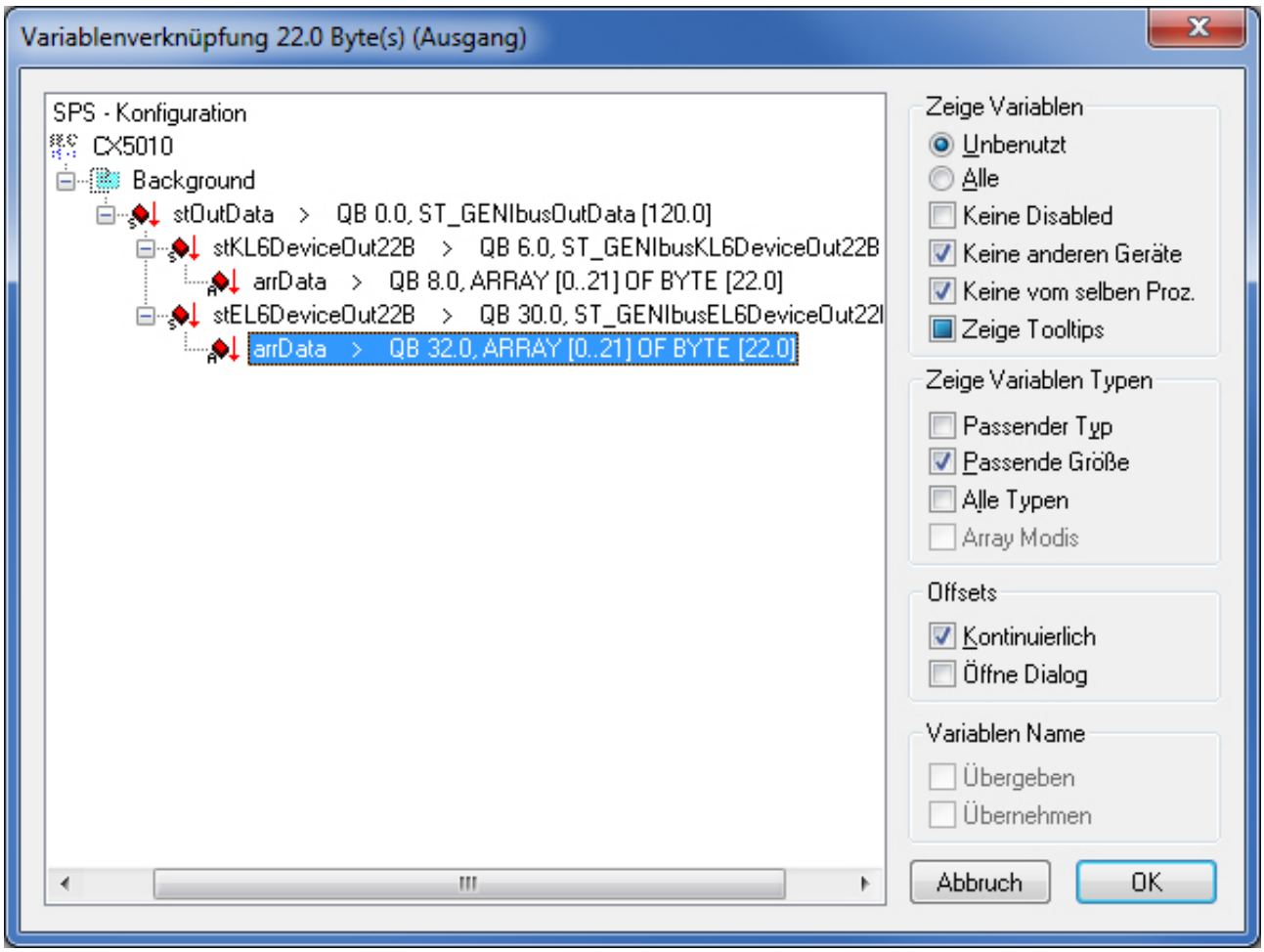
Die Ausgangsvariablen sind analog zu verknüpfen:


CX5010.tsm - TwinCAT System Manager - 'CX-0E2778'
Datei Bearbeiten Aktionen Ansicht Optionen Hilfe
SYSTEM - Konfiguration
SPS - Konfiguration
E/A - Konfiguration
E/A Geräte
Device 1 (EtherCAT)
Device 1-Image
Device 1-Image-2
Device 1-Image-Info
Inputs
Outputs
InfoData
Term 1 (EK1200)
Term 2 (EL6021)
Inputs
Outputs
WcState
InfoData
Term 3 (EL1008)
Term 4 (EL9011)
Zuordnungen
Name Online Typ Größe >Adre... Ein/A... User ID
Ctrl 0x0000 (0) UINT 2.0 26.0 Ausg... 0
Data Out 0 0x00 (0) USINT 1.0 28.0 Ausg... 0
Data Out 1 0x00 (0) USINT 1.0 29.0 Ausg... 0
Data Out 2 0x00 (0) USINT 1.0 30.0 Ausg... 0
Data Out 3 0x00 (0) USINT 1.0 31.0 Ausg... 0
Data Out 4 0x00 (0) USINT 1.0 32.0 Ausg... 0
Data Out 5 0x00 (0) USINT 1.0 33.0 Ausg... 0
Data Out 6 0x00 (0) USINT 1.0 34.0 Ausg... 0
Data Out 7 0x00 (0) USINT 1.0 35.0 Ausg... 0
Data Out 8 0x00 (0) USINT 1.0 36.0 Ausg... 0
Data Out 9 0x00 (0) USINT 1.0 37.0 Ausg... 0
Data Out 10 0x00 (0) USINT 1.0 38.0 Ausg... 0
Data Out 11 0x00 (0) USINT 1.0 39.0 Ausg... 0
Data Out 12 0x00 (0) USINT 1.0 40.0 Ausg... 0
Data Out 13 0x00 (0) USINT 1.0 41.0 Ausg... 0
Data Out 14 0x00 (0) USINT 1.0 42.0 Ausg... 0
Data Out 15 0x00 (0) USINT 1.0 43.0 Ausg... 0
Data Out 16 0x00 (0) USINT 1.0 44.0 Ausg... 0
Data Out 17 0x00 (0) USINT 1.0 45.0 Ausg... 0
Data Out 18 0x00 (0) USINT 1.0 46.0 Ausg... 0
Data Out 19 0x00 (0) USINT 1.0 47.0 Ausg... 0
Data Out 20 0x00 (0) USINT 1.0 48.0 Ausg... 0
Data Out 21 0x00 (0) USINT 1.0 49.0 Ausg... 0

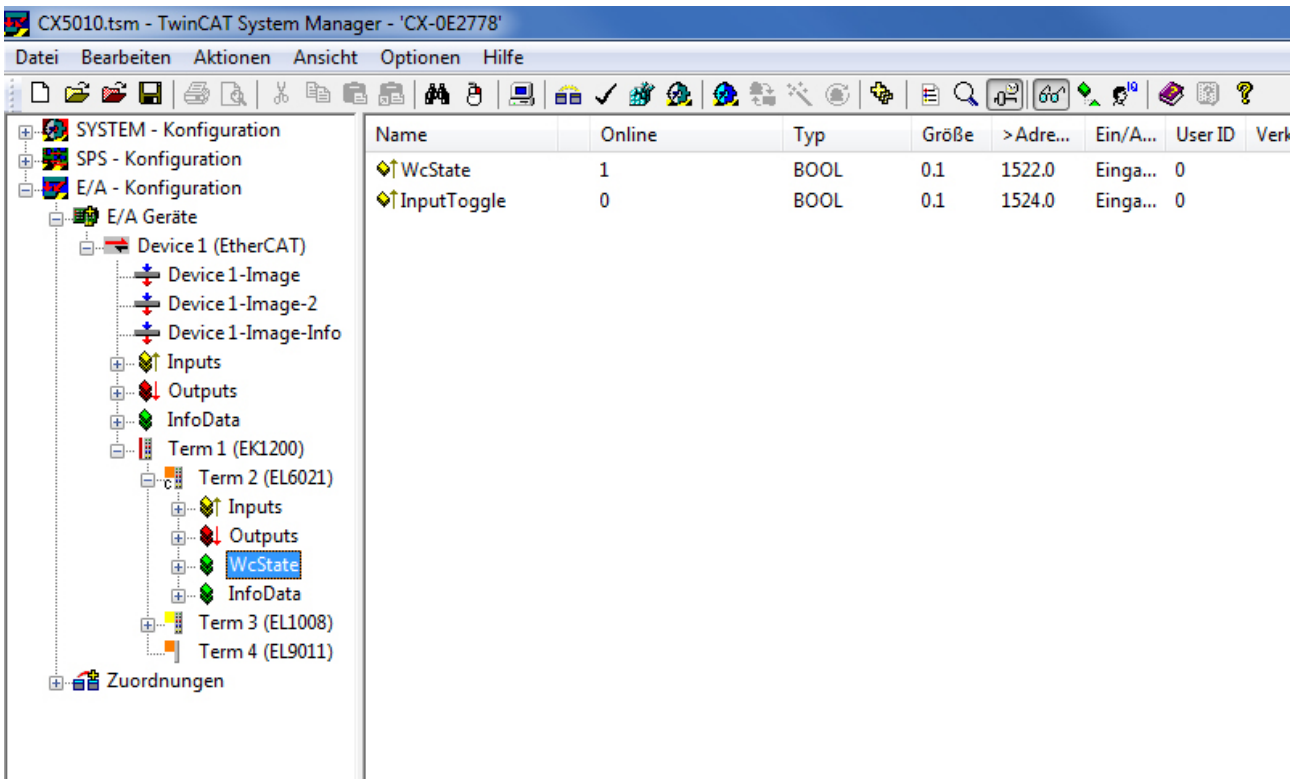

CX5010.tsm - TwinCAT System Manager - 'CX-0E2778'
Datei Bearbeiten Aktionen Ansicht Optionen Hilfe
SYSTEM - Konfiguration
SPS - Konfiguration
E/A - Konfiguration
E/A Geräte
Device 1 (EtherCAT)
Device 1-Image
Device 1-Image-2
Device 1-Image-Info
Inputs
Outputs
InfoData
Term 1 (EK1200)
Term 2 (EL6021)
Inputs
Outputs
WcState
InfoData
Term 3 (EL1008)
Term 4 (EL9011)
Zuordnungen
Name Online Typ Größe >Adre... Ein/A... User ID
Verknüpfung Ändern...
Verknüpfungen Löschen
Variablen Löschen...
Gehe zu Variable
Adressen Verschieben...
Online Schreiben...
Online Forcen...
Forcen Zurücknehmen
Zum Watchfenster hinzufügen
Aus dem Watchfenster entfernen
Liste Drucken... Strg+P
Liste Kopieren Strg+C
Liste Exportieren...
Verknüpfte Variable Erzeugen in

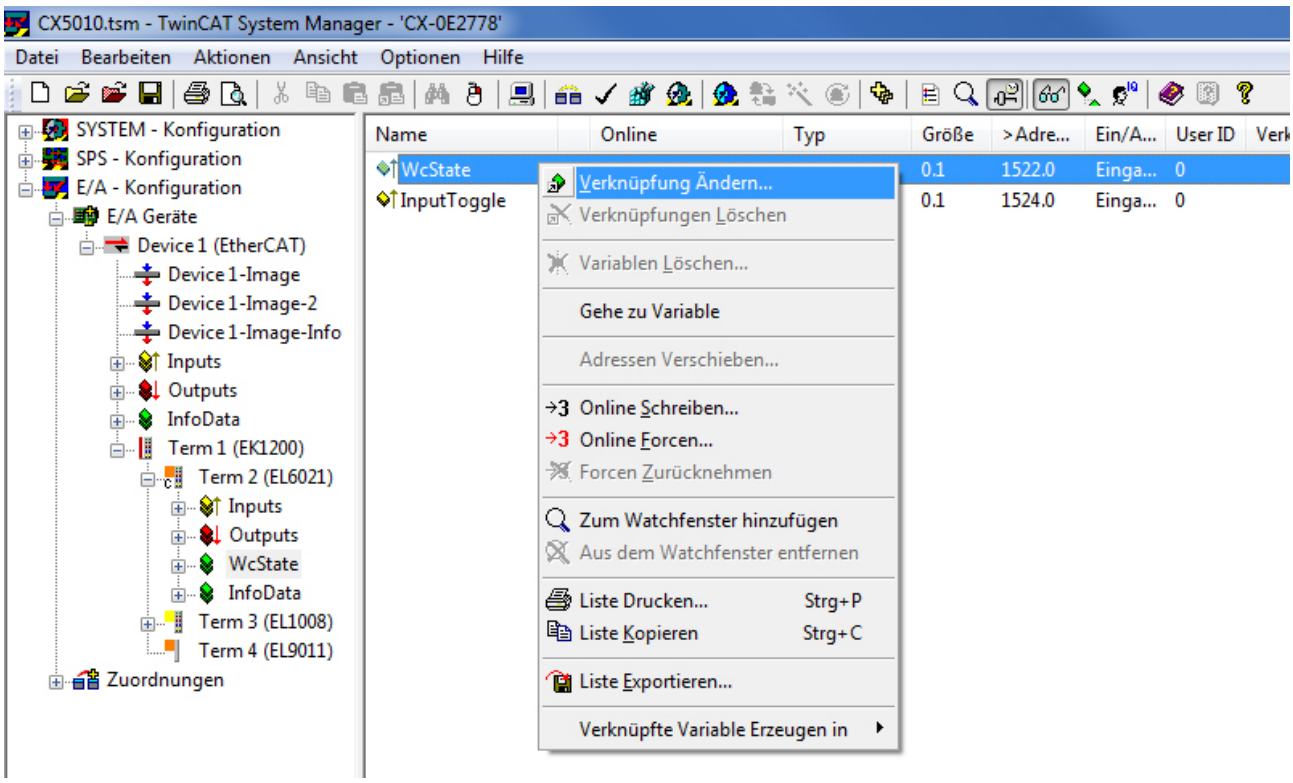

Variablenverknüpfung Ctrl (Ausgang)
SPS - Konfiguration
CX5010
Background
stOutData > QB 0.0, ST_GENIbusOutData [120.0]
stKL6DeviceOut22B > QB 6.0, ST_GENIbusKL6DeviceOut22B
wCtrl > QB 6.0, WORD [2.0]
stEL6DeviceOut22B > QB 30.0, ST_GENIbusEL6DeviceOut22I
wCtrl > QB 30.0, WORD [2.0]
stPcComDeviceOut > QB 54.0, ST_GENIbusPcComDeviceOutI
wCtrl > QB 54.0, WORD [2.0]
Zeige Variablen
Unbenutzt
Alle
Keine Disabled
Keine anderen Geräte
Keine vom selben Proz.
Zeige Tooltips
Zeige Variablen Typen
Passender Typ
Passende Größe
Alle Typen
Array Modis
Offsets
Kontinuierlich
Öffne Dialog
Variablen Name
Übergeben
Übernehmen
Abbruch
OK

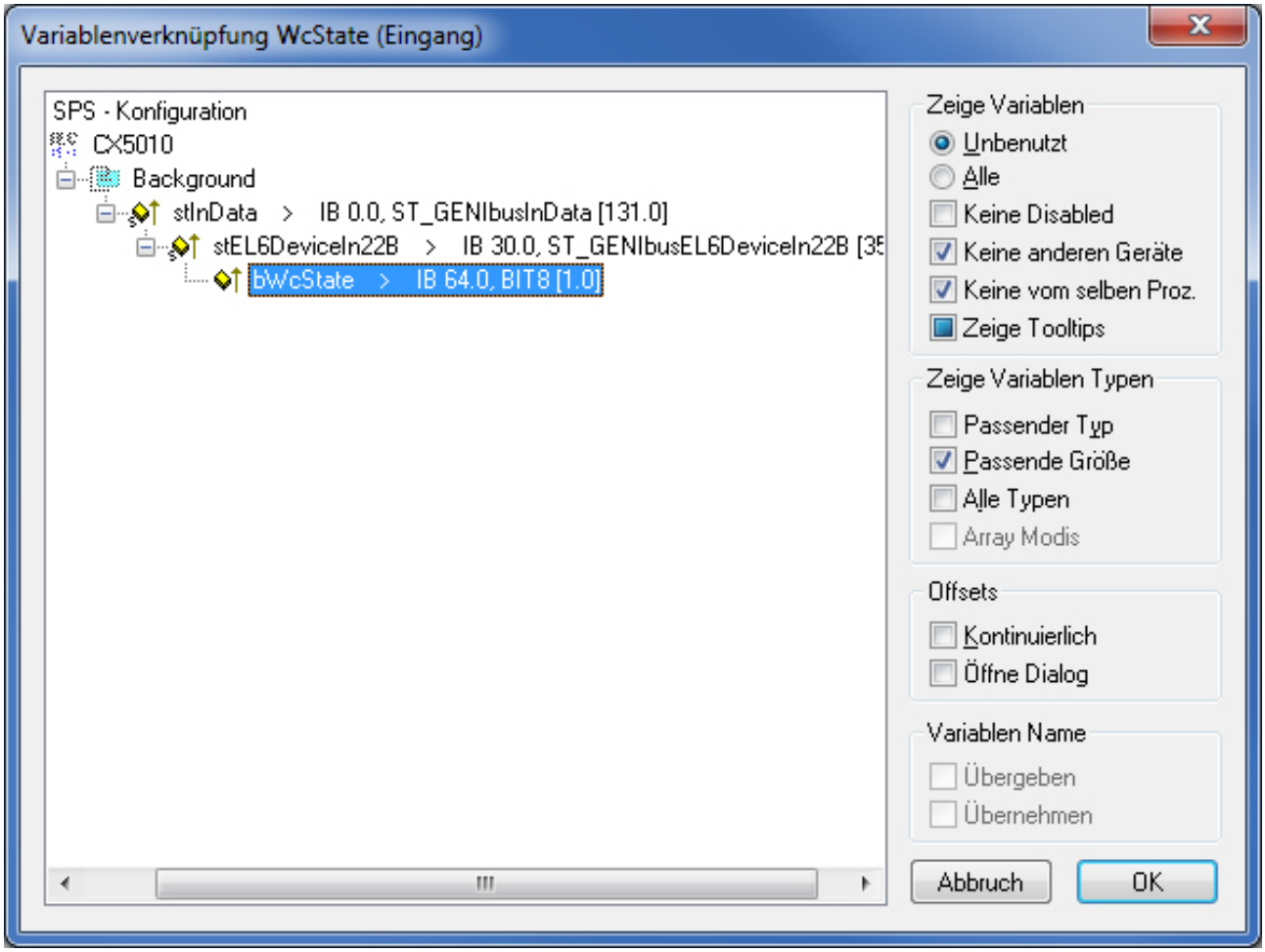

CX5010.tsm - TwinCAT System Manager - 'CX-0E2778'
Datei Bearbeiten Aktionen Ansicht Optionen Hilfe
SYSTEM - Konfiguration
SPS - Konfiguration
E/A - Konfiguration
E/A Geräte
Device 1 (EtherCAT)
Device 1-Image
Device 1-Image-2
Device 1-Image-Info
Inputs
Outputs
InfoData
Term 1 (EK1200)
Term 2 (EL6021)
Inputs
Outputs
WcState
InfoData
Term 3 (EL1008)
Term 4 (EL9011)
Zuordnungen
Name Online Typ Größe >Adre... Ein/A... User ID
Ctrl X 0x0000 (0) UINT 2.0 26.0 Ausg... 0
Data Out 0 0x00 (0) USINT 1.0 28.0 Ausg... 0
Data Out 1 0x00 (0) Ausg... 0
Data Out 2 0x00 (0) Ausg... 0
Data Out 3 0x00 (0) Ausg... 0
Data Out 4 0x00 (0) Ausg... 0
Data Out 5 0x00 (0) Ausg... 0
Data Out 6 0x00 (0) Ausg... 0
Data Out 7 0x00 (0) Ausg... 0
Data Out 8 0x00 (0) Ausg... 0
Data Out 9 0x00 (0) Ausg... 0
Data Out 10 0x00 (0) Ausg... 0
Data Out 11 0x00 (0) Ausg... 0
Data Out 12 0x00 (0) Ausg... 0
Data Out 13 0x00 (0) Ausg... 0
Data Out 14 0x00 (0) Ausg... 0
Data Out 15 0x00 (0) Ausg... 0
Data Out 16 0x00 (0) Ausg... 0
Data Out 17 0x00 (0) Ausg... 0
Data Out 18 0x00 (0) Ausg... 0
Data Out 19 0x00 (0) Ausg... 0
Data Out 20 0x00 (0) USINT 1.0 48.0 Ausg... 0
Data Out 21 0x00 (0) USINT 1.0 49.0 Ausg... 0
Einzel-Verknüpfung Ändern...
Multi-Verknüpfung Ändern...
Verknüpfungen Löschen
Variablen Löschen...
Adressen Verschieben...
Online Schreiben...
Online Forcen...
Forcen Zurücknehmen
Merge Variables
Zum Watchfenster hinzufügen
Aus dem Watchfenster entfernen
Liste Drucken... Strg+P
Liste Kopieren Strg+C
Liste Exportieren...

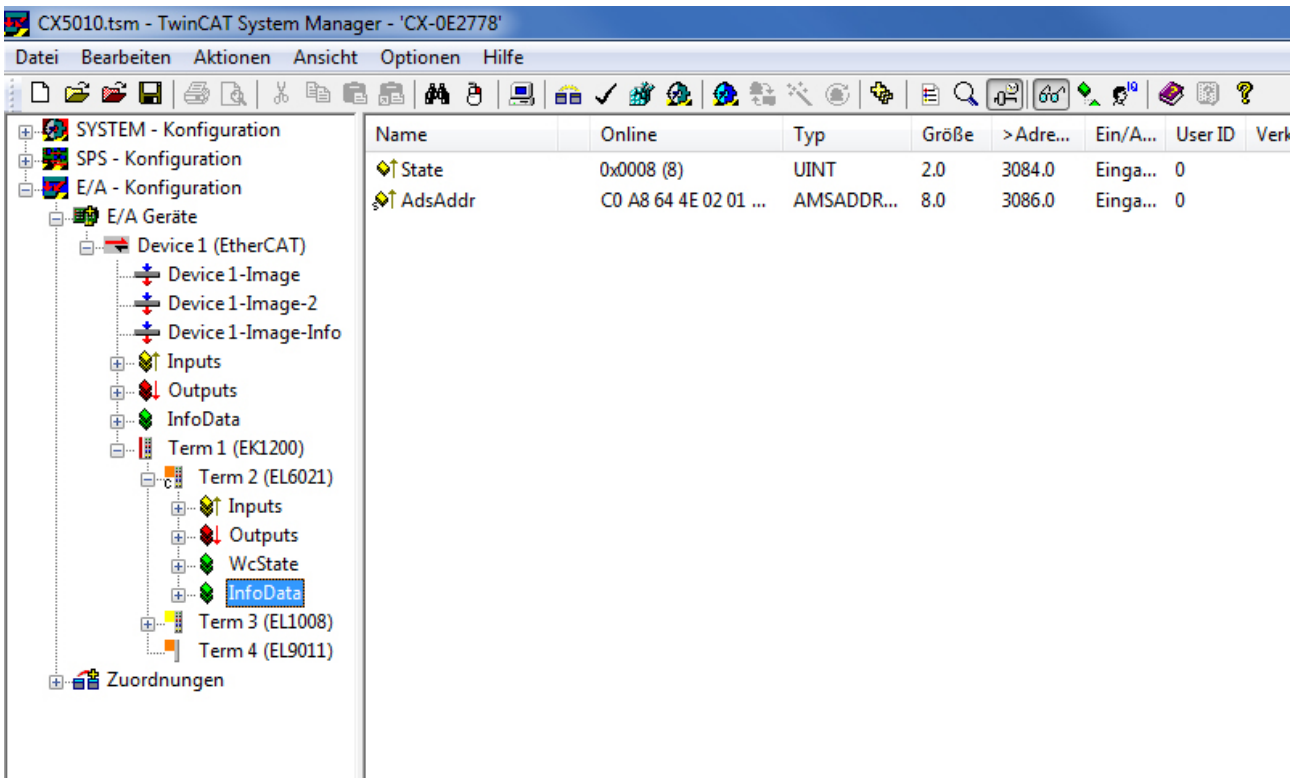


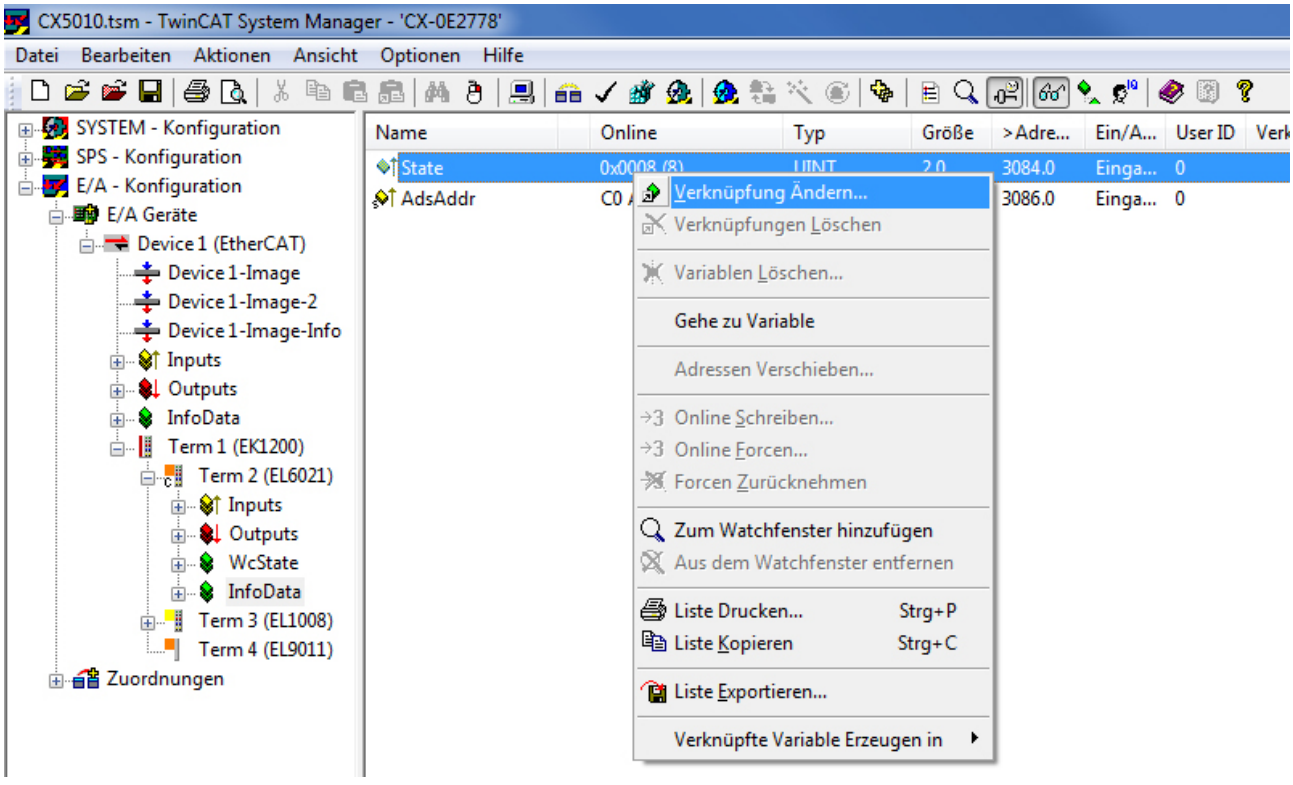
Des Weiteren müssen noch die Variablen der Gruppen *WcState* und *InfoData* verknüpft werden:

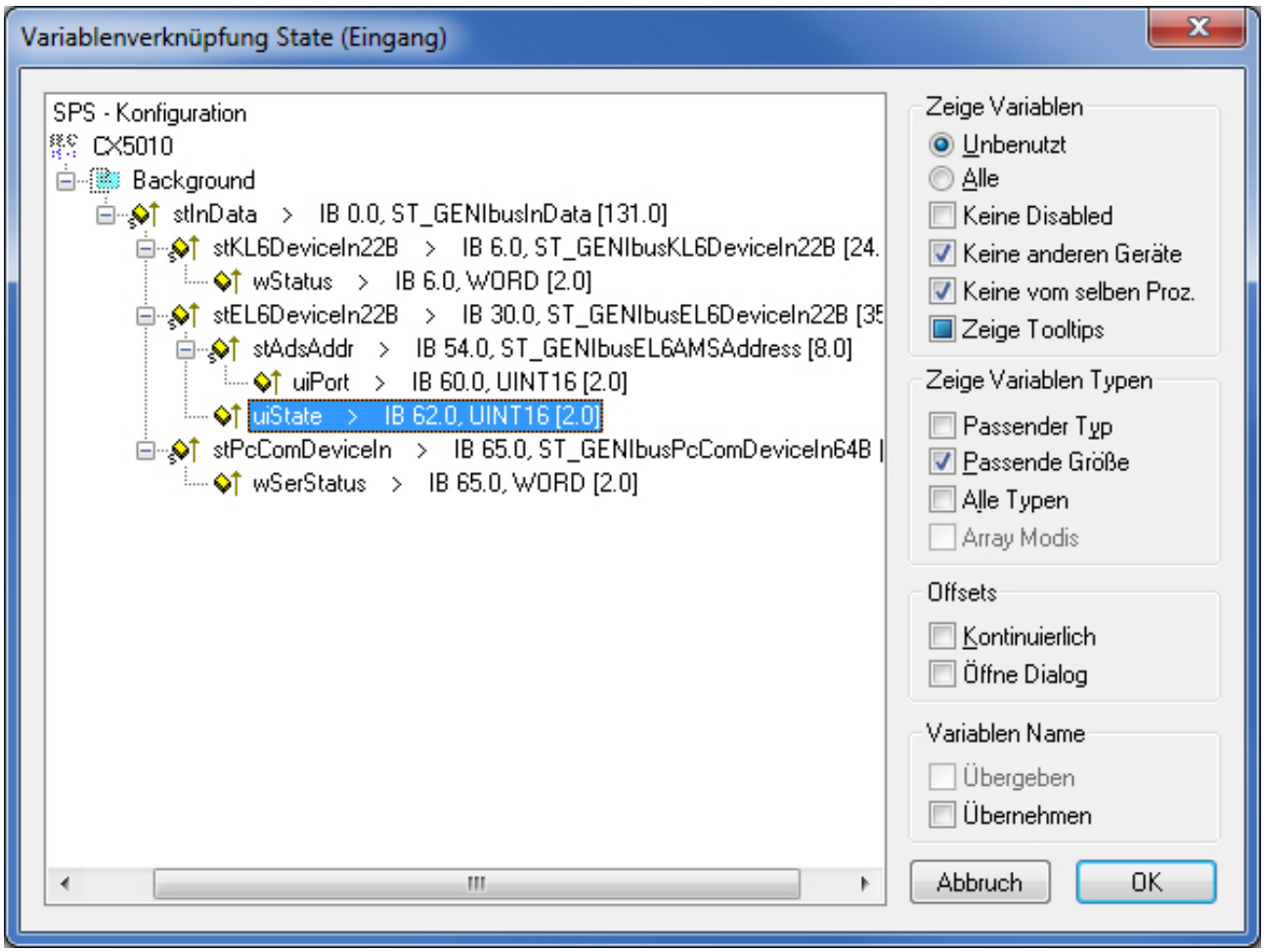


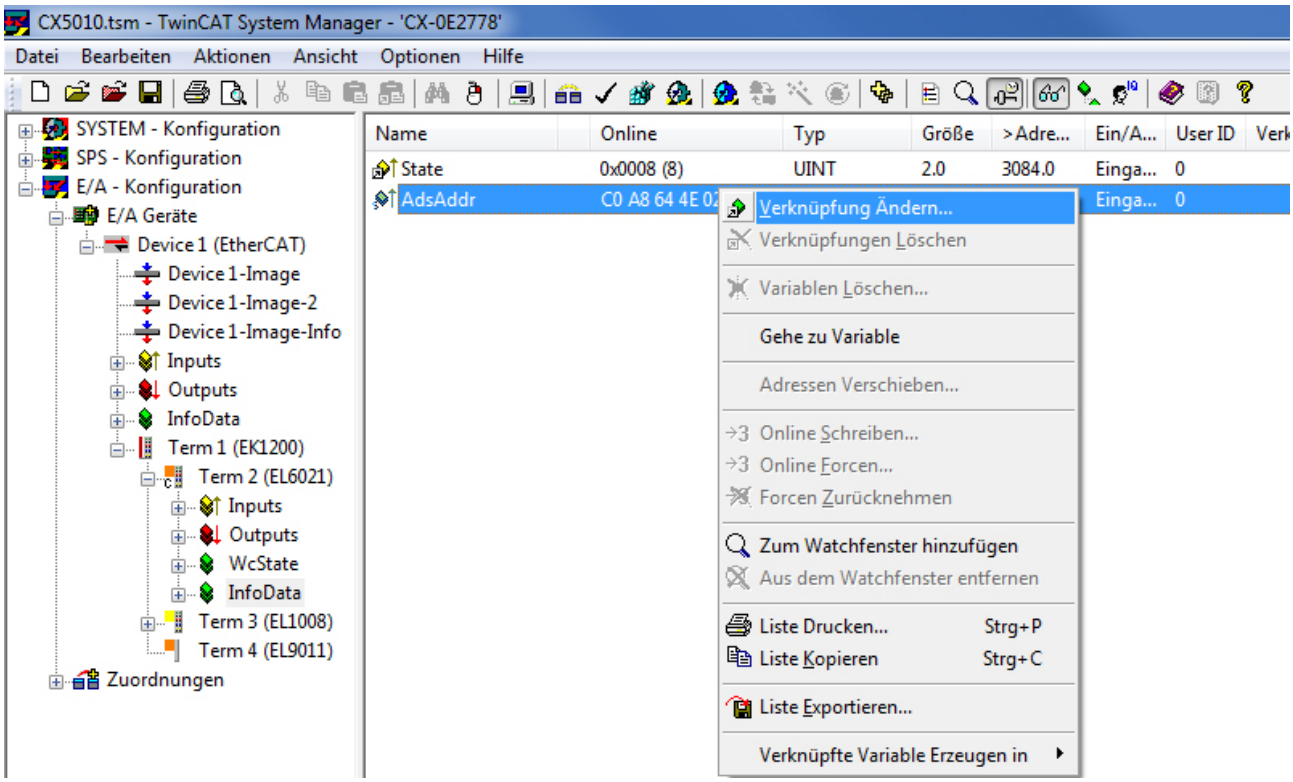

CX5010.tsm - TwinCAT System Manager - 'CX-0E2778'
Datei Bearbeiten Aktionen Ansicht Optionen Hilfe
SYSTEM - Konfiguration
SPS - Konfiguration
E/A - Konfiguration
E/A Geräte
Device 1 (EtherCAT)
Device 1-Image
Device 1-Image-2
Device 1-Image-Info
Inputs
Outputs
InfoData
Term 1 (EK1200)
Term 2 (EL6021)
Inputs
Outputs
WcState
InfoData
Term 3 (EL1008)
Term 4 (EL9011)
Zuordnungen
Name
Online
Typ
Größe
>Adre...
Ein/A...
User ID
WcState
0.1
1522.0
Einga...
0
InputToggle
0.1
1524.0
Einga...
0
Verknüpfung Ändern...
Verknüpfungen Löschen
Variablen Löschen...
Gehe zu Variable
Adressen Verschieben...
Online Schreiben...
Online Forcen...
Forcen Zurücknehmen
Zum Watchfenster hinzufügen
Aus dem Watchfenster entfernen
Liste Drucken... Strg+P
Liste Kopieren Strg+C
Liste Exportieren...
Verknüpfte Variable Erzeugen in

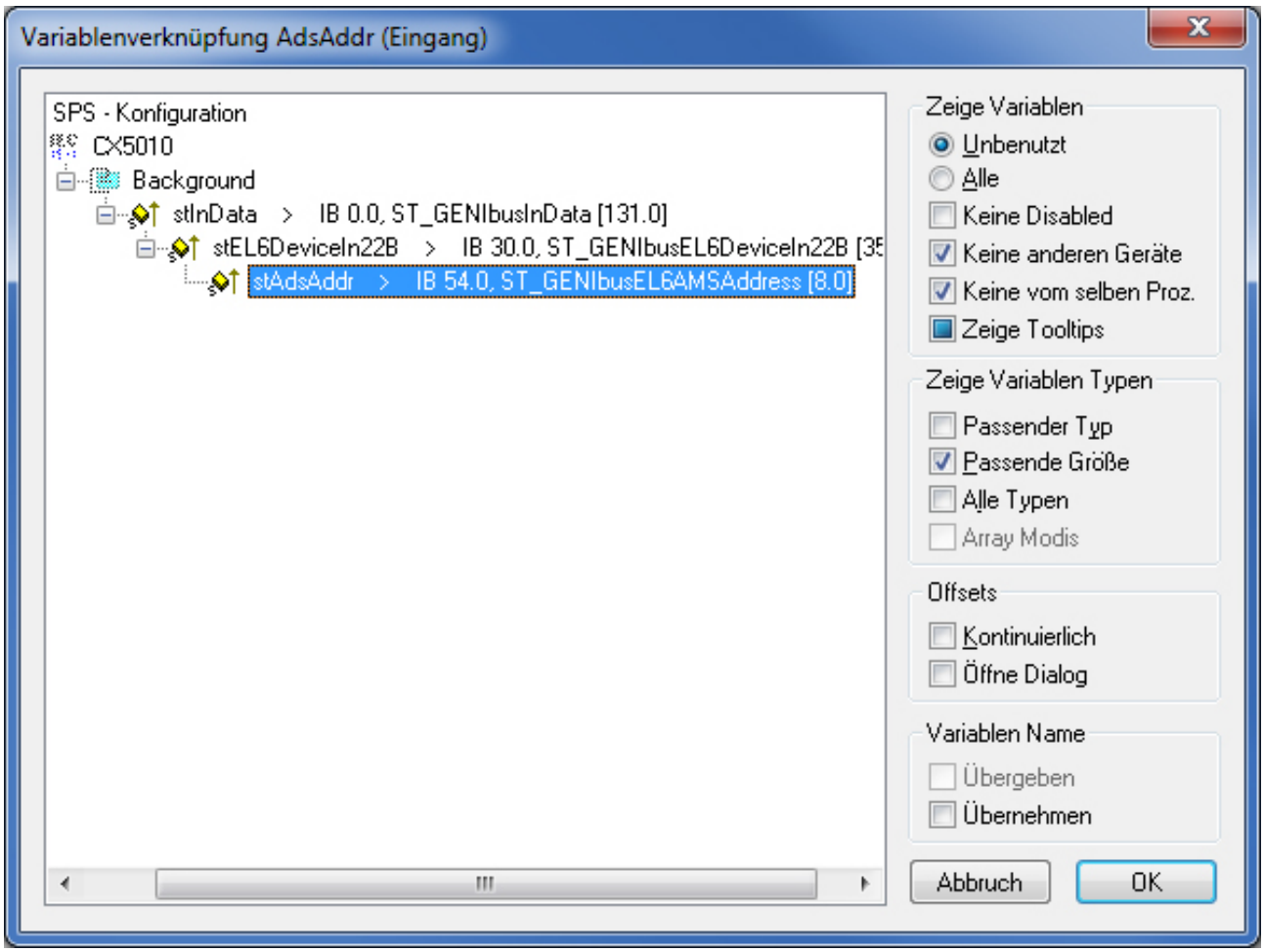

Variablenverknüpfung WcState (Eingang)
SPS - Konfiguration
CX5010
Background
stInData > IB 0.0, ST_GENIbusInData [131.0]
stEL6DeviceIn22B > IB 30.0, ST_GENIbusEL6DeviceIn22B [35
bWcState > IB 64.0, BIT8 [1.0]
Zeige Variablen
Unbenutzt
Alle
Keine Disabled
Keine anderen Geräte
Keine vom selben Proz.
Zeige Tooltips
Zeige Variablen Typen
Passender Typ
Passende Größe
Alle Typen
Array Modis
Offsets
Kontinuierlich
Öffne Dialog
Variablen Name
Übergeben
Übernehmen
Abbruch
OK


CX5010.tsm - TwinCAT System Manager - 'CX-0E2778'
Datei Bearbeiten Aktionen Ansicht Optionen Hilfe
SYSTEM - Konfiguration
SPS - Konfiguration
E/A - Konfiguration
E/A Geräte
Device 1 (EtherCAT)
Device 1-Image
Device 1-Image-2
Device 1-Image-Info
Inputs
Outputs
InfoData
Term 1 (EK1200)
Term 2 (EL6021)
Inputs
Outputs
WcState
InfoData
Term 3 (EL1008)
Term 4 (EL9011)
Zuordnungen
Name Online Typ Größe >Adre... Ein/A... User ID
State 0x0008 (8) UINT 2.0 3084.0 Einga... 0
AdsAddr C0 A8 64 4E 02 01 ... AMSADDR... 8.0 3086.0 Einga... 0


CX5010.tsm - TwinCAT System Manager - 'CX-0E2778'
Datei Bearbeiten Aktionen Ansicht Optionen Hilfe
SYSTEM - Konfiguration
SPS - Konfiguration
E/A - Konfiguration
E/A Geräte
Device 1 (EtherCAT)
Device 1-Image
Device 1-Image-2
Device 1-Image-Info
Inputs
Outputs
InfoData
Term 1 (EK1200)
Term 2 (EL6021)
Inputs
Outputs
WcState
InfoData
Term 3 (EL1008)
Term 4 (EL9011)
Zuordnungen
Name Online Typ Größe >Adre... Ein/A... User ID
State 0x0008 (8) UINT 2.0 3084.0 Einga... 0
AdsAddr C0 A 3086.0 Einga... 0
Verknüpfung Ändern...
Verknüpfungen Löschen
Variablen Löschen...
Gehe zu Variable
Adressen Verschieben...
Online Schreiben...
Online Forcen...
Forcen Zurücknehmen
Zum Watchfenster hinzufügen
Aus dem Watchfenster entfernen
Liste Drucken... Strg+P
Liste Kopieren Strg+C
Liste Exportieren...
Verknüpfte Variable Erzeugen in


Variablenverknüpfung State (Eingang)
SPS - Konfiguration
CX5010
Background
stInData > IB 0.0, ST_GENIbusInData [131.0]
stKL6DeviceIn22B > IB 6.0, ST_GENIbusKL6DeviceIn22B [24.
wStatus > IB 6.0, WORD [2.0]
stEL6DeviceIn22B > IB 30.0, ST_GENIbusEL6DeviceIn22B [35
stAdsAddr > IB 54.0, ST_GENIbusEL6AMSAddress [8.0]
uiPort > IB 60.0, UINT16 [2.0]
uiState > IB 62.0, UINT16 [2.0]
stPcComDeviceIn > IB 65.0, ST_GENIbusPcComDeviceIn64B
wSerStatus > IB 65.0, WORD [2.0]
Zeige Variablen
Unbenutzt
Alle
Keine Disabled
Keine anderen Geräte
Keine vom selben Proz.
Zeige Tooltips
Zeige Variablen Typen
Passender Typ
Passende Größe
Alle Typen
Array Modis
Offsets
Kontinuierlich
Öffne Dialog
Variablen Name
Übergeben
Übernehmen
Abbruch
OK


CX5010.tsm - TwinCAT System Manager - 'CX-0E2778'
Datei Bearbeiten Aktionen Ansicht Optionen Hilfe
SYSTEM - Konfiguration
SPS - Konfiguration
E/A - Konfiguration
E/A Geräte
Device 1 (EtherCAT)
Device 1-Image
Device 1-Image-2
Device 1-Image-Info
Inputs
Outputs
InfoData
Term 1 (EK1200)
Term 2 (EL6021)
Inputs
Outputs
WcState
InfoData
Term 3 (EL1008)
Term 4 (EL9011)
Zuordnungen
Name Online Typ Größe >Adre... Ein/A... User ID
State 0x0008 (8) UINT 2.0 3084.0 Einga... 0
AdsAddr C0 A8 64 4E 02 Einga... 0
Verknüpfung Ändern...
Verknüpfungen Löschen
Variablen Löschen...
Gehe zu Variable
Adressen Verschieben...
Online Schreiben...
Online Forcen...
Forcen Zurücknehmen
Zum Watchfenster hinzufügen
Aus dem Watchfenster entfernen
Liste Drucken... Strg+P
Liste Kopieren Strg+C
Liste Exportieren...
Verknüpfte Variable Erzeugen in


Variablenverknüpfung AdsAddr (Eingang)
SPS - Konfiguration
CX5010
Background
stInData > IB 0.0, ST_GENIbusInData [131.0]
stEL6DeviceIn22B > IB 30.0, ST_GENIbusEL6DeviceIn22B [35
stAdsAddr > IB 54.0, ST_GENIbusEL6AMSAddress [8.0]
Zeige Variablen
Unbenutzt
Alle
Keine Disabled
Keine anderen Geräte
Keine vom selben Proz.
Zeige Tooltips
Zeige Variablen Typen
Passender Typ
Passende Größe
Alle Typen
Array Modis
Offsets
Kontinuierlich
Öffne Dialog
Variablen Name
Übergeben
Übernehmen
Abbruch
OK

5 Programmierung

5.1 Allgemeine Informationen

Installation

Ab TwinCAT 2.11 Build 2253 (R3 und x64 Engineering) wird die Bibliothek "TcGENIbus.lib" standardmäßig mitinstalliert.

Weitere erforderliche Bibliotheken

Für PC-Systeme (x86) und Embedded-PCs (CXxxxx):

- Standard.lib
- TcBase.lib
- TcSystem.lib
- TcUtilities.lib
- TcEtherCAT.lib

Speicherauslastung

Durch das Einbinden der Bibliothek wird bereits SPS-Programmspeicher verbraucht. Abhängig vom Applikationsprogramm ist es daher möglich, dass der verbleibende Speicher nicht ausreichend ist.

5.2 Funktionsbausteine

Basisbefehle

Name	Beschreibung
FB_GENIbusCommunication [▶ 45]	Liest sequentiell die GENIbus-Befehle aus den internen Puffern der SPS-Bibliothek aus und gibt diese zu der zu der seriellen Schnittstelle weiter.

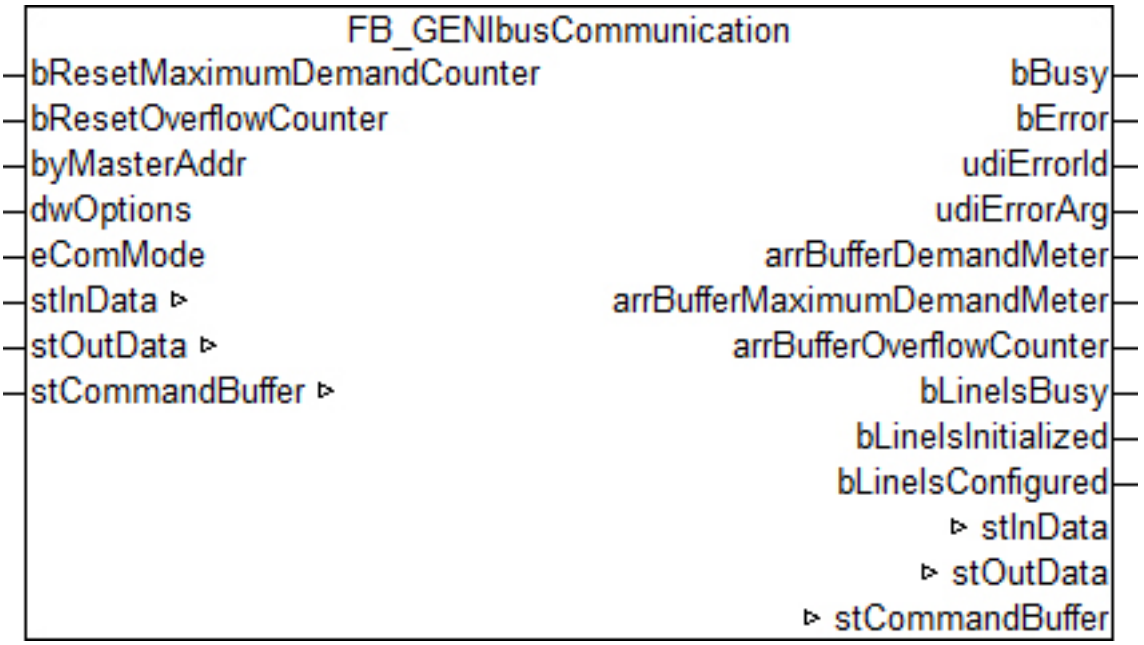
Grundbefehle

Name	Beschreibung
FB_GENIbusGet [▶ 47]	Liest einen Wert aus einem GENIbus-Gerät aus.
FB_GENIbusSet [▶ 48]	Schreibt einen Wert in ein GENIbus-Gerät oder führt ein Kommando (IDs der Klasse 3) aus.
FB_GENIbusInfo [▶ 50]	Liest den Informationsbereich einer ID aus.
FB_GENIbusGetMValue [▶ 51]	Liest einen Messwert aus einem GENIbus-Gerät aus.

Pumpen

Name	Beschreibung
FB_GENIbusMagnaPump [▶ 53]	Stellt eine universelle Applikation für eine Grundfos-Magna-Pumpe dar.

5.2.1 FB_GENIbusCommunication



Die Bausteine für die GENIbus-Befehle greifen nicht direkt auf das Prozessabbild der gewählten seriellen Schnittstelle zu, sondern legen die einzelnen GENIbus-Befehle in drei verschiedene Puffer ab. Der Baustein FB_GENIbusCommunication() liest sequentiell die GENIbus-Befehle aus diesen drei Puffern aus und gibt die GENIbus-Befehle zu der seriellen Schnittstelle weiter. Hierdurch wird sichergestellt, dass nicht mehrere Bausteine gleichzeitig auf das Prozessabbild der seriellen Schnittstelle zugreifen. Jeder dieser drei Puffer wird mit einer anderen Priorität (hoch, mittel oder niedrig) abgearbeitet. Der Anwender der SPS-Library kann durch den Parameter eCommandPriority [▶ 57], den es bei den meisten Bausteinen gibt, beeinflussen, mit welcher Priorität der jeweilige GENIbus-Befehl von dem Baustein FB_GENIbusCommunication() bearbeitet werden soll.

Die Puffer, in denen die GENIbus-Befehle abgelegt werden, sind alle in einer Variablen vom Typ ST_GENIbusCommandBuffer [▶ 58] enthalten. Pro serieller Schnittstelle gibt es eine Instanz vom Baustein FB_GENIbusCommunication() und eine Variable vom Typ ST_GENIbusCommandBuffer [▶ 58]. Der Baustein FB_GENIbusCommunication() sollte, wenn möglich, in einer separaten, schnelleren Task aufgerufen werden.

Über die Ausgänge des Bausteins kann ermittelt werden, wie stark die Puffer ausgelastet sind. Hierzu werden drei Arrays ausgegeben, bei dem jedes Element (0, 1 oder 2) für einen der drei Puffer (hoch, mittel oder niedrig) steht. Sollten Sie feststellen, dass einer der drei Puffer regelmäßig überläuft, so sollten Sie folgende Maßnahmen in Betracht ziehen:

- Wie stark sind die einzelnen SPS-Task ausgelastet? Der TwinCAT System Manager bietet zur Analyse entsprechende Hilfsmittel an.
- Versuchen Sie die Zykluszeit der Task, in der der Baustein FB_GENIbusCommunication() aufgerufen wird, zu verringern. Der Wert sollte nicht größer als 6 ms sein, optimal sind 2 ms.
- Überprüfen Sie die Zykluszeit der SPS-Task, in der die Bausteine für die einzelnen GENIbus-Befehle aufgerufen werden. Dieser Wert sollte zwischen 10 ms und 60 ms liegen.
- Vermeiden Sie möglichst das Pollen (regelmäßiges Auslesen) von Werten. Lesen Sie nur dann Werte aus, wenn diese auch benötigt werden.

VAR_INPUT

```
bResetMaximumDemandCounter  : BOOL;
bResetOverflowCounter       : BOOL;
byMasterAddr                : BYTE;
dwOptions                   : DWORD := 0;
eComMode                    : E_GENIbusComMode;
```

bResetMaximumDemandCounter: Eine steigende Flanke setzt den gespeicherten Wert der maximalen Befehlspuffer-Auslastung, *arrBufferMaximumDemandMeter* (0 - 100%, siehe VAR_OUTPUT), zurück.

bResetOverflowCounter: Eine steigende Flanke setzt den gespeicherten Wert der Anzahl der Befehlspuffer-Überläufe, *arrBufferOverflowCounter* (siehe VAR_OUTPUT), zurück.

byMasterAddr: Gibt die Adresse an, welche die TwinCAT-Steuerung innerhalb der GENIbus-Linie haben soll. Möglicher Eingabebereich: 0 - 31.

dwOptions: Reserviert für zukünftige Anwendungen.

eComMode: An diesem Parameter [▶ 57] muss die Auswahl der gewählten seriellen Kommunikationsschnittstelle eingetragen werden. Bei Verwendung einer KL-Klemme oder einer EtherCAT-Klemme wird dann intern automatisch eine Konfiguration der Verbindungsparameter gestartet:

- Baud Rate: 9600
- Data Bits: 8
- Parity: None
- Stop-Bits: 1

Für PC-basierte Schnittstellen ist dieses leider nicht möglich, dort müssen diese Parameter im TwinCAT System Manager direkt eingetragen werden.

VAR_OUTPUT

```
bBusy                       : BOOL;
bError                      : BOOL;
udiErrorId                  : UDINT;
udiErrorArg                 : UDINT;
arrBufferDemandMeter        : ARRAY[0..2] OF BYTE;
arrBufferMaximumDemandMeter : ARRAY[0..2] OF BYTE;
arrBufferOverflowCounter    : ARRAY[0..2] OF UINT;
bLineIsBusy                 : BOOL;
bLineIsInitialized          : BOOL;
bLineIsConfigured           : BOOL;
```

bBusy: Ein GENIbus-Befehl wird in ein serielles Telegramm gewandelt und versendet. Nach erfolgreicher Antwort oder Abbruch nach Fehler wird dieser Merker wieder zurück gesetzt.

bError: Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten.

udiErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Wird durch das erneute Aktivieren des Bausteins über den Eingang *bStart* wieder auf 0 zurückgesetzt. Siehe Fehlercodes [▶ 65].

udiErrorArg: Enthält ggf. eine erweiterte Beschreibung des Fehlercodes.

arrBufferDemandMeter: Belegung des jeweiligen Puffers (0 - 100%).

arrBufferMaximumDemandMeter: Bisherige maximale Auslastung des jeweiligen Puffers (0 - 100%).

arrBufferOverflowCounter: Bisherige Anzahl der Pufferüberläufe.

bLineIsBusy: Solange die Serielle Kommunikation aktiv ist, ist dieser Ausgang gesetzt.

bLineIsInitialized: Wird der Baustein das erste Mal aufgerufen (z. B. beim Starten der Steuerung), so wird eine Initialisierung durchgeführt. Während dieser Zeit können keine GENIbus-Befehle bearbeitet werden.

bLineIsConfigured: Dieser Ausgang zeigt mit TRUE an, dass die Klemme erfolgreich mit den o.a. seriellen Parametern konfiguriert wurde. Handelt es sich um eine PC-Schnittstelle, so ist dieser Ausgang automatisch gesetzt, da der Anwender die Parameter im TwinCAT System Manager selbst eintragen muss.

Da ein Fehler die Abarbeitung des Bausteines nicht unterbrechen darf, werden bError, udiErrorId und udiErrorArg in jedem SPS-Zyklus zunächst wieder zurückgesetzt und dann neu beurteilt.

VAR_IN_OUT

```
stInData              : ST_GENIbusInData;
stOutData             : ST_GENIbusOutData;
stCommandBuffer       : ST_GENIbusCommandBuffer;
```

stInData: Verweis auf die Struktur [▶ 60], welche das Eingangs-Prozessabbild zur Kommunikation mit der seriellen Schnittstelle enthält.

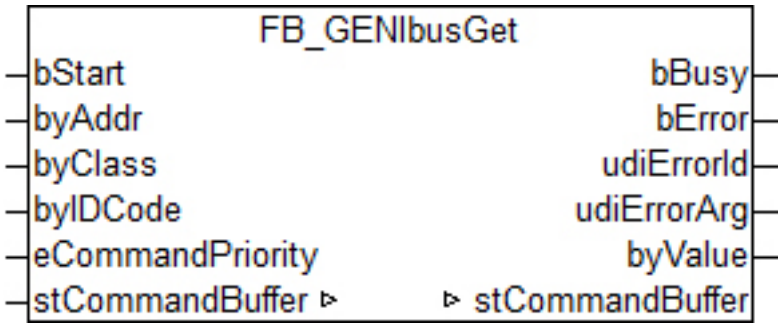
stOutData: Verweis auf die Struktur [▶ 62], welche das Ausgangs-Prozessabbild zur Kommunikation mit der seriellen Schnittstelle enthält.

stCommandBuffer: Verweis auf die Struktur [▶ 58] zur Kommunikation (Puffer) mit den GENIbus-Bausteinen.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	erforderliche Bibliotheken
TwinCAT v2.11 R3/x64 ab Build 2253	PC/CX	TcGENIbus-Bibliothek ab V1.0.0

5.2.2 FB_GENIbusGet



Dieser Baustein liest einen Wert aus einem GENIbus-Gerät aus.

VAR_INPUT

```
bStart              : BOOL;
byAddr              : BYTE := 0;
byClass             : BYTE := 2;
byIDCode            : BYTE := 0;
eCommandPriority    : E_GENIbusCommandPriority := eGENIbusCommandPriorityMiddle;
```

bStart: Eine steigende Flanke an diesem Eingang startet den Leseprozess.

byAdress : Adresse des anzusprechenden GENIbus-Gerätes: Gültige Eingaben: 1 - 200. Das entspricht der Einstellung, wie sie am GENIbus-Gerät direkt eingestellt wird. Eine Umrechnung auf den tatsächlichen Adressbereich von 32 - 231, siehe GENIbus-Standard, erfolgt Baustein-intern. Ein Broadcast-Befehl über die Adresse 255 ist naturgemäß nicht erlaubt.

byClass/byIDCode: Klasse und ID-Code des zu lesenden Speicherplatzes. GET-Befehle sind nur für die Klassen 2, 4, 5 und 7 zulässig - für alle anderen Einträge wird ein Fehler ausgegeben. Eine Einschränkung des ID-Code-Eintrages hingegen gibt es nicht, da diese Bereiche nicht lückenlos sind und unter Umständen erweitert werden.

eCommandPriority: Priorität [▶ 57] (hoch, mittel oder niedrig), mit der der Befehl von der SPS-Bibliothek abgearbeitet wird.

VAR_OUTPUT

```
bBusy          : BOOL;
bError         : BOOL;
udiErrorId     : UDINT;
udiErrorArg    : UDINT;
byValue        : BYTE;
```

bBusy: Beginnend mit der Flanke an *bStart* ist dieser Ausgang so lange auf TRUE, bis der Befehl abgearbeitet wurde.

bError: Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten.

udiErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Wird durch das erneute Aktivieren des Bausteins über den Eingang *bStart* wieder auf 0 zurückgesetzt. Siehe Fehlercodes [▶ 65].

udiErrorArg: Enthält ggf. eine erweiterte Beschreibung des Fehlercodes.

byValue: Ausgabe des gelesenen Wertes.

VAR_IN_OUT

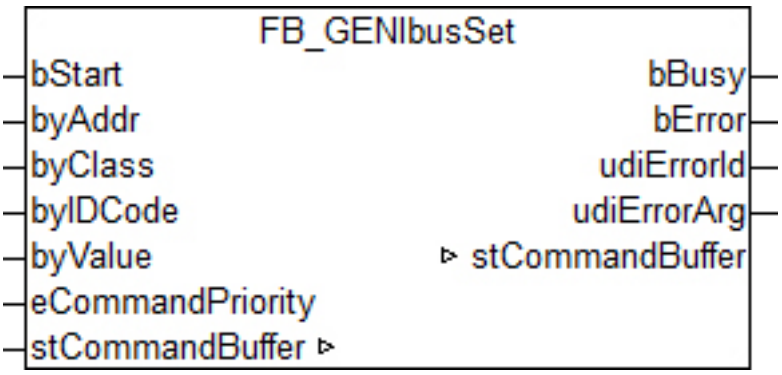
```
stCommandBuffer    : ST_GENIbusCommandBuffer;
```

stCommandBuffer: Verweis auf die Struktur [▶ 58] zur Kommunikation (Puffer) mit dem FB_GENIbusCommunication [▶ 45]()-Baustein.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	erforderliche Bibliotheken
TwinCAT v2.11 R3/x64 ab Build 2253	PC/CX	TcGENIbus-Bibliothek ab V1.0.0

5.2.3 FB_GENIbusSet



Dieser Baustein schreibt einen Wert in ein GENIbus-Gerät oder führt ein Kommando (IDs der Klasse 3) aus.

VAR_INPUT

```
bStart             : BOOL;
byAddr             : BYTE := 0;
byClass            : BYTE := 2;
byIDCode           : BYTE := 0;
byValue            : BYTE;
eCommandPriority   : E_GENIbusCommandPriority := eGENIbusCommandPriorityMiddle;
```

bStart: Eine steigende Flanke an diesem Eingang startet den Setzprozess.

byAdress : Adresse des anzusprechenden GENIbus-Gerätes: Gültige Eingaben: 1 - 200. Das entspricht der Einstellung, wie sie am GENIbus-Gerät direkt eingestellt wird. Eine Umrechnung auf den tatsächlichen Adressbereich von 32 - 231, siehe GENIbus-Standard, erfolgt Baustein-intern.

byClass/byIDCode: Klasse und ID-Code des zu schreibenden Speicherplatzes. SET-Befehle sind nur für die Klassen 3, 4 und 5 zulässig - für alle anderen Einträge wird ein Fehler ausgegeben. Eine Einschränkung des ID-Code-Eintrages hingegen gibt es nicht, da diese Bereiche nicht lückenlos sind und unter Umständen erweitert werden.

byValue: Wert der geschrieben werden soll. Im Falle von Kommando-IDs der Klasse 3 bleibt dieser Eintrag unberücksichtigt.

eCommandPriority: Priorität [▶ 57] (hoch, mittel oder niedrig), mit der der Befehl von der SPS-Bibliothek abgearbeitet wird.

VAR_OUTPUT

```
bBusy         : BOOL;
bError        : BOOL;
udiErrorId    : UDINT;
udiErrorArg   : UDINT;
```

bBusy: Beginnend mit der Flanke an *bStart* ist dieser Ausgang so lange auf TRUE, bis der Befehl abgearbeitet wurde.

bError: Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten.

udiErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Wird durch das erneute Aktivieren des Bausteins über den Eingang *bStart* wieder auf 0 zurückgesetzt. Siehe Fehlercodes [▶ 65].

udiErrorArg: Enthält ggf. eine erweiterte Beschreibung des Fehlercodes.

VAR_IN_OUT

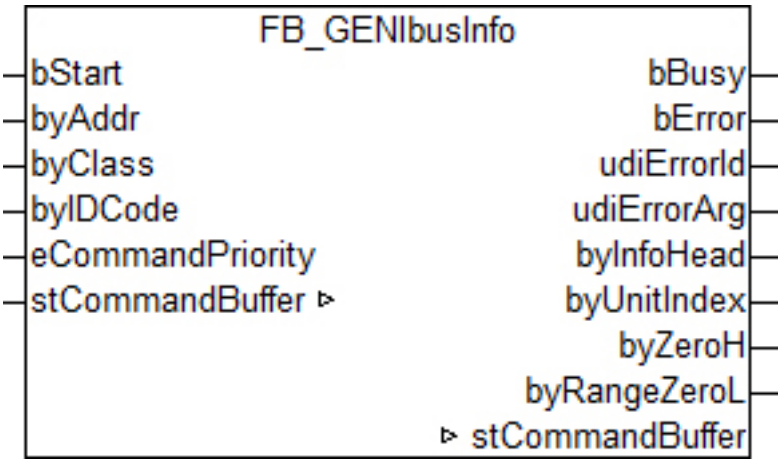
```
stCommandBuffer   : ST_GENIbusCommandBuffer;
```

stCommandBuffer: Verweis auf die Struktur [▶ 58] zur Kommunikation (Puffer) mit dem FB_GENIbusCommunication [▶ 45]()-Baustein.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	erforderliche Bibliotheken
TwinCAT v2.11 R3/x64 ab Build 2253	PC/CX	TcGENIbus-Bibliothek ab V1.0.0

5.2.4 FB_GENIbusInfo



Dieser Baustein liest den Informationsbereich einer ID aus.

VAR_INPUT

```
bStart            : BOOL;
byAddr            : BYTE := 0;
byClass           : BYTE := 2;
byIDCode          : BYTE := 0;
eCommandPriority  : E_GENIbusCommandPriority := eGENIbusCommandPriorityMiddle;
```

bStart: Eine steigende Flanke an diesem Eingang startet den Leseprozess.

byAdress : Adresse des anzusprechenden GENIbus-Gerätes: Gültige Eingaben: 1 - 200. Das entspricht der Einstellung, wie sie am GENIbus-Gerät direkt eingestellt wird. Eine Umrechnung auf den tatsächlichen Adressbereich von 32 - 231, siehe GENIbus-Standard, erfolgt Baustein-intern. Ein Broadcast-Befehl über die Adresse 255 ist naturgemäß nicht erlaubt.

byClass/byIDCode: Klasse und ID-Code des zu lesenden Speicherplatzes. INFO-Befehle sind nur für die Klassen 2, 3, 4 und 5 zulässig - für alle anderen Einträge wird ein Fehler ausgegeben. Eine Einschränkung des ID-Code-Eintrages hingegen gibt es nicht, da diese Bereiche nicht lückenlos sind und unter Umständen erweitert werden.

eCommandPriority: Priorität [▶ 57] (hoch, mittel oder niedrig), mit der der Befehl von der SPS-Bibliothek abgearbeitet wird.

VAR_OUTPUT

```
bBusy         : BOOL;
bError        : BOOL;
udiErrorId    : UDINT;
udiErrorArg   : UDINT;
byInfoHead    : BYTE;
byUnitIndex   : BYTE;
byZeroH       : BYTE;
byRangeZeroL  : BYTE;
```

bBusy: Beginnend mit der Flanke an *bStart* ist dieser Ausgang so lange auf TRUE, bis der Befehl abgearbeitet wurde.

bError: Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten.

udiErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Wird durch das erneute Aktivieren des Bausteins über den Eingang *bStart* wieder auf 0 zurückgesetzt. Siehe Fehlercodes [▶ 65].

udiErrorArg: Enthält ggf. eine erweiterte Beschreibung des Fehlercodes.

byInfoHead: Skalierungsinformation

byUnitIndex: Vorzeichen und Einheit - kodiert.

byZeroH: Nullpunkt bei normaler Bereichs-Nullpunkt-Skalierung ODER High-Byte Nullpunkt bei erweiterter Skalierung.

byRangeZeroL: Bereich bei normaler Bereichs-Nullpunkt-Skalierung ODER Low-Byte Nullpunkt bei erweiterter Skalierung.

VAR_IN_OUT

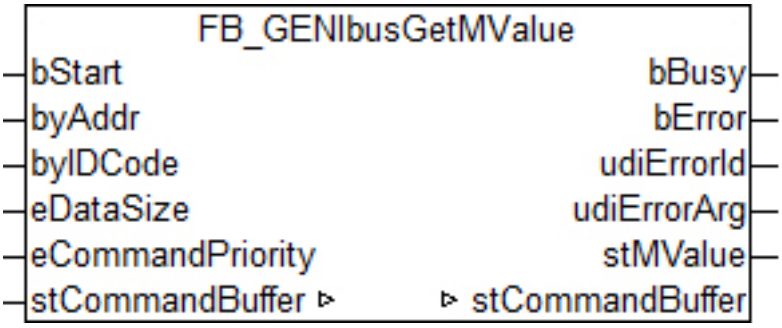
```
stCommandBuffer   : ST_GENIbusCommandBuffer;
```

stCommandBuffer: Verweis auf die Struktur [▶ 58] zur Kommunikation (Puffer) mit dem FB_GENIbusCommunication [▶ 45]()-Baustein.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	erforderliche Bibliotheken
TwinCAT v2.11 R3/x64 ab Build 2253	PC/CX	TcGENIbus-Bibliothek ab V1.0.0

5.2.5 FB_GENIbusGetMValue



Dieser Baustein liest einen Messwert aus einem GENIbus-Gerät aus. Die Operation beschränkt sich dabei ausschließlich auf Werte der Klasse 2. Es ist lediglich der ID-Code des Hi-Bytes und die Länge des Messwertes vorzugeben, die Art der Skalierung und die Messwerteinheit wird durch eine interne INFO-Abfrage ermittelt. Am Ausgang *stMValue* wird durch eine Struktur alle wichtigen Informationen über den Wert zur Verfügung gestellt.

VAR_INPUT

```
bStart            : BOOL;
byAddr            : BYTE := 0;
byIDCode          : BYTE := 0;
eDataSize         : E_GENIBusMDataSize;
eCommandPriority  : E_GENIbusCommandPriority := eGENIbusCommandPriorityMiddle;
```

bStart: Eine steigende Flanke an diesem Eingang startet den Leseprozess.

byAddr : Adresse des anzusprechenden GENIbus-Gerätes: Gültige Eingaben: 1 - 200. Das entspricht der Einstellung, wie sie am GENIbus-Gerät direkt eingestellt wird. Eine Umrechnung auf den tatsächlichen Adressbereich von 32 - 231, siehe GENIbus-Standard, erfolgt Baustein-intern. Ein Broadcast-Befehl über die Adresse 255 ist naturgemäß nicht erlaubt.

byIDCode: ID-Code des zu lesenden Wertes. Bei 16- 24 und 32-Bit-Werten ist hier die ID des Hi-Bytes anzugeben, wobei immer von folgender Reihenfolge ausgegangen wird: ID = hi, ID+1 = lo1, ID+2 = lo2, ID+3 = lo3.

eDataSize: Datengröße [▶ 57] des Messwertes: 8, 16, 24 oder 32 Byte.

eCommandPriority: Priorität [▶ 57] (hoch, mittel oder niedrig), mit der der Befehl von der SPS-Bibliothek abgearbeitet wird.

Beispiel: Auslesen des gesamten gepumpten Wasservolumens. Für diesen Fall ist:

- byIdCode = 121
- eDataSize = eGENIbusMSize32Bit

temp_in_3 '	2.118			R	Temperature input 3 (113) value
t_bear_de	2.119		5	R	Motor bearing temperature Drive End (DE)
t_bear_nde	2.120		5	R	Motor bearing temp. None Drive End (NDE)
volume_hi	2.121	1 m^3	5	R	Pumped volume (accumulated value of actual pump flow). Reset by command **RESET_HIST**
volume_lo1	2.122				
volume_lo2	2.123				
volume_lo3	2.124				
spec_energy_hi	2.125	1 Wh/m^3	5	R	Specific energy consumption
spec_energy_lo	2.126		5		
grf_sensor_press	2.127	INFO		R	Grundfos sensor pressure measurement GSP
grf_sensor_temp	2.128	INFO		R	Grundfos sensor temperature measurement GST

Quelle: Grundfos-Dokumentation "Operating the MAGNA3 and MGE model H/I via the GENIpro interface - Edition 01.00.35 - April 2015" .

VAR_OUTPUT

```
bBusy          : BOOL;
bError         : BOOL;
udiErrorId     : UDINT;
udiErrorArg    : UDINT;
stMValue       : ST_GENIbusMValue;
```

bBusy: Beginnend mit der Flanke an *bStart* ist dieser Ausgang so lange auf TRUE, bis der Befehl abgearbeitet wurde.

bError: Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten.

udiErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Wird durch das erneute Aktivieren des Bausteins über den Eingang *bStart* wieder auf 0 zurückgesetzt. Siehe Fehlercodes [▶ 65].

udiErrorArg: Enthält ggf. eine erweiterte Beschreibung des Fehlercodes.

stMValue : Ausgabe [▶ 62] des gelesenen Wertes.

VAR_IN_OUT

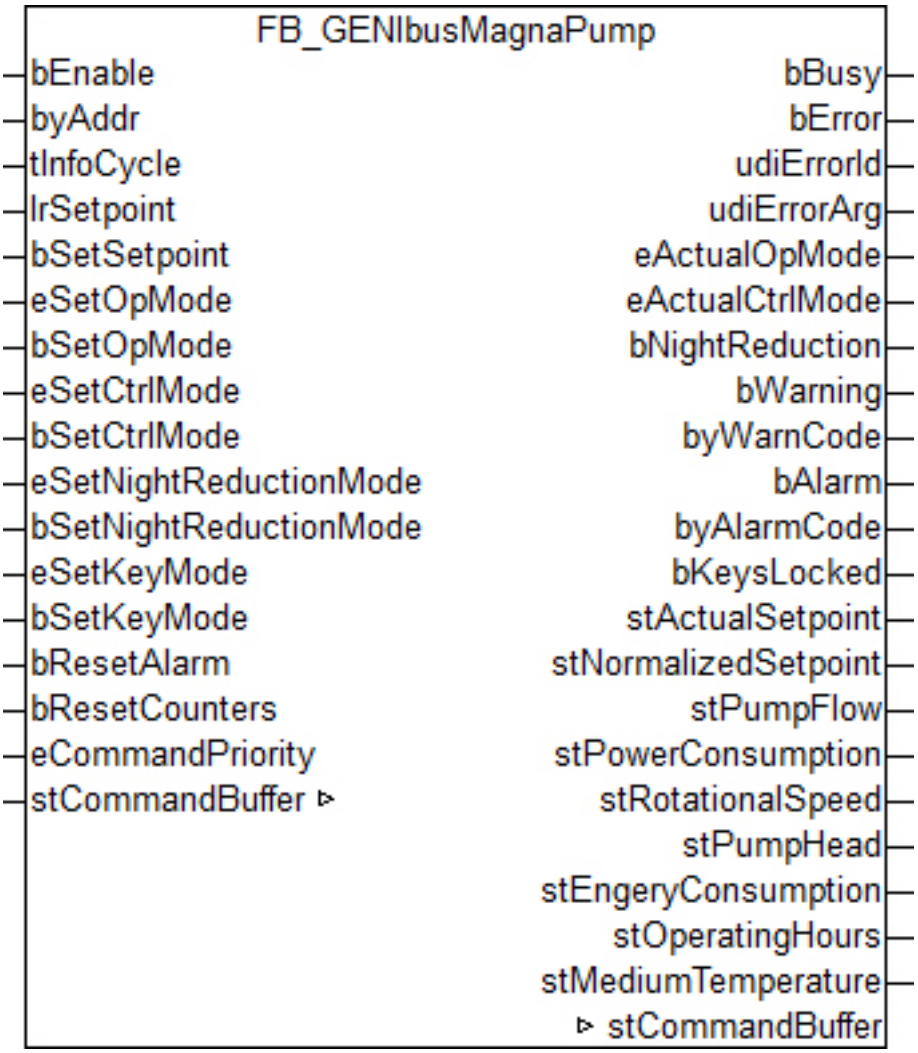
```
stCommandBuffer    : ST_GENIbusCommandBuffer;
```

stCommandBuffer: Verweis auf die Struktur [▶ 58] zur Kommunikation (Puffer) mit dem FB_GENIbusCommunication [▶ 45]()-Baustein.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	erforderliche Bibliotheken
TwinCAT v2.11 R3/x64 ab Build 2253	PC/CX	TcGENIbus-Bibliothek ab V1.0.0

5.2.6 FB_GENIbusMagnaPump



Dieser Baustein stellt eine universelle Applikation für eine Grundfos-Magna-Pumpe dar. Es können die grundlegenden Betriebsarten eingestellt und wesentliche Parameter ausgelesen werden.

Die im Folgenden fett und in eckigen Klammern dargestellten Werte stellen die Klasse und ID dar, mit denen die Kommandos ausgeführt bzw. die Informationen gewonnen werden. Diese Werte sind in der Grundfos-Dokumentation "Operating the MAGNA3 and MGE model H/I via the GENIpro interface - Edition 01.00.35 - April 2015" aufgeführt.

VAR_INPUT

```
bEnable                 : BOOL;
byAddr                  : BYTE := 0;
tInfoCycle              : TIME := t#5s;
lrSetpoint              : LREAL;
bSetSetpoint            : BOOL;
eSetOpMode              : E_GENIbusOpMode := eGENIbusOpModeStop;
bSetOpMode              : BOOL;
eSetCtrlMode            : E_GENIbusCtrlMode := eGENIbusCtrlModeConstFreq;
bSetCtrlMode            : BOOL;
eSetNightReductionMode  : E_GENIbusNightReductionMode := eGENIbusNightReductionModeOff;
bSetNightReductionMode  : BOOL;
eSetKeyMode             : E_GENIbusKeyMode;
bSetKeyMode             : BOOL;
bResetAlarm             : BOOL;
bResetCounters          : BOOL;
eCommandPriority        : E_GENIbusCommandPriority := eGENIbusCommandPriorityMiddle;
```

bEnable: Ist dieser Eingang gesetzt, wird der Baustein aktiviert.

byAdress : Adresse des anzusprechenden GENIbus-Gerätes: Gültige Eingaben: 1 - 200. Das entspricht der Einstellung, wie sie am GENIbus-Gerät direkt eingestellt wird. Eine Umrechnung auf den tatsächlichen Adressbereich von 32 - 231, siehe GENIbus-Standard, erfolgt Baustein-intern.

Möglich ist auch ein Broadcast- bzw. Sammelbefehl an mehrere Pumpen. Der Wert an diesem Eingang muss dann 255 sein. Für den Fall des Broadcast-Befehls sind die Werte-Abfragen deaktiviert.

tInfoCycle: Gibt an, in welchem Intervall die Wert-Abfrage-Befehle ausgegeben werden sollen. Dieser Eintrag ist nach unten hin auf 1s begrenzt. Der Eintrag "0s" hingegen ist zulässig und bedeutet, dass keine Abfrage statt findet.

lrSetpoint: Eingabe Sollwert **[5, 1]**. Die Eingabe ist in Prozent und bezieht sich je nach Regelungsart auf die vorgegebenen Grenzen. Eine genauere Beschreibung ist in der entsprechenden Dokumentation der Firma Grundfos gegeben.

bSetSetpoint: Eine steigende Flanke an diesem Eingang überträgt den eingestellten Sollwert.

eSetOpMode: Dieser Eingang dient der Anwahl eines der folgenden Betriebsmodi [▶ 58]:

- Stop **[3, 5]**
- Start **[3, 6]**
- Minimal-Kurve **[3, 25]**
- Maximal-Kurve **[3, 26]**

bSetOpMode: Eine steigende Flanke an diesem Eingang überträgt den eingestellten Betriebsmodus.

eSetCtrlMode: Dieser Eingang dient der Anwahl eines der folgenden Regelungsmodi [▶ 57]:

- Konstante Frequenz **[3, 22]**
- Proportionaler Druck **[3, 23]**
- Konstanter Druck **[3, 24]**
- Auto-adaptierend **[3, 52]**

bSetCtrlMode: Eine steigende Flanke an diesem Eingang überträgt den eingestellten Regelungsmodus [▶ 57].

eSetNightReductionMode: Dieser Eingang dient der An- bzw. Abwahl des Nachtmodus [▶ 58]. **[4, 170]**

bSetNightReductionMode: Eine steigende Flanke an diesem Eingang überträgt die eingestellte Anwahl [▶ 58].

eSetKeyMode: Mit Hilfe dieses Eingangs kann eine Verriegelung der Handbedienung [▶ 57] an der Pumpe angewählt werden. Die Verriegelung sperrt lediglich die Parametriermenüs, nicht die Tasten an sich. **[3, 30/31]**

bSetKeyMode: Eine steigende Flanke an diesem Eingang überträgt die eingestellte Anwahl [▶ 57].

bResetAlarm: Eine steigende Flanke an diesem Eingang setzt den aktuell anliegenden Alarm am Gerät zurück. **[3, 2]**

bResetCounters: Eine steigende Flanke an diesem Eingang setzt Zähler, wie beispielsweise Betriebsstunden oder Energie zurück. **[3, 36]**

eCommandPriority: Priorität [▶ 57] (hoch, mittel oder niedrig), mit der der Befehl von der SPS-Bibliothek abgearbeitet wird.

VAR_OUTPUT

```
bBusy                : BOOL;
bError               : BOOL;
udiErrorId           : UDINT;
udiErrorArg          : UDINT;
eActualOpMode        : E_GENIbusActOpMode;
eActualCtrlMode      : E_GENIbusActCtrlMode;
bNightReduction      : BOOL;
bWarning             : BOOL;
```

```
byWarnCode            : BYTE;
bAlarm                : BOOL;
byAlarmCode           : BYTE;
bKeysLocked           : BOOL;
stActualSetpoint      : ST_GENIbusMValue;
stNormalizedSetpoint  : ST_GENIbusMValue;
stPumpFlow            : ST_GENIbusMValue;
stPowerConsumption    : ST_GENIbusMValue;
stRotationalSpeed     : ST_GENIbusMValue;
stPumpHead            : ST_GENIbusMValue;
stEngeryConsumption   : ST_GENIbusMValue;
stOperatingHours      : ST_GENIbusMValue;
stMediumTemperature   : ST_GENIbusMValue;
```

bBusy: Dieser Ausgang ist immer dann TRUE, wenn ein Befehl oder eine Abfrage abgearbeitet wird.

bError: Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten.

udiErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Wird durch das erneute Aktivieren des Bausteins über den Eingang *bStart* wieder auf 0 zurückgesetzt. Siehe Fehlercodes [▶ 65].

udiErrorArg: Enthält ggf. eine erweiterte Beschreibung des Fehlercodes.

eActualOpMode: Aktuell gültiger Betriebsmodus [▶ 56]. **[2, 81]**

eActualCtrlMode: Aktuell gültiger Regelungsmodus [▶ 56]. **[2, 81]**

bNightReduction: Nachtabsenkung ist angewählt. **[2, 84]**

bWarning: Eine Warnmeldung liegt an.

byWarnCode: Code der aktuellen Warnmeldung. **[2, 156]**

bAlarm: Ein Alarm liegt an.

byAlarmCode: Code des aktuellen Alarmes. **[2, 158]**

bKeysLocked: Die Verriegelung der Handbedienung an der Pumpe ist aktiviert. **[4, 170]**

stActualSetpoint: Aktuell eingestellter Sollwert [▶ 62], die angezeigte Einheit richtet sich dabei nach dem Regelungsmodus. **[2, 48]**

stNormalizedSetpoint: Aktueller normierter Sollwert. **[2, 49]**

stPumpFlow: Durchflussmenge. **[2, 39]**

stPowerConsumption: Leistungsaufnahme. **[2, 34]**

stRotationalSpeed: Drehzahl. **[2, 35/36]**

stPumpHead: Förderhöhe. **[2, 37]**

stEngeryConsumption: Energieverbrauch. **[2, 152/153]**

stOperatingHours: Betriebsstundenzähler. **[2, 24/25]**

stMediumTemperature: Wasser- (medium-) Temperatur. **[2, 58]**

Da ein Fehler die Abarbeitung des Bausteines nicht unterbrechen darf, werden bError, udiErrorId und udiErrorArg in jedem SPS-Zyklus zunächst wieder zurückgesetzt und dann neu beurteilt. Zur Ermittlung von sporadisch auftretenden Fehlern muss daher Baustein-extern ein Fehlerspeicher programmiert werden.

VAR_IN_OUT

```
stCommandBuffer       : ST_GENIbusCommandBuffer;
```

stCommandBuffer: Verweis auf die Struktur [▶ 58] zur Kommunikation (Puffer) mit dem FB_GENIbusCommunication [▶ 45]()-Baustein.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	erforderliche Bibliotheken
TwinCAT v2.11 R3/x64 ab Build 2253	PC/CX	TcGENIbus-Bibliothek ab V1.0.0

5.3 Datentypen

5.3.1 E_GENIbusACK

ACK (Acknowledge-Code) aus dem Antworttelegramm.

```
TYPE E_GENIbusACK :
(
  eGENIbusACKOk           := 0,
  eGENIbusACKUnknownClass := 1,
  eGENIbusACKUnknownId    := 2,
  eGENIbusACKIllegalOp    := 3
);
END_TYPE
```

5.3.2 E_GENIbusActCtrlMode

Ausgelesener aktueller Regelungsmodus.

```
TYPE E_GENIbusActCtrlMode :
(
  eGENIbusActCtrlModeUnknown    := 0,
  eGENIbusActCtrlModeConstFreq  := 1,
  eGENIbusActCtrlModeConstPress := 2,
  eGENIbusActCtrlModePropPress  := 3,
  eGENIbusActCtrlModeAutoAdapt  := 4
);
END_TYPE
```

5.3.3 E_GENIbusActOpMode

Ausgelesener aktueller Betriebsmodus.

```
TYPE E_GENIbusActOpMode :
(
  eGENIbusActOpModeUnknown := 0,
  eGENIbusActOpModeStop    := 1,
  eGENIbusActOpModeStart   := 2,
  eGENIbusActOpModeMin     := 3,
  eGENIbusActOpModeMax     := 4,
  eGENIbusActOpModeHand    := 5,
  eGENIbusActOpUserDef     := 6
);
END_TYPE
```

5.3.4 E_GENIbusAddrType

Adressierungstyp.

```
TYPE E_GENIbusAddrType :
(
  eGENIbusAddrTypeSingle    := 0,
  eGENIbusAddrTypeMulti     := 1,
```

```
  eGENIbusAddrTypeBroadcast := 2
);
END_TYPE
```

5.3.5 E_GENIbusCommandPriority

Befehlspriorität.

```
TYPE E_GENIbusCommandPriority :
(
  eGENIbusCommandPriorityHigh   := 0,
  eGENIbusCommandPriorityMiddle := 1,
  eGENIbusCommandPriorityLow    := 2
);
END_TYPE
```

5.3.6 E_GENIbusComMode

Auswahl der seriellen Kommunikations-Schnittstelle.

```
TYPE E_GENIbusComMode :
(
  eGENIbusComMode_Unknown := 0,
  eGENIbusComMode_KL6_5B  := 1,
  eGENIbusComMode_KL6_22B := 2,
  eGENIbusComMode_EL6_22B := 3,
  eGENIbusComMode_PC_64B  := 4
);
END_TYPE
```

5.3.7 E_GENIbusCtrlMode

Einstellbare Regelungsmodi.

```
TYPE E_GENIbusCtrlMode :
(
  eGENIbusCtrlModeUnknown    := 0,
  eGENIbusCtrlModeConstFreq  := 1,
  eGENIbusCtrlModeConstPress := 2,
  eGENIbusCtrlModePropPress  := 3,
  eGENIbusCtrlModeAutoAdapt  := 4
);
END_TYPE
```

5.3.8 E_GENIbusKeyMode

Abschaltung der Parametriermöglichkeit an dem GENIbus-Gerät.

```
TYPE E_GENIbusKeyMode :
(
  eGENIbusKeyModeLocked   := 0,
  eGENIbusKeyModeUnlocked := 1
);
END_TYPE
```

5.3.9 E_GENIBusMDataSize

Bit-Größe des auszulesenden Wertes aus dem GENIbus-Gerät.

```
TYPE E_GENIBusMDataSize :
(
  eGENIbusMSize8Bit  := 0,
  eGENIbusMSize16Bit := 1,
  eGENIbusMSize24Bit := 2,
  eGENIbusMSize32Bit := 3
);
END_TYPE
```

5.3.10 E_GENIbusNightReductionMode

Modus Nachtabsenkung Ein/Aus.

```
TYPE E_GENIbusNightReductionMode :
(
  eGENIbusNightReductionModeOff := 0,
  eGENIbusNightReductionModeOn  := 1
);
END_TYPE
```

5.3.11 E_GENIbusOpMode

Einstellbare Steuerungsmodi.

```
TYPE E_GENIbusOpMode :
(
  eGENIbusOpModeUnknown := 0,
  eGENIbusOpModeStop    := 1,
  eGENIbusOpModeStart   := 2,
  eGENIbusOpModeMin     := 3,
  eGENIbusOpModeMax     := 4
);
END_TYPE
```

5.3.12 E_GENIbusOS

OS (Operation-Specifier) im Befehlstelegramm.

```
TYPE E_GENIbusOS :
(
  eGENIbusGET  := 0,
  eGENIbusSET  := 1,
  eGENIbusINFO := 2
);
END_TYPE
```

5.3.13 E_GENIbusSD

SD (Start-Delimiter) im Befehls- oder Antworttelegramm.

```
TYPE E_GENIbusSD :
(
  eGENIbusNull        := 16#0,
  eGENIbusDatareply   := 16#24,
  eGENIbusDatamessage := 16#26,
  eGENIbusDatarequest := 16#27
);
END_TYPE
```

5.3.14 E_GENIbusSIF

SIF (Scale Information Format) im Antworttelegramm.

```
TYPE E_GENIbusSIF :
(
eGENIbusNoScaleInfo     := 0,
  eGENIbusBitWiseScaled := 1,
  eGENIbusScaled816     := 2,
  eGENIbusScaledExt     := 3
);
END_TYPE
```

5.3.15 ST_GENIbusCommandBuffer

Globaler Kommandopuffer für Befehle und deren Antworten.

```
TYPE ST_GENIbusCommandBuffer :
STRUCT
  arrMessageQueue  : ARRAY[0..2] OF ST_GENIbusMessageQueue;
  stResponseTable  : ST_GENIbusResponseTable;
  udiMessageHandle : UDINT;
END_STRUCT
END_TYPE
```

arrMessageQueue: Eingangspuffer der Befehle [▶ 61]. Durch die Felddeklaration stehen dabei 3 verschiedene Puffer zur Auswahl: für hohe, mittlere und niedrige Priorität.

stResponseTable: Puffer [▶ 64] für die Befehlsantwort.

udiMessageHandle: Zeiger auf das aktuelle Pufferelement.

5.3.16 ST_GENIbusComRegisterData

Registeradresse und -inhalt zur Parametrierung von Klemmen.

```
TYPE ST_GENIbusComRegisterData :
STRUCT
  byRegister : BYTE;
  wValue     : WORD;
END_STRUCT
END_TYPE
```

byRegister: Registeradresse.

wValue: Registerinhalt.

5.3.17 ST_GENIbusEL6AMSAddress

Struktur zur Verknüpfung im Eingangs-Prozessabbild, sollte zur Kommunikation eine EL6xxx-Klemme verwendet werden.

```
TYPE ST_GENIbusEL6AMSAddress :
STRUCT
  arrNetId : ARRAY[0..5] OF USINT;
  uiPort   : UINT;
END_STRUCT
END_TYPE
```

5.3.18 ST_GENIbusEL6DeviceIn22B

Struktur zur Verknüpfung im Eingangs-Prozessabbild, muss zur Kommunikation einer EL6xxx-Klemme verwendet werden.

```
TYPE ST_GENIbusEL6DeviceIn22B :
STRUCT
  wStatus   : WORD;
  arrData   : ARRAY[0..21] OF BYTE;
  stAdsAddr : ST_GENIbusEL6AMSAddress;
  uiState   : UINT;
  bWcState  : BOOL;
END_STRUCT
END_TYPE
```

Sehen Sie dazu auch

- ST_GENIbusEL6AMSAddress [▶ 59]

5.3.19 ST_GENIbusEL6DeviceOut22B

Struktur zur Verknüpfung im Ausgangs-Prozessabbild, muss zur Kommunikation einer EL6xxx-Klemme verwendet werden.

```
TYPE ST_GENIbusEL6DeviceOut22B :
STRUCT
  wCtrl   : WORD;
  arrData : ARRAY[0..21] OF BYTE;
END_TYPE
```

5.3.20 ST_GENIbusInData

Struktur zur Verknüpfung des Eingangsabbildes der Prozessvariablen. Es stehen 4 verschiedene mögliche Strukturen zur Auswahl, von denen letztendlich nur eine zu verknüpfen ist:

```
TYPE ST_GENIbusInData :
STRUCT
  stKL6DeviceIn5B  : ST_GENIbusKL6DeviceIn5B;
  stKL6DeviceIn22B : ST_GENIbusKL6DeviceIn22B;
  stEL6DeviceIn22B : ST_GENIbusEL6DeviceIn22B;
  stPcComDeviceIn  : ST_GENIbusPcComDeviceIn64B;
END_STRUCT
END_TYPE
```

stKL6DeviceIn5B: Eingangsprozessabbild einer 5-Byte Datenklemme [▶ 60] mit Standard-Kommunikationsbus, z. B. KL6021.

stKL6DeviceIn22B: Eingangsprozessabbild einer 22-Byte Datenklemme [▶ 60] mit Standard-Kommunikationsbus, z. B. KL6041.

stEL6DeviceIn22B: Eingangsprozessabbild einer 22-Byte EtherCAT-Datenklemme [▶ 59], z. B. EL6021.

stPcComDeviceIn: Eingangsprozessabbild [▶ 63] einer seriellen PC-Schnittstelle.

5.3.21 ST_GENIbusKL6DeviceIn22B

Struktur zur Verknüpfung im Eingangs-Prozessabbild, muss zur Kommunikation einer KL6xxx-Klemme mit 22-Byte-Prozessabbild verwendet werden.

```
TYPE ST_GENIbusKL6DeviceIn22B :
STRUCT
  wStatus : WORD;
  arrData : ARRAY[0..21] OF BYTE;
END_TYPE
```

5.3.22 ST_GENIbusKL6DeviceIn5B

Struktur zur Verknüpfung im Eingangs-Prozessabbild, muss zur Kommunikation einer KL6xxx-Klemme mit 5-Byte-Prozessabbild verwendet werden.

```
TYPE ST_GENIbusKL6DeviceIn5B :
STRUCT
  byStatus : BYTE;
  arrData  : ARRAY[0..4] OF BYTE;
END_TYPE
```

5.3.23 ST_GENIbusKL6DeviceOut22B

Struktur zur Verknüpfung im Ausgangs-Prozessabbild, muss zur Kommunikation einer KL6xxx-Klemme mit 22-Byte-Prozessabbild verwendet werden.

```
TYPE ST_GENIbusKL6DeviceOut22B :
STRUCT
  wCtrl   : WORD;
  arrData : ARRAY[0..21] OF BYTE;
END_TYPE
```

5.3.24 ST_GENIbusKL6DeviceOut5B

Struktur zur Verknüpfung im Ausgangs-Prozessabbild, muss zur Kommunikation einer KL6xxx-Klemme mit 5-Byte-Prozessabbild verwendet werden.

```
TYPE ST_GENIbusKL6DeviceOut5B :
STRUCT
  byCtrl  : BYTE;
  arrData : ARRAY[0..4] OF BYTE;
END_TYPE
```

5.3.25 ST_GENIbusMessageQueue

Befehlspuffer.

```
TYPE ST_GENIbusMessageQueue :
STRUCT
  arrBuffer                   : ARRAY[1..GENIBUS_COMMAND_BUFFER_ENTRIES] OF ST_GENIbusMessageQueueI
tem;
  byBufferReadPointer         : BYTE;
  byBufferWritePointer        : BYTE;
  byBufferDemandCounter       : BYTE;
  byBufferMaximumDemandCounter : BYTE;
  uiBufferOverflowCounter     : UINT;
  bLockSemaphore              : BOOL;
END_STRUCT
END_TYPE
```

arrBuffer: Befehlspuffer.

byBufferReadPointer: Zeiger auf das aktuelle Pufferelement des Befehlsspeichers.

byBufferWritePointer: Zeiger auf das aktuelle Pufferelement des Empfangsspeichers.

byBufferDemandCounter: Aktuelle Pufferauslastung.

byBufferMaximumDemandCounter: maximale Pufferauslastung.

uiBufferOverflowCounter: Anzahl Pufferüberläufe.

bLockSemaphore: Schreibschutz während der Abarbeitung eines Befehls.

Sehen Sie dazu auch

- ST_GENIbusMessageQueueItem [▶ 61]

5.3.26 ST_GENIbusMessageQueueItem

Einzelnes Element im Befehlspuffer.

```
TYPE ST_GENIbusMessageQueueItem :
STRUCT
  byAddr           : BYTE;
  eAddrType        : E_GENIbusAddrType;
  eSD              : E_GENIbusSD;
  arrAPDUs         : ARRAY[1..GENIBUS_MAX_APDU_NUMBER] OF ST_GENIbusRequestClassEntry;
  byRFS            : BYTE;
  udiMessageHandle : UDINT;
END_STRUCT
END_TYPE
```

byAddr: Geräte-Zieladresse.

eAddrType: Einzel- Multi- oder Sammelbefehl [▶ 56].

eSD: Start-Delimiter [▶ 58] des Telegrammes.

arrAPDUs: Sammlung der zu übertragenen APDU [▶ 64]s (Application Program Data Unit).

byRFS: Noch nicht benutzt: "Request from Slave".

udiMessageHandle: Zeiger auf das aktuelle Pufferelement.

5.3.27 ST_GENIbusMValue

Struktur mit Inhalten eines gelesenen Gerätewertes, z.B. Durchfluss oder Drehzahl.

```
TYPE ST_GENIbusMValue :
STRUCT
  lrValue      : LREAL;
  lrPrefix     : LREAL;
  sUnit        : STRING(8);
  eDataSize    : E_GENIBusMDataSize;
  byValueH     : BYTE;
  byValueL1    : BYTE;
  byValueL2    : BYTE;
  byValueL3    : BYTE;
  byInfoHead   : BYTE;
  byUnitIndex  : BYTE;
  byZeroH      : BYTE;
  byRangeZeroL : BYTE;
END_STRUCT
END_TYPE
```

lrValue: Aus den Rohdaten ermittelter Endwert.

lrPrefix: Vorzeichen und Teilung (+/- und z.B. 0.1).

sUnit: Einheit.

eDataSize: Größe [▶ 57] des Messwertes (8, 16, 24 oder 32 Byte).

byValueH: High-Byte des Messwertes.

byValueL1: Low-Byte.

byValueL2: Low-Byte.

byValueL3: Low-Byte.

byInfoHead: Skalierungsinformation

byUnitIndex: Vorzeichen und Einheit - kodiert.

byZeroH: Nullpunkt bei normaler Bereichs-Nullpunkt-Skalierung ODER High-Byte Nullpunkt bei erweiterter Skalierung.

byRangeZeroL: Bereich bei normaler Bereichs-Nullpunkt-Skalierung ODER Low-Byte Nullpunkt bei erweiterter Skalierung.

5.3.28 ST_GENIbusOutData

Struktur zur Verknüpfung des Ausgangsabbildes der Prozessvariablen. Es stehen 4 verschiedene mögliche Strukturen zur Auswahl, von denen letztendlich nur eine zu verknüpfen ist:

```
TYPE ST_GENIbusOutData :
STRUCT
  stKL6DeviceOut5B  : ST_GENIbusKL6DeviceOut5B;
  stKL6DeviceOut22B : ST_GENIbusKL6DeviceOut22B;
  stEL6DeviceOut22B : ST_GENIbusEL6DeviceOut22B;
  stPcComDeviceOut  : ST_GENIbusPcComDeviceOut64B;
END_STRUCT
END_TYPE
```

stKL6DeviceOut5B: Ausgangsprozessabbild einer 5-Byte Datenklemme [▶ 61] mit Standard-Kommunikationsbus, z. B. KL6021.

stKL6DeviceOut22B: Ausgangsprozessabbild einer 22-Byte Datenklemme [▶ 60] mit Standard-Kommunikationsbus, z. B. KL6041.

stEL6DeviceOut22B: Ausgangsprozessabbild einer 22-Byte EtherCAT-Datenklemme [▶ 59], z. B. EL6021.

stPcComDeviceOut: Ausgangsprozessabbild [▶ 63] einer seriellen PC-Schnittstelle.

5.3.29 ST_GENIbusPcComDeviceIn64B

Struktur zur Verknüpfung im Eingangs-Prozessabbild, muss zur Kommunikation einer seriellen PC-Schnittstelle verwendet werden.

```
TYPE ST_GENIbusPcComDeviceIn64B :
STRUCT
  wStatus : WORD;
  arrData : ARRAY[0..63] OF BYTE;
END_TYPE
```

5.3.30 ST_GENIbusPcComDeviceOut64B

Struktur zur Verknüpfung im Eingangs-Prozessabbild, muss zur Kommunikation einer seriellen PC-Schnittstelle verwendet werden.

```
TYPE ST_GENIbusPcComDeviceOut64B :
STRUCT
  wCtrl   : WORD;
  arrData : ARRAY[0..63] OF BYTE;
END_TYPE
```

5.3.31 ST_GENIbusReplyClassEntry

Antwort-Struktur, welche zur Bearbeitung innerhalb der Bibliothek die Daten eines Antwort-APDUs enthält.

```
TYPE ST_GENIbusReplyClassEntry :
STRUCT
  byClass      : BYTE;
  eACK         : E_GENIbusACK;
  eOS          : E_GENIbusOS;
  iEntryCount  : INT;
  arrEntry     : ARRAY[0..GENIBUS_MAX_APDU_LENGTH] OF ST_GENIbusReplyDataEntry;
  sASCIIString : STRING(64);
END_STRUCT
END_TYPE
```

byClass: Datenklasse.

eACK: Quittierung [▶ 56] des GENIbus-Gerätes.

eOS: Operationsanzeiger [▶ 58] (GET/SET/INFO).

iEntryCount: Anzahl der verwendeten Datenpunkten (ID-Codes) innerhalb des APDUs.

arrEntry: Inhalte der Datenpunkte [▶ 63] (ID-Codes).

sASCIIString: String-Auswertung bei Datenklasse 7.

5.3.32 ST_GENIbusReplyDataEntry

Inhalt eines Elements eines Antwort-APDUs: Wert und Information.

```
TYPE ST_GENIbusReplyDataEntry :
STRUCT
  byValue      : BYTE;
  byInfoHead   : BYTE;
  byUnitIndex  : BYTE;
  byZeroH      : BYTE;
  byRangeZeroL : BYTE;
END_STRUCT
END_TYPE
```

byValue: Rohwert.

byInfoHead: Informations-Kopf, welche unter anderem die Skalierinformation enthält.

byUnitIndex: Vorzeichen und Einheitscode.

byZeroH: Nullpunkt bei normaler Bereichs-Nullpunkt-Skalierung ODER High-Byte Nullpunkt bei erweiterter Skalierung.

byRangeZeroL: Bereich bei normaler Bereichs-Nullpunkt-Skalierung ODER Low-Byte Nullpunkt bei erweiterter Skalierung.

5.3.33 ST_GENIbusRequestClassEntry

Befehls-, bzw. Abfrage-Struktur, welche zur Bearbeitung innerhalb der Bibliothek die Daten eines Anfrage-APDUs enthält.

```
TYPE ST_GENIbusRequestClassEntry :
STRUCT
  byClass      : BYTE;
  eOS          : E_GENIbusOS;
  byEntryCount : BYTE;
  arrEntry     : ARRAY[0..GENIBUS_MAX_APDU_LENGTH] OF ST_GENIbusRequestDataEntry;
END_STRUCT
END_TYPE
```

byClass: Datenklasse.

eOS: Operationsanzeiger [▶ 58] (GET/SET/INFO).

byEntryCount: Anzahl der verwendeten Datenpunkten (ID-Codes) innerhalb des APDUs.

arrEntry: Feld mit Adressen der Datenpunkte [▶ 64] (ID-Codes) und ggf. die zu schreibenden Werte.

5.3.34 ST_GENIbusRequestDataEntry

Adresse und ggf. zu schreibender Wert innerhalb eines Anfrage-APDUs.

```
TYPE ST_GENIbusRequestDataEntry :
STRUCT
  byIDCode : BYTE;
  byValue  : BYTE;
END_STRUCT
END_TYPE
```

byIDCode: Adresse.

byValue: Zu schreibender Wert.

5.3.35 ST_GENIbusResponseTable

Antwortpuffer.

```
TYPE ST_GENIbusResponseTable :
STRUCT
  arrResponseTableItem           : ARRAY[1..GENIBUS_COMMAND_BUFFER_ENTRIES] OF ST_GENIbusResponseTab
leItem;
  byResponseTableCounter         : BYTE;
  byResponseTableMaxCounter      : BYTE;
  uiResponseTableOverflowCounter : UINT;
  bLockSemaphore                 : BOOL;
END_STRUCT
END_TYPE
```

arrResponseTableItem: Antwort-Puffer [▶ 65].

byResponseTableCounter: Aktuelle Pufferauslastung.

byResponseTableMaxCounter: Maximale Pufferauslastung.

uiResponseTableOverflowCounter: Anzahl Pufferüberläufe.

bLockSemaphore: Schreibschutz während der Abarbeitung eines Befehls.

5.3.36 ST_GENIbusResponseTableItem

Einzelnes Element im Antwortpuffer.

```
TYPE ST_GENIbusResponseTableItem :
STRUCT
  byAddr           : BYTE;
  byLength         : BYTE;
  eSD              : E_GENIbusSD;
  arrAPDUs         : ARRAY[1..GENIBUS_MAX_APDU_NUMBER] OF ST_GENIbusReplyClassEntry;
  byRFS            : BYTE;
  udiMessageHandle : UDINT;
  udiErrorId       : UDINT;
END_STRUCT
END_TYPE
```

byAddr: Geräte-Zieladresse.

eAddrType: Einzel- Multi- oder Sammelbefehl.

eSD: Start-Delimiter [▶ 58] des Telegrammes.

arrAPDUs: Sammlung der zu übertragenen APDU [▶ 63]s (Application Program Data Unit).

byRFS: Noch nicht benutzt: "Request from Slave".

udiMessageHandle: Zeiger auf das aktuelle Pufferelement.

udiErrorId: Sollte ein Fehler im Baustein FB_GENIbusCommunication() [▶ 45] aufgetreten sein, so wird hier der entsprechende Fehlercode zur weiteren Auswertung hinterlegt.

5.3.37 ST_GENIbusSerComBuffer

Serieller Kommunikationspuffer - für Senden und Empfangen gleichermaßen.

```
TYPE ST_GENIbusSerComBuffer :
STRUCT
  arrBuffer    : ARRAY[0..GENIBUS_MAX_TELEGRAM_LENGTH] OF BYTE;
  uiDataLength : UINT;
  bBlocked     : BOOL;
END_STRUCT
END_TYPE
```

5.4 Fehlercodes

Wert (hex)	Wert (dez)	ErrArg	Beschreibung
0x0000	0	n/a	Kein Fehler.
0x8001	32769	n/a	interner Fehler: Es wird keine AMS-Net-ID ausgelesen. Möglicherweise ist das Prozessabbild nicht richtig verknüpft.
0x8002	32770	n/a	Fehlerhafter Eintrag bei der Baudrate.
0x8003	32771	Fehlernummer Sub-Baustein	interner Fehler beim Schreiben der Konfigurationsdaten. *udiErrorArg* enthält die Fehlernummer des Schreibbausteines FB_EcCoESdoWrite() der intern verwendeten Bibliothek TcEtherCAT.lib.
0x8004	32772	n/a	interner Fehler: Fehlerhafte Pointerzuweisung *pRegComIn*/*pRegComOut*. Einer der beiden Pointer verweist auf die Adresse 0.

Wert (hex)	Wert (dez)	ErrArg	Beschreibung
0x8005	32773	n/a	Timeout-Fehler bei der Registerkommunikation. Der Versuch, die Klemme auf die benötigten seriellen Parameter zu konfigurieren ist gescheitert. Ursache kann zum einen eine defekte Klemme aber auch eine fehlerhafte Kommunikationsverknüpfung oder eine K-Bus-Überlastung sein. Überprüfen Sie die Variablenverknüpfung mit dem Beispiel. Stellen Sie sicher, dass die Kommunikationsvariablen im Systemmanager alle der schnellen Task zugeordnet sind. Bei Verwendung von Analogklemmen erhöhen Sie die Zykluszeit der schnellen Task auf zunächst 5ms. Beim neu-Einlesen des PLC-Projektes denken Sie bitte daran wiederum zu prüfen, ob die Kommunikationsvariablen im Systemmanager alle der schnellen Task zugeordnet sind.
0x8019	32793	n/a	Ungültige Master-Adresse. Gültiger Bereich: 0 - 31.
0x8020	32800	Fehlernummer Sub-Baustein	Fehler beim Konfigurieren einer KL6xxx (5 Byte Daten). *udiErrorArg* enthält die Fehlernummer des internen Konfigurations-Bausteines.
0x8021	32801	Fehlernummer Sub-Baustein	Fehler beim Konfigurieren einer KL6xxx (22 Byte Daten). *udiErrorArg* enthält die Fehlernummer des internen Konfigurations-Bausteines.
0x8022	32802	Fehlernummer Sub-Baustein	Fehler beim Konfigurieren einer EL6xxx (22 Byte Daten). *udiErrorArg* enthält die Fehlernummer des internen Konfigurations-Bausteines.
0x8023	32803	1	Fehlerhafter Kommunikationstyp (Eingang *eGENIbusComMode*).
		2	Fehlerhafte Pointerzuweisung. Einer der beiden Adressen der gewählten Ein-/Ausgangsvariablen (*stGENIbusInData*/*stGENIbusOutData*) verweist auf die Adresse 0.
		3	Die Kommunikation über eine EtherCAT-Klemme ist angewählt. Dabei ist die Klemme EL6xxx nicht im "OP-State".
		4	Die Klemme EL6xxx erhält Fehlerhafte Daten. Das wird dadurch signalisiert, dass die Eingangsvariable "WC-State" auf 1 steht.
0x8024	32804	Fehlernummer Sub-Baustein	Fehler beim Erstellen des seriellen Telegrammes. *udiErrorArg* enthält die Fehlernummer des internen Bausteins.
0x8025	32805	Fehlernummer Sub-Baustein	Fehler bei der seriellen Datenübertragung. *udiErrorArg* enthält die Fehlernummer des internen Bausteins.
0x8026	32806	Fehlernummer Sub-Baustein	Fehler bei der Auswertung des seriellen Telegrammes. *udiErrorArg* enthält die Fehlernummer internen Bausteins.
0x8027	32807	n/a	Timeout-Fehler beim Senden-Empfangen-Zyklus.
0x8030	32816	n/a	Indexfehler beim Senden des Telegrammes.
0x8031	32817	n/a	Indexfehler beim Empfang des Telegrammes.
0x8032	32818	n/a	Falsche Datenlänge beim Empfang des Telegrammes.
0x8033	32819	n/a	Timeout-Fehler beim Empfang des Telegrammes.
0x8034	32820	n/a	Timeout-Fehler beim Senden des Telegrammes. Es wurde 100 Zyklen gewartet, dass, die Klemme die Daten sendet, was jedoch nicht erfolgt ist. Das Telegramm wurde verworfen.

Wert (hex)	Wert (dez)	ErrArg	Beschreibung
0x8040	32832	Fehlerhafter OS	Das Antworttelegramm enthält einen unbekannten "Operation-Specifier" (OS), siehe *GENIbus Protocol Specification.*
0x8041	32833	n/a	Fehler Telegrammlänge.
0x8042	32834	n/a	Fehler Telegramm-CRC-Prüfung.
0x8045	32837	maximale Anzahl APDUs	Fehler bei der Konvertierung in ein Telegramm: zu viele APDU-Einträge. *udiErrorArg* zeigt die maximal mögliche Anzahl der APDU-Einträge.
0x8049	32841	n/a	Ungültige Geräte-(Slave-) Adresse. Gültiger Bereich: 1 - 200.
0x8050	32848	n/a	Fehlerhafter Klasseneintrag *byClass.*
0x8051	32849	n/a	Fehlerhafter Eintrag *eCommandPriority.*
0x8052	32850	n/a	Fehlerhafter Eintrag *eSetOpMode.*
0x8053	32851	n/a	Fehlerhafter Eintrag *eSetCtrlMode.*
0x8054	32852	n/a	Fehlerhafter Eintrag *eSetNightReductionMode.*
0x8055	32853	n/a	Fehlerhafter Eintrag *eSetKeyMode.*
0x8056	32854	n/a	Kommandopuffer-Überlauf (*stCommandBuffer*): es sind noch nicht alle zuvor gesendeten Befehle abgearbeitet worden.
0x8057	32855	n/a	Timeout-Fehler (Laufzeitüberwachung) beim Antworttelegramm.
0x8058	32856	n/a	Das Antwort-Telegramm des GENIbus-Gerätes meldet "unbekannte Datenklasse / Data Class Unknown", siehe *GENIbus Protocol Specification*, Rückmeldeeintrag "*ACK*".
0x8059	32857	n/a	Das Antwort-Telegramm des GENIbus-Gerätes meldet "unbekannte Daten-ID / Data Item ID Unknown", siehe *GENIbus Protocol Specification*, Rückmeldeeintrag "*ACK*".
0x805A	32858	n/a	Das Antwort-Telegramm des GENIbus-Gerätes meldet "ungültiger Befehl oder Sendepuffer-Überlauf / Operation illegal or Data Class write buffer is full", siehe *GENIbus Protocol Specification*, Rückmeldeeintrag "*ACK*".
0x805B	32859	n/a	Unbekannter ACK-Eintrag im Antworttelegramm.
0x805C	32860	übertragene Fehlernummer	Der Baustein FB_GENIbusCommunication() hat bereits einen Fehler erkannt und diesen in die Antwortstruktur *stResponseTableItem* eingetragen. *udiErrorArg* enthält die Fehlernummer des Bausteines FB_GENIbusCommunication().
0x805D	32861	übertragene Fehlernummer	Es ist ein interner Fehler bei der Skalierung aufgetreten. *udiErrorArg* enthält die interne Fehlernummer.
0x8060	32864	n/a	Datengröße (*eDataSize*) ungültig.
0x8061	32865	n/a	Ungültiger Parameter Scale-Info im Telegramm (*eSIF*), siehe *GENIbus Protocol Specification*, Rückmeldeeintrag "SIF".
0x8062	32866	n/a	Ungültige Kombination von Datengröße und Scale-Info.
0x8063	32867	n/a	Keine Info-Daten verfügbar.
0x8064	32868	n/a	Der ausgelesene Unit-Index ist keiner Einheit zugewiesen, d.h. nicht in der internen Tabelle vorhanden.

6 Anhang

6.1 Support und Service

Beckhoff und seine weltweiten Partnerfirmen bieten einen umfassenden Support und Service, der eine schnelle und kompetente Unterstützung bei allen Fragen zu Beckhoff Produkten und Systemlösungen zur Verfügung stellt.

Beckhoff Niederlassungen und Vertretungen

Wenden Sie sich bitte an Ihre Beckhoff Niederlassung oder Ihre Vertretung für den lokalen Support und Service zu Beckhoff Produkten!

Die Adressen der weltweiten Beckhoff Niederlassungen und Vertretungen entnehmen Sie bitte unseren Internetseiten: https://www.beckhoff.de

Dort finden Sie auch weitere Dokumentationen zu Beckhoff Komponenten.

Beckhoff Support

Der Support bietet Ihnen einen umfangreichen technischen Support, der Sie nicht nur bei dem Einsatz einzelner Beckhoff Produkte, sondern auch bei weiteren umfassenden Dienstleistungen unterstützt:

- Support
- Planung, Programmierung und Inbetriebnahme komplexer Automatisierungssysteme
- umfangreiches Schulungsprogramm für Beckhoff Systemkomponenten

Hotline:	+49(0)5246 963 157
Fax:	+49(0)5246 963 9157
E-Mail:	support@beckhoff.com

Beckhoff Service

Das Beckhoff Service-Center unterstützt Sie rund um den After-Sales-Service:

- Vor-Ort-Service
- Reparaturservice
- Ersatzteilservice
- Hotline-Service

Hotline:	+49(0)5246 963 460
Fax:	+49(0)5246 963 479
E-Mail:	service@beckhoff.com

Beckhoff Firmenzentrale

Beckhoff Automation GmbH & Co. KG

Hülshorstweg 20
33415 Verl
Deutschland

Telefon:	+49(0)5246 963 0
Fax:	+49(0)5246 963 198
E-Mail:	info@beckhoff.com
Internet:	https://www.beckhoff.de

Mehr Informationen:
www.beckhoff.de/tx1200

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Deutschland
Telefon: +49 5246 9630
info@beckhoff.de
www.beckhoff.de