BECKHOFF New Automation Technology

Dokumentation | DE

BX9000

Busklemmen-Controller für Ethernet

07.02.2022 | Version: 2.4.0

Inhaltsverzeichnis

1 Vorwort

1.1 Hinweise zur Dokumentation

Zielgruppe

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 veröffentlichte 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® 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 Hinweise

In der vorliegenden Dokumentation werden die folgenden Hinweise verwendet.
Diese Hinweise sind aufmerksam zu lesen und unbedingt zu befolgen!

⚠ GEFAHR

Akute Verletzungsgefahr!

Wenn dieser Sicherheitshinweis nicht beachtet wird, besteht unmittelbare Gefahr für Leben und Gesundheit von Personen!

⚠ WARNUNG

Verletzungsgefahr!

Wenn dieser Sicherheitshinweis nicht beachtet wird, besteht Gefahr für Leben und Gesundheit von Personen!

⚠ VORSICHT

Schädigung von Personen!

Wenn dieser Sicherheitshinweis nicht beachtet wird, können Personen geschädigt werden!

HINWEIS

Schädigung von Umwelt/Geräten oder Datenverlust

Wenn dieser Hinweis nicht beachtet wird, können Umweltschäden, Gerätebeschädigungen oder Datenverlust entstehen.

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.

1.4 Ausgabestände der Dokumentation

Version	Änderungen
2.4.0	• Kapitel *Technische Daten - BX* aktualisiert • Kapitel *Hinweise zum ESD-Schutz* hinzugefügt • Kapitel *Entsorgung* hinzugefügt • Kapitel *Hinweise zur Informationssicherheit* hinzugefügt • Neue Titelseite
2.3.0	• Kapitel *Adresseinstellung über BootP-Server* aktualisiert • Struktur-Update
2.2.0	• Kapitel *Technische Daten* aktualisiert
2.1.0	• Download-Links aktualisiert • Gestaltung der Sicherheitshinweise an IEC 82079-1 angepasst.
2.0.0	• Migration • Strukturupdate
1.2.0	• Aktualisierung zur Firmware-Version 1.20, ab Hardware 3.5
1.1.7	• Aktualisierung zur Firmware-Version 1.17
1.1.5	• Aktualisierung zur Firmware-Version 1.15
1.1.2	• Aktualisierung zur Firmware-Version 1.12
1.0.8	• Aktualisierung zur Firmware-Version 1.08
1.0.0	• Erste Version

Firmware-Version des BX9000

Die aktuelle Firmware wird Ihnen beim Einschalten des BX-Controllers für ca. drei Sekunden angezeigt. Zum Update Ihrer Firmware auf dem BX9000 verwenden die den TwinCAT System Manager Firmware mit TWINCAT System Manager [▶ 205].

Firmware	Beschreibung
1.25	• SSB Reset-Problem bei kurzgeschlossener CAN-Leitung gelöst
1.24	• SSB fix wenn der CAN beim starten des BX nicht angeschlossen ist
1.22	• Large Model implementiert
1.20	• Firmware für BX-Controller ab Hardware-Version 3.5 • Umschaltung der COM 2 Schnittstelle zwischen RS232 und RS485 geändert • Im DHCP- und BootP-Modus wird eine Private IP Adresse generiert wenn der DHCP- oder BootP-Server nicht antwortet. **Achtung** Die Firmware-Version 1.20 läuft nicht auf älteren Hardware-Versionen (kleiner als 3.5). Die Hardware-Version Ihres BX-Controllers entnehmen Sie bitte dessen Aufkleber.
1.17	• Neu: Unterstützung der TwinSAFE-Busklemmen: maximal eine Logik-Klemme am K-Bus mit maximal 7 Verbindungen erlaubt (weitere Informationen) [▶ 138] • Neu: Zusätzlich zu den 2k Byte remanenten Daten 1000 Byte persistente Daten [▶ 97] eingebaut.
1.15	• Neu: Modbus Client [▶ 158] • Neu: TCP/IP und UDP/IP Bausteine implementiert [▶ 146] • Neu: HTML Page • Beleuchtung des Displays kann ein und ausgeschaltet werden • CRC des Programms kann beim Booten anzeigen werden
1.12	• New: SSB Sync support • New: ADS indication • Größere TwinCAT Files für die TwinCAT Config möglich • ADS Read Client 2 Byte Offset behoben • Neue Bausteine implementiert [▶ 111]
1.08	• Für diese Firmware Version ist TwinCAT 2.10 ab Build 1251 notwendig
1.00	• Erste Version

2 Produktübersicht

2.1 Busklemmen-Controller der BX-Serie

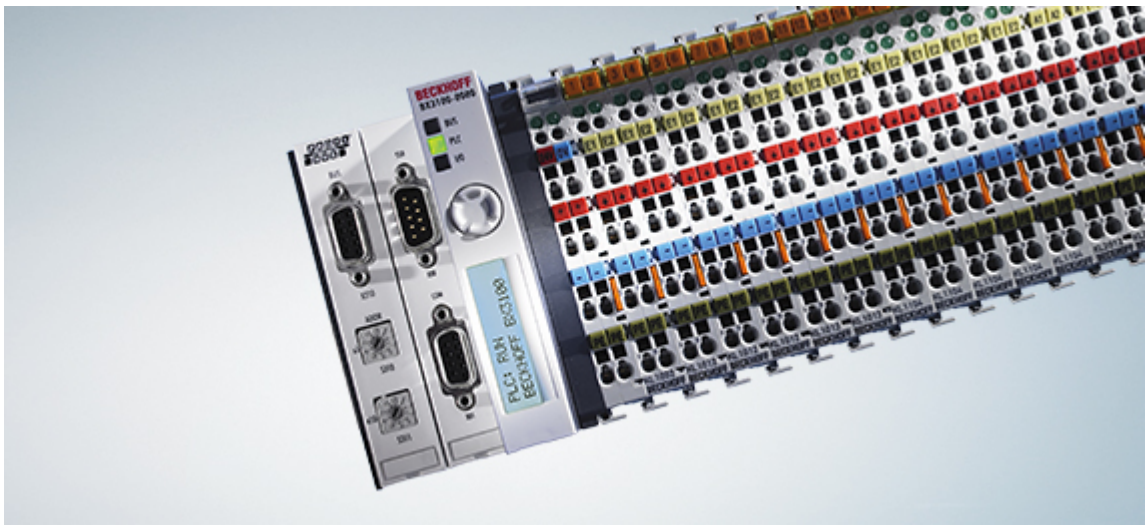


Abb. 1: Busklemmen-Controller der BX-Serie

Die Busklemmen-Controller der BX-Serie (BX-Controller) sind eine Steuerungsfamilie mit hohem Flexibilitätsgrad. Vom Ausstattungs- und Leistungsspektrum ist die BX-Serie zwischen den Busklemmen-Controller der BC-Serie und dem Embedded-PC CX1000 positioniert. Von der BC-Serie wurde das Konzept der autarken Steuerung in Kombination dem Anschluss für ein übergeordnetes Feldbussystem übernommen. Das Gehäusekonzept stammt vom CX1000. Die Hauptunterscheidungsmerkmale zwischen BC- und BX-Serie sind die größere Speicherausstattung und die erweiterten Schnittstellen der BX-Serie.

Die BX-Controller bestehen aus einem programmierbaren IEC 61131-3 Controller, einem Anschluss für das übergeordnete Feldbussystem und dem K-Bus-Interface zum Anschluss der Beckhoff Busklemmen. Zusätzlich verfügen die BX-Controller über zwei serielle Schnittstellen: eine für die Programmierung, die andere zur freien Nutzung. Im Gerät selbst enthalten sind ein beleuchtetes LC-Display (2 Zeilen mit je 16 Zeichen) mit Joystickschalter sowie eine Echtzeit-Uhr. Über den integrierten Beckhoff Smart System Bus (SSB) können weitere Peripheriegeräte, z. B. Displays, angeschlossen werden.

Die Busklemmen werden wie gewohnt auf der rechten Seite des BX-Controllers angesteckt. Durch das umfangreiche Spektrum an verschiedenen E/As kann jedes Eingangssignal gelesen und jedes benötigte Ausgangssignal erzeugt werden. Dadurch sind mit den BX-Controllern vielfältige Automatisierungsaufgaben lösbar, von der Garagentorsteuerung bis hin zur autarken Temperaturregelung an einer Spritzgussmaschine. Auch im Hinblick auf ein modulares Maschinenkonzept sind die BX-Controller hervorragend einzusetzen. Im Verbund kann der BX-Controller über die Feldbusschnittstelle mit anderen Maschinenteilen Daten austauschen. Die Echtzeit-Uhr ermöglicht auch einen dezentralen Einsatz, bei dem der Wochentag oder die Uhrzeit eine wichtige Rolle spielen.

Die Einsatzgebiete der BX-Serie sind denen der BC-Serie ähnlich, jedoch lassen sich mit dem BX, aufgrund der großen Speicherausstattung, wesentlich komplexere, größere Programme abarbeiten und lokal mehr Daten verwalten (z. B. Historie- und Trenddatenaufzeichnung), die dann sukzessive über den Feldbus abgeholt werden.

Busklemme und Endklemme erforderlich

Zum Betrieb eines BX-Controllers müssen an dessen K-Bus mindestens eine Busklemme mit Prozessabbild und die Endklemme gesteckt sein!

Feldbus-Interface

Die Varianten der Busklemmen-Controller der BX-Serie unterscheiden sich durch die unterschiedlichen Feldbusschnittstellen. Zusätzlich sind zwei serielle Schnittstellen für die Programmierung und für den Anschluss weiterer serieller Geräte integriert. Fünf verschiedene Ausführungen decken die wichtigsten Feldbussysteme ab:

- BX3100: PROFIBUS DP
- BX5100: CANopen
- BX5200: DeviceNet
- BX8000: RS232 oder RS485 (ohne Feldbusinterface)
- BX9000: Ethernet, ModbusTCP/ADS-TCP/UDP

Programmierung

Programmiert werden die BX-Controller nach der leistungsfähigen IEC 61131-3 Norm. Wie auch bei allen anderen Beckhoff Steuerungen ist die Automatisierungssoftware TwinCAT Grundlage für die Parametrierung und Programmierung. Dem Anwender stehen also die gewohnten TwinCAT Werkzeuge, wie z. B. SPS-Programmieroberfläche, System Manager und TwinCAT Scope zur Verfügung. Der Datenaustausch erfolgt wahlweise über die serielle Schnittstelle (COM1) oder über den Feldbus via Beckhoff PC-Feldbuskarten FCxxxx.

Konfiguration

Die Konfiguration erfolgt ebenfalls mit TwinCAT. Über den System Manager können das Feldbusinterface, der SSB-Bus und die Echtzeit-Uhr konfiguriert und parametriert werden. Alle angeschlossenen Geräte und Busklemmen können vom System Manager ausgelesen werden. Die Konfiguration wird nach der Parametrierung über die serielle Schnittstelle auf den BX gespeichert. Diese erstellte Konfiguration kann auch wieder ausgelesen werden.

2.2 BX9000 - Einführung

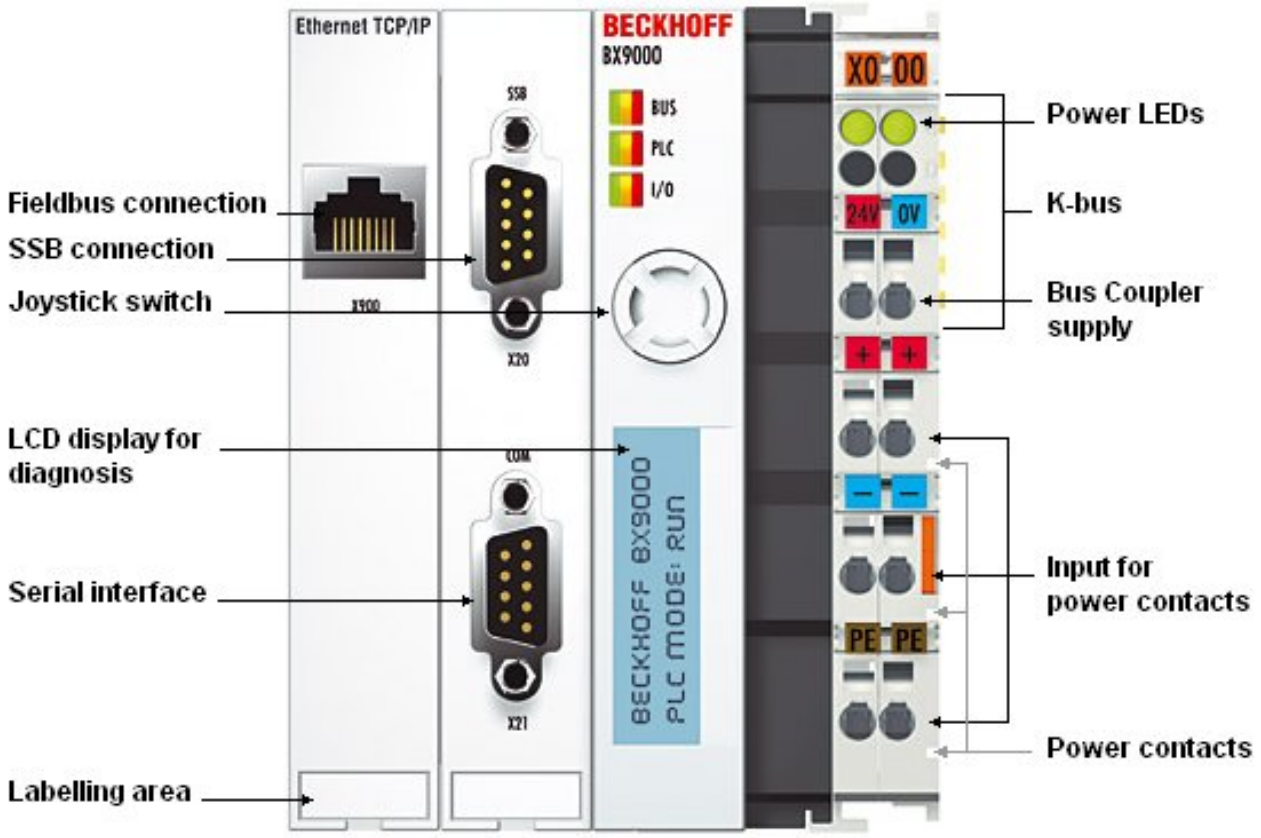


Abb. 2: BX9000 - Ethernet ModbusTCP/ADS-TCP/UDP

Der Busklemmen Controller BX9000 besitzt eine Ethernet-Slave-/Masterschnittstelle und verfügt über eine automatische Baudratenerkennung bis 100 MBaud. Die Adresse ist wahlweise über DHCP-, BootP-, ARP-Eintrag oder Joystickschalter einstellbar. Es können bis zu 2-kByte-Eingänge und 2-kByte-Ausgänge mit der Steuerung ausgetauscht werden. Implementiert sind das ModbusTCP-, das ADS/TCP- und das ADS/UDP-Protokoll.

2.3 Technische Daten

2.3.1 Technische Daten - BX

Technische Daten	BX3100	BX5100	BX5200	BX8000	BX9000
Prozessor	16 Bit Mikrocontroller				
Diagnose LEDs	2 x Spannungsversorgung, 2 x K-Bus				
Display	FSTN 2 x 16 Zeilen Display für Diagnose oder eigene Texte, beleuchtet				
Schalter	Joystickschalter für Parametrierung und Diagnose				
Uhr	interne akkugepufferte Uhr für Zeit und Datum				
Konfigurations- und Programmiersoftware	TwinCAT PLC				
Feldbus-Interface	PROFIBUS-DP	CANopen	DeviceNet	-	Ethernet
Feldbus-Anschluss	D-Sub, 9-pin	Open Style Connector, 5-pin		-	RJ45
Serielle Schnittstellen	COM1 (RS232 für Konfiguration und Programmierung, automatische Baudratenerkennung 9600/19200/38400/57600 Baud) COM2 (RS232 oder RS485) für den Anschluss serieller Geräte				
SSB	CANopen basierende Sub-Bus Schnittstelle				
Klemmenbus (K-Bus)	64 (255 mit K-Bus-Verlängerung)				
Digitale Peripheriesignale	2040 Ein-/Ausgänge				
Analoge Peripheriesignale	1024 Ein-/Ausgänge				
Konfigurationsmöglichkeit	über TwinCAT oder die Steuerung				
maximale Byte-Anzahl Feldbus	feldbusabhängig				
maximale Byte-Anzahl SPS	2048 Byte Eingangsdaten, 2048 Byte Ausgangsdaten				

Versorgung	BX3100	BX5100	BX5200	BX8000	BX9000
Spannungsversorgung (Us)	24 V_{DC} (-15%/+20%), siehe UL-Anforderungen				
Eingangsstrom (Us)	180 mA + (ges. K-Bus Strom)/4				
Einschaltstrom (Us)	ca. 2,5 x Dauerstrom				
K-Bus-Strom (5 V)	maximal 1450 mA				
Spannung Powerkontakt (Up)	maximal 24 V_{DC}				
Stromlast Powerkontakt (Up)	maximal 10 A, siehe UL-Anforderungen				
Spannungsfestigkeit	500 V (Powerkontakt/Versorgungsspannung/Feldbus)				

⚠ VORSICHT

UL-Anforderungen

Für die Spannungsversorgungen des BX-Controllers (Us) und der Powerkontakte (Up) benutzen Sie eine 4 A Sicherung oder eine Spannungsversorgung, die *NEC Class 2* entspricht um die UL-Anforderungen zu erfüllen!

Technische Daten	BX3100	BX5100	BX5200	BX8000	BX9000
zulässiger Umgebungstemperaturbereich im Betrieb	0°C ... +55°C (vor Hardware-Version 4.4) -25°C ... +60°C (ab Hardware-Version 4.4)				0°C ... +55°C
zulässiger Umgebungstemperaturbereich bei Lagerung	-20°C ... +85°C (vor Hardware-Version 4.4) -40°C ... +85°C (ab Hardware-Version 4.4)				-20°C ... +85°C
Relative Feuchte	95% ohne Betauung				
Vibrations-/ Schockfestigkeit	gemäß EN 60068-2-6 / EN 60068-2-27				
EMV-Festigkeit / Aussendung	gemäß EN 61000-6-2 / EN 61000-6-4				
Schutzart	IP20				
Zulassungen/Kennzeichnungen*	CE, UKCA, cULus, EAC				

*) Real zutreffende Zulassungen/Kennzeichnungen siehe seitliches Typenschild (Produktbeschriftung).

Mechanische Daten	BX3100	BX5100	BX5200	BX8000	BX9000
Gewicht	ca. 170 g				
Abmessungen (B x H x T)	ca. 83 mm x 100 mm x 90 mm (BX8000: ca. 65 mm x 100 mm x 90 mm)				
Montage	mit Verriegelung, auf Tragschiene (35 mm Hutschiene)				
Einbaulage	Beliebig				
Anschlussquerschnitt	0,08 mm² ... 2,5 mm² AWG 28 ... 14 8 ... 9 mm Abisolierlänge				

2.3.2 Technische Daten - Ethernet

Systemdaten	Ethernet (BX9000)
Anzahl der E/A-Module	steuerungsabhängig
Anzahl der E/A-Punkte	steuerungsabhängig
Übertragungsmedium	4 x 2 Twisted-Pair-Kupferkabel Kategorie 5 (100 MBaud)
Leitungslänge	100 m vom Switch bis BX9000
Übertragungsrate	10/100 MBaud
Topologie	sternförmige Verkabelung

2.3.3 Technische Daten - SSB-Interface

Systemdaten	SSB-Interface
Max. Anzahl an Slaves	8
Max. Anzahl an PDOs	32 RxPODs / 32 TxPDOs
Baudrate	10 k ... 1 MBaud
Erlaubte Slave-Adressen	1 bis 64

2.3.4 Technische Daten - SPS

SPS- Daten	BX3100	BX5100	BX5200	BX8000	BX9000
Programmiermöglichkeit	über Programmierschnittstelle (COM1 oder COM2) oder über Feldbus				
Programmspeicher	256 kByte				
Source Code Speicher	256 kByte				
Datenspeicher	256 kByte				
Remanente Merker	2 kByte				
SPS-Zykluszeit	ca. 0,85 ms für 1000 AWL Befehle (ohne E/A Zyklus)				
Programmiersprachen	IEC 6-1131-3 (AWL, KOP, FUP, ST, AS)				
Laufzeit	1 SPS Task (2. Task in Vorbereitung)				
Online Change	Ja				
Up/Down Load Code	Ja/Ja				

2.4 Prinzip der Busklemme

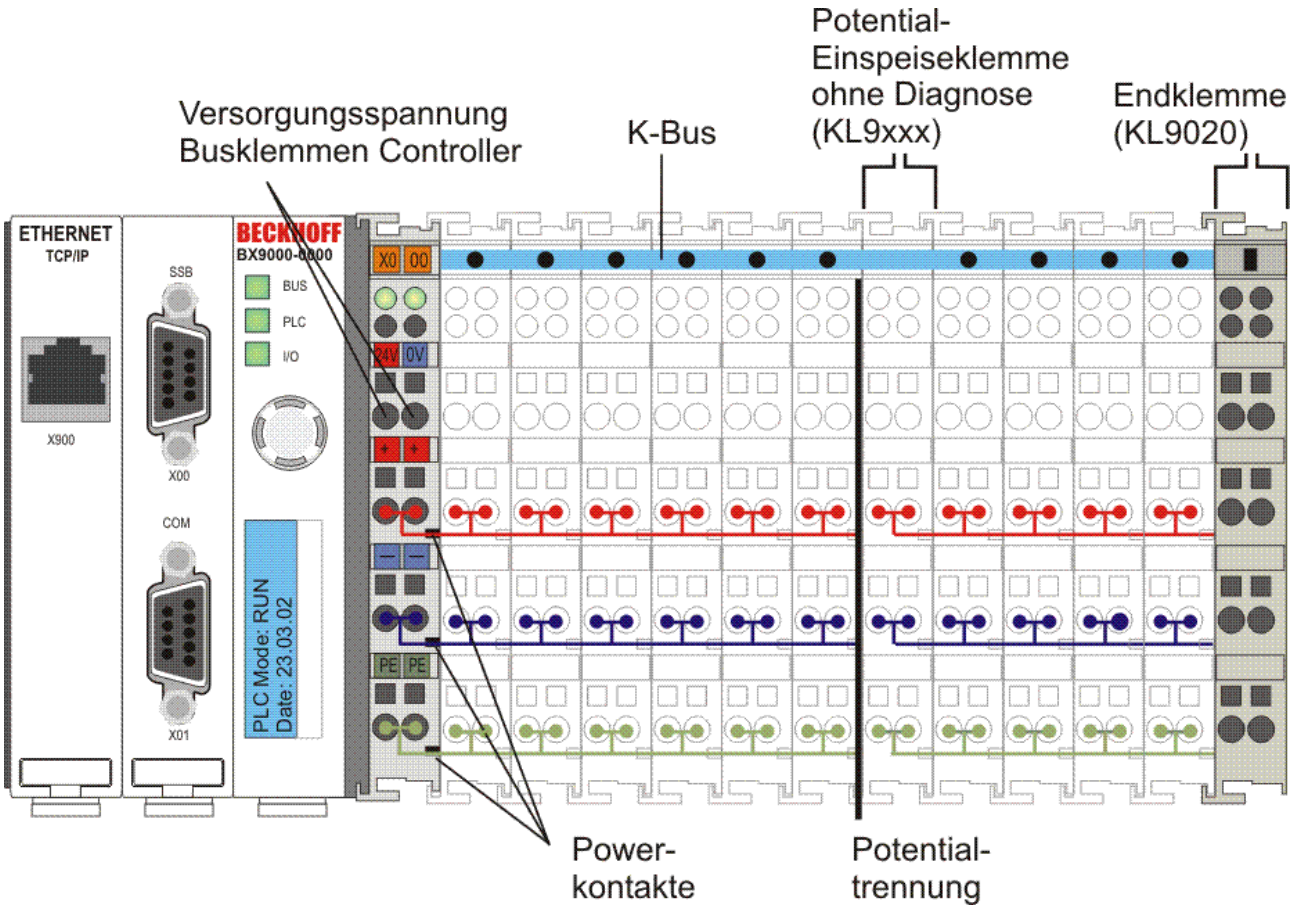


Abb. 3: Prinzip der Busklemme

2.5 Das Beckhoff Busklemmensystem

Bis zu 256 Busklemmen mit ein bis 16 E/A-Kanälen für jede Signalform

Das Busklemmen-System ist das universelle Bindeglied zwischen einem Feldbus-System und der Sensor / Aktuator - Ebene. Eine Einheit besteht aus einem Buskoppler als Kopfstation und bis zu 64 elektronischen Reihenklemmen, wovon die letzte eine Endklemme ist. Mit der K-Bus Erweiterung können bis zu 255 Busklemmen angeschlossen werden. Für jede technische Signalform stehen Klemmen mit ein, zwei, vier oder acht E/A-Kanälen zur Verfügung, die beliebig gemischt werden können. Dabei haben alle Klemmentypen die gleiche Bauform, wodurch der Projektierungsaufwand sehr gering gehalten wird. Bauhöhe und Tiefe sind auf kompakte Klemmenkästen abgestimmt.

Dezentrale Verdrahtung der E/A-Ebene

Die Feldbustechnik erlaubt den Einsatz kompakter Steuerungsbauformen. Die E/A-Ebene muss nicht bis zur Steuerung geführt werden. Die Verdrahtung der Sensoren und Aktuatoren ist dezentral mit minimalen Kabellängen durchführbar. Der Installationsstandort der Steuerung kann im Bereich der Anlage beliebig gewählt werden.

Industrie-PCs als Steuerung

Durch den Einsatz eines Industrie-PCs als Steuerung lässt sich das Bedien- und Beobachtungselement in der Hardware der Steuerung realisieren. Der Standort der Steuerung kann deshalb ein Bedienpult, eine Leitwarte oder ähnliches sein. Die Busklemmen stellen die dezentrale Ein-/Ausgabeebene der Steuerung im Schaltschrank und untergeordneten Klemmenkästen dar. Neben der Sensor/Aktuator-Ebene wird auch der Leistungsteil der Anlage über das Bussystem gesteuert. Die Busklemme ersetzt die konventionelle Reihenklemme als Verdrahtungsebene im Schaltschrank. Der Schaltschrank kann kleiner dimensioniert werden.

Buskoppler für alle gängigen Bussysteme

Das Beckhoff Busklemmen-System vereint die Vorteile eines Bussystems mit den Möglichkeiten der kompakten Reihenklemme. Busklemmen können an allen gängigen Bussystemen betrieben werden und verringern so die Teilevielfalt in der Steuerung. Dabei verhalten sich Busklemmen wie herkömmliche Anschaltungen dieses Bussystems. Alle Leistungsmerkmale des jeweiligen Bussystems werden unterstützt.

Montage auf genormten Tragschienen

Die einfache und platzsparende Montage auf einer genormten Tragschiene (EN 60715, 35 mm) und die direkte Verdrahtung von Aktoren und Sensoren ohne Querverbindungen zwischen den Klemmen standardisiert die Installation. Dazu trägt auch das einheitliche Beschriftungskonzept bei.

Die geringe Baugröße und die große Flexibilität des Busklemmen-Systems ermöglichen den Einsatz überall dort, wo auch eine Reihenklemme zur Anwendung kommt. Jede Art von Ankopplung, wie analoge, digitale, serielle oder der Direktanschluss von Sensoren kann realisiert werden.

Modularität

Die modulare Zusammenstellung der Klemmleiste mit Busklemmen verschiedener Funktionen begrenzt die Zahl der ungenutzten Kanäle auf maximal einen pro Funktion. Die Anzahl von zwei Kanälen in einer Klemme trifft das Optimum zwischen der Zahl der ungenutzten Kanäle und den Kosten pro Kanal. Auch die Möglichkeit der Potentialtrennung durch Einspeiseklemmen hilft, die Anzahl der ungenutzten Kanäle gering zu halten.

Anzeige des Kanalzustands

Die integrierten Leuchtdioden zeigen in Sensor/Aktuator-Nähe den Zustand des entsprechenden Kanals an.

K-Bus

Der K-Bus ist der Datenweg innerhalb der Klemmleiste. Über sechs Kontakte an den Seitenwänden der Klemmen wird der K-Bus vom Buskoppler durch alle Klemmen geführt. Die Endklemme schließt den K-Bus ab. Der Benutzer muss sich keinerlei Wissen über die Funktion des K-Bus oder die interne Arbeitsweise von Klemmen und Buskoppler aneignen. Viele lieferbare Software-Tools erlauben eine komfortable Projektierung, Konfiguration und Bedienung.

Potential-Einspeiseklemmen für potentialgetrennte Gruppen

Über drei Powerkontakte wird die Betriebsspannung an die nachfolgenden Klemmen weitergegeben. Durch den Einsatz von Potential-Einspeiseklemmen, können Sie die Klemmleiste in beliebige potentialgetrennte Gruppen gliedern. Die Potential-Einspeiseklemmen werden bei der Ansteuerung der Klemmen nicht berücksichtigt, sie dürfen an beliebiger Stelle in die Klemmleiste eingereiht werden.

In einem Klemmenblock können Sie bis zu 64 Busklemmen einsetzen und diesen über die K-Busverlängerung auf bis zu 256 Busklemmen erweitern. Dabei werden Potential-Einspeiseklemmen mitgezählt, die Endklemme nicht.

Buskoppler für verschiedene Feldbus-Systeme

Verschiedene Buskoppler lassen sich einsetzen, um die elektronische Klemmleiste schnell und einfach an unterschiedliche Feldbus-Systeme anzukoppeln. Auch eine nachträgliche Umrüstung auf ein anderes Feldbus-System ist möglich. Der Buskoppler übernimmt alle Kontroll- und Steuerungsaufgaben, die für den Betrieb der angeschlossenen Busklemmen notwendig sind. Die Bedienung und Konfiguration der Busklemmen wird ausschließlich über den Buskoppler durchgeführt. Die eingestellten Parameter werden jedoch spannungsausfallsicher in den jeweiligen Busklemmen gespeichert. Feldbus, K-Bus und E/A-Ebene sind galvanisch getrennt.

Wenn der Datenaustausch über den Feldbus zeitweise gestört ist oder ausfällt, bleiben Registerinhalte (wie z. B. Zählerstände) erhalten, digitale Ausgänge werden gelöscht und analoge Ausgänge nehmen einen Wert an, der bei der Inbetriebnahme für jeden Ausgang konfigurierbar ist. Die Default-Einstellung der analogen Ausgänge ist 0 V bzw. 0 mA. Digitale Ausgänge fallen in einen inaktiven Zustand zurück. Die Timeout-Zeiten der Buskoppler entsprechen den für das Feldbus-System üblichen Zeiten. Bei der Umstellung auf ein anderes Bussystem beachten Sie im Falle großer Zykluszeiten des Bussystems die Änderung der Timeout-Zeiten.

Die Schnittstellen

Ein Buskoppler besitzt sechs unterschiedliche Anschlussmöglichkeiten. Diese Schnittstellen sind als Steckverbindungen und Federkraftklemmen ausgelegt.

3 Montage und Verdrahtung

3.1 Hinweise zum ESD-Schutz

HINWEIS

Zerstörung der Geräte durch elektrostatische Aufladung möglich!

Die Geräte enthalten elektrostatisch gefährdete Bauelemente, die durch unsachgemäße Behandlung beschädigt werden können.

- Sie müssen beim Umgang mit den Komponenten elektrostatisch entladen sein; vermeiden Sie außerdem die Federkontakte (s. Abb.) direkt zu berühren.
- Vermeiden Sie den Kontakt mit hoch isolierenden Stoffen (Kunstfaser, Kunststofffolien etc.)
- Beim Umgang mit den Komponenten ist auf gute Erdung der Umgebung zu achten (Arbeitsplatz, Verpackung und Personen)
- Jede Busstation muss auf der rechten Seite mit der Endklemme KL9010 abgeschlossen werden, um Schutzart und ESD-Schutz sicher zu stellen.

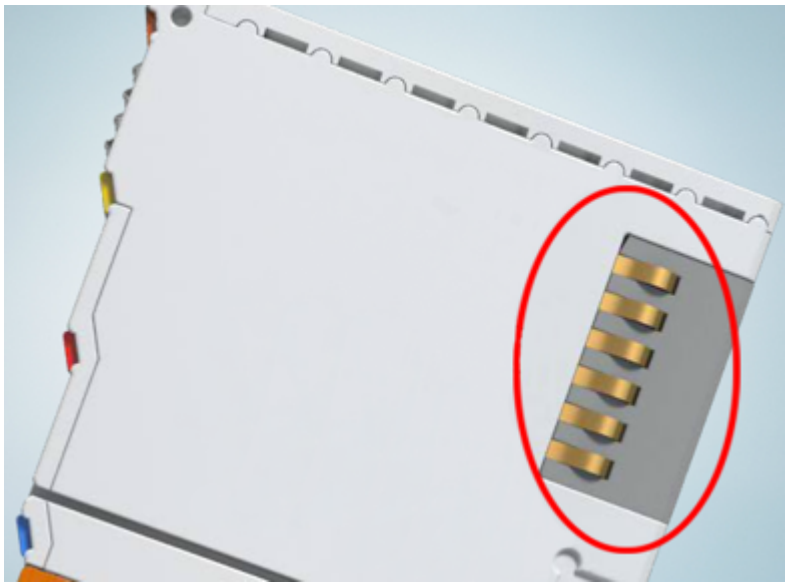


Abb. 4: Federkontakte der Beckhoff I/O-Komponenten

3.2 Montage

⚠ WARNUNG

Verletzungsgefahr durch Stromschlag und Beschädigung des Gerätes möglich!

Setzen Sie das Busklemmen-System in einen sicheren, spannungslosen Zustand,
bevor Sie mit der Montage, Demontage oder Verdrahtung der Komponenten beginnen!

3.2.1 Abmessungen

Das Beckhoff Busklemmen-System zeichnet sich durch geringes Bauvolumen und hohe Modularität aus. Für die Projektierung muss ein Buskoppler und eine Anzahl von Busklemmen vorgesehen werden. Die Abmessungen der Busklemmen-Controller sind unabhängig vom Feldbus-System.

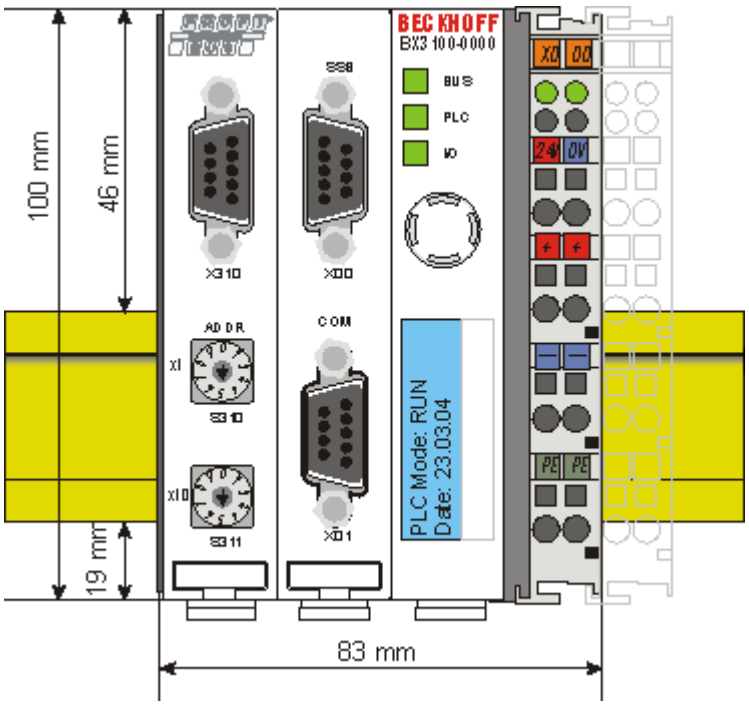


Abb. 5: BX3100, BX5100, BX5200, BX9000

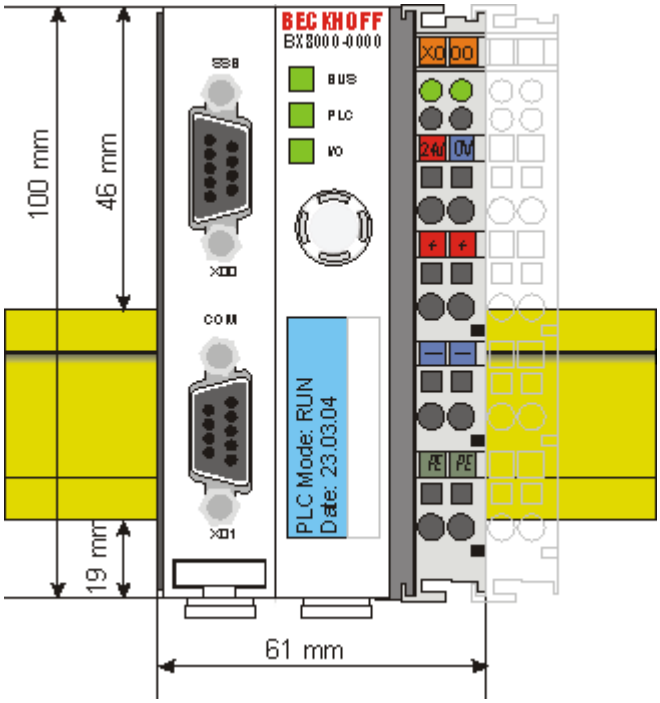


Abb. 6: BX8000

Die Gesamtbreite der Feldbusstation setzt sich aus der Breite des Busklemmen-Controllers und der Breite der verwendeten Busklemmen (incl. Busendklemme KL9010) zusammen. Die Busklemmen sind je nach Bauform 12 mm oder 24 mm breit. Die Die Höhe beträgt 100 mm.

Die Busklemmen-Controller der BX-Serie sind bis zu 83 mm breit und 91 mm tief.

Gesamttiefe beachten

Beachten sie, dass ein Busklemmen-Controller mit Hutschiene und angeschlossenen Steckern meist tiefer aufbaut als die angegebenen 91 mm. Beispiel:
BX3100 + ZB3100 + Hutschiene = 105 mm

3.2.2 Tragschienenmontage

Montage

1. Auf der Unterseite der BX-Controller befinden sich weiße Zuglaschen, die mit einem Rastmechanismus verbunden sind.
   Ziehen Sie diese Zuglaschen nach unten, bevor Sie den BX-Controller auf die Tragschiene drücken.

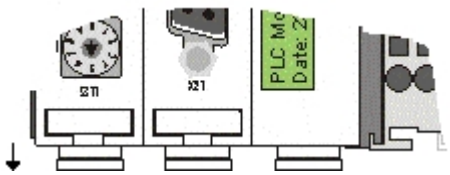


Abb. 7: Entriegelter BX-Controller

HINWEIS

Display bei der Montage nicht beschädigen!

Achten Sie darauf, nicht auf das Display zu drücken, wenn Sie den BX-Controller auf die Tragschiene drücken. Andernfalls kann das Display beschädigt werden.

2. Drücken Sie den BX-Controller nun auf die Tragschiene.
3. Nach dem Aufrasten auf die Tragschiene, schieben Sie die Zuglaschen wieder in die Ausgangsstellung zurück.

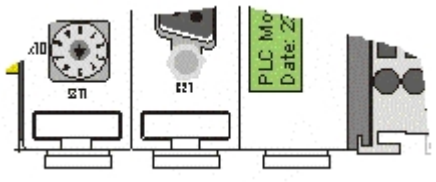


Abb. 8: Verriegelter BX-Controller

Demontage

1. Entriegeln Sie zuerst alle Zuglaschen an der Unterseite des BX-Controllers.
2. Ziehen Sie dann an der orangen Lasche neben der Einspeisung für die Powerkontakte.

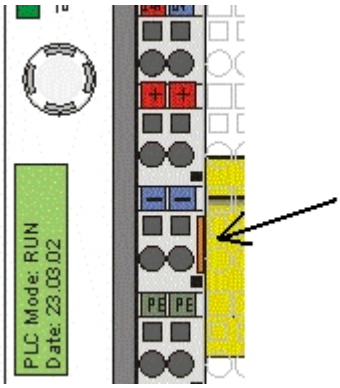


Abb. 9: Demontage

3.3 Entsorgung

Mit einer durchgestrichenen Abfalltonne gekennzeichnete Produkte dürfen nicht in den Hausmüll. Das Gerät gilt bei der Entsorgung als Elektro- und Elektronik-Altgerät. Die nationalen Vorgaben zur Entsorgung von Elektro- und Elektronik-Altgeräten sind zu beachten.

3.4 Verdrahtung

⚠ WARNUNG

Verletzungsgefahr durch Stromschlag und Beschädigung des Gerätes möglich!

Setzen Sie das Busklemmen-System in einen sicheren, spannungslosen Zustand, bevor Sie mit der Montage, Demontage oder Verdrahtung der Komponenten beginnen!

3.4.1 Potentialgruppen, Isolationsprüfung und PE

Potentialgruppen

Ein Beckhoff Busklemmenblock verfügen in der Regel über drei verschiedene Potentialgruppen:

- Die Feldbusschnittstelle ist (außer bei einzelnen Low Cost Kopplern) galvanisch getrennt und bildet die erste Potentialgruppe.
- Buskoppler- / Busklemmen-Controller-Logik, K-Bus und Klemmenlogik bilden eine zweite galvanisch getrennte Potentialgruppe.
- Die Ein- und Ausgänge werden über die Powerkontakte gespeist und bilden weitere Potentialgruppen.

Gruppen von E/A-Klemmen lassen sich durch Potentialeinspeiseklemmen oder Trennklemmen zu weiteren Potentialgruppen zusammenfassen.

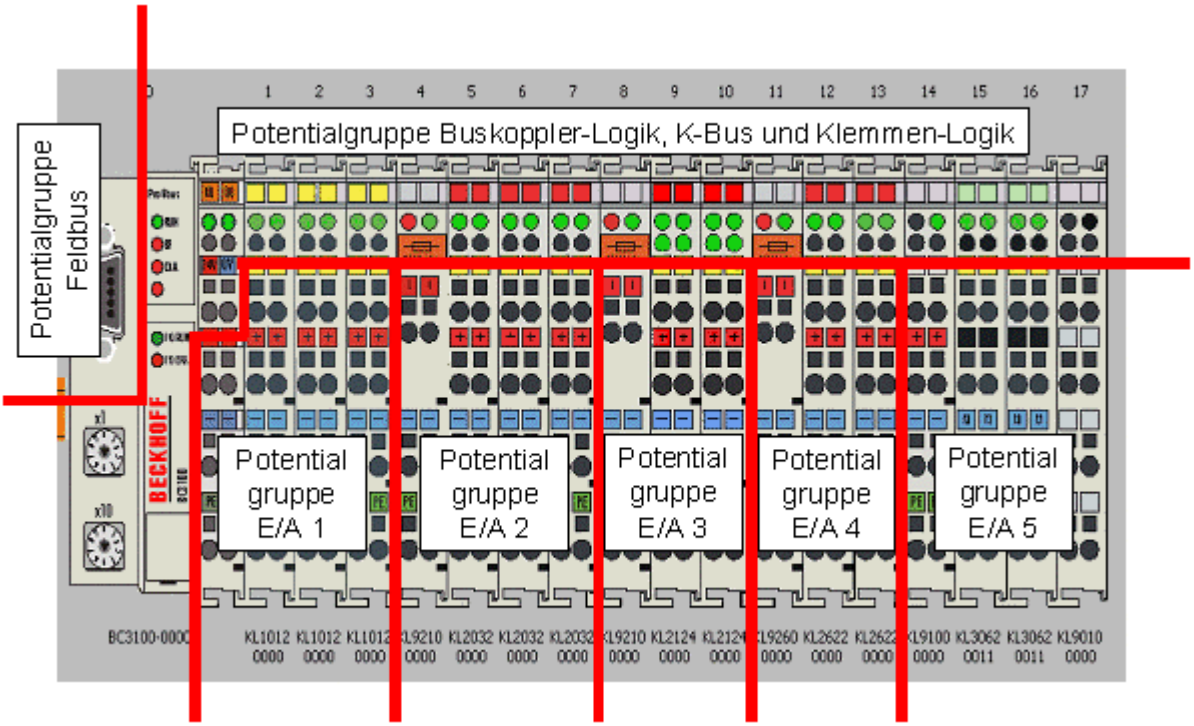


Abb. 10: Potentialgruppen eines Busklemmenblocks

Isolationsprüfung

Die Verbindung zwischen Buskoppler- / Busklemmen-Controller und Busklemmen wird durch das Zusammenstecken der Komponenten automatisch realisiert. Die Übertragung der Daten und die Versorgungsspannung der intelligenten Elektronik der Busklemmen übernimmt der K-Bus. Die Versorgung der Feldelektronik wird über die Powerkontakte durchgeführt. Die Powerkontakte stellen durch das Zusammenstecken eine Versorgungsschiene dar. Da einige Busklemmen (z. B. analoge Busklemmen oder digitale Vierkanal-Busklemmen) diese Powerkontakte nicht oder nicht vollständig durchschleifen, sind die Kontaktbelegungen der Busklemmen zu beachten.

Die Einspeiseklemmen unterbrechen die Powerkontakte und stellen den Anfang einer neuen Versorgungsschiene dar. Der Buskoppler- / Busklemmen-Controller kann auch zur Einspeisung der Powerkontakte eingesetzt werden.

PE-Powerkontakte

Der Powerkontakt mit der Bezeichnung PE kann als Schutzerde eingesetzt werden. Der Kontakt ist aus Sicherheitsgründen beim Zusammenstecken voreilend und kann Kurzschlussströme bis 125 A ableiten.

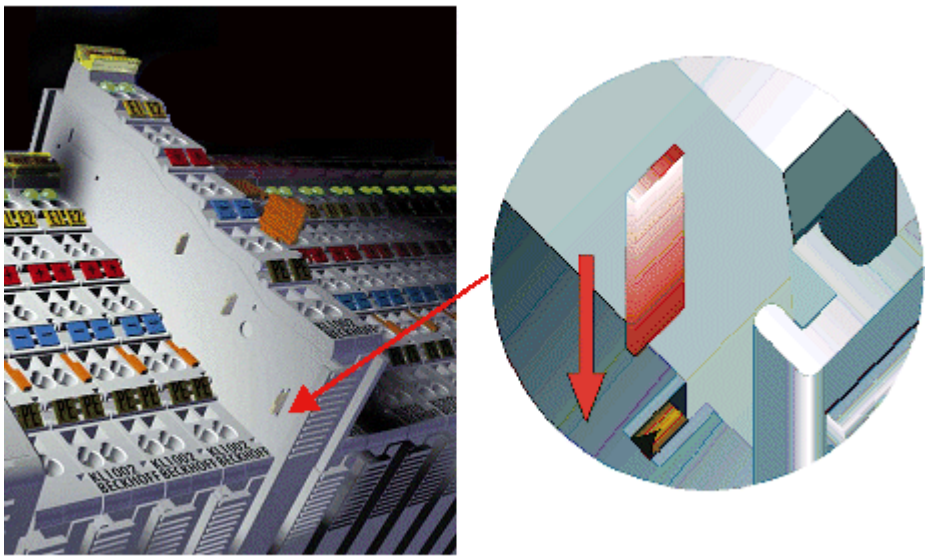


Abb. 11: Linksseitiger Powerkontakt

Es ist zu beachten, dass aus EMV-Gründen die PE-Kontakte kapazitiv mit der Tragschiene verbunden sind. Das kann zu falschen Ergebnissen und auch zur Beschädigung der Klemme bei der Isolationsprüfung führen (z. B. Isolationsdurchschlag an einem 230 V-Verbraucher zur PE-Leitung). Die PE-Zuleitung am Buskoppler- / Busklemmen-Controller muss zur Isolationsprüfung abgeklemmt werden. Um weitere Einspeisestellen für die Prüfung zu entkoppeln, können die Einspeiseklemmen aus dem Verbund der übrigen Klemmen mindestens 10 mm herausgezogen werden. Die PE-Zuleitungen müssen in diesem Fall nicht abgeklemmt werden.

Der Powerkontakt mit der Bezeichnung PE darf nicht für andere Potentiale verwendet werden.

3.4.2 Spannungsversorgung

⚠ VORSICHT

Beachten Sie die UL-Anforderungen für die Spannungsversorgung!

Diese UL-Anforderungen gelten für alle Versorgungsspannungen der BX-Controller (Us und Up)!

Zur Einhaltung der UL-Anforderungen dürfen die BX-Controller nur mit Versorgungsspannungen (24 V_{DC}) versorgt werden, die

- von einer isolierten, mit einer Sicherung (entsprechend UL248) von maximal 4 A geschützten Quelle, oder
- von einer Spannungsquelle die NEC class 2 entspricht stammen.
  Eine Spannungsquelle entsprechend NEC class 2 darf nicht seriell oder parallel mit einer anderen NEC class 2 entsprechenden Spannungsquelle verbunden werden!

⚠ VORSICHT

Keine unbegrenzten Spannungsquellen!

Zur Einhaltung der UL-Anforderungen dürfen die BX-Controller nicht mit unbegrenzten Spannungsquellen verbunden werden!

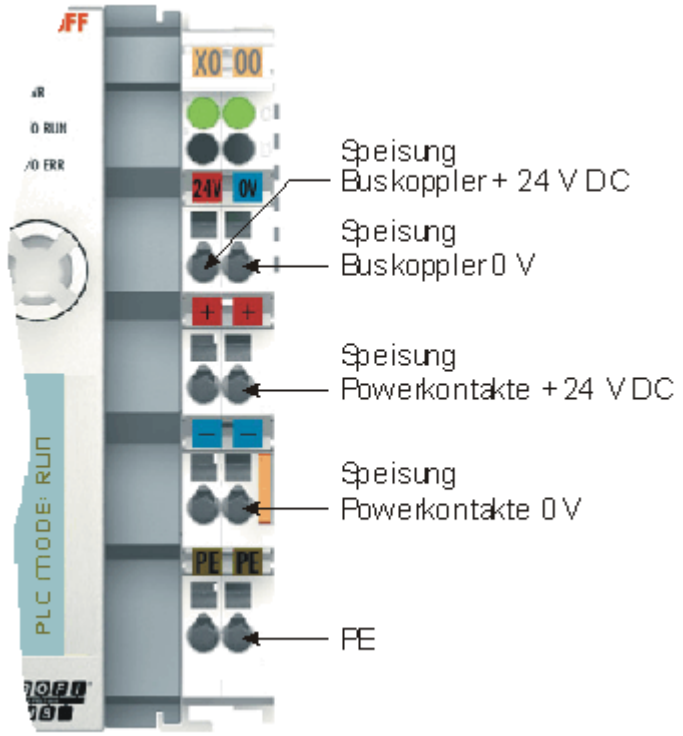


Abb. 12: Klemmstellen zur Versorgung des Busklemmen-Controllers

Abb. 13: UL-Kennzeichnung der BX-Controller

Versorgung von Busklemmen-Controller und Busklemmen (Us)

Der Busklemmen-Controller benötigt zum Betrieb eine Versorgungsspannung von 24 V_{DC}.

Der Anschluss findet beim BX-Controller über die oberen Klemmstellen mit der Bezeichnung *24 V* und *0 V* statt. Diese Versorgungsspannung versorgt die Elektronik des Busklemmen-Controllers sowie über den K-Bus die Elektronik der Busklemmen. Sie ist galvanisch von der Spannung der Feldebene getrennt.

Versorgung der Powerkontakte (Up)

Die unteren sechs Anschlüsse mit Federkraft - Klemmen können zur Einspeisung der Peripherieversorgung benutzt werden. Die Federkraftklemmen sind paarweise mit einem Powerkontakt verbunden. Die Einspeisung zu den Powerkontakten besitzt keine Verbindung zur Spannungsversorgung der BX-Elektronik. Die Auslegung der Einspeisung lässt Spannungen bis zu 24 V zu.

Die Federkraftklemmen sind für Drähte von 0,08 mm^2 bis 2,5 mm^2 Querschnitt ausgelegt.

Die paarweise Anordnung und die elektrische Verbindung zwischen den Speiseklemmkontakten ermöglicht das Durchschliefen der Anschlussdrähte zu unterschiedlichen Klemmpunkten. Die Strombelastung über den Powerkontakt darf 10 A nicht dauerhaft überschreiten. Die Strombelastbarkeit zwischen zwei Federkraftklemmen ist mit der Belastbarkeit der Verbindungsdrähte identisch.

Powerkontakte

An der rechten Seitenfläche des Busklemmen Controllers befinden sich drei Federkontakte der Powerkontaktverbindungen. Die Federkontakte sind in Schlitzen verborgen um einen Berührungsschutz sicher zu stellen. Durch das Anreihen einer Busklemme werden die Messerkontakte auf der linken Seite der Busklemme mit den Federkontakten verbunden. Die Nut/Federführung an der Ober- und Unterseite der Busklemmen Controller und Busklemmen garantiert eine sichere Führung der Powerkontakte.

3.4.3 Programmierkabel für COM1

Für die Programmierung der BX-Controller können Sie ein 1:1 Kabel nehmen (Buchse/Stecker und nur die unten angegebenen PINs verdrahten). Auf der BX-Seite benötigen Sie einen neunpoligen Stecker und auf der PC-Seite meist eine neunpolige Buchse. Die Verdrahtung ist 1:1 und die benötigten PINs sind aus der unten angegebenen Tabelle zu entnehmen. Die Länge des Kabels darf 5 Meter nicht überschreiten!

Beschreibung	BX COM Port 1	PC COM Port RS 232 Serielle Schnittstelle
Kabel	Stecker, Pin	Buchse, Pin
RS 232 RxD/TxD	2	2
RS 232 RxD/TxD	3	3
GND	5	5

HINWEIS

Alle in der Tabelle nicht aufgeführten Pins sind reserviert

Beachten Sie, dass an diesem COM Port [▶ 27] die hier nicht aufgeführten Pins nicht frei verfügbar, sondern mit weiteren Signalen belegt sind.

ZK1000-0030

Das Programmierkabel bietet die Möglichkeit über die Schnittstelle COM 1 den BX-Controller zu programmieren und an der Schnittstelle COM 2 ein weiteres serielles Gerät anzuschließen. Achten Sie im eingebauten Zustand auf die maximale Bauhöhe des Steckers.

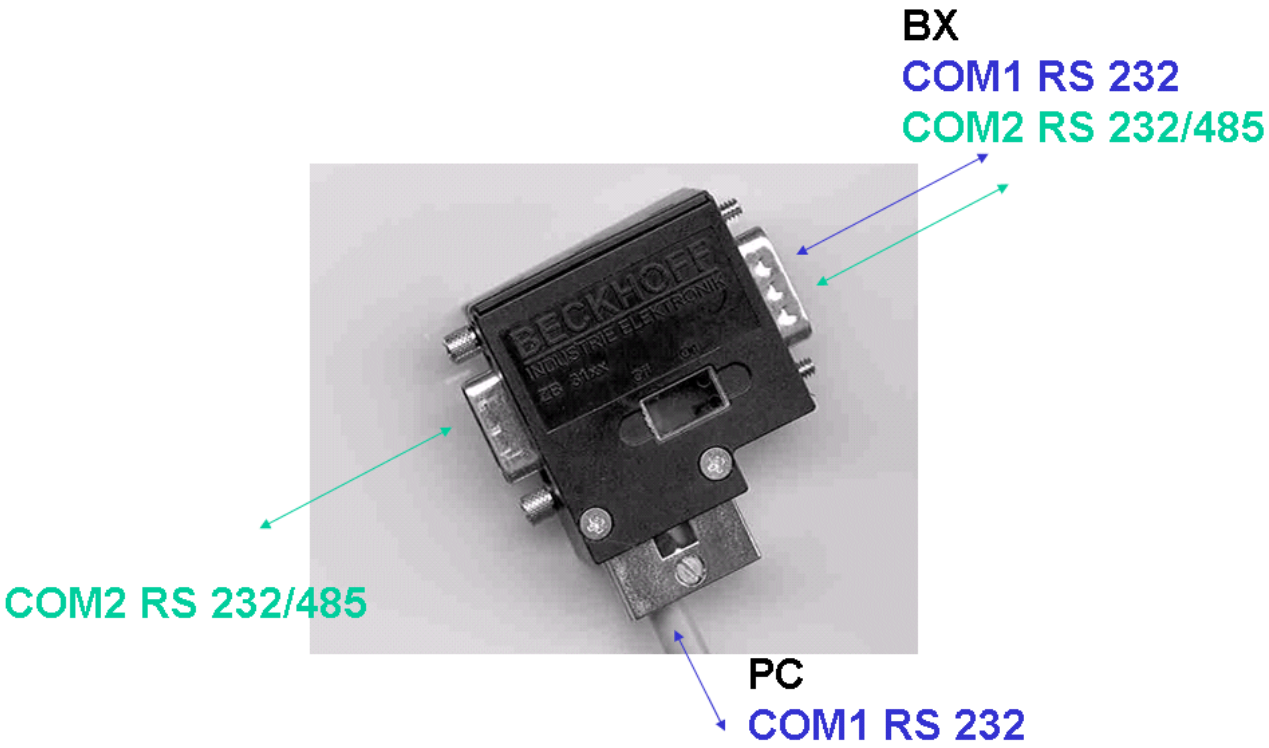
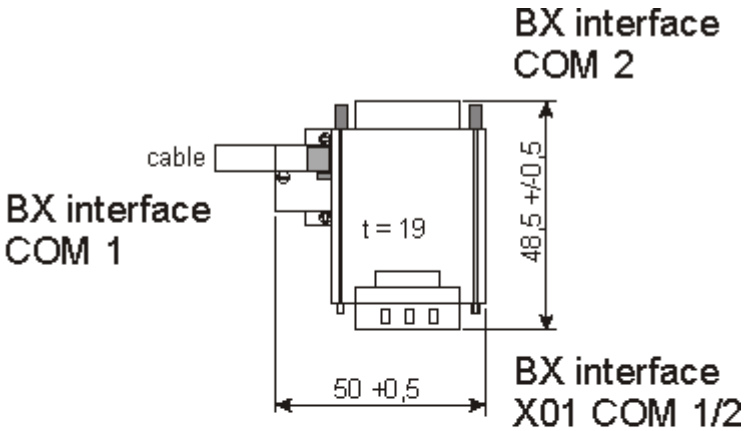


Abb. 14: Programmierkabel ZK1000-0030 - COM 1 und COM 2



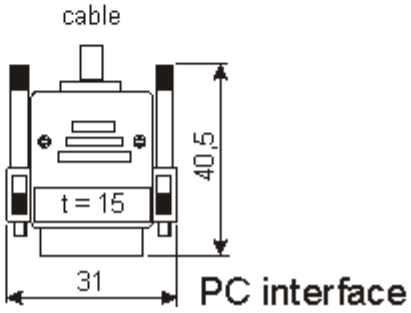


Abb. 15: Programmierkabel ZK1000-0030 - Abmessungen der Stecker

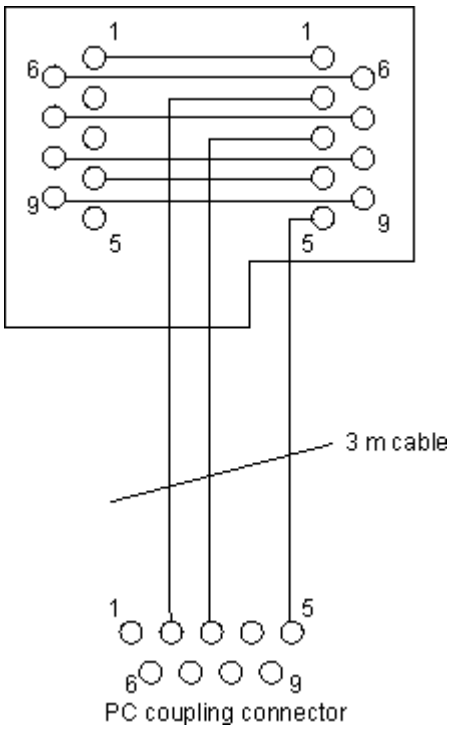


Abb. 16: Programmierkabel ZK1000-0030 - Pinning

3.4.4 SSB- und COM-Schnittstellen

Das Grundmodul der BX-Controller besitzt die Schnittstellen COM1, COM2 und SSB (Smart System. COM1 und COM2 sind auf einer D-Sub-Buchse untergebracht. Ein spezielles Programmierkabel (ZK1000-0030) mit dem Sie beide Schnittstellen verwenden können, können Sie bei Beckhoff bestellen. Die Schnittstelle COM2 ist für den Anschluss serieller Geräte vorgesehen. Sie können bei der Schnittstelle COM2 zwischen RS232 oder RS485 wählen.
Für die serielle Schnittstelle COM2 stehen Bibliotheken [▶ 116] zur Verfügung.

SSB-Schnittstelle

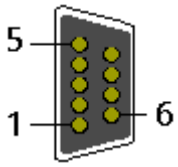


Abb. 17: SSB-Schnittstelle

Belegung der SSB-Schnittstelle (Stecker X00)

PIN	Signal
1	reserviert
2	CAN low
3	GND
4	reserviert
5	Shield
6	GND
7	CAN high
8	reserviert
9	reserviert

COM1 (RS 232) und COM2 (RS 232/485) Schnittstelle

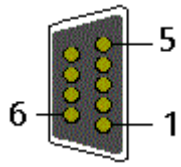


Abb. 18: COM1- (RS 232) und COM2-Schnittstelle (RS 232/485)

Belegung der COM-Schnittstellen (Buchse X01)

PIN	Schnittstelle	Signal
1	COM2	RS485 D+
2	COM1	RS232 TxD
3	COM1	RS232 RxD
4	VCC +5 V	VCC
5	GND	GND
6	COM2	RS485 D-
7	COM2	RS232 RxD
8	COM2	RS232 TxD
9	GND	GND

3.4.5 Ethernet-Anschluss

Der Anschluss an den Ethernet-Bus erfolgt über einen RJ45-Stecker (Westernstecker).

Abb. 19: RJ45-Stecker

Verkabelung

Ethernet-Verbindung über Hub oder Switch

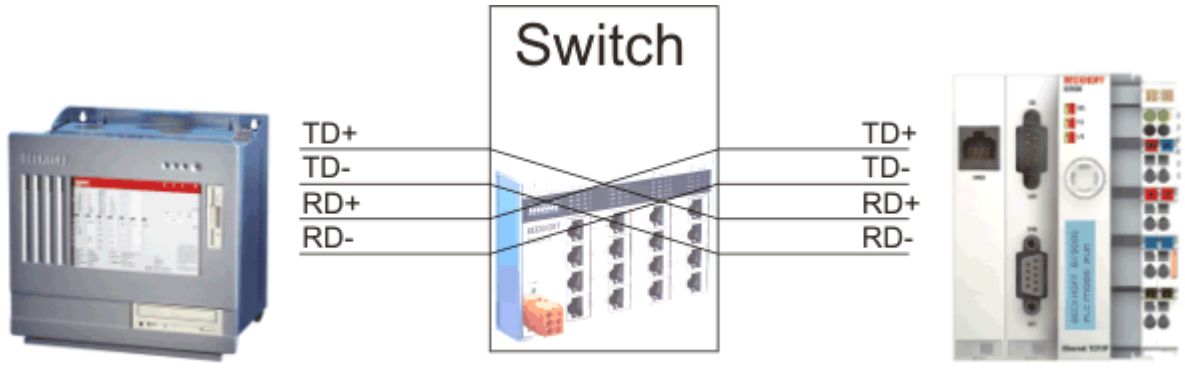


Abb. 20: Ethernet-Verbindung zwischen PC und BX9000 über Hub oder Switch

Verbinden Sie die Netzwerkkarte des PCs über ein Standart-Ethernet-Kabel mit dem Hub/Switch und den Hub/Switch ebenfalls über ein Standard-Ethernet-Kabel mit dem Busklemmen Controller.

Ethernet-Verbindung über Cross-Over-Kabel

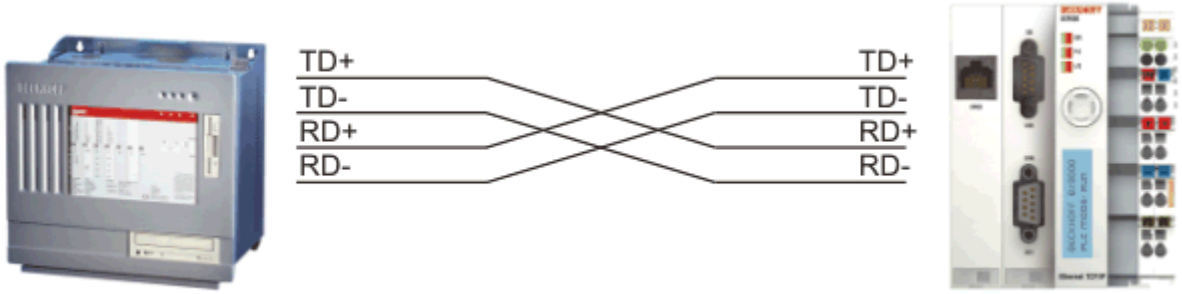


Abb. 21: Ethernet-Verbindung zwischen PC und BX9000 über Cross-Over-Kabel

Um den PC direkt mit dem Busklemmen-Controller zu verbinden, müssen Sie ein Ethernet-Kabel mit gekreuzten Aderpaaren (Cross-Over-Kabel) verwenden.

Belegung des RJ45-Steckers

PIN	Signal	Beschreibung
1	TD +	Transmit +
2	TD -	Transmit -
3	RD +	Receive +
4	-	reserviert
5	-	reserviert
6	RD -	Receive -
7	-	reserviert
8	-	reserviert

4 Parametrierung und Inbetriebnahme

4.1 Anlaufverhalten des Busklemmen-Controllers

Nach dem Einschalten prüft der Busklemmen-Controller seinen Zustand, konfiguriert den K-Bus, erstellt anhand der gesteckten Busklemmen eine Aufbauliste und startet seine lokale SPS.
Beim Hochlaufen des Busklemmen-Controllers leuchten und blinken die I/O-LEDs. Im fehlerfreien Zustand sollte nach ca. 2 bis 3 Sekunden keine I/O-LED mehr Blinken. Im Fehlerfall hängt es von der Fehlerart ab, welche LED blinkt (siehe Kapitel *Diagnose-LEDs*).

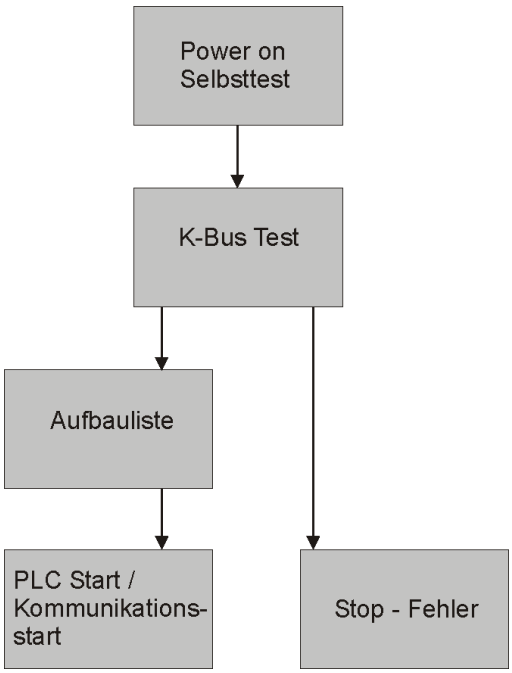


Abb. 22: Anlaufverhalten des Busklemmen-Controllers

4.2 Einstellung der IP-Adresse

Die Einstellung der IP-Adresse kann mit vier unterschiedlichen Verfahren durchgeführt werden, die im Folgenden genauer beschrieben werden.

BX9000: Das Adressierungsverfahren wird per Navigationsschalter und Display oder über die TwinCAT Config vorgegeben.

Verfahren	Erläuterung	Benötigte Komponenten
Manuell	Adressierung über den Navigationsschalter [▶ 198] (nur BX9000)	keine
TwinCAT	Adressierung über TwinCAT System Manager [▶ 31]	PC mit Netzwerk und TwinCAT
BootP*	Adressierung über BootP-Server [▶ 32]	BootP-Server
DHCP*	Adressierung über DHCP-Server [▶ 33]	DHCP-Server
Lokale IP-Adresse*	Lokale IP-Adresse	Falls kein BootP- oder DHCP-Server antwortet oder vorhanden ist

*) BX9000 ab Firmware-Version 1.20

4.2.1 Adresseinstellung über den TwinCAT System Manager

Voraussetzungen

- ab BX-Firmware 1.08 (siehe Anzeige auf dem Display des BX9000 nach Einschalten der Betriebsspannung).
- alle BC9050, BC9020 und BC9120
- ab TwinCAT 2.10, Build 1322

Für die Einstellung der IP-Adresse über den System Manager ist eine funktionsfähige ADS-Verbindung notwendig. Diese kann Seriell oder über Ethernet erfolgen (siehe Kapitel Busklemmen-Controller mit dem TwinCAT System Manager suchen [▶ 44]).

Bei den BC9x20 müssen die DIP-Schalter 9 und 10 auf ON stehen,
beim BC9050 müssen die Schalter 1 und 2 des 2-poligen DIP-Schalter auf ON stehen, um die IP-Adressierung über den System Manager zuzulassen.
Dann sind die Einstellungen, die über den System Manager gemacht wurden, gültig.

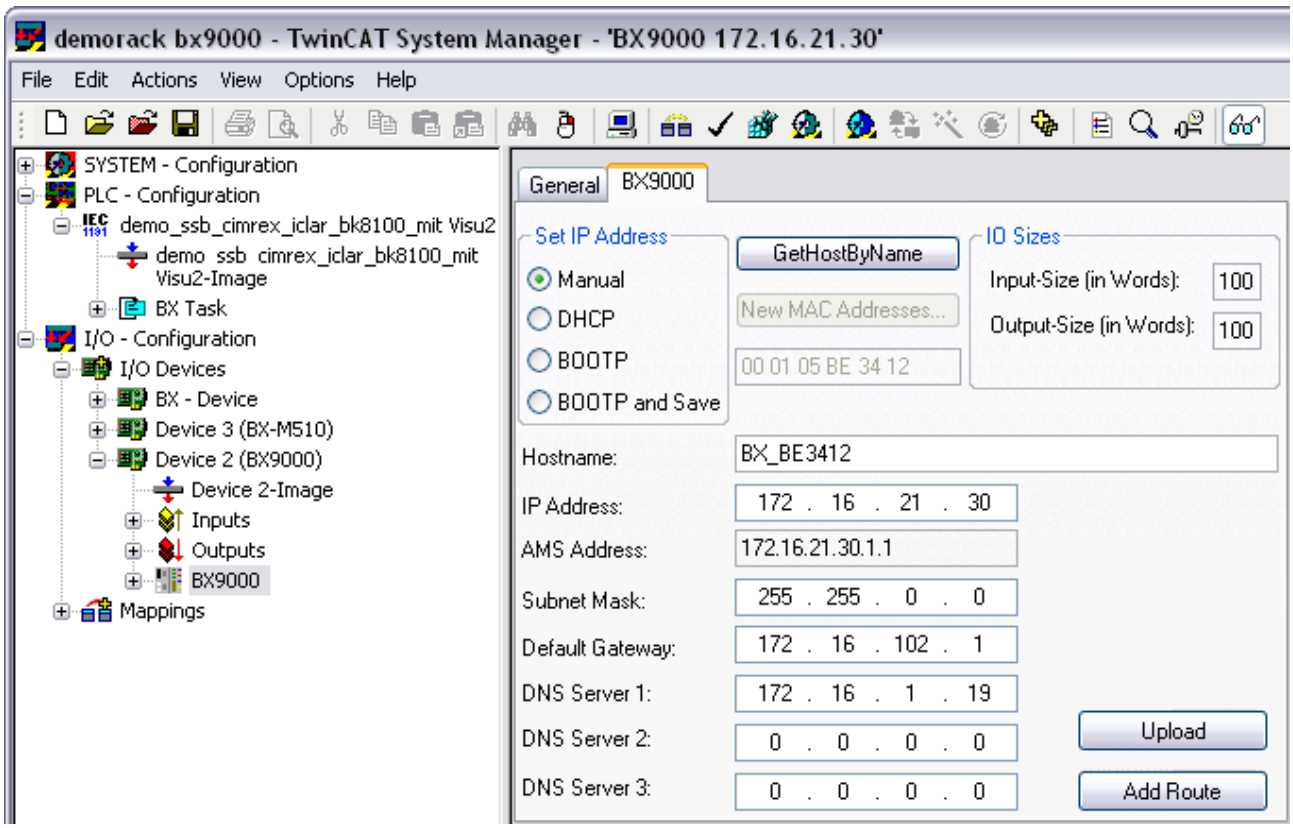


Abb. 23: Adresseinstellung über den TwinCAT System Manager

Mit Upload können Sie jetzt die aktuellen Einstellungen auslesen. Sollten Sie den Busklemmen-Controller offline Konfiguriert haben können Sie durch mit der Schaltfläche *Add Route* eine Verbindung zum Busklemmen-Controller aufbauen. Editieren Sie Ihre gewünschten Einstellungen, aktivieren Sie die Konfiguration  und starten Sie Ihren Busklemmen-Controller mit dem das grünen TwinCAT Icon (Tastenkombination [Ctrl] [F4]) neu. Sollten Sie dem Busklemmen-Controller eine neue IP-Adresse gegeben haben so müssen Sie erneut die neue Route eintragen (siehe Kapitel Busklemmen-Controller mit dem TwinCAT System Manager suchen [▶ 44]).

4.2.2 Adresseinstellung über BootP-Server

BX9000: Die Vergabe der IP-Adresse über einen BootP-Server wird beim BX9000 über den Navigationsschalter des Controllers aktiviert (siehe Kapitel BX-Menü [▶ 87], oder First Steps [▶ 198]).

BC9xx0: Die Vergabe der IP-Adresse über einen BootP-Server wird beim BC9xx0 über den DIP-Schalter aktiviert.

Speicherverhalten der IP-Adresse

BootP & Save

Bei *BootP & Save* wird die vom BootP-Server vergebene IP-Adresse auf dem BX-Controllergespeichert und der BootP-Service wird beim nächsten Kaltstart nicht erneut angefragt.
Die Adresse kann durch Reaktivierung der Herstellereinstellungen (mittels Konfigurationssoftware KS2000 oder DIP-Schalter und Endklemme) wieder gelöscht werden.

BootP

Bei BootP ist die vom BootP-Server vergebene IP-Adresse nur bis zum Ausschalten des Busklemmen-Controller gültig. Beim nächsten Neustart muss der BootP-Server dem Busklemmen-Controller eine neue IP-Adresse vergeben.
Bei einem Software-Reset des Busklemmen-Controller bleibt die Adresse allerdings erhalten.

Beckhoff BootP-Server

Beckhoff bietet einen BootP-Server für Windows 98, ME, NT4.0, NT2000 und XP an. Sie finden die Installationsversion auf der Beckhoff CD *Software Products* im Ordner *Unsupported Utilities* oder im Internet unter https://download.beckhoff.com/

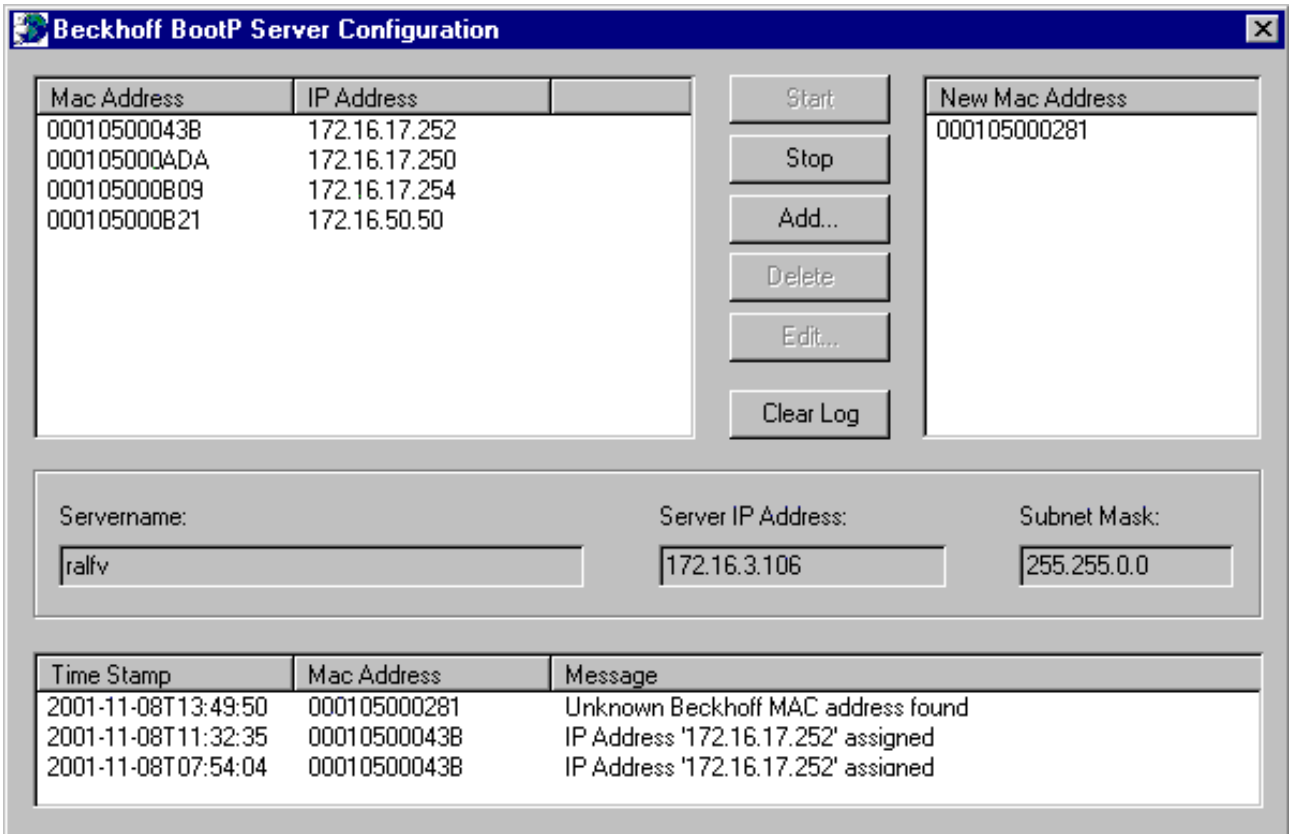


Abb. 24: Adresseinstellung über BootP-Server

Sobald der BootP-Server gestartet wird zeigt das Fenster *New MAC Address* alle Beckhoff-Knoten an, die im BootP-Betrieb arbeiten und noch keine IP-Adresse bekommen haben. Die Zuweisung der MAC-ID zur IP-Adresse erfolgt mit der Schaltfläche [<<]. Eine erfolgreiche Zuweisung wird im Log-Fenster angezeigt.
Um den BootP-Server beim Booten Ihres PCs automatisch zu starten reicht eine Verknüpfung im Windows Autostart-Ordner. Fügen Sie hierzu im Verknüpfungspfad den Parameter */Start* an (.../TcBootPDlg.exe/start).

4.2.3 Adresseinstellung über DHCP-Server

Die Vergabe der IP-Adresse über einen DHCP-Server wird beim BX9000 über den Navigationsschalter des Busklemmen-Controllers aktiviert (siehe dazu BX-Menü, [▶ 87] oder First Steps [▶ 198]).
Die Vergabe der IP-Adresse über einen DHCP-Server wird beim BC9xx0 über den Dipschalter des Busklemmen-Controllers aktiviert (siehe Kapitel *DIP-Schalter*).

Ist DHCP eingeschaltet bekommt der Busklemmen-Controller automatisch eine IP-Nummer vom DHCP-Server zugewiesen. Der DHCP-Server muss hierfür die MAC-ID des Busklemmen-Controller kennen. Die IP-Adresse sollte statisch eingestellt werden.

Der DNS-Name wird aus dem Typ und den letzten 3 Byte der MAC-ID gebildet. Die MAC-ID steht auf dem Produktionsaufkleber des Busklemmen Controllers.

Beispiel zum BX9000

- MAC ID: 00-01-05-01-02-03
- DNS-Name: BX_010203

Beispiel zum BC9050

- MAC ID: 00-01-05-03-02-01
- DNS-Name: BC_030201

4.2.4 Subnetz-Maske

Die Subnetz-Maske unterliegt der Kontrolle des Netzwerkverwalters und legt die Struktur der Subnetze fest.

Kleine Netze ohne Router benötigen keine Subnetz-Maske. Das gleiche gilt, wenn Sie keine registrierten IP-Nummern verwenden. Sie können die Subnetz-Maske dazu verwenden, anstelle des Gebrauchs vieler Netznummern das Netz mit dieser Maske zu unterteilen.

Die Subnetz-Maske ist eine 32-Bit Ziffer:

- Einsen in der Maske kennzeichnen den Subnetz-Nummernteil eines Adressraums.
- Nullen kennzeichnen den Teil des Adressraums, der für die Host-IDs zur Verfügung steht.

Beschreibung	Binäre Darstellung	Dezimale Darstellung
IP-Adresse	10101100.00010000.00010001.11001000	172.16.17.200
Subnetz-Maske	11111111.11111111.00010100.00000000	255.255.20.0
Netz-ID	10101100.00010000.00010000.00000000	172.16.16.0
Host-ID	00000000.00000000.00000001.11001000	0.0.1.200

Standard Subnetz-Maske

Adressklasse	Standard Subnetz-Maske (dezimal)	Standard Subnetz-Maske (hex)
A	255.0.0.0	FF.00.00.00
B	255.255.0.0	FF.FF.00.00
C	255.255.255.0	FF.FF.FF.00

Beachten Sie zu Subnetzen und zur Hostnummer

Die Subnetze 0 und das nur aus Einsen bestehende Subnetz dürfen nicht verwendet werden!
Die Host-Nummer 0 und die aus nur Einsen bestehende Host-Nummer dürfen nicht verwendet werden!
Wenn die IP-Adresse über die Konfigurationssoftware KS2000 eingestellt wurde, muss auch die Subnetz-Maske mit der Konfigurationssoftware KS2000 geändert werden.
Bei ARP-Adressierung wird anhand der IP-Adresse die dazugehörige Standard Subnetz-Maske eingetragen.
Bei BootP und DHCP wird die Subnetz-Maske mit vom Server übertragen.

4.3 Konfiguration

4.3.1 Überblick

Konfigurationsarten

Bei den Busklemmen-Controllern der Serien BCxx50, BCxx20 und BXxx00 gibt es zwei verschiedene Arten der Konfiguration, die DEFAULT CONFIG und die TWINCAT CONFIG.

DEFAULT-CONFIG

Die Busklemmen mappen sich in der Reihenfolge wie diese gesteckt sind, erst die komplexen Busklemmen, dann die digitalen Busklemmen.

Das Mapping der komplexen Busklemmen ist:

- Word-Alignment
- komplexe Darstellung

⚠ VORSICHT

Prozessabbild ist abhängig von angesteckten Klemmen!

Das Prozessabbild verändert sich, sobald eine Klemme dazu gesteckt wird oder entfernt wird!

Die Daten der Feldbus-Slave Schnittstelle werden SPS-Variablen genannt. Die SPS-Variablen befinden sich ab der Adresse %QB1000 und %IB1000.

Weiterhin kann die DEFAULT-CONFIG ohne SPS-Programm für das Schreiben und Testen der Angeschlossenen Busklemmen verwendet werden. Dafür muss im System Manager der Busklemmen-Controller gescannt werden und die Betriebsart FreeRun aktiviert werden (um diese Funktion zu nutzen darf kein SPS-Programm auf dem Busklemmen-Controller sein).

TWINCAT-CONFIG

In der TWINCAT-CONFIG können die Busklemmen und die SPS-Variablen frei verknüpft sein (TwinCAT System Manager-File notwendig). Die Konfiguration wird mit Hilfe des System Managers per ADS zum Koppler übertragen.

Für die TwinCAT Config (TC-File) benötigen Sie folgendes:

- Über der Feldbus (PROFIBUS, CANopen, Ethernet)
  PROFIBUS: (BC3150, BX3100)
  - PC mit FC310x ab Version 2.0 und TwinCAT 2.9 Build 1000
  - BX3100 mit CIF60 oder CP5412
  - TwinCAT 2.9 Build 946
    (**ACHTUNG:** bei den PROFIBUS Karten von Hilscher ist nur eine ADS-Kommunikation erlaubt, d.h. entweder System Manager oder PLC Control)
    CANopen: (BC5150, BX5100)
  - PC mit FC510x ab Version 1.76 TwinCAT Build 1030
    DeviceNet: (BC5250, BX5200)
  - Auf Anfrage
    Ethernet: (BC9050, BC9020, BC9120, BX9000)
  - PC mit TwinCAT 2.10 Build 1322
- Über das serielle ADS TwinCAT 2.9 Build 1010
  - BX3100 Version 1.00
  - BX5100 Version 1.00
  - BX5200 Version 1.10
  - BX8000 Version 1.00
  - BC3150, BC5150, BC5250, BC9050, BC9020, BC9120 ab Firmware B0
  - Für BC8150 ab TwinCAT 2.10 Build 1243

BCxx50 und BXxx00 können über den System Manager des TwinCAT Programms parametriert werden.

- Variables I/O Mapping
- Typ gerechte PROFIBUS Daten (nur BC3150 und BX3100)
- RTC (Real Time Clock) (nur BX-Serie)
- SSB Bus (Smart System Bus) (nur BX-Serie)
- PLC Einstellungen

- K-Bus Einstellungen

Die Konfiguration kann per Feldbus ADS-Protokoll oder seriellem ADS-Protokoll zum BCxx50 oder BXxx00 übertragen werden.

Mit der TwinCAT Konfiguration kann man Variablen, I/Os und Daten verknüpfen. Folgendes ist möglich:

- PLC - K-BUS
- PLC - Feldbus (z. B. PROFIBUS Slave Schnittstelle zur PLC)
- K-Bus - Feldbus (nur bei den BX-Controllern)
- Unterstützung der TwinSAFE-Klemmen (nur BX-Controller ab Firmware 1.17)

Zusätzlich können mit der TwinCAT Konfiguration spezielle verhalten parametriert werden, zum Beispiel ob bei einem Feldbus Fehler die Daten erhalten bleiben oder auf "0" gesetzt werden sollen.
Die Echtzeituhr kann man über einen Karteireiter im System Manager einstellen.

Arbeitsschritte

1. Feldbus Adresse einstellen
2. System Manager öffnen und TC-File anlegen
3. Feldbus Daten in dem TC-File konfigurieren
4. TC-File speichern
5. Neuer System Manager öffnen und PC-File anlegen und gespeichertes TC-File einlesen
6. Verknüpfen zu einer SPS-Task herstellen
7. Speichern der Konfiguration
8. Starten des TwinCAT Systems
9. System Manager des TC-File öffnen, fertig konfigurieren und zum BCxx50, BCxx20 oder BXxx00 übertragen
10. Programm zum BCxx50, BCxx20 oder BXxx00 übertragen
11. Bootprojekt erzeugen

4.3.2 Anlegen einer TwinCAT-Konfiguration

Um einen Busklemmen-Controller der Serien BCxx50, BCxx20, BXxx00 oder BC9191 zu konfigurieren muss im System-Manager ein BX-File angelegt werden. Zur Vereinfachung sind die Grundgeräte schon als File vorbereitet. Dazu öffnen Sie mit *New from Template* den entsprechenden Busklemmen-Controller.

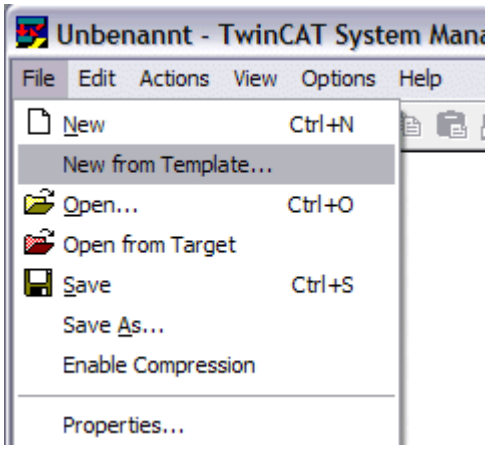


Abb. 25: Anlegen einer TwinCAT-Konfiguration

Wählen Sie den entsprechenden Busklemmen-Controller aus.

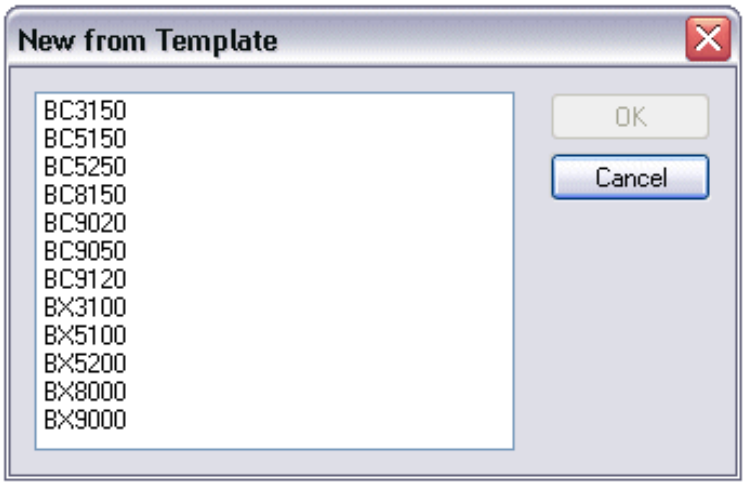


Abb. 26: Auswahl des Busklemmen-Controllers

Nun sind alle Komponenten des Busklemmen-Controllers vorhanden:

- Feldbusschnittstelle
- K-Bus Interface [▶ 49]
- PLC Programm [▶ 51]
- SSB [▶ 54] (nur Busklemmen-Controller der BX-Serie)

Die Konfiguration der Geräte entnehmen Sie den entsprechenden Kapiteln.

4.3.3 Download einer TwinCAT-Konfiguration

Die TwinCAT-Konfiguration wird per ADS-Protokoll zum Busklemmen-Controller geladen.

Serielles ADS-Protokoll

(alle Busklemmen-Controller der Serien BXxx00 und BCxx50)

Tragen Sie die serielle ADS-Verbindung ein, wie unter dem Kapitel Serielles ADS [▶ 42] beschrieben ist.

ADS-Protokoll über den Feldbus

(nur BC3150, BC5150, BC9x20, BC9050, BX3100, BX5100, BX9000, BC9191)

Vorraussetzung hierfür ist, dass TwinCAT als Master arbeitet und sich im Datenaustausch befindet, d.h. die physikalische, wie auch die Feldbus-Konfiguration muss abgeschlossen sein und der Datenaustausch vom Master (z. B. Feldbus-Master-Karte) zum Busklemmen-Controller stattfinden.

Auswahl des Zielsystems

Wählen Sie den Busklemmen-Controller aus, auf den Sie die Konfiguration laden wollen. Mit der Funktionstaste F8 öffnet sich der Dialog, mit indem Sie Ihr File auf das entsprechende Gerät herunterladen können.

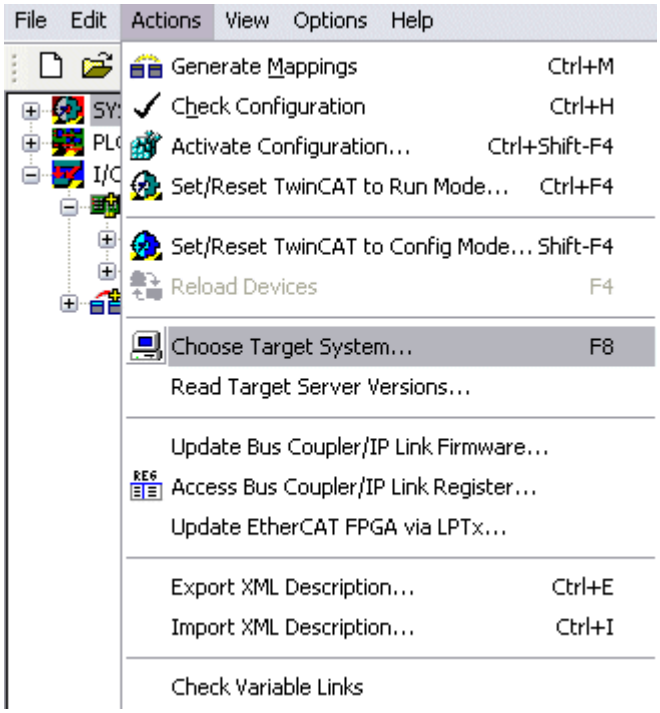


Abb. 27: Download einer TwinCAT-Konfiguration

Wählen Sie den entsprechenden Busklemmen-Controller aus.

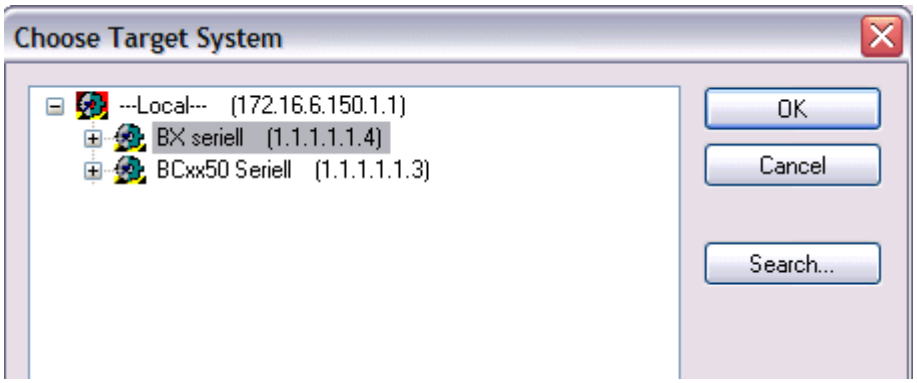


Abb. 28: Auswahl des Busklemmen-Controllers

Den Zustand des Busklemmen-Controllers wird unten rechts im System-Manager angezeigt.

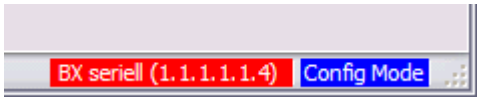


Abb. 29: Zustand des Busklemmen-Controllers

Im *Config Mode / FreeRun* kann man jetzt die Konfiguration zum Busklemmen-Controller herunterspielen. Wenn der Busklemmen-Controller im *Stop Mode* ist, ist die ADS-Kommunikation noch nicht aktiviert. Ein Download der Konfiguration ist dann nicht möglich.

Zum Aktivieren der TwinCAT-Konfiguration wählen Sie Ctrl+Shift+F4 oder *Activate Configuration*.

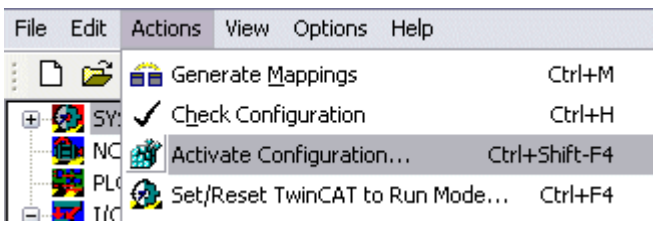


Abb. 30: Aktivieren der TwinCAT-Konfiguration

Die aktuelle Konfiguration wird in dem Busklemmen-Controller geladen. Im Display erscheint *Store Config* und die BUS und I/O LED blinken. Nachdem die Konfiguration erfolgreich im Busklemmen-Controller ist, sollte ein BXxx00 im Display *TwinCAT Config* anzeigen. Jetzt kann man das entsprechende Programm zum Busklemmen-Controller übertragen (Programm-Download über den Feldbus) [▶ 173].

4.3.4 Upload einer TwinCAT-Konfiguration

Die TwinCAT-Konfiguration wird per ADS-Protokoll zum Busklemmen-Controller geladen.

Serielles ADS-Protokoll

(alle Busklemmen-Controller der Serien BCxx50, BCxx20 und BXxx00)

Tragen Sie die serielle ADS-Verbindung ein, wie unter dem Kapitel Serielles ADS [▶ 42] beschrieben ist.

ADS-Protokoll über den Feldbus

(nur BC3150, BC5150, BC9x20, BC9050, BX3100, BX5100, BX9000, BC9191)

Voraussetzung hierfür ist, dass TwinCAT als Master arbeitet und sich im Datenaustausch befindet, d.h. die physikalische, wie auch die Feldbus Konfiguration muss abgeschlossen sein und der Datenaustausch vom Master (z. B. Feldbus-Karte) zum Busklemmen-Controller stattfindet.

Auswahl des Zielsystems

Wählen Sie den Busklemmen-Controller aus, auf den Sie die Konfiguration laden wollen. Mit der Funktionstaste [F8] öffnet sich der Dialog, mit indem Sie Ihr File auf das entsprechende Gerät herunterladen können.

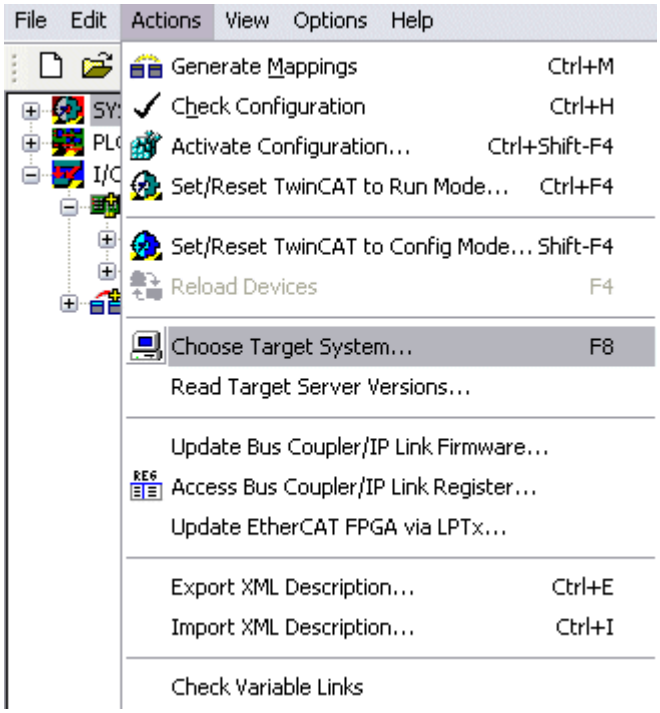


Abb. 31: Auswahl des Zielsystems

Wählen Sie den entsprechenden Busklemmen-Controller aus.

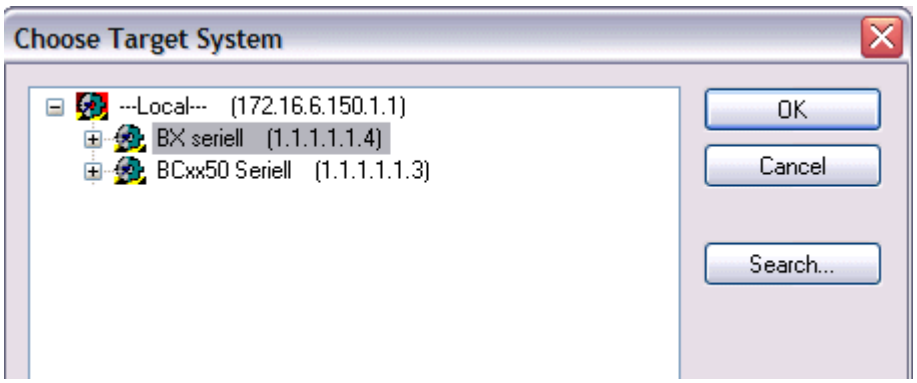


Abb. 32: Auswahl des Busklemmen-Controllers

Der Zustand des Busklemmen-Controllers wird unten rechts im System-Manager angezeigt.

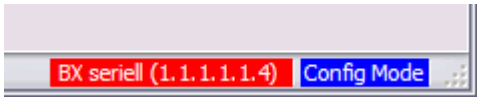


Abb. 33: Zustand des Busklemmen-Controllers

Klicken Sie den roten Ordner an. Die TwinCAT-Konfiguration wird jetzt hochgeladen.

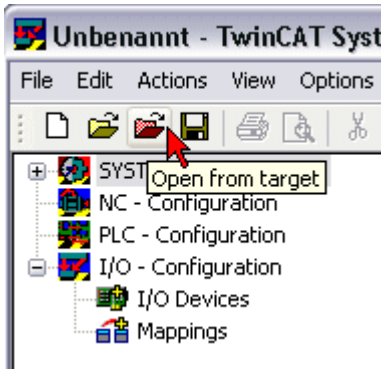


Abb. 34: Hochladen der TwinCAT-Konfiguration

4.3.5 Ressourcen im Busklemmen-Controller

Die im Busklemmen-Controller belegten Speicher-Ressourcen zeigt der System Manager auf dem Karteireiter *Resources* des Busklemmen-Controller an.

Mapping Code

Der Mapping Code wird für die Berechnung der TwinCAT Konfiguration benötigt (siehe Abb. *Speicher für das Code Mapping*). Die Prozentzahlen werden hier zusammen addiert, in dem Beispiel aus Abb. *Speicher für das Code Mapping* sind 8% des Speichers für die Mapping-Berechnung belegt.

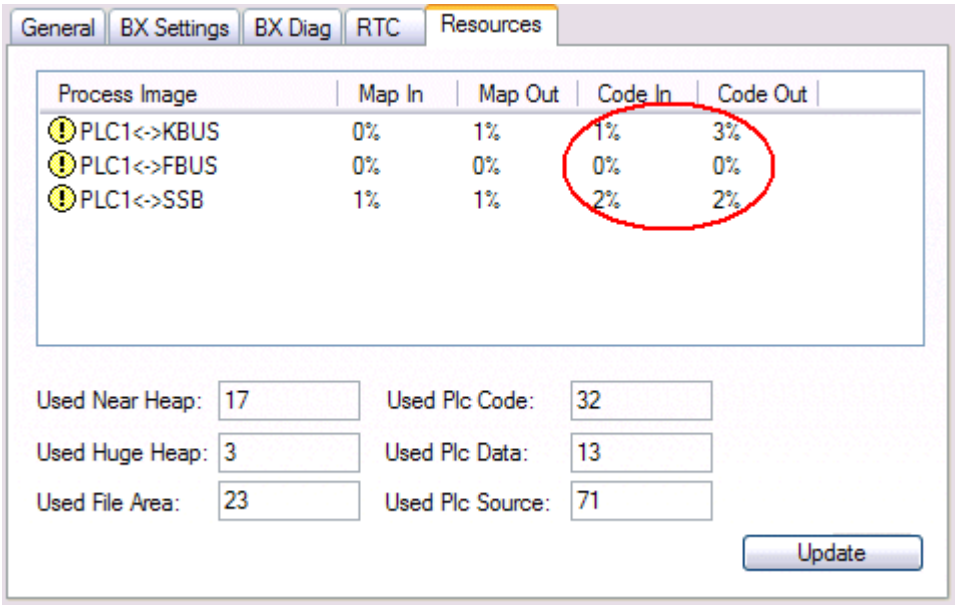


Abb. 35: Speicher für das Code Mapping

Daten Speicher Mapping

Daten Speicher für die Mappings. Die Werte sind einzeln zu betrachten, das bedeutet das jeder der Werte bis zu 100% betragen kann.

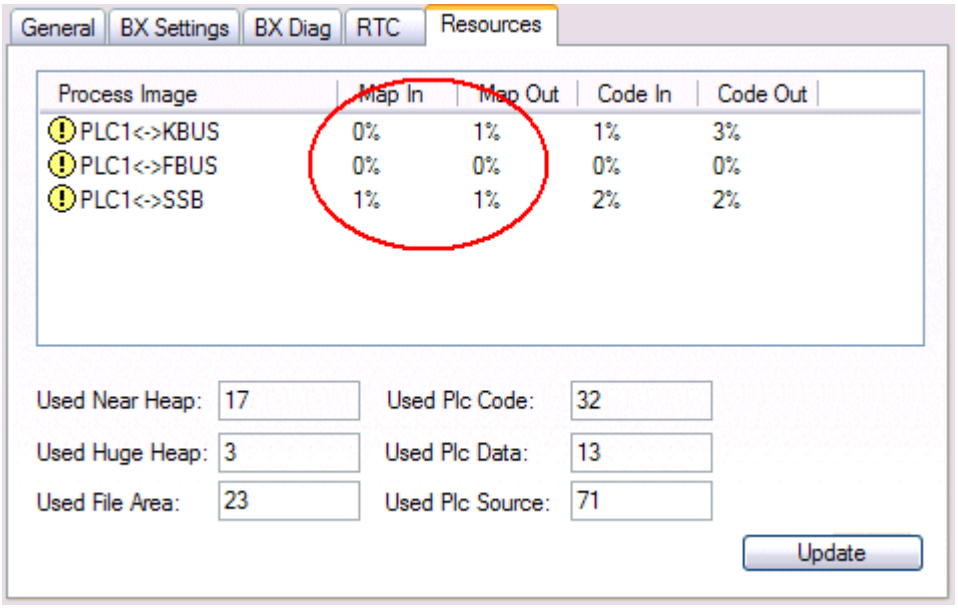


Abb. 36: Daten Speicher Mapping

Verbrauchter Code und Daten Speicher

Abb. *Code und Daten Speicher* (1) "Used Plc Code" verbrauchter PLC Code, Angabe in %.
Abb. *Code und Daten Speicher* (2) "Used Plc Data" verbrauchter PLC Daten, Angabe Speicher in %.
Abb. *Code und Daten Speicher* (3) "Used Plc Source" verbrauchter Source Code, Angabe in %.

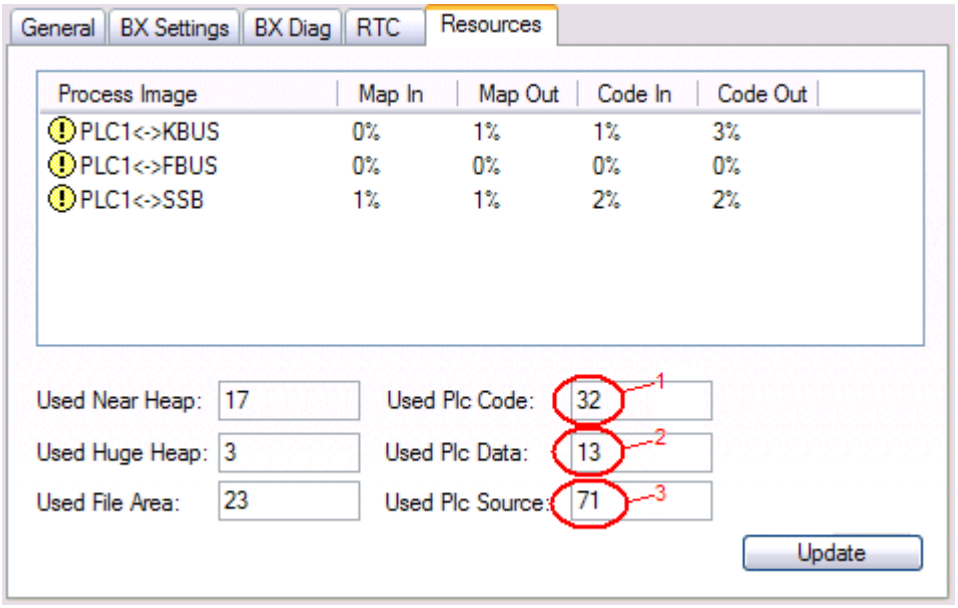


Abb. 37: Code und Daten Speicher

Sonstiger Speicher

Abb. *Sonstiger Speicher* (1) "Used Near Heap" wird für die COM Schnittstelle und SSB benötigt. Angabe in %.
Abb. *Sonstiger Speicher* (2) "Used Huge Heap" wird für die ADS Kommunikation benötigt. Angabe in %. Dieser Wert sollte kleiner 30 % betragen.
Abb. *Sonstiger Speicher* (3) "Used File Area" wird für die TwinCAT Konfiguration, dem TSM-File und dem 16 kByte Flash Zugriff benötigt. Angabe in %.

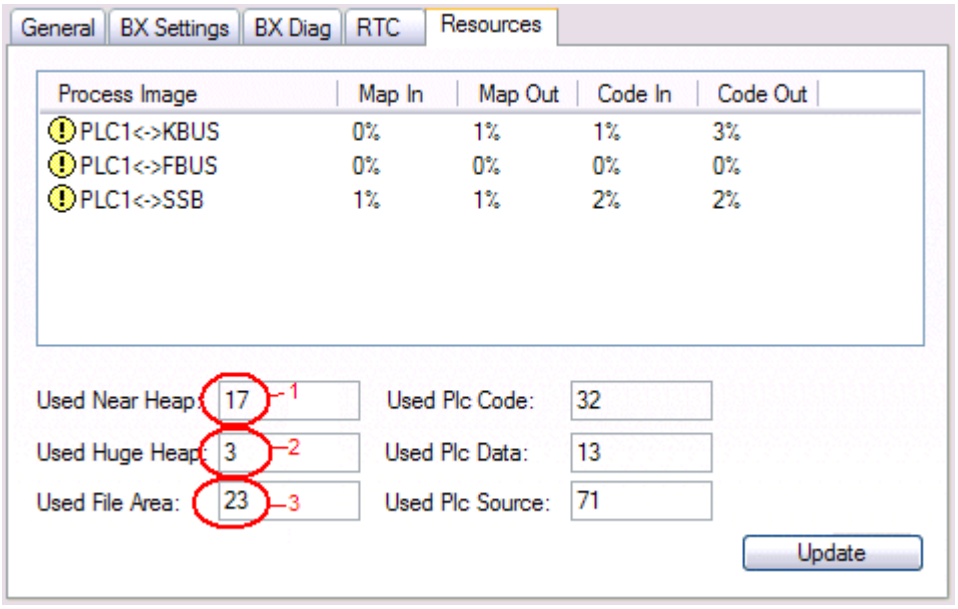


Abb. 38: Sonstiger Speicher

4.3.6 ADS-Verbindung über die serielle Schnittstelle

(ab Firmware-Version 1.xx oder 0.99x - Busklemmen-Controller der BX-Serie und bei allen BCxx50)

Ab TwinCAT 2.9 Build 1020 (TwnCAT Level PLC, NC oder NCI)

Verwenden Sie nur eine serielle Verbindung

Um die problemlose Funktion der ADS-Verbindung über die serielle Schnittstelle zu garantieren, ist nur dabei eine serielle Verbindung zum BX-Controller erlaubt.
Nach einer erfolgreichen Konfiguration mit dem System Manager schließen Sie diesen, bevor Sie die Programmierung starten.

AMS-Net-ID im Auslieferungszustand (Default)

Für BX9000

Die Default AMS-Net-ID lautet 172.16.21.20.1.1. Wird die IP-Adresse des BX9000 verändert, so verändert sich auch die AMS-Net-ID des BX9000. Die aktuelle AMS-Net-ID können Sie mit Hilfe des Menüs sich auf dem Display anzeigen lassen.
Beispiel: Sie ändern die IP-Adresse auf 10.2.3.7 so ändert sich die AMS-Net-ID auf 10.2.3.7.1.1.

Für BC9050, BC9020, BC9120

Die Default AMS-Net-ID lautet 172.16.xxx.[DIP-Switch].1.1. Wird die IP-Adresse des BC9xxx verändert, so verändert sich auch die AMS-Net-ID des BC9xxx.
Beispiel: Sie ändern die IP-Adresse auf 10.2.3.7 so ändert sich die AMS-Net-ID auf 10.2.3.7.1.1.
BC9050: DEFAULT 172.16.21.[DIP-Switch].1.1
BC9020: DEFAULT 172.16.22.[DIP-Switch].1.1
BC9120: DEFAULT 172.16.23.[DIP-Switch].1.1

ADS-Verbindung initialisieren

Tragen Sie den Busklemmen-Controller unter TwinCAT in die Remote-Verbindung ein. Klicken Sie dazu auf das TwinCAT-Icon und öffnen Sie das Eigenschafts-Menü. Unter dem Karteireiter >AMS Remote< können Sie dann folgende Einträge vornehmen.

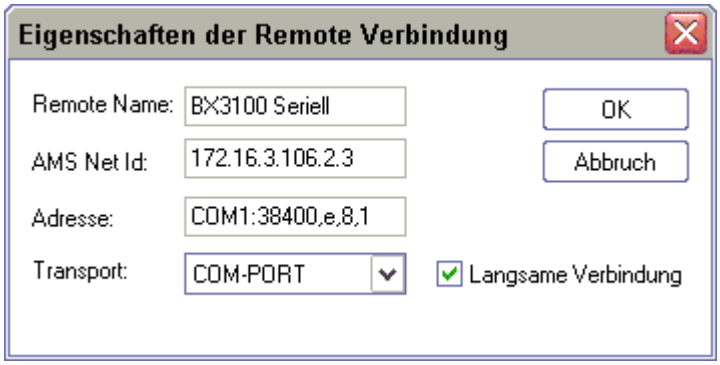


Abb. 39: Eigenschaften der Remote-Verbindung

Remote Name: Beliebig
AMS-Net-ID: 1.1.1.1.1.1 (Default)
Adresse: COM Port: Baudrate, Parity, Datenbits, Stop Bits
Transport: Hier ist "COM-PORT" anzuwählen

Wird der Busklemmen-Controller eingeschaltet, so hat dieser per Default immer die AMS-Net-ID "1.1.1.1.1.1" (außer alle Ethernet Controller).
Sie können diese AMS-Net-ID beliebig ändern. Die neue AMS-Net-ID kann dann allerdings nicht noch einmal geändert werden.

Wenn Sie die AMS-Net-ID erneut ändern möchten, müssen Sie den Busklemmen-Controller zuerst neu starten, so dass die AMS Net Id wieder auf die Default AMS-Net-ID "1.1.1.1.1.1" zurückgesetzt wird.
Nun können sie die AMS-Net-ID wiederum einmal ändern.

Erst beim zweiten Aufruf können Sie Strings eingeben

Beim ersten Aufruf des Dialogs (siehe oben) können Sie unter Adresse keine Strings eingeben. Tragen Sie Namen, AMS-Net-ID und Transport-Typ ein und schließen Sie den Dialog. Beim zweiten Aufruf können Sie dann Ihre COM-Schnittstelle eintragen.

Die Kommunikation startet sobald TwinCAT im Config- (TwinCAT-Icon ist blau) oder RUN-Modus (TwinCAT-Icon ist grün) ist. Die COM-Schnittstelle wird erst mit einem TwinCAT Stopp (TwinCAT-Icon ist rot) geschlossen und ist dann wieder für andere Programme verfügbar. Sollte die COM-Schnittstelle bei einem TwinCAT-Neustart von einem anderen Programm (z. B. von der Konfigurations-Software KS2000) verwendet werden, so wird kein Fehlerhinweis ausgegeben.

AMS-Net-ID nach ADS-Verbindung über den Feldbus

Sollte Sie vor oder während der Nutzung der seriellen ADS-Verbindung den Busklemmen-Controller mit einer ADS-Verbindung über den Feldbus angesprochen haben, ist die AMS-Net-ID vom System Manager automatisch verändert worden. Eine erneute serielle ADS-Verbindung ist dann nur möglich, wenn die AMS-Net-ID angepasst wird.

BX-Serie: auslesen der AMS-Net-ID

Die aktuelle AMS-Net-ID kann aus dem Menü über das Display des Busklemmen-Controller der BX-Serie ausgelesen werden.

AMS	AMS-Net-ID
1.1.1.1.1.1	

4.3.7 ADS-Verbindung über Ethernet

4.3.7.1 Busklemmen-Controller mit dem TwinCAT System Manager suchen

Voraussetzung BX9000:

- ab BX-Firmware 1.08 (siehe Anzeige auf dem Display des BX9000 nach Einschalten der Betriebsspannung)
- ab TwinCAT 2.10 Build 1251

Voraussetzung BC9050, BC9020, BC9120:

- ab Firmware B1
- ab TwinCAT 2.10 Build 1322

Zur Einstellung der IP-Adresse über den System Manager ist eine funktionsfähige ADS-Verbindung notwendig. Diese kann Seriell oder über Ethernet erfolgen.

Für die ADS-Verbindung über Ethernet ist eine funktionierende Ethernet Verbindung notwendig. Sie können die IP-Verbindung mit dem Befehl PING testen. Per Default ist der Busklemmen-Controller auf 172.16.21.20 eingestellt mit der Sub-Netzmaske 255.255.0.0. Stellen Sie Ihren PC auf die gleiche Netzwerkklasse ein, zum Beispiel 172.16.200.100 (Sub Net Mask 255.255.0.0) ein.

Prüfen Sie nun mit PING ob eine Verbindung existiert:

Starten Sie nun den TwinCAT System Manager und suchen Sie den Busklemmen-Controller mit Hilfe der im Bild markierten Schaltfläche oder drücken Sie [F8]:

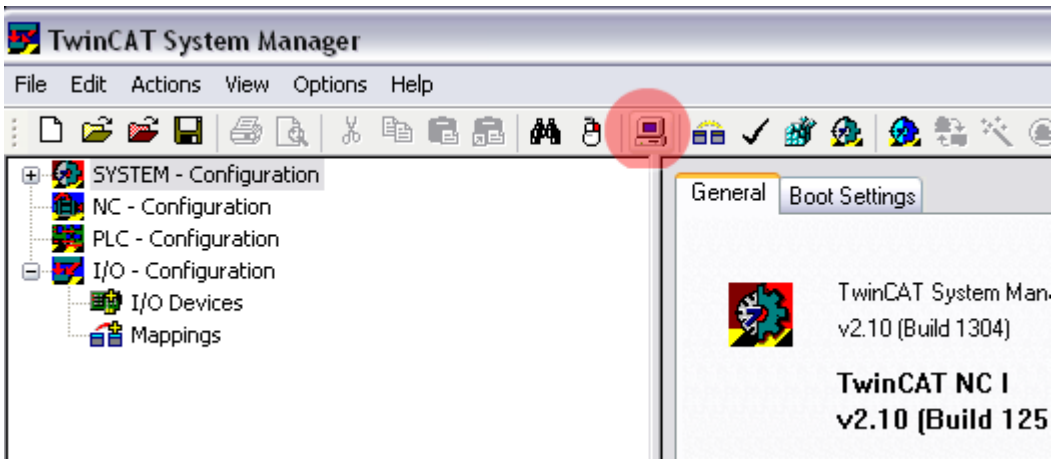


Abb. 40: Festlegen des Zielsystems

Nun suchen Sie über Ethernet den Busklemmen-Controller:

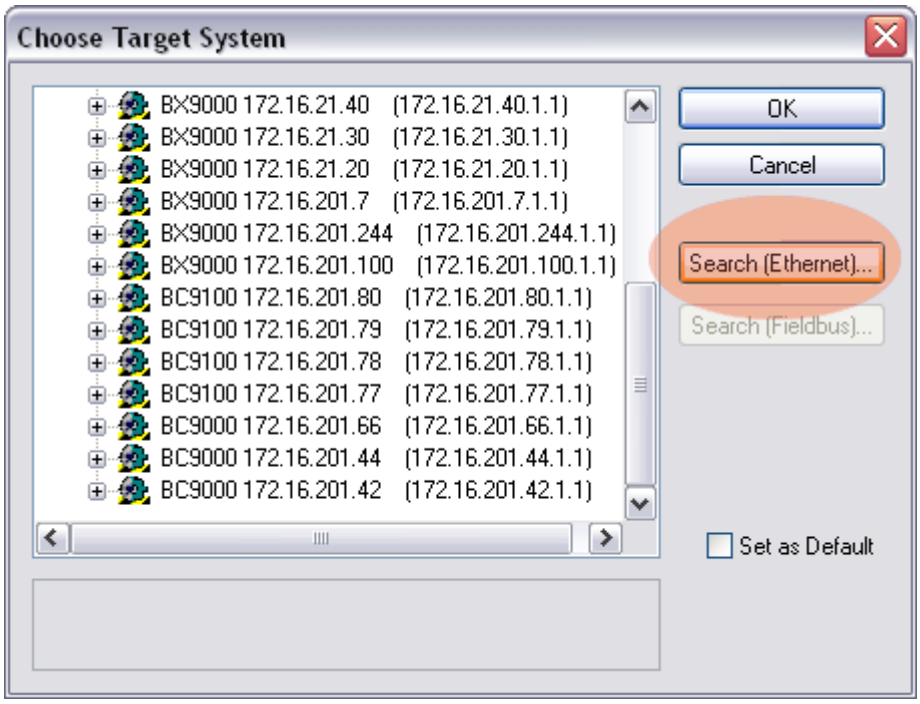


Abb. 41: Suchen des Busklemmen-Controllers

Suchen Sie nun über *Broadcast Search* Ihren Busklemmen-Controller. Haben Sie mehrere Busklemmen-Controller in Ihrem Netzwerk, können Sie diese mit Hilfe des Namens unterscheiden.
Der Name setzt sich aus den Zeichen "BX_" oder "BC_" und den letzten drei Bytes der MAC-ID zusammen. Die MAC-ID ist beim BX9000 auf der Seite und bei den BC-Controllern auf der Unterseite aufgedruckt.
Beispiel: MAC-ID: 00-01-05-00-1D-C3, dann ist der Default-Name BX_001DC3.

Sollte über den Broadcast Search kein Gerät gefunden werden, überprüfen Sie die Ethernet-Verbindung oder die Firmware des Controllers.

Ist der Busklemmen-Controller über DHCP adressiert worden, können Sie den Busklemmen-Controller über die Schaltfläche *Add Route* im Dialog *Host Name* in ihre Verbindung aufnehmen.

Ist der Busklemmen-Controller manuell oder über BootP Adressiert worden so wählen sie vergebene *IP-Adresse* an und über die Schaltfläche *Add Route* nehmen Sie die Verbindung auf.

Quittieren bei Abfrage eines Passwortes diesen Dialog einfach ohne Eingabe. Beim Busklemmen-Controller ist kein Passwort notwendig, es wird auch nicht vom den Busklemmen-Controllern unterstützt.

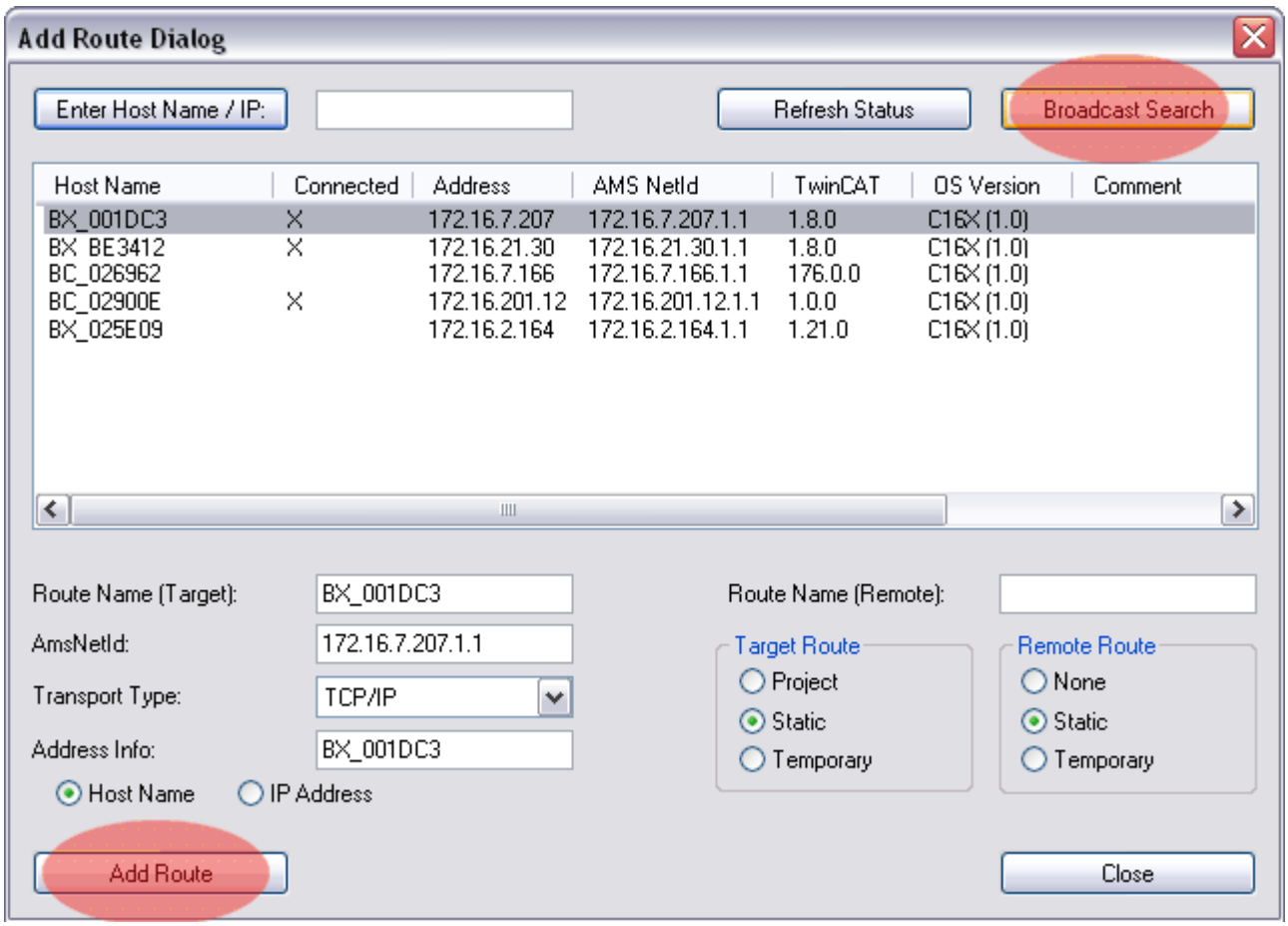


Abb. 42: Auswahl des Busklemmen-Controllers

Wählen Sie nun den Busklemmen-Controller aus, mit dem Sie sich verbinden wollen und Scannen Sie die daran angeschlossenen Geräte (SSB, K-Bus). Der Busklemmen-Controller muss sich dazu im Config Mode befinden ([Shift] [F4]).

4.3.7.2 TwinCAT PLC als Master

Ein PC oder ein CX kann als Master für den BC9x20/BC9050/BX9000 arbeiten. Er pollt dabei die SPS-Variablen in Abhängigkeit der eingestellten Taskzeit. Damit ist es möglich Daten von einem Leitsystem zum BC9x20/BC9050/BX9000 zu senden wie auch zu empfangen. Folgende Kommunikationsmöglichkeiten sind erlaubt:

- ADS TCP Zyklisch oder Azyklisch aus der SPS mit den ADS READ und WRITE Bausteinen
- ADS UDP Zyklisch
- ModbusTCP (mit TC-Modbus-Client)
- Busklemmen-Controller sendet oder liest Daten aus der SPS mit Hilfe der ADS READ und WRITE Bausteinen

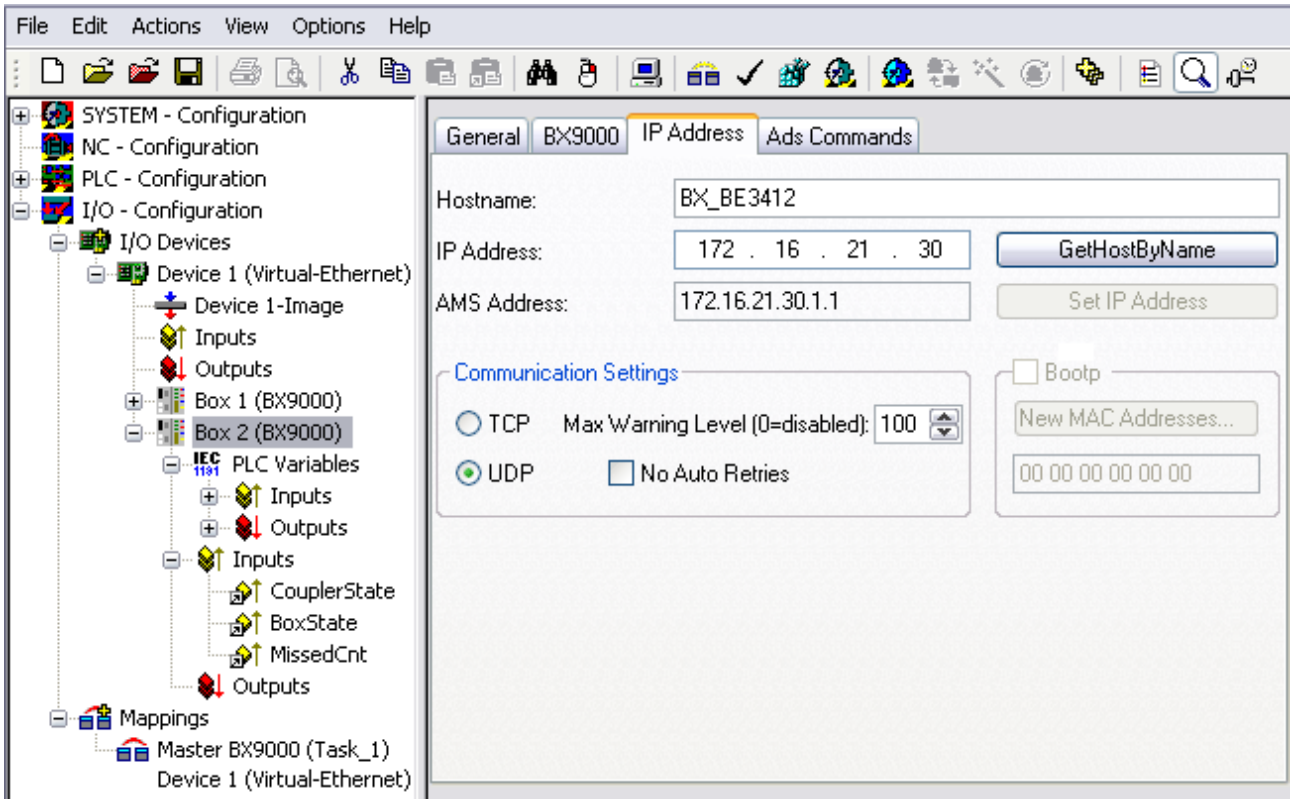


Abb. 43: IP-Adresse

GetHostByName: Diese Funktion erlaubt die Suche der IP-Adresse über den Namen (Funktioniert nur wenn der Busklemmen-Controller seine IP-Adresse über DHCP erhalten hat)

PLC Variablen: Daten für die zyklische Datenverbindung. Diese müssen mit mindestens einer Task verbunden werden. Maximal 256 Worte Ein- oder Ausgänge. Sollten mehr Daten für die Übertragung notwendig sein können diese Azyklisch über den Merkerbereich des Busklemmen-Controller gelesen oder geschrieben werden.

Diagnose Daten:

Coupler State: Sollte immer null sein. "1" wird gesetzt wenn zum Beispiel der K-Bus einen Fehler meldet
BoxState: siehe Comment im Dialog
MissedCnt: Sollte möglichst nicht hochzählen. Da das TwinCAT in Echtzeit läuft, TCP oder UDP aber keine Echtzeit Protokolle sind, ist es nicht auszuschließen das unter Umständen der Zähler sich erhöht. Der Zähler zählt immer dann um eins hoch, wenn die Daten, die er bei Task-Anfang gesendet hat, beim nächsten Task-Anfang noch nicht wieder eingetroffen sind.

Die Task Zeit sollte wie folgt eingestellt werden

Bei ADS TCP Zyklisch

Messen Sie die benötigte PLC Zeit auf dem Busklemmen-Controller und addieren sie 20-30 % drauf und stellen Sie so die Task-Zeit auf dem Busklemmen-Controller ein. Nun nehmen Sie die Task-Zeit mal drei und dies entspricht dann der Task-Zeit auf ihrer Mastersteuerung. Beispiel 5 ms gemessene PLC Zeit, stellen Sie 7 ms Task-Zeit für den Busklemmen-Controller ein und 3 x 7 ms = 21 ms für die Master SPS. Bei mehreren Busklemmen-Controller, ist der mit der langsamsten PLC Zeit der entscheidende, der die Task-Zeit auf der Master-Steuerung vorgibt.

Bei ADS UDP Zyklisch

Messen Sie die benötigte PLC Zeit auf dem Busklemmen-Controller und addieren sie 20-30 % drauf und stellen Sie so die Task-Zeit auf dem Busklemmen-Controller ein. Nun nehmen Sie die Task-Zeit mal zwei und dies entspricht dann der Task-Zeit auf ihrer Mastersteuerung. Beispiel 5 ms gemessene PLC Zeit,

stellen Sie 7 ms Task-Zeit für den Busklemmen-Controller ein und 2 x 7 ms = 14 ms für die Master SPS. Bei mehreren Busklemmen-Controller, ist der mit der langsamsten PLC Zeit der entscheidende, der die Task-Zeit auf der Master-Steuerung vorgibt.

Bei ModbusTCP Zyklisch

Messen Sie die benötigte PLC Zeit auf dem Busklemmen-Controller und addieren sie 20-30 % drauf und stellen Sie so die Task-Zeit auf dem Busklemmen-Controller ein. Nun nehmen Sie die Task-Zeit mal zwei und dies entspricht dann der Task-Zeit auf ihrer Mastersteuerung. Beispiel 5 ms gemessene PLC Zeit, stellen Sie 7 ms Task-Zeit für den Busklemmen-Controller ein und 2 x 7 ms = 14 ms für die Master SPS. Bei mehreren Busklemmen-Controller, ist der mit der langsamsten PLC Zeit der entscheidende, der die Task-Zeit auf der Master-Steuerung vorgibt.

Unterschiedliche SPS-Zykluszeiten

Wenn die Busklemmen-Controller Ihrer Anlage unterschiedlich lange Zykluszeiten für ihre lokale SPS-Abarbeitung benötigen, können Sie die Zeit nach der die übergeordnete TwinCAT PLC jeden einzelnen Busklemmen-Controller abfragt individuell anpassen.

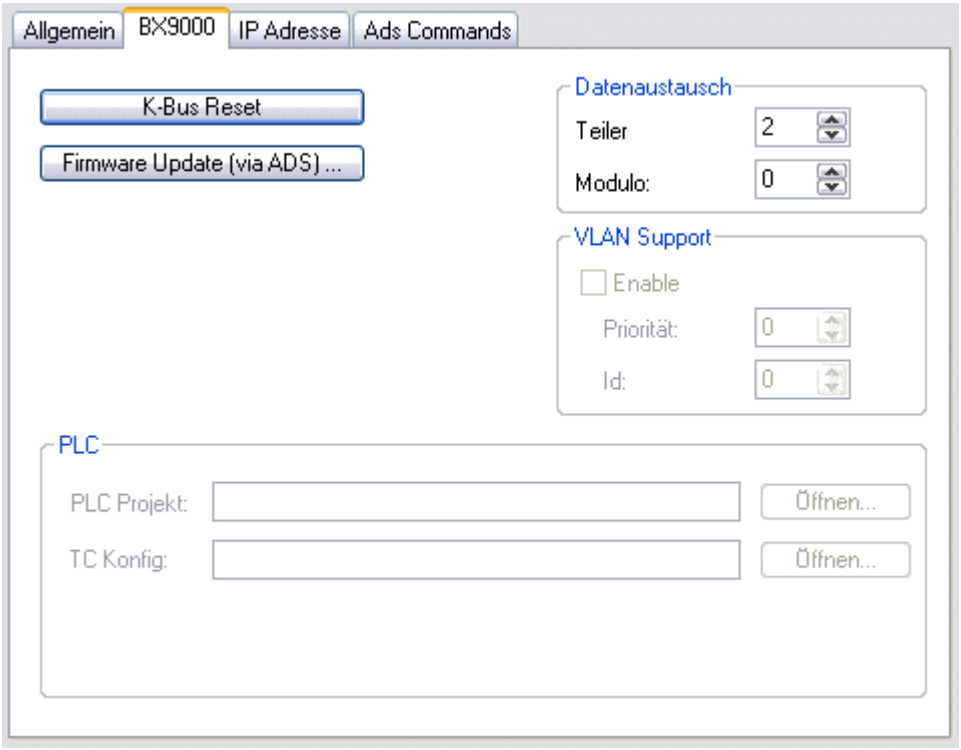


Abb. 44: Unterschiedliche SPS-Zykluszeiten

Teiler

Benutzen Sie hierfür den Teiler. Dieser nimmt als Grundlage die Zykluszeit der übergeordneten TwinCAT PLC, zum Beispiel 10 ms. Wird der Teiler nun auf 2 gestellt so wird alle 2 x 10 ms also alle 20 ms ein Telegramm zum Busklemmen-Controller gesendet.

Modulo

Mit Modulo können Sie zusätzlich einstellen, wann das übergeordnete TwinCAT PLC dies machen soll.
Beispiel:
Teiler 3, Modulo 0, bedeutet nach dem 1. Task-Zyklus und dann nach jedem 3. Task-Zyklus wird ein Telegramm versendet.
Steht der Modulo auf 1 wird erst nach dem 2. Task-Zyklus ein Telegramm versendet und dann nach jedem 3. Taskzyklus + 1.
So können Sie bei vielen Ethernet-Knoten die Anzahl der Ethernet-Pakete besser verteilen, haben eine gleichmäßigere Netzwerkauslastung und keine Spitzen in der Netzwerklast.

4.3.8 K-Bus

Busklemme und Endklemme erforderlich

Zum Betrieb eines Busklemmen-Controllers der BC- oder BX-Serie müssen an dessen K-Bus mindestens eine Busklemme mit Prozessabbild und die Endklemme gesteckt sein!

Karteireiter BX Settings

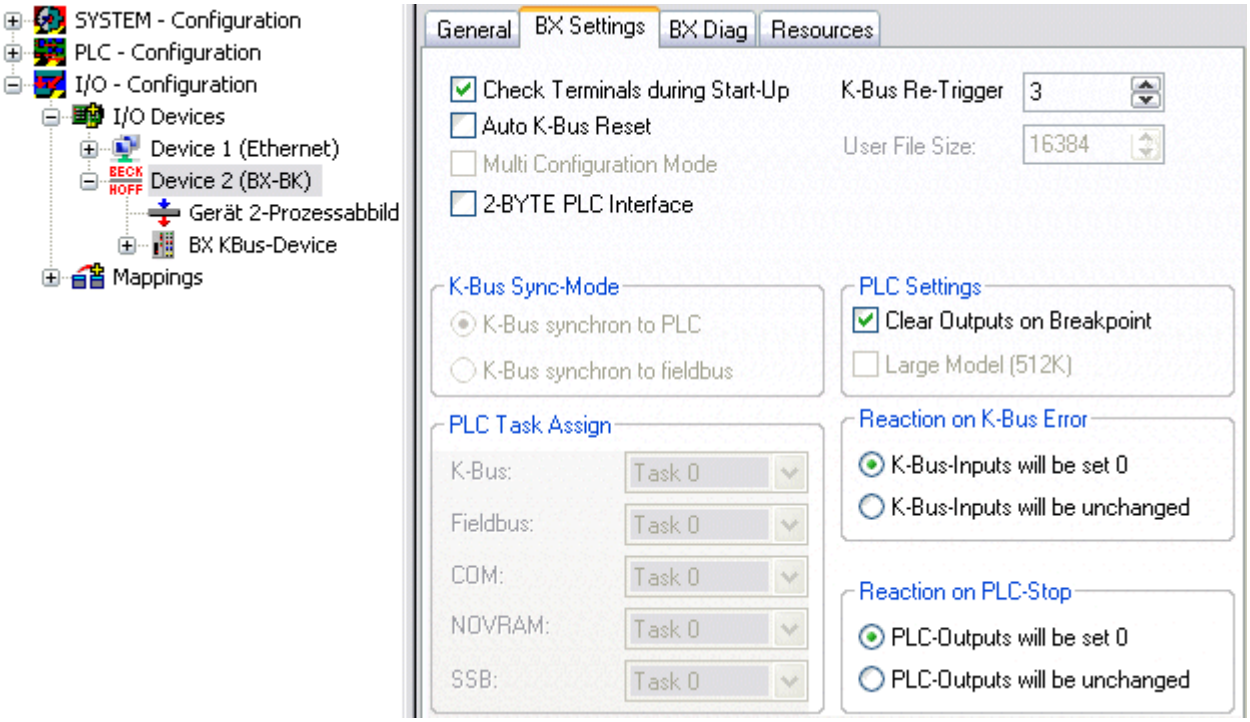


Abb. 45: Karteireiter BX Settings

Check Terminals during Start up

Beim Erzeugen eines Bootprojektes wird die aktuelle Busklemmenkonfiguration gespeichert. Beim Erneuten starten des Busklemmen-Controllers werden die angeschlossenen Busklemmen überprüft. Ist diese Option angewählt, geht der Busklemmen-Controller nicht in den Datenaustausch. Das PLC Projekt wird nicht gestartet.

Auto K-Bus Reset

Nach Behebung eines K-Bus-Fehlers geht der Busklemmen-Controller automatisch wieder in den Datenaustausch.

⚠ VORSICHT

Nach Behebung eines K-Bus Fehlers werden die Ausgänge sofort wieder aktiv!

Achten Sie darauf, dass die Ausgänge dann sofort wieder aktiv werden und analoge Ausgänge ihren programmierten Wert erhalten, wenn dies nicht in Ihrem PLC-Projekt beachtet wurde.

Clear Outputs on Breakpoint

Wenn Breakpoints im PLC Control gesetzt werden, wird der K-Bus nicht mehr bearbeitet, das heißt die Ausgänge werden in den sichern Zustand, sprich null, gesetzt.

K-Bus Sync Mode

Das Schreiben und lesen der Busklemmen kann synchron zur Task1 oder dem Feldbus stattfinden.

K-Bus Re-Trigger

Sollte das PLC Projekt oder der SSB länger den Prozessor belasten, kann der K-Bus eine Zeit nicht mehr bearbeitet werden. Das führt dazu, dass der Watchdog der Busklemmen zuschlägt und Ausgänge abfallen. Der Busklemmen-Controller ist so eingestellt, das nach 85 ms der K-Bus Watchdog nachgetriggert wird und das 3 mal. Danach würde der K-Bus Watchdog zuschlagen.
K-Bus Re-Trigger 0: 100 ms
K-Bus Re-Trigger 1: 2 x 85 ms = 170 ms
K-Bus Re-Trigger 2: 3 x 85 ms = 255 ms
K-Bus Re-Trigger 3: 4 x 85 ms = 340 ms

Reaktion auf K-Bus Fehler

Bei K-Bus Fehler werden die K-Bus Eingänge auf "0" geschrieben oder behalten Ihren letzten Stand.

Reaktion auf PLC-Stop

Wird das PLC Projekt gestoppt, kann man Einstellen, wie sich die PLC Feldbus Ausgangsdaten verhalten sollen. Im Master sind diese Daten dann Eingangsdaten. Die Daten können bei PLC Stop auf "0" geschrieben oder unverändert gelassen werden.

Karteireiter BX Diag

Anzeige der Zykluszeit für Task 1, K-Bus, Bearbeitung Feldbus und die Auslastung des SSB.

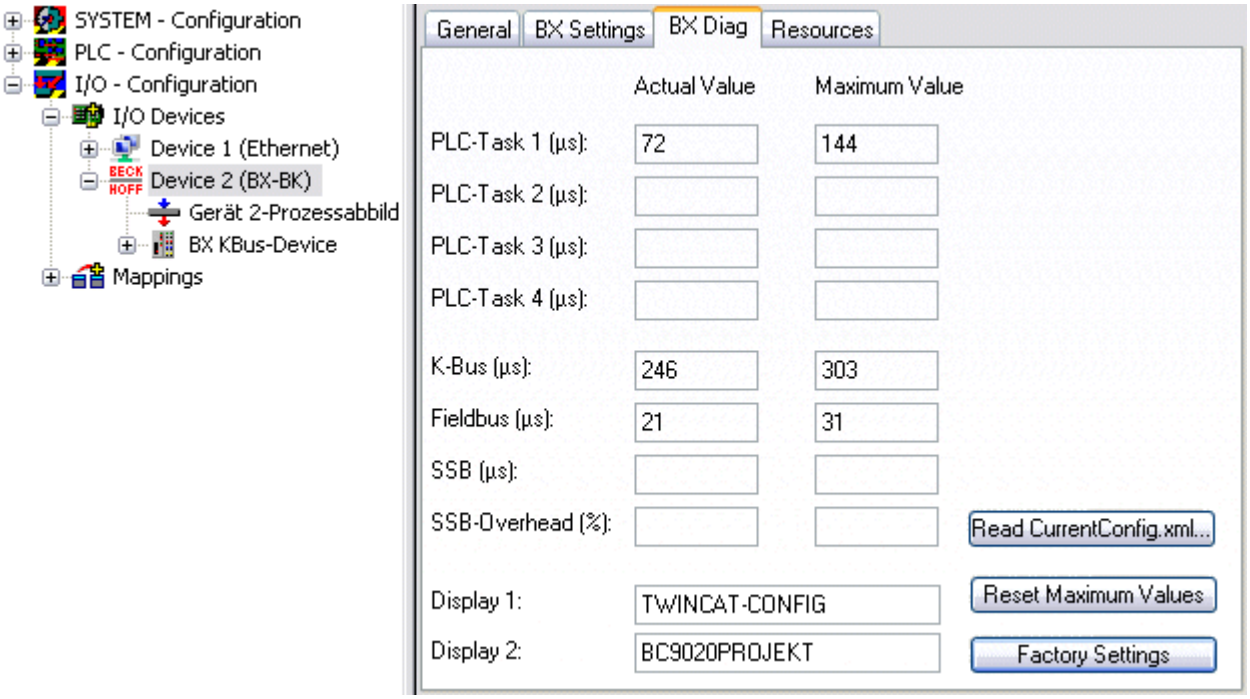
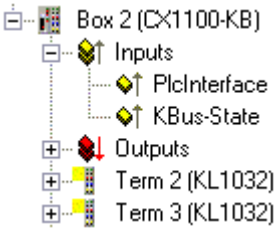


Abb. 46: Karteireiter BX Diag

Factory Settings - Der Busklemmen-Controller wird in seinen Auslieferungszustand gesetzt. Mit Restart System oder aus- und einschalten sind diese Einstellungen wieder gültig (Display DEFAULT-CONFIG).
Reset Maximum Values - löscht die Maximalen Werte

K-Bus Variablen



PLC-Interface: Wird nicht unterstützt (nur enthalten um CX oder BX Projekte zu verschieben)

K-Bus-State: Siehe Diagnose [▶ 192]

4.3.9 PLC

4.3.9.1 Einfügen eines PLC-Projektes

Für ein variables Mapping muss im System Manager die Konfiguration festgelegt sein. Hier vereinbart man die Verknüpfung zwischen PLC und der Hardware. Die Variablen können Bit, Byte, Wort oder Datenstrukturen verarbeiten und verknüpfen. Eine automatische Adressierung durch den System Manager ist möglich, sollte aber auf ihren Offset überprüft werden.

Word Alignment - Byte Orientierung

Achten Sie bei Datenstrukturen darauf, dass der Busklemmen-Controller die Daten in Word Alignment speichert und der System Manager Byte orientiert arbeitet (siehe Datenstrukturen [▶ 103])

Im PLC-Control muss ein gültiges Projekt übersetzt und gespeichert sein. Diese Daten werden als *.tpy Datei abgespeichert. Um ein PLC-Projekt einzufügen klicken Sie mit der rechten Maustaste auf *PLC-Configuration*. Wählen Sie Ihr aktuelles PLC Projekt aus.

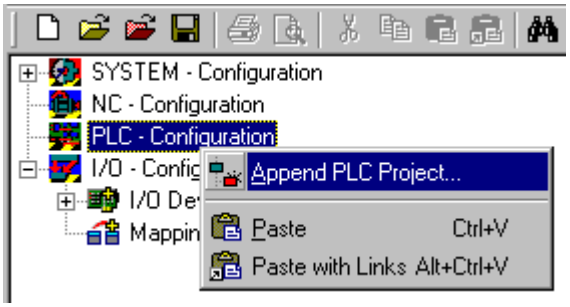


Abb. 47: Auswahl des PLC-Projekts

Verbinden Sie die PLC-Variable mit der Hardware (z. B. digitale Busklemme).

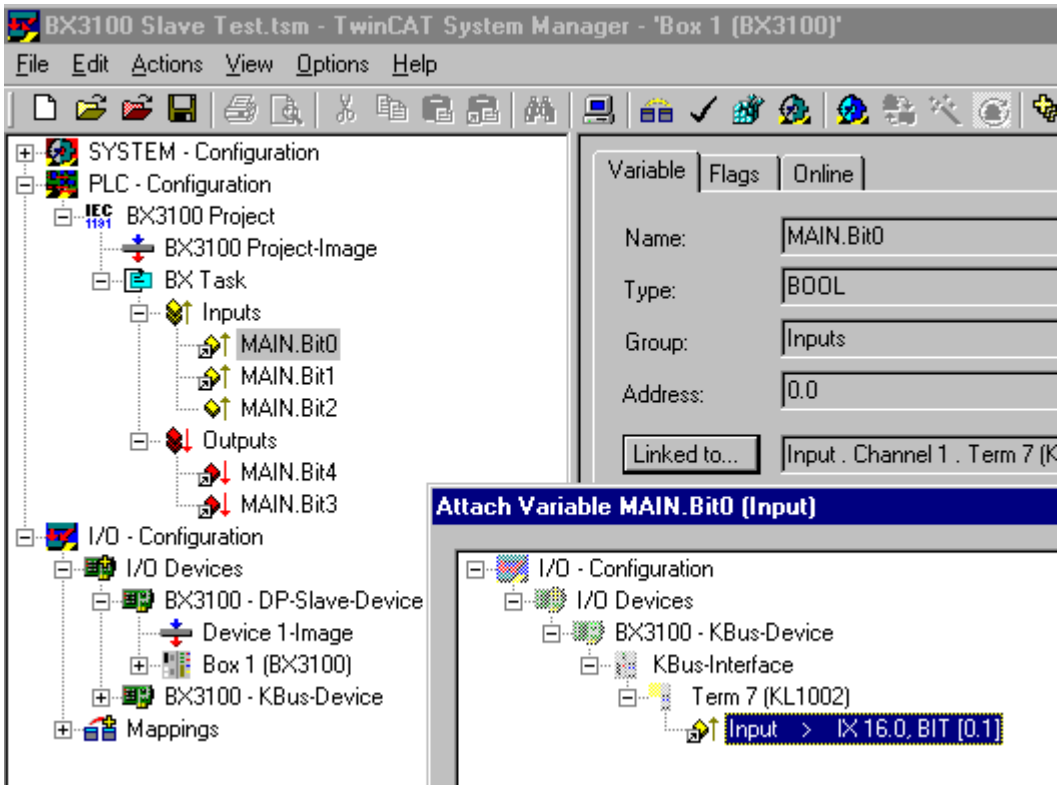


Abb. 48: Verbinden vom PLC-Variable und Hardware

Nachdem alle Verknüpfungen erstellt sind aktivieren Sie die Konfiguration *Actions/Activate Configuration* (Ctrl+Shift+F4) und Starten Sie TwinCAT *Set/Reset TwinCAT to Run Mode*. Achten Sie darauf, dass Sie das richtige Zielsystem angewählt haben (unten rechts im Fenster des System-Managers).

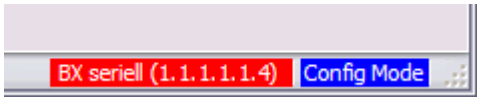


Abb. 49: Anzeige des Ziel-Systems

4.3.9.2 Messen der PLC-Zykluszeit

Die Task-Zeit wird im PLC Control eingestellt. Die Default Einstellung beträgt 20 ms.

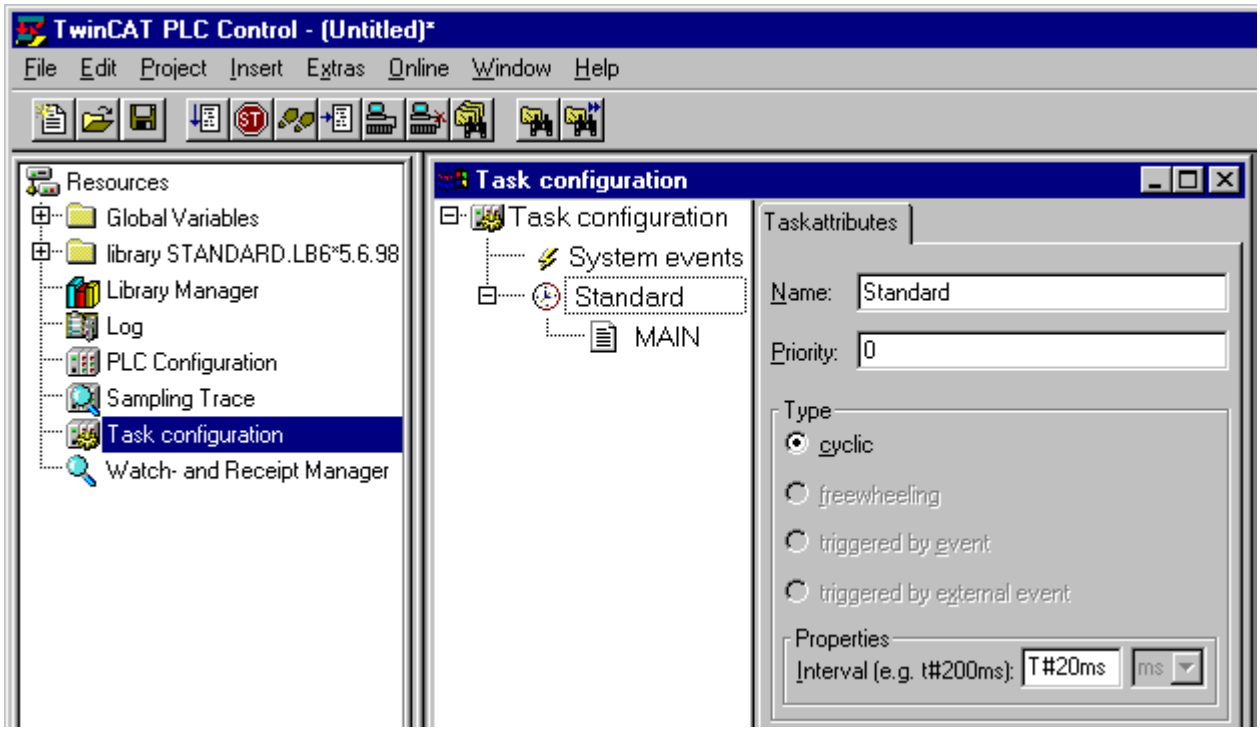


Abb. 50: Einstellen der Task-Zeit

Das PLC-Programm wird in der Default-Einstellung alle 20 ms erneut aufgerufen, solange die allgemeine Zykluszeit keiner als 20 ms ist. Um die Auslastung ihres Systems zu ermitteln kann im System Manager die Zykluszeit der PLC gemessen werden. Um einen Problemlosen Betrieb zu garantieren, muss die eingestellte Task-Zeit 20 bis 30 % höher sein als die gemessene gesamt Zykluszeit. Eine genauere Aufschlüsselung der Zykluszeit fingen Sie unter der Beschreibung K-Bus-Reiter [▶ 49]. Die gesamte Zykluszeit wird mit der TcBase Bibliothek angezeigt (siehe TcBase.lbx oder TcBaseBCxx50.lbx).

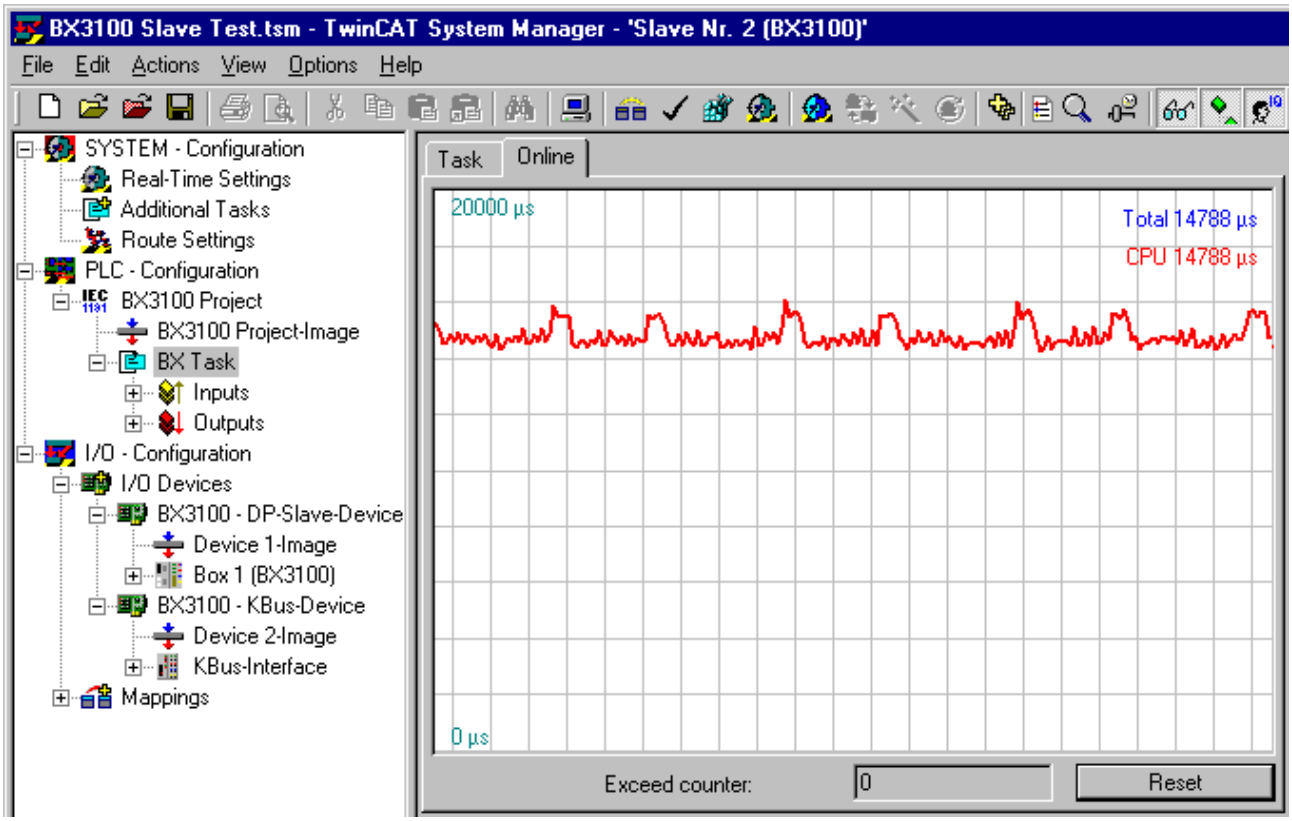


Abb. 51: Anzeige der PLC-Zykluszeit

4.3.10 SSB

4.3.10.1 SSB - Überblick

Der SSB (Smart System Bus) ist ein auf CANopen basierendes Sub-Bussystem. Es handelt sich dabei um einen CANopen-Master mit eingeschränkter Funktionalität. CANopen-Slaves können an dieser Schnittstelle angeschlossen sein, um dezentrale I/Os einzulesen oder zu beschreiben. Über ein Startup-Fenster können dem Slave parametrierungs-SDOs (Service Data Objects) geschickt werden.

Konfiguration

Der SSB wird mit Hilfe des TwinCAT System Manager projektiert (siehe TwinCAT config). Die Konfiguration wird dann per ADS auf den BX-Controller gespielt.

Technische Daten

SSB	Daten
Max. Anzahl an Slaves	8
Max. Anzahl an PDOs	32 RxPODs / 32 TxPDOs
Baudrate	10 kBaud bis 1 MBaud
Erlaubte Slave Adressen	1 bis 64

Sync-Telegramm

Das Sync-Telegramm wird in Abhängigkeit zur PLC-Task-Zeit übertragen. Ist eine Taskzeit von 20 ms eingestellt wird auch das Sync-Telegramm alle 20 ms asynchron zur PLC Laufzeit versendet. Das Sync-Telegramm wird erst erzeugt, wenn ein Teilnehmer dieses Benötigt und dieser Konfiguriert wird.
Das Sync-Telegramm wird ab der Firmware 1.12 unterstützt.

Guarding

Das Guarding wird unterstützt und wird nach einer konfigurierbaren Zeit versendet.

4.3.10.2 CANopen Verkabelung

Hinweise für die Überprüfung der CAN-Verdrahtung finden sich im Kapitel Fehlersuche / Trouble Shooting [▶ 68].

4.3.10.2.1 CAN-Topologie

CAN ist ein 2-Draht-Bussystem, an dem alle Teilnehmer parallel (d.h. mit kurzen Stichleitungen) angeschlossen werden. Der Bus muss an jedem Ende mit einem Abschlusswiderstand von 120 (bzw. 121) Ohm abgeschlossen werden, um Reflexionen zu vermeiden. Dies ist auch bei sehr kurzen Leitungslängen erforderlich!

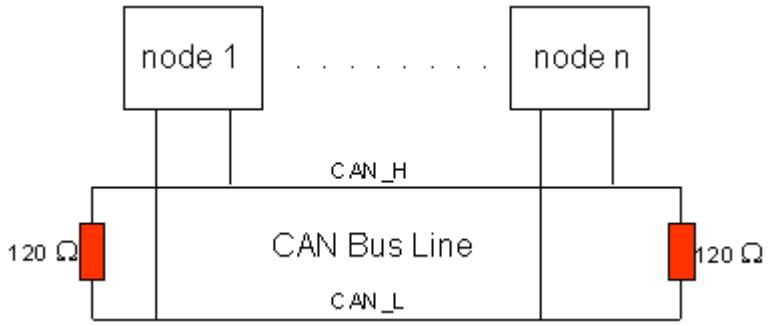


Abb. 52: Abschluss des Busses mit Abschlusswiderstand 120 Ohm

Da die CAN-Signale als Differenzpegel auf dem Bus dargestellt werden, ist die CAN-Leitung vergleichsweise unempfindlich gegen eingeprägte Störungen (EMI). Es sind jeweils beide Leitungen betroffen, somit verändert die Störung den Differenzpegel kaum.

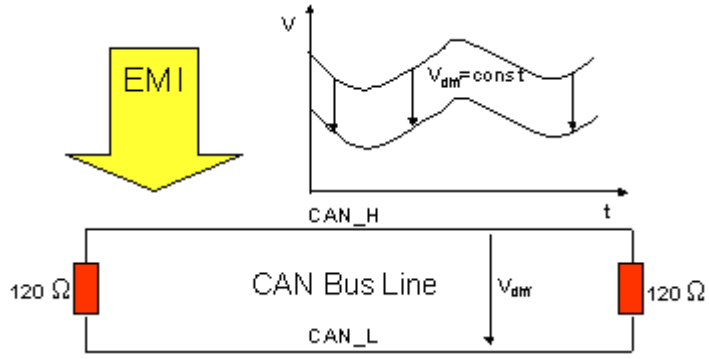


Abb. 53: Unempfindlichkeit gegen eingeprägte Störungen

4.3.10.2.2 Buslänge

Die maximale Buslänge wird bei CAN vorwiegend durch die Signallaufzeit beschränkt. Das Multi-Master-Buszugriffsverfahren (Arbitrierung) erfordert, dass die Signale quasi gleichzeitig (vor der Abtastung innerhalb einer Bitzeit) an allen Knoten anliegen. Da die Signallaufzeit in den CAN-Anschaltungen (Transceiver, Optokoppler, CAN-Controller) nahezu konstant sind, muss die Leitungslänge an die Bit-Rate angepasst werden.

Bit-Rate	Buslänge
1 MBit/s	< 20 m*
500 kBit/s	< 100 m
250 kBit/s	< 250 m
125 kBit/s	< 500 m
50 kBit/s	< 1000 m
20 kBit/s	< 2500 m
10 kBit/s	< 5000 m

*) Häufig findet man in der Literatur für CAN die Angabe 40 m bei 1 MBit/s. Dies gilt jedoch nicht für Netze mit optoentkoppelten CAN-Controllern. Die worst case Berechnung mit Optokopplern ergibt bei 1 MBit/s eine maximale Buslänge von 5 m - erfahrungsgemäß sind jedoch 20 m problemlos erreichbar.

Bei Buslängen über 1000 m kann der Einsatz von Repeatern notwendig werden.

4.3.10.2.3 Stichleitungen

Stichleitungen ("drop lines") sind nach Möglichkeit zu vermeiden, da sie grundsätzlich zu Signalreflexionen führen. Die durch Stichleitungen hervorgerufenen Reflexionen sind jedoch in der Regel unkritisch, wenn sie vor dem Abtastzeitpunkt vollständig abgeklungen sind. Bei den in den Buskopplern gewählten Bit-Timing-Einstellungen kann dies angenommen werden, wenn folgende Stichleitungslängen nicht überschritten werden:

Bit-Rate	Länge Stichleitung	gesamte Länge aller Stichleitungen
1 MBit/s	< 1 m	< 5 m
500 kBit/s	< 5 m	< 25 m
250 kBit/s	< 10 m	< 50 m
125 kBit/s	< 20 m	< 100 m
50 kBit/s	< 50 m	< 250 m

Stichleitungen dürfen nicht mit Abschlusswiderständen versehen werden.

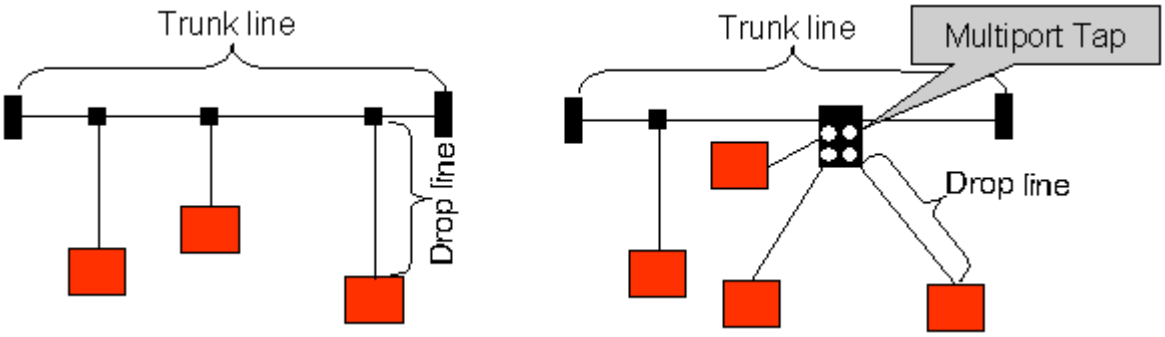


Abb. 54: Beispieltopologie Stichleitungen

4.3.10.2.4 Sternverteiler (Multiport Tap)

Beim Einsatz von passiven Verteilern ("Multiport Taps"), z. B. der Beckhoff Verteilerbox ZS5052-4500 sind kürzere Stichleitungslängen einzuhalten. Die folgende Tabelle gibt die maximalen Stichleitungslängen und die maximale Länge der Trunk Line (ohne Stichleitungen) an:

Bit-Rate	Länge Stichleitung bei Multi-port Topologie	Länge Trunk Line (ohne Stichleitungen)
1 MBit/s	< 0,3 m	< 25 m
500 kBit/s	< 1,2 m	< 66 m
250 kBit/s	< 2,4 m	< 120 m
125 kBit/s	< 4,8 m	< 310 m

4.3.10.2.5 CAN-Kabel

Für die CAN-Verdrahtung wird die Verwendung von paarig verdrillten, geschirmten Kabeln (2x2) mit einem Wellenwiderstand von 108...132 Ohm empfohlen. Wenn das Bezugspotential der CAN-Transceiver (CAN-Ground) nicht verbunden werden soll, so kann auf das zweite Adernpaar verzichtet werden (nur bei kleinen Netzausdehnungen mit gemeinsamer Speisung aller Teilnehmer empfehlenswert).

ZB5100 CAN-Kabel

Beckhoff hat ein hochwertiges CAN-Kabel mit folgenden Eigenschaften im Programm:

- 2 x 2 x 0,25 mm² (AWG 24) paarig verseilt, Kabelfarben: rot/schwarz + weiß/schwarz
- doppelt geschirmt
- Schirmgeflecht mit Beilauflitze (kann direkt auf Pin3 der 5-pol Anschlussklemme aufgelegt werden)
- flexibel (Mindestbiegeradius 35 mm bei einmaligem Biegen, 70 mm bei mehrmaligem Biegen)
- Wellenwiderstand (60 kHz): 120 Ohm
- Leiterwiderstand < 80 Ohm/km
- Mantel: PVC grau, Außendurchmesser 7,3 +/- 0,4 mm
- Gewicht: 64 kg/km.
- Bedruckt mit "Beckhoff ZB5100 CAN-BUS 2x2x0.25" und Metermarkierung (Längenangaben, alle 20cm)

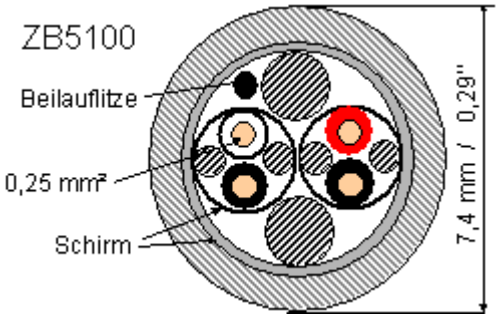


Abb. 55: Aufbau CAN-Kabel ZB5100

ZB5200 CAN/DeviceNet-Kabel

Das Kabelmaterial ZB5200 entspricht der DeviceNet Spezifikation und eignet sich ebenso für CANopen Systeme. Aus diesem Kabelmaterial sind auch die vorkonfektionierten Busleitungen ZK1052-xxxx-xxxx für die Feldbus Box Baugruppen gefertigt. Es hat folgende Spezifikation:

- 2 x 2 x 0,34 mm² (AWG 22) paarig verseilt
- doppelt geschirmt, Schirmgeflecht mit Beilauflitze
- Wellenwiderstand (1 MHz): 126 Ohm
- Leiterwiderstand 54 Ohm/km
- Mantel: PVC grau, Außendurchmesser 7,3 mm
- Bedruckt mit "InterlinkBT DeviceNet Type 572" sowie UL und CSA Ratings
- Litzenfarben entsprechen DeviceNet Spezifikation
- UL anerkanntes AWM Type 2476 Rating

- CSA AWM I/II A/B 80°C 300V FT1
- Entspricht DeviceNet "Thin Cable" Spezifikation

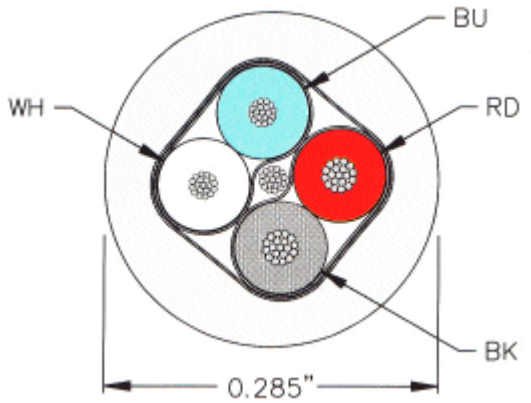


Abb. 56: Aufbau CAN-/DeviceNet-Kabel ZB5200

4.3.10.2.6 Schirmung

Der Schirm ist über das gesamte Buskabel zu verbinden und nur an einer Stelle galvanisch zu erden um Masseschleifen zu vermeiden.
Das Schirmungskonzept mit HF-Ableitung von Störungen über R/C-Glieder auf die Tragschiene geht davon aus, dass die Tragschiene entsprechend geerdet und störungsfrei ist. Sollte dies nicht der Fall sein, so kann es vorkommen, dass HF-Störpegel über die Tragschiene auf den Schirm des Buskabels übertragen werden. In diesem Fall sollte der Schirm an den Kopplern nicht aufgelegt werden - aber dennoch komplett durchverbunden sein.

Hinweise für die Überprüfung der CAN-Verdrahtung finden sich im Kapitel Fehlersuche / Trouble Shooting [▶ 68].

4.3.10.2.7 Kabelfarben

Vorschlag für die Verwendung der Beckhoff CAN-Kabel an Busklemme und Feldbus Box:

Pin BK51x0 PIN BX5100 (X510)	Pin BK5151 CX8050, CX8051, CXxxxx-B510/M510	Pin Feld-bus Box	Pin FC51xx	Funktion	Kabelfarbe ZB5100	Kabelfarbe ZB5200
1	3	3	3	CAN Ground	**schwarz** /(rot)	**schwarz**
2	2	5	2	CAN Low	**schwarz**	**blau**
3	5	1	5	Schirm	Beilauflitze	Beilauflitze
4	7	4	7	CAN high	**weiß**	**weiß**
5	9	2	9	nicht benutzt	**(rot)**	**(rot)**

4.3.10.2.8 BK5151, FC51xx, CX mit CAN Interface und EL6751: D-Sub 9polig

Die CAN Busleitung wird an die FC51x1, FC51x2 CANopen Karten und bei der EL6751 CANopen Master-/Slaveklemme über 9polige Sub-D-Buchsen mit folgender Steckerbelegung angeschlossen.

Pin	Belegung
2	CAN low (CAN-)
3	CAN Ground (intern verbunden mit Pin 6)
6	CAN Ground (intern verbunden mit Pin 3)
7	CAN high (CAN+)

Die nicht aufgeführten Pins sind nicht verbunden.
Die Tragschienenkontaktfeder und der Steckerschirm sind durchverbunden.

Hinweis: an Pin 9 darf eine Hilfsspannung bis 30 V_{DC} angeschlossen sein (wird von manchen CAN Geräten zur Versorgung der Transceiver genutzt).

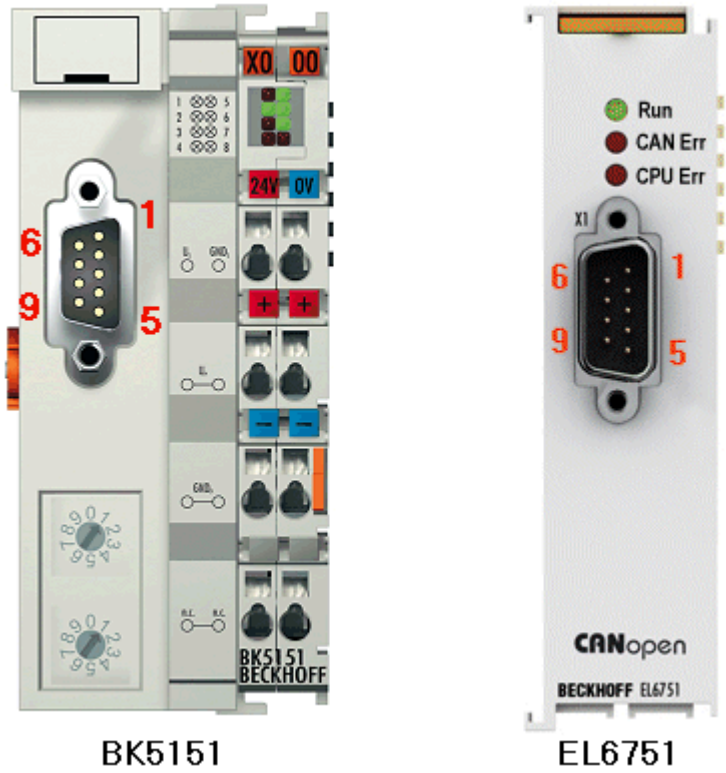


Abb. 57: Pinbelegung BK5151, EL6751

FC51x2:

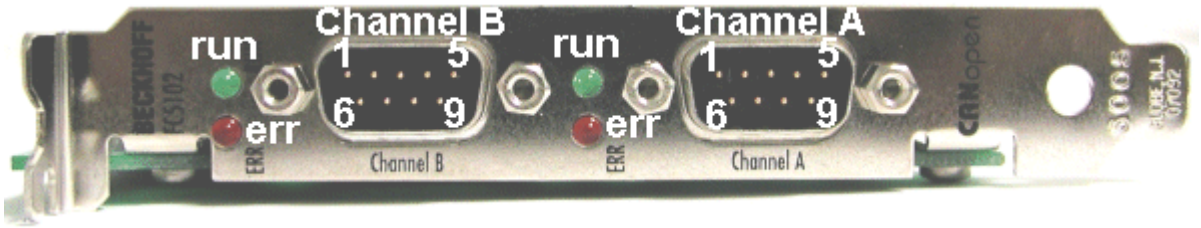


Abb. 58: FC51x2

4.3.10.2.9 BK51x0/BX5100: 5poliger Open Style Connector

Bei den BK51x0/BX5100 (X510) Buskopplern befindet sich auf der linken Seite eine abgesenkte Frontfläche mit einem 5poligen Stecker.
Hier kann die mitgelieferte CANopen- Verbindungsbuchse eingesteckt werden.

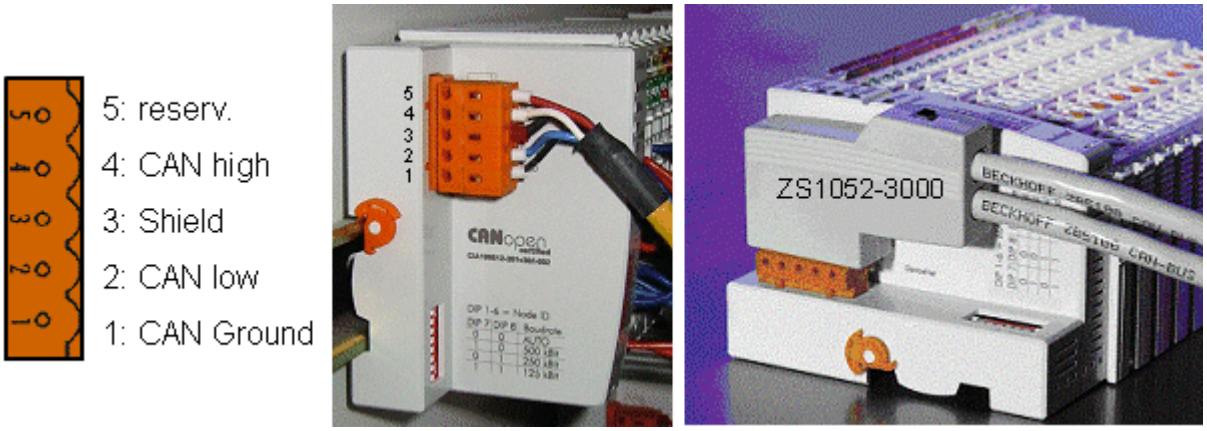


Abb. 59: Belegung Verbindungsbuchse BK51x0/BX5100

Das linke Bild zeigt die Buchse im Buskoppler BK51x0/BX5100. Pin 5 ist dabei der oberste Pin auf der Steckerleiste. Pin 5 ist nicht benutzt. Pin 4 ist die CAN-High-Leitung, Pin 2 die CAN-Low-Leitung und an Pin 3 wird der Schirm aufgelegt (ist über eine R/C-Schaltung mit der Tragschiene verbunden). Optional kann am Pin 1 CAN-GND angeschlossen werden. Wenn alle CAN-Ground Pins verbunden sind ergibt dies ein gemeinsames Bezugspotential für die CAN Transceiver im Netz. Es empfiehlt sich, CAN-GND an einer Stelle zu erden, damit das gemeinsame CAN Bezugspotential nahe beim Versorgungs-Potential liegt. Da die CANopen Buskoppler BK51X0/BX5100 über eine vollständige galvanische Trennung des Busanschlusses verfügen, kann u.U. auf die Verdrahtung von CAN-Ground verzichtet werden.

Businterface Connector ZS1052-3000

Alternativ zum mitgelieferten Stecker kann der CAN Interface Connector ZS1052-3000 eingesetzt werden. Dieser vereinfacht die Verdrahtung erheblich. Für die ankommenden und abgehenden Leitungen stehen separate Klemmen zur Verfügung, der Schirm wird durch die Zugentlastung großflächig angeschlossen. Der integrierte Abschlusswiderstand kann von außen zugeschaltet werden. Ist er eingeschaltet, so wird die abgehende Busleitung elektrisch abgetrennt - damit lassen sich Verdrahtungsfehler schnell lokalisieren, und es ist gewährleistet, dass nicht mehr als zwei Widerstände im Netz aktiv sind.

4.3.10.2.10 LC5100: Busanschluss über Federkraftklemmen

Beim Low-Cost-Koppler LC5100 wird die CAN-Leitung direkt auf die Klemmstellen 1 (CAN-H, gekennzeichnet mit C+) bzw. 5 (CAN-L, gekennzeichnet mit C-) aufgelegt. Der Schirm kann optional auf die Klemmstellen 4 bzw. 8 aufgelegt werden, diese sind über eine R/C-Schaltung mit der Tragschiene verbunden.

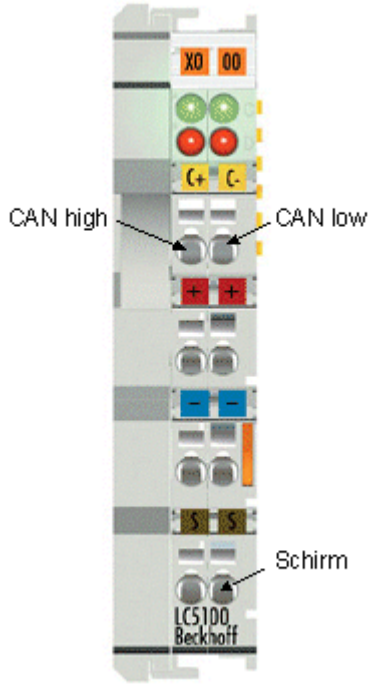


Abb. 60: LC5100

HINWEIS

Beschädigung der Geräte möglich!

Durch die nicht vorhandene galvanische Trennung kann durch unsachgemäße Verkabelung der CAN Treiber zerstört oder geschädigt werden. Verkabeln sie immer im ausgeschalteten Zustand.
Verkabeln Sie erst die Spannungsversorgung und legen sie erst dann den CAN auf. Überprüfen Sie die Verkabelung und schalten dann erst die Spannung ein.

4.3.10.2.11 Feldbus Box: M12 CAN Buchse

Bei der Feldbus Box IPxxxx-B510, IL230x-B510 und IL230x-C510 wird der Busanschluss mit 5poligen M12 Steckverbindern ausgeführt.

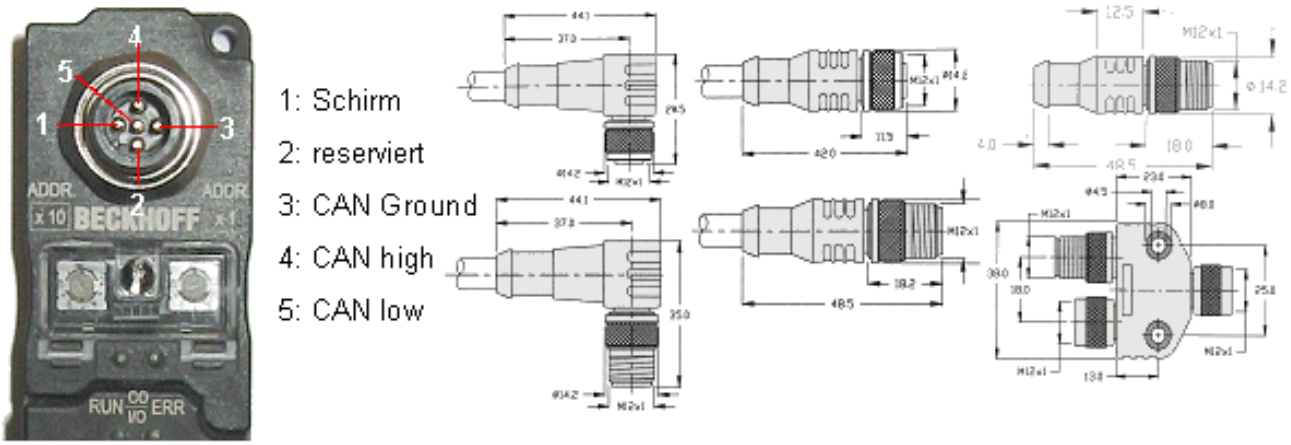


Abb. 61: Pinbelegung M12 Stecker Feldbus Box

Für das Feldbus Box System bietet Beckhoff feldkonfektionierbare Stecker, Passivverteiler, Abschlusswiderstände sowie eine große Auswahl an vorkonfektionierten Kabeln an. Details finden sich im Katalog oder unter www.beckhoff.de.

4.3.10.3 SSB - Konfiguration

Der SSB wird im System Manager konfiguriert. Öffnen Sie Ihre bestehende Konfiguration oder öffnen Sie eine neue Konfiguration in der Sie das PLC-Projekt, K-Bus und den übergeordneten Feldbus bereits konfiguriert haben.
Sie können nun unter I/O Devices (linke Maustaste) ein weiteres Gerät anfügen.

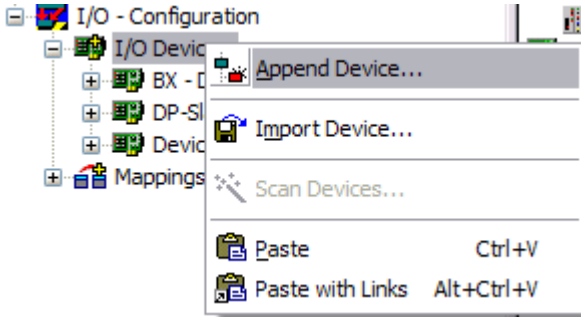


Abb. 62: Anfügen eines weiteren Gerätes

Wählen Sie den CANopen Master SSB aus und bestätigen Sie mit OK.

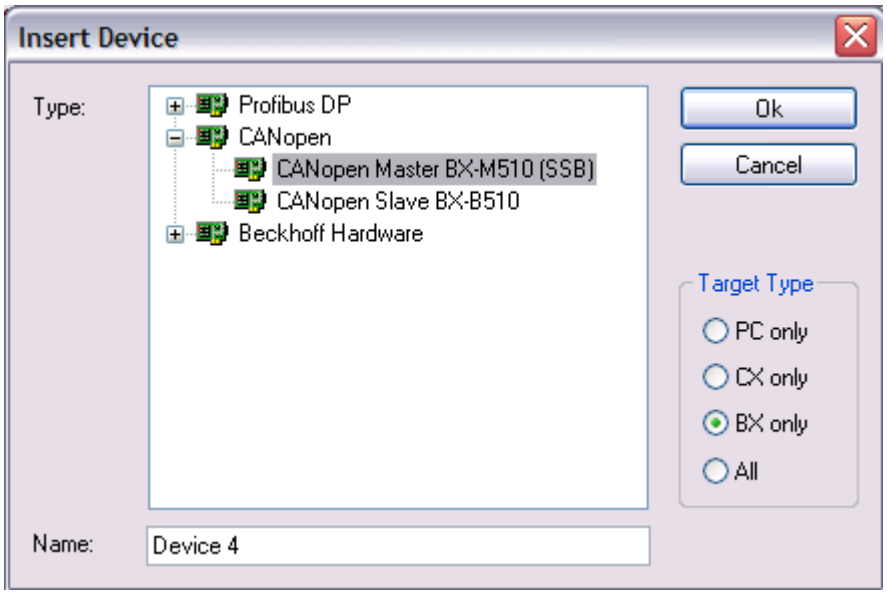


Abb. 63: Auswahl des CANopen Masters SSB

Als nächstes kann mit der linken Maustaste auf dem SSB-Device ein CANopen-Knoten angewählt werden.

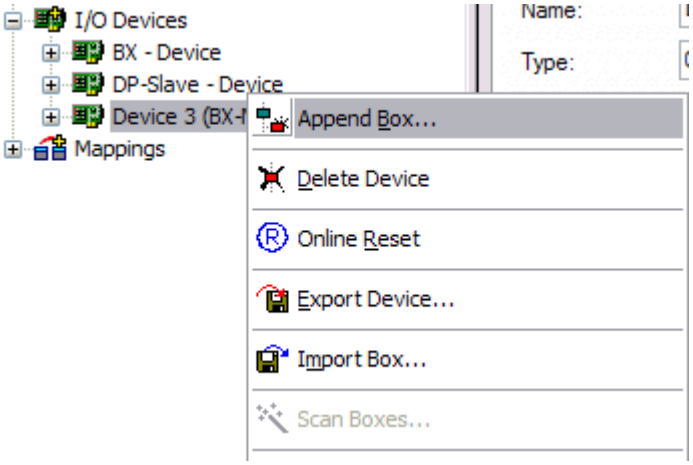


Abb. 64: Anfügen eines CANopen Geräts

Es stehen hier alle Beckhoff CAN-Knoten zur Verfügung sowie ein allgemeiner CANopen-Node für CANopen-Geräte anderer Hersteller.

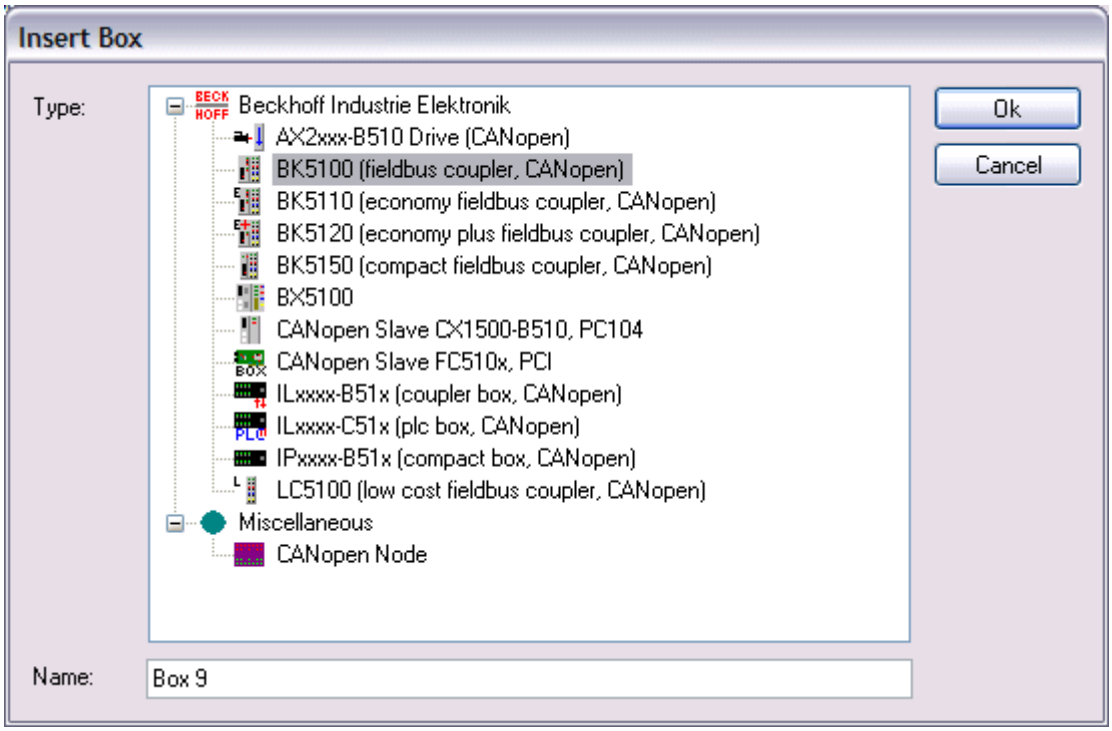


Abb. 65: Auswahl eines CANopen-Knotens

Verknüpfen Sie nun die PLC Variablen mit Ihren CAN-Knoten. Nach erfolgreicher Konfiguration laden Sie diese in den BX.

4.3.10.4 SSB - SDO-Kommunikation

CANopen SDO-Kommunikation (Service Daten Objekt) dient zum Auslesen bzw. Beschreiben beliebiger Parameter im Objektverzeichnis des CANopen Busknotens. Der SSB benutzt die SDO-Kommunikation zur Konfiguration der Kommunikationsparameter beim Aufstarten.

Download von anwendungsspezifischen Parametern beim Aufstarten

Hierzu sind die entsprechenden Parameter im System Manager bei dem entsprechenden Knoten im Karteireiter SDO einzugeben. In eckigen Klammern erscheinen die Objekte, die sich aus den Konfigurationen im Griff CAN Node ergeben. Anschließend können beliebige Objektverzeichniseinträge angefügt werden.

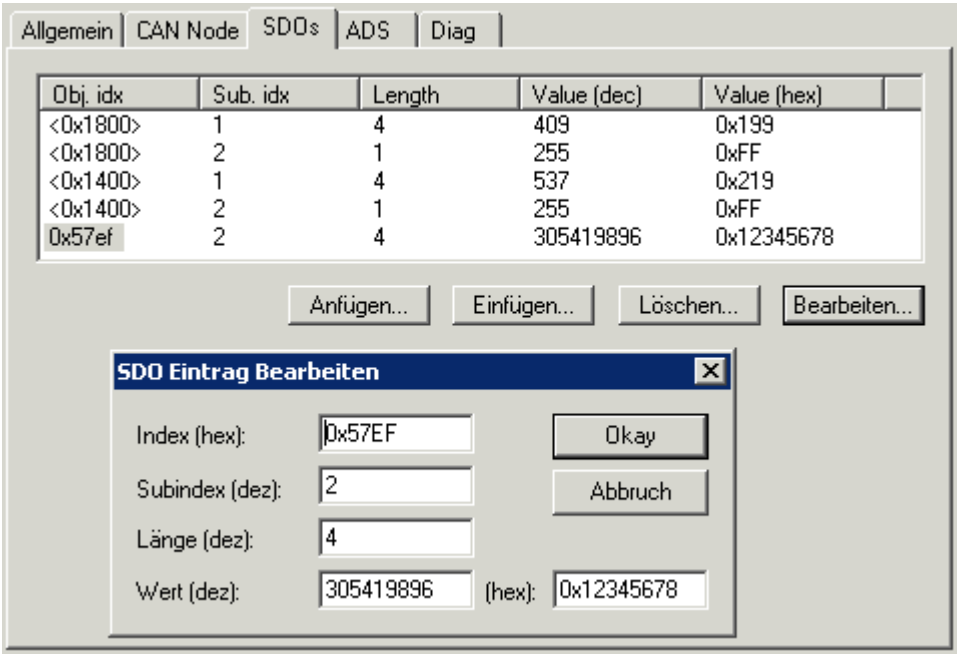


Abb. 66: Anfügen/Bearbeiten von Objektverzeichniseinträgen

Der SSB erwartet eine positive Quittierung des Parameterdownloads vom jeweiligen Busteilnehmer. Falls ein Parameter nicht geschrieben werden konnte (SDO-Abbruch durch Busteilnehmer), so versucht die Karte den entsprechenden Wert auszulesen und mit dem zu schreibenden Wert zu vergleichen - es könnte sich ja z. B. um einen schreibgeschützten Wert (Read only) handeln, der bereits korrekt im Busteilnehmer konfiguriert ist. Bei Übereinstimmung geht die Karte zum nächsten Parametereintrag.

4.3.10.5 SDO-Kommunikation aus der PLC

Für die SDO-Kommunikation aus der PLC heraus verwendet man die ADS-Bausteine. Mit diesen Bausteinen ist es möglich SDO-Telegramme zu versenden und die Antwort des Slaves zu empfangen (ADSWRITE/ADSREAD).

Eingangsparameter	Beschreibung
NETID	lokale NetId des BX oder leer lassen zum Beispiel mit ''
Port Nummer	$0x1000_{hex}$ + NodeId (Slave Nummer)
IDXGRP	SDO Index
IDXOFFS	SDO Subindex
LEN	Länge der SDO Daten (1...4)

Download BX (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207257611.zip)

Setzen einzelner oder aller Knoten in den Pre-Operational oder Operational Zustand

Mit dem ADSWRTCTL Baustein können Sie einzelne CANopen Knoten oder alle Slaves in den Pre-Operational oder Operational Zustand versetzen.

Eingangsparameter	Beschreibung
NETID	lokale NetId des BX oder leer lassen zum Beispiel mit ''
Port Nummer	$0x1000_{hex}$ + NodeId (Slave Nummer) / 153_{dez} (alle Knoten)
ADSSTATE	ADSSTATE_RUN
DEVSTATE	1 - Pre / 0 - Operational
LEN	0
SRCADDR	0

Download BX (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207259787.zip)

SSB Interface neu Starten

Mit dem ADSWRTCTL Baustein kann der SSB gestoppt und neu gestartet werden. Führen Sie als erstes ein Stopp aus und als nächstes einen Start aus.

Eingangsparameter	Beschreibung
NETID	lokale NetId des BX oder leer lassen zum Beispiel mit ''
Port Nummer	153_{dez}
ADSSTATE	ADSSTATE_STOP, ADSSTATE_RUN
DEVSTATE	0
LEN	0
SRCADDR	0

oder

Eingangsparameter	Beschreibung (ab Software Version 1.16 bei allen BX Controllern)
NETID	lokale NetId des BX oder leer lassen zum Beispiel mit ''
Port Nummer	300_{dez}
ADSSTATE	ADSSTATE_RESET
DEVSTATE	0
LEN	4
SRCADDR	ADR auf eine DWORD Variable mit der ID des SSB Device (die ID ist aus dem System Manger File zu entnehmen und ist typischerweise ein Wert von 1...3)

4.3.10.6 Emergency-Telegramme und Diagnose

Über den NodeState wird der Status des CAN-Slaves angezeigt. Das DiagFlag wird gesetzt wenn ein Emergency-Telegramm empfangen wurde. Der EmergencyCounter zählt bei jedem Emergency-Telegramm eins hoch.

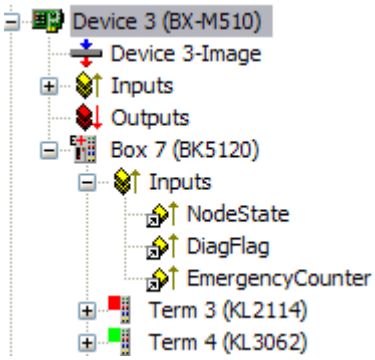


Abb. 67: NodeState, DiagFlag und EmergencyCounter

Wert NodeState	Beschreibung
0	No error
1	Node deactivated
2	Node not found
4	SDO syntax error at Start Up
5	SDO data mismatch at Start Up
8	Node start up in progress
11	SSB Bus off
12	Pre-Operational
13	Servere bus fault
14	Guarding: toggle error
20	TxPDO too short
22	Expected TxPDO is missing
23	Node is Operational but not all TxPDOs were received

ADS Port 153

Auslesen der Emergency Telegramme mit AdsRead

Eingangsparameter	Beschreibung
NETID	lokale NetId des BX
Port Nummer	153
IDXGRP	16#xxxxF180 (xxxx) NodeId, das Diag Flag wird nur beim Auslesen von mindesten 106 Byte zurückgesetzt 16#xxxxF181 (xxxx) NodeId, das Diag Flag wird sofort zurückgesetzt
IDXOFFS	Byte Offset

Beschreibung des Arrays

Offset	Bit	Wert / Beschreibung
0 - 1	Bit 0	reserviert
	Bit 1	Boot-Up-Message nicht empfangen oder fehlerhaft
	Bit 2	Emergency-Overflow
	Bit 3 - 15	reserviert
2 - 3	Bit 0 - 14	TX-PDO (i+1) empfangen
	Bit 15	alle TX-PDOs 16-n empfangen
4 - 5	Bit 0 - 4	1: falsche TX-PDO-Länge
		2: synchrone TX-PDO fehlt
		3: Node meldet PRE-OPERATIONAL
		4: Event-Timer bei einer TX-PDO abgelaufen
		5: keine Antwort beim Guarden
		6: mehrmals kein Toggeln beim Guarden
	Bit 5 - 15	zugehörige COB-ID
6	Bit 0 - 7	1: falscher Wert bei einem SDO-Upload
		2: falsche Länge bei einem SDO-Upload
		3: Abort bei einem SDO-Up-/Download
		4: falsches Datum bei einer Boot-Up-Message
		5: Timeout beim Warten auf Boot-Up-Message
7	Bit 0 - 7	2: falscher SDO-Command specifier
		3: SDO-Toggle-Bit hat sich nicht geändert
		4: SDO-Länge zu groß
		5: SDO-Abort
		6: SDO-Timeout
8 - 9	Bit 0 - 7	Index des SDO-Up/Downloads
10	Bit 0 - 7	Subindex des SDO-Up/Downloads
11	Bit 0 - 7	reserviert
12	Bit 0 - 7	errorClass des Aborts
13	Bit 0 - 7	errorCode des Aborts
14 - 15	Bit 0 - 15	additionalCode des Aborts
16 - 19		gelesener Wert (falls Offset 6 = 1)
20 - 23		erwarteter Wert (falls Offset 6 = 1)
24 - 25		Anzahl der folgenden Emergencies
26 - n		Emergencies (jeweils 8 Byte)

Download BX (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207261963.zip)

Download Beispiel System-Manager File BX (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207264139.zip)

Auslesen der Anzahl der PDO Telegramme mit AdsRead

Eingangsparameter	Beschreibung
NETID	lokale NetId des BX
Port Nummer	153
IDXGRP	16#xxxxF930 (xxxx) NodeId
IDXOFFS	0

Download BX (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207266315.zip)

Konfiguration der Node-ID erforderlich

Die Node-ID muss vor dem Aufruf des ADS-Baustein in der TwinCAT-Konfiguration konfiguriert worden sein.

Beliebige CAN Nachricht verschicken

Mit diesem ADSWRITE Befehl ist es möglich eine beliebige CAN Nachricht zu versenden.

Eingangsparameter	Beschreibung
NETID	lokale NetId des BX
Port Nummer	153
IDXGRP	16#0000F921
IDXOFFS	0
LEN	11 Bytes
SRCADDR	Pointer auf ein ARRAY von 11 Byte

Aufbau der 11 Byte CAN Daten

Byte	Beschreibung	Beispiel Node 7 SDO 0x607 Len 8 Download Request 0x2100 (Index) Sub Index 1 - Value "1"
1	COB-ID LowByte	0x06 (SDO Low Byte)
2	COB-ID HighByte	0x07 (SDO High Byte)
3	LEN (Länge)	0x08 (Len, kann hier auch 5 sein)
4	Daten[1]	0x22 (Download Request)
5	Daten[2]	0x00 (Index Low Byte)
6	Daten[3]	0x21 (Index High Byte)
7	Daten[4]	0x01 (Sub Index)
8	Daten[5]	0x01 (Value "1")
9	Daten[6]	0x00
10	Daten[7]	0x00
11	Daten[8]	0x00

4.3.10.7 CANopen Trouble Shooting

Error Frames

Fehler in der CAN-Verkabelung, der Adressvergabe und der Baud-Rateneinstellung zeigen sich u.a. durch eine erhöhte Anzahl an Error Frames: die Diagnose LEDs zeigen dann *Warning Limit wird überschritten* oder *Bus-Off-Zustand erreicht*.

Error Frames

Überschrittenes Warning Limit, Error Passive oder Bus-Off Zustand werden zunächst bei demjenigen Knoten angezeigt, der die meisten Fehler entdeckt hat. Dieser Knoten muss nicht unbedingt die Ursache für das Auftreten dieser Error Frames sein!
Wenn z. B. ein Knoten überdurchschnittlich stark zum Busverkehr beiträgt (z. B. weil er als einziger über analoge Eingänge verfügt, deren Daten in kurzen Abständen ereignisgesteuerte PDOs auslösen), so werden auch seine Telegramme mit großer Wahrscheinlichkeit zunächst gestört - entsprechend erreicht sein Fehlerzähler als erster kritische Zustände.

Node-ID / Baud Rate Einstellung

Es muss sorgfältig darauf geachtet werden, dass keine Knotenadresse doppelt vergeben ist: für jedes CAN-Datentelegramm darf es nur einen Sender geben.

Test 1

Knotenadressen überprüfen. Falls die CAN Kommunikation wenigstens zeitweise funktioniert und alle Geräte die Boot-Up-Nachricht unterstützen, so kann die Adressvergabe auch durch Aufzeichnen der Boot-Up-Nachrichten nach dem Einschalten der Geräte überprüft werden - hierdurch wird aber kein Vertauschen von Knotenadressen erkannt.

Test 2

Überprüfen, ob überall die gleiche Baud-Rate eingestellt ist. Bei Sondergeräten: Wenn Bittiming Parameter zugänglich, stimmen diese mit den CANopen-Definitionen überein (Abtastzeitpunkt, SJW, Oszillator).

Test der CAN-Verkabelung

Diese Tests nicht ausführen, wenn das Netzwerk aktiv ist: Während der Tests sollte keine Kommunikation stattfinden. Die folgenden Tests sollten in der angegebenen Reihenfolge ausgeführt werden, da manche Tests davon ausgehen, dass der vorhergehende Test erfolgreich war. In der Regel sind nicht alle Tests notwendig.

Netzwerkabschluss und Signalleitungen

Für diesen Test sollten die Knoten ausgeschaltet oder die CAN-Leitung abgesteckt sein, da die Messergebnisse sonst durch die aktiven CAN-Transceiver verfälscht werden können.

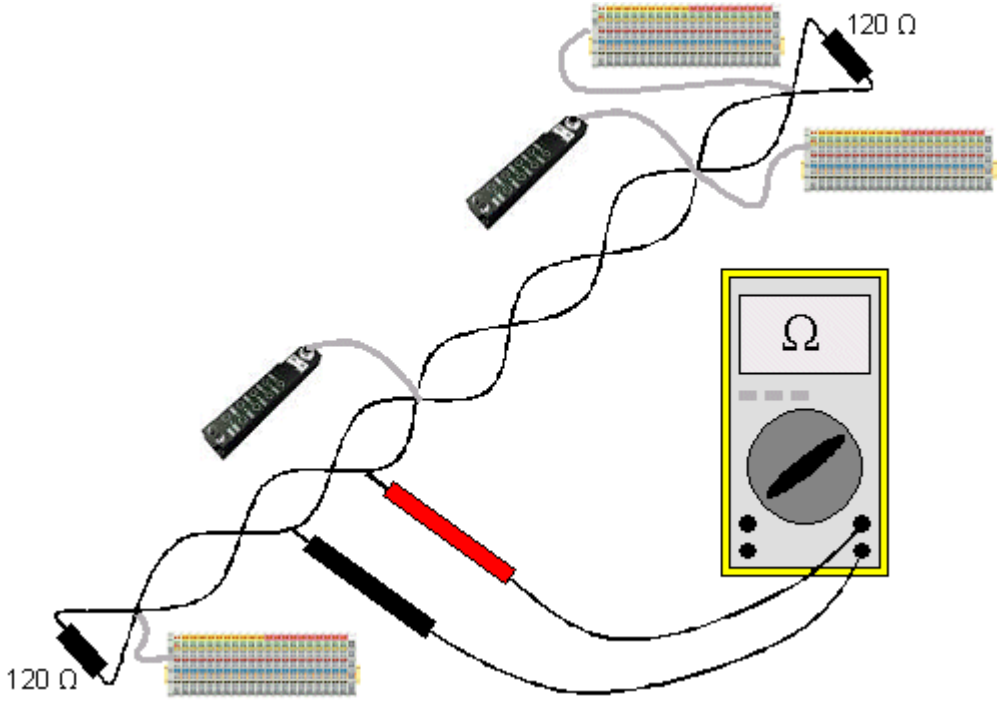


Abb. 68: Verdrahtungsplan für Testaufbau

Test 3

Widerstand zwischen CAN-high und CAN-low ermitteln - ggf. bei jedem Gerät.

Wenn der Messwert über 65 Ohm liegt, deutet dies auf fehlende Abschlusswiderstände oder den Bruch einer Signalleitung hin. Wenn der Messwert kleiner 50 Ohm ist, nach Kurzschluss zwischen CAN-Leitung, überzähligen Abschlusswiderständen oder fehlerhaften Transceivern suchen.

Test 4

Auf Kurzschluss zwischen CAN-Ground und den Signalleitungen sowie zwischen Schirm und Signalleitungen prüfen.

Test 5

Erdung von CAN-Ground und Schirm auftrennen. Auf Kurzschluss zwischen CAN-Ground und Schirm prüfen.

Topologie

Die Leitungslänge bei CAN Netzwerken hängt stark von der gewählten Baud-Rate ab. CAN toleriert dabei kurze Stichleitungen - ebenfalls in Abhängigkeit von der Baud-Rate. Die erlaubte Stichleitungslänge sollte nicht überschritten werden. Häufig wird die verlegte Leitungslänge unterschätzt - die Schätzung liegt teilweise Faktor 10 unter der tatsächlichen Länge. Deshalb empfiehlt sich folgender Test:

Test 6

Die Stichleitungslängen sowie die Busgesamtlänge nachmessen (nicht nur grob schätzen!) und mit den Topologieregeln (Baud-Ratenabhängig) vergleichen.

Schirmung und Erdung

Stromversorgung und Schirm sollten sorgfältig, einmalig und niederohmig beim Netzteil geerdet werden. Alle Verbindungsstellen, Abzweige etc. im CAN-Kabel müssen neben den Signalleitungen (und evtl. CAN-GND) auch den Schirm durchverbinden. In den Beckhoff IP20 Buskopplern wird der Schirm über ein R/C-Glied hochfrequenzmäßig geerdet.

Test 7

Mit DC-Strommessgerät (16 Amp max.) Strom zwischen Spannungsversorgungs-Masse und Schirm am vom Netzteil entfernten Ende des Netzes messen. Es sollte ein Ausgleichsstrom vorhanden sein. Wenn kein Strom vorhanden ist, so ist der Schirm nicht durchgängig verbunden oder das Netzteil ist nicht richtig geerdet. Wenn das Netzteil in der Mitte des Netzwerkes angeordnet ist, so sollte an beiden Enden gemessen werden. Dieser Test kann u.U. auch an den Stichleitungsenden durchgeführt werden.

Test 8

Den Schirm an mehreren Stellen auftrennen und den Verbindungsstrom messen. Wenn ein Stromfluss vorhanden ist, so ist der Schirm an mehreren Stellen geerdet (Erdschleife)

Potentialunterschiede

Der Schirm muss für diesen Test durchgängig sein und darf keinen Strom führen (vorher getestet).

Test 9

Spannung zwischen Schirm und Spannungsversorgungs-Erde an jedem Knoten ermitteln und notieren. Der maximale Potentialunterschied zwischen zwei beliebigen Geräten sollte kleiner als 5 Volt sein.

Fehler erkennen und lokalisieren

Am besten funktioniert in der Regel der "Low-tech-Ansatz": Teile des Netzes abhängen und beobachten, wann der Fehler verschwindet.

Aber: Dies funktioniert nicht gut bei Problemen wie zu großen Potentialunterschieden, Masseschleifen, EMV und Signalverfälschung da die Verkleinerung des Netzes häufig das Problem löst, ohne dass der „fehlende“ Teil ursächlich war. Auch die Buslast ändert sich beim Verkleinern des Netzes - damit können externe Störungen seltener CAN-Telegramme "treffen".

Die Diagnose mittels Oszilloskop führt meist nicht zum Erfolg: CAN Signale sehen auch im ungestörten Zustand teilweise recht wirr aus. Unter Umständen kann mit einem Speicheroszilloskop auf Error Frames getriggert werden - diese Art der Diagnose ist aber Messtechnik-Experten vorbehalten.

Protokollprobleme

In seltenen Fällen sind auch Protokollprobleme (z. B. fehlerhafte oder unvollständige CANopen-Implementierung, unglückliches Timing im Boot-Up etc.) Ursache von Störungen. Hier ist dann ein Mitschrieb (Trace) des Busverkehrs mit anschließender Auswertung durch CANopen Experten erforderlich - das Beckhoff Support Team kann hier helfen.
Für solch einen Trace eignet sich ein freier Kanal einer Beckhoff FC5102 CANopen PCI-Karte - die erforderliche Trace-Software stellt Beckhoff im Internet zur Verfügung. Alternativ kann selbstverständlich auch ein handelsübliches CAN Analysetool eingesetzt werden.

Protokollprobleme lassen sich vermeiden, indem auf den Einsatz von Geräten verzichtet wird, die nicht Conformance getestet sind. Der offizielle CANopen Conformance Test und das entsprechende Zertifikat sind beim CAN in Automation Verband (http://www.can-cia.de) erhältlich.

4.3.10.8 Beispiele

4.3.10.8.1 BK5120 am SSB

Notwendiges Material:

- TwinCAT 2.9 Build 953 oder Größer
- BX3100 Version 0.80 oder Größer, BX5100 Version 0.13, BX8000 Version 0.04
- 1 x KL1xx4
- 1 x KL2xx4
- 1 x KL9010
- 1 x BK5120
- 1 x KL1xx4
- 1 x KL2xx4
- 1 x KL9010
- Verkabelungsmaterial sowie Spannungsversorgung
- TwinCAT System Manager File (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207268491.zip)

   (Das System Manager File muss per ADS auf den BX-Controller geladen werden).
- BX-Programm File (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207270667.zip)

Für den Konfigurations-Download über ADS benötigt man entweder einen BECKHOFF Feldbus-Master oder eine freie serielle Schnittstelle.

4.3.10.8.2 Kommunikation von BX-Controller zu BX-Controller (über SSB)

Über den SSB können 2 oder mehrere BX-Controller untereinander Daten austauschen. Verwenden Sie im System Manager bei der Projektierung dieses Datenaustausches CAN-Schicht 2 Telegramme.

Über die COB Id wird die Kommunikation des CAN-Telegramms festgelegt. Um welchen BX-Typ es sich dabei handelt spielt keine Rolle, da der SSB auf jedem BX-Controller vorhanden ist, sich auch gleich verhält und projektiert wird.

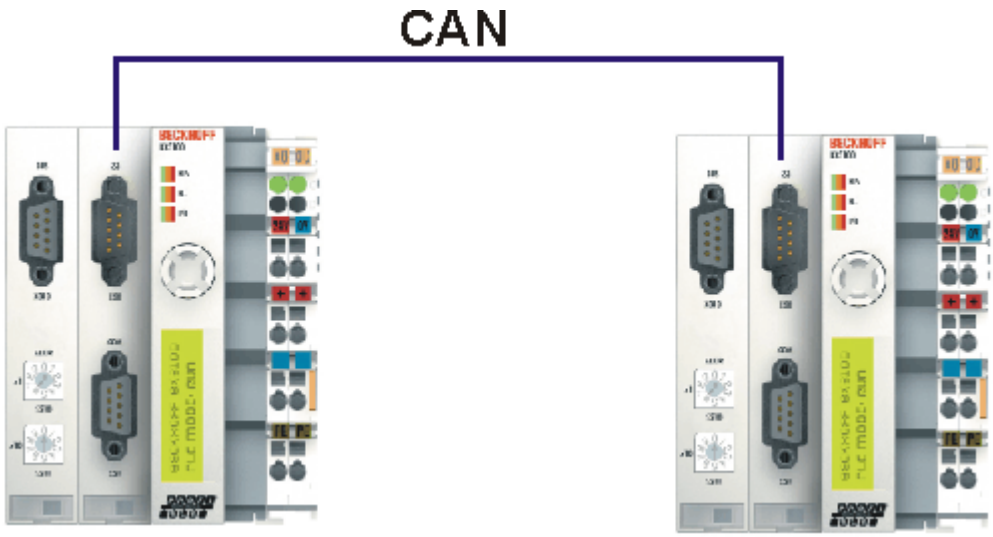


Abb. 69: Kommunikation von BX-Controller zu BX-Controller (über SSB)

Beispielkonfiguration

BX_ONE:
Node Id 2
CAN_Out AT %QB100: ARRAY[0..7] OF BYTE
COD Id 514 0x202
CAN_In AT %IB100: ARRAY[0..7] OF BYTE
COD Id 386 0x182

BX_TWO:
Node Id 2
CAN_Out AT %QB100: ARRAY[0..7] OF BYTE
COD Id 386 0x182
CAN_In AT %IB100: ARRAY[0..7] OF BYTE
COD Id 514 0x202

Beispiel Konfiguration und Programm:

Notwendiges Material

- TwinCAT 2.9 Build 959 oder Größer
- 2 x BXxx00
- Verkabelungsmaterial sowie Spannungsversorgung
- TwinCAT System Manager File BX_ONE (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207272843.zip)

- Programm File BX_ONE (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207275019.zip)

- TwinCAT System Manager File BX_TWO (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207277195.zip)

- Programm File BX_ONE (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207279371.zip)

Für den Konfigurations-Download über ADS benötigt man entweder einen BECKHOFF Master (FC310x, FC510x, FC520x) oder eine freie serielle Schnittstelle.

4.3.10.8.3 AX2000 am SSB

Abb. 70: AX2000

Notwendiges Material:

- TwinCAT 2.9 Build 953 oder Größer
- BX3100 Version 0.80 oder Größer
- 1 x KL1xx4
- 1 x KL2xx4
- 1 x KL9010
- 1 x AX2000 mit folgende Einstellungen: Slave Adresse 4, Baudrate 500 kByte
- Verkabelungsmaterial sowie Spannungsversorgung
- Beispiel Programm und Konfiguration auf dem BX-Controller
  - TwinCAT System Manager File (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207281547.zip)

    (Das System Manager File muss per ADS zum BX-Controller geladen werden).
  - BX-Programm-File (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207283723.zip)

Für den Konfigurations-Download über ADS benötigt man entweder einen BECKHOFF Feldbus-Master oder eine freie serielle Schnittstelle.

AX2000 Beschreibung

Folgende Abschnitte sind Auszüge aus der Dokumentation des AX2000 Drive Handbuch. Weitere Informationen finden Sie unter der Internet Adresse http://www.Beckhoff.de.

Hardware und Schnittstellen

Einstellen der Stationsadresse

Die Stationsadresse (Geräteadresse am CAN-Bus) des Servoverstärkers können Sie auf drei Arten einstellen:

- Mit der Tastatur in der Frontplatte (siehe Installationsanleitung AX2000)
- In der Inbetriebnahme-Software DRIVE.EXE auf der Bildschirmseite "Basiseinstellungen"
- Über die serielle Schnittstelle mit der Abfolge der ASCII-Kommandos:

ADDR nn > SAVE > COLDSTART (mit nn = Adresse)

Der Adressbereich kann mit Hilfe des ASCII – Objektes MDRV von 1..63 auf 1..127 expandiert werden.

Einstellen der Baudrate

Die CAN-Übertragungsgeschwindigkeit (Baudrate) können Sie auf drei Arten einstellen:

- Mit der Tastatur in der Frontplatte (siehe Installationsanleitung AX2000)
- In der Inbetriebnahme-Software DRIVE.EXE auf der Bildschirmseite "Basiseinstellungen"
- Über die serielle Schnittstelle mit der Abfolge der ASCII - Kommandos:
  CBAUD bb > SAVE > COLDSTART (mit bb = Baudrate in kB)

Mögliche Baudraten sind 10, 20, 50, 100, 125, 250, 333, 500 (default), 666, 800 und 1000 kBaud.

CANopen Interface (X6)

Interface zum Anschluss an den CAN Bus (default 500 kBaud). Das integrierte Profil basiert auf dem Kommunikationsprofil CANopen DS301 und dem Antriebsprofil DSP402. Im Zusammenhang mit dem Lageregler werden u.a. folgende Funktionen bereitgestellt:

Tippen mit variabler Geschwindigkeit, Referenzfahren, Fahrauftrag starten, Direktfahrauftrag starten, digitale Sollwertvorgabe, Datentransferfunktionen und viele andere.
Detaillierte Informationen finden Sie im CANopen-Handbuch. Die Schnittstelle ist über Optokoppler galvanisch getrennt und liegt auf dem gleichen Potential wie das RS232-Interface. Die analogen Sollwerteingänge sind weiterhin nutzbar. Mit der optionalen Erweiterungskarte -2CAN- werden die beiden Schnittstellen RS232 und CAN, die denselben Stecker X6 belegen, auf zwei Stecker verteilt.

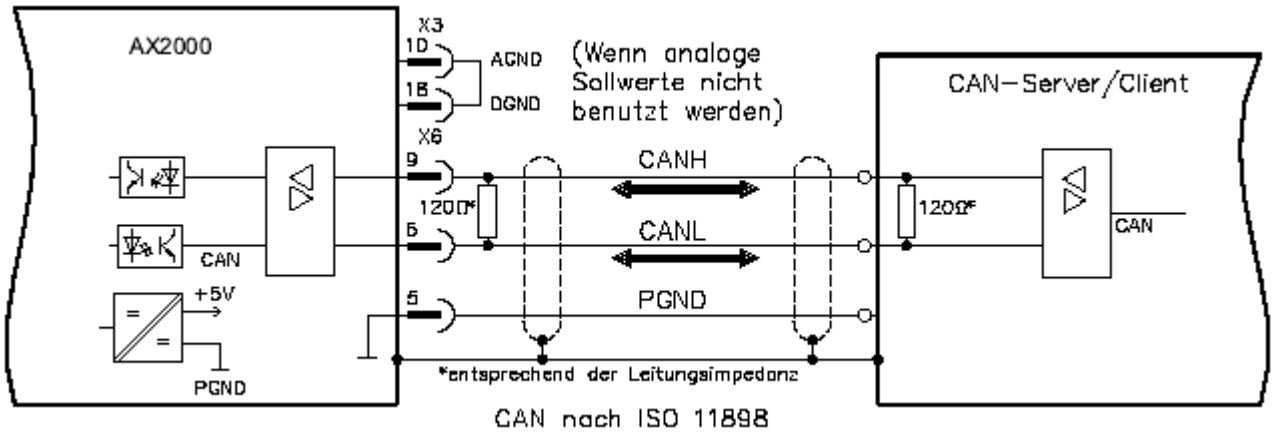


Abb. 71: CANopen Interface (X6)

4.3.10.8.4 Cimrex-Panel am SSB des BX-Controllers

Die CAN-Schnittstelle des BX-Controllers kann auch dazu benutzt werden, ein Bedien-Panel anzuschließen. In diesem Beispiel wird ein Panel der Firma Beijers angeschlossen. Weitere Informationen zu dem Panel finden Sie unter http://www.beijerelectronics.de.

Abb. 72: Cimrex-Panel am SSB des BX-Controllers

Benötigte Komponenten

- 1 x BX3100
- Einige Busklemmen für den K-Bus (hier 3 x KL2114, kann aber in der System-Manager-Datei angepasst werden)
- 1 x Cimrex 41
- 1 x CAB 15 CAN Adapter
- Beispielprogramm in ST für den BX: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207285899.zip)
- Beispielkonfiguration für den BX: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207288075.zip)
- Beispiel für Cimrex 41: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207290251.zip)
  - Baudrate 500 kBaud
  - CAN Slave Adresse 10

4.3.10.8.5 IclA-Drive am SSB

Abb. 73: IclA-Drive am SSB

Notwendiges Material:

- TwinCAT 2.9 Build 953 oder Größer
- BX3100 Version 0.80 oder Größer
- 1 x KL1xx4
- 1 x KL2xx4
- 1 x KL9010
- 1 x IclA D065 Folgende Einstellungen Slave Adresse 10, Baudrate 500 kByte (Achtung: Dies sind nicht die Default Parameter des Antriebs)
- Verkabelungsmaterial sowie Spannungsversorgung

Für den Konfigurations-Download über ADS benötigt man entweder einen BECKHOFF Feldbus-Master oder eine freie serielle Schnittstelle.

Umkonfigurierungsbeispiel für TwinCAT mit der CANopen-Masterkarte FC510x

Um den Antrieb umzustellen, kann dies mit folgendem Beispiel erfolgen.

- TwinCAT-System-Manager-File (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207292427.zip)

- TwinCAT-PLC-File (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207294603.zip)

Beispiel Programm und Konfiguration auf dem BX-Controller

- TwinCAT-System-Manager-File (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207296779.zip)

  (Das System-Manager-File muss per ADS zum BX-Controller geladen werden).

- BX-Programm-File (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207298955.zip)

IclA D065 - Beschreibung

Folgende Abschnitte sind Auszüge aus der Dokumentation des IclA-Drive-Handbuchs. Diese wurden uns von der Firma SIG Positec Automation GmbH für die Beschreibung der grundliegenden Parameter zur Verfügung gestellt. Weitere Informationen finden Sie unter der Internet Adresse http://www.sig-positec.de.

Hardware und Schnittstellen

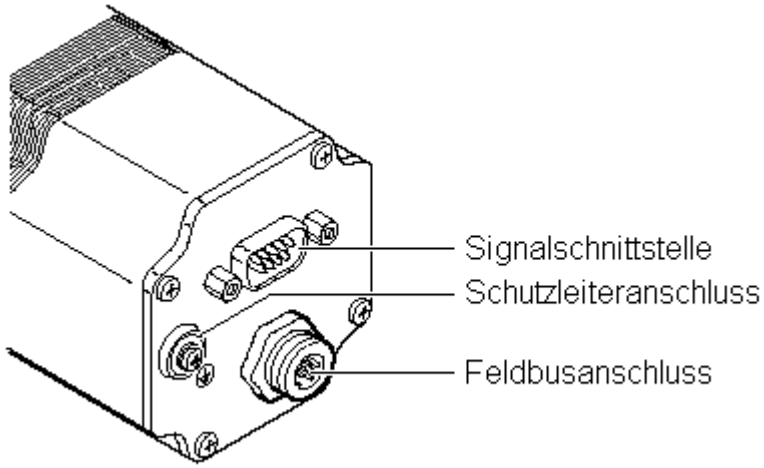


Abb. 74: Anschlüsse des IclA-Drives

- Signalschnittstelle für
  - Versorgungsspannung
  - Steuersignale für Manuellbetrieb
  - Anschluss für Not-Aus-Signal
- Schutzleiteranschluss für die Erdung über PE-Sammelschiene
- Feldbusanschluss für den Anschluss des Feldbuskabels.

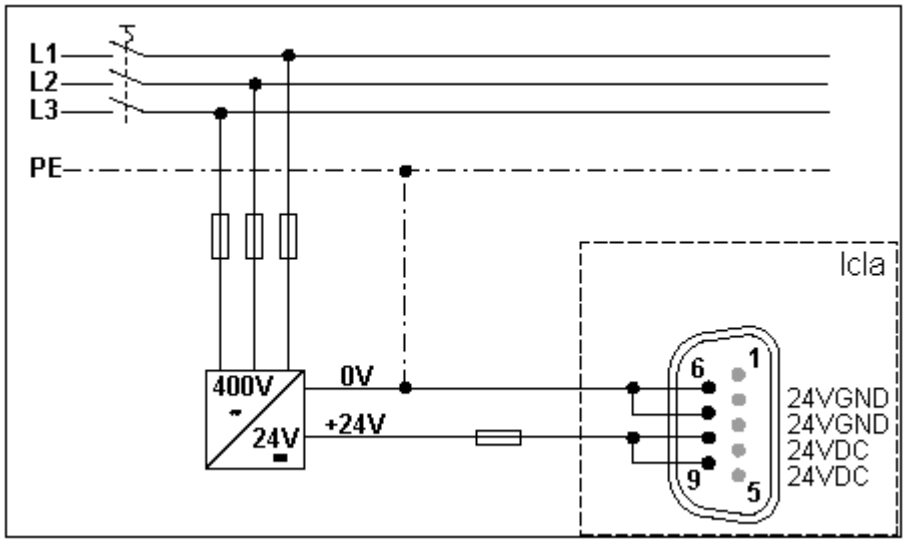


Abb. 75: Signalschnittstelle

Wird die Not-Aus-Funktion nicht benötigt, verbinden Sie Pin 2 mit Pin 8 oder 9 (24 V_{DC}).

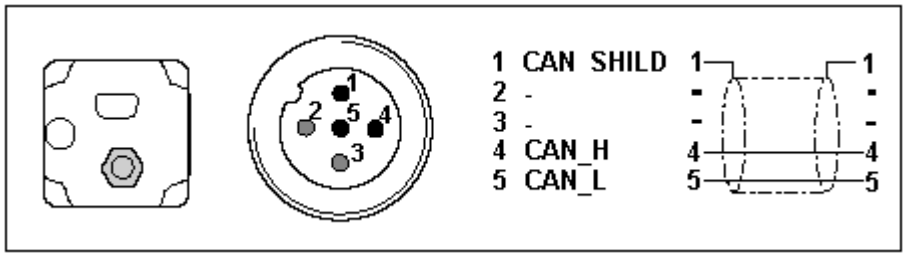


Abb. 76: Feldbusanschluss

Control-Wort 0x6040

Das Objekt stellt das Steuerungswort des Geräts dar. Mit dem Control-Wort werden mehrere Steuerungsaufgaben erfüllt:

- Wechsel zwischen den verschiedenen Betriebszuständen. Die möglichen Zustände und Übergänge finden Sie unter dem Index-Schlagwort „Zustandsmaschine". Relevant für einen Zustandswechsel sind die Bits 0 bis 3 und Bit 7.
- Starten und Unterbrechen von betriebsartenspezifischen Funktionen, z. B. Starten eines Fahrauftrags über das Bit 4. Die Bits 4 bis 6 werden für betriebsartenspezifische Einstellungen benutzt. Einzelheiten finden Sie unter den Schlagworten „Betriebsart, starten", „Betriebsart, überwachen" und bei der Beschreibung der jeweiligen Betriebsarten in den Kapiteln „Manuellbetrieb" und „Positionierbetrieb".
- Stoppen des Positionierantriebs aus einem laufenden Fahrbetrieb. Zum Anhalten wird das Bit 8 „Halt" benutzt Einzelheiten finden Sie unter den Schlagworten „Betriebsart, starten" und „Betriebsart, überwachen".

Objektbeschreibung	Wertebeschreibung
Index	6040h
Objektname	Control-Wort
Datentyp	Integned16
Subindex	00h, Control-Wort
Zugriff	read-write
PDO-Mapping	R_PDO

Bit	Bezeichnung	Bedeutung
11..15	Manufacturer specific	nicht benutzt
9, 10	-	reserviert
8	Halt	Motor stoppen
7	Reset fault	Fehler rücksetzen
4..6	-	betriebsartenabhängig,
3	Enable operation	Betriebsart ausführen
2	Quick Stop (low aktiv)	Abbremsen mit Quick Stop-Rampe
1	Disable voltage (low aktiv)	Spannung ausschalten
0	Switch on	betriebsbereit schalten

Statusword 0x6041

Das Objekt beschreibt den aktuellen Betriebszustand des Geräts. Mit dem Status-Wort führen Sie folgende Überwachungsfunktionen durch:

- Prüfen des Betriebszustands der Positioniersteuerung. Dazu sind die Bits 0 bis 3, 5 und 6 relevant.
- Das Bit 4 zeigt an, ob die Endstufe bereit ist, einen Fahrauftrag zu bearbeiten.
- Die Bits 7 bis 15 werden zur Überwachung des Fahrbetriebs und zur Statusüberwachung gerätespezifischer Zustände benutzt.

Einzelheiten zur Überwachung des Fahrbetriebs finden Sie unter den Schlagworten „Betriebsart, starten", „Betriebsart, überwachen" und bei der Beschreibung der jeweiligen Betriebsarten in den Kapiteln „Manuellbetrieb" und „Positionierbetrieb". Die Bits zur Statusüberwachung des Geräts sind im Kapitel „Diagnose und Fehlerbehebung" beschrieben.
Das Steuerungswort ist in den ersten beiden Byte der R_PDOs abgebildet.

Objektbeschreibung	Wertebeschreibung
Index	6041h
Objektname	Status-Wort
Datentyp	Unsigned16
Subindex	00h, Status-Wort
Zugriff	read-only
PDO-Mapping	T_PDO

Bit	Bezeichnung	Bedeutung
15	Out of security area	Sicherheitsbereich verlassen 0->1: Endschalterposition S0 oder S1überfahren
14	Out of drive area	Fahrbereich verlassen 0->1: Endschalterposition D0 oder D1überfahren
12..13	-	betriebsartenabhängige Bedeutung
11	Internal limit active	Arbeitsbereich verlassen
10	Target reached	Ziel erreicht 1->0: Neue Zielposition übergeben 0->1: Angeforderte Zielposition erreicht oder Motorstillstand nach Halt-Anforderung
9	Remote	0: Manuellbetrieb 1: kein Manuellbetrieb
8	Right out of drive area	Nur gültig, wenn Bit 11 = 1 - 0: Endschalterposition W1 überfahren - 1: Endschalterposition W0 überfahren
7	Warning	Warnung
6	Switch on disabled	nicht betriebsbereit
5	Quick Stop	Quick Stop aktiv
4	Voltage disabled	Spannung ausgeschaltet
3	Fault	Fehler aufgetreten
2	Operation enabled	Betriebsart aktiviert
1	Switched on	betriebsbereit
0	Ready to switch on	einschaltbereit

Referenzierungsbereiche

Eine gültige Referenzierung wird über drei Endschalterzonen definiert, die im möglichen Verfahrbereich des Antriebs liegen müssen. Die Endschalter schützen Antrieb und Anlage vor einer Beschädigung.

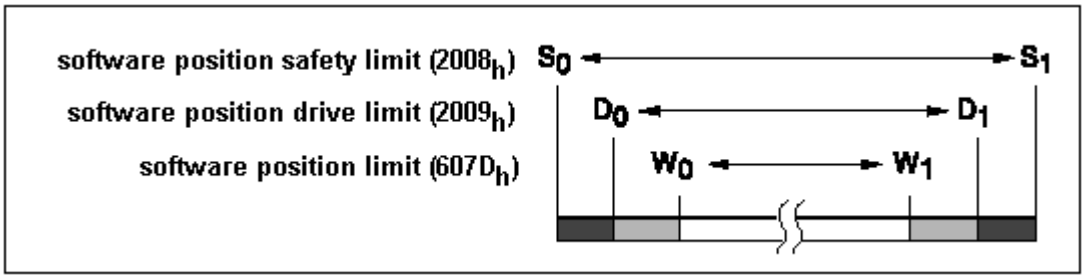


Abb. 77: Referenzierungsbereiche

- Arbeitsbereich W0 - W1 für den Positionierbetrieb.

- Fahrbereich D0 - D1. Aus den Bereichen D0 - W0 und D1 - W1 kann
- der Antrieb nur in Richtung Arbeitsbereich zurückgefahren werden.
- Sicherheitsbereich S0 - S1. Aus den Bereichen S0 - D0 und S1 - D1 kann der Antrieb nur manuell zurückbewegt werden.
- CANopen-Objekte Für die Einrichtung der Endschalter werden drei CANopen-Objekte eingesetzt, in denen die Positionswerte für die obere und untere Bereichsgrenze eingetragen werden.
- Grenzen des Arbeitsbereichs in software position limit ($607D_{hex}$)
- Grenzen des Fahrbereichs in software position drive limit (2009_{hex})
- Grenzen des Sicherheitsbereichs in software position safety limit (2008_{hex})

Bespiel für eine Referenzierung

Das folgende Listing zeigt die Eingabe der Referenzierungswerte. Die Knotenadresse des Positionierantriebs ist auf 01_{hex} eingestellt.

COB-ID	Daten	Bedeutung
601	2F 60 60 00 06	R_SDO: Umschalten in Homing Mode
581	60 60 60 00 xx	T_SDO: OK
601	23 08 20 02 0C 7B 00 00	R_SDO: max. Wert Sicherheitsbereich S_1: 7B0Ch
581	60 08 20 02 xx xx xx xx	T_SDO: OK
601	23 08 20 01 00 00 00 00	R_SDO: min. Wert Sicherheitsbereich S_0: 0000h
581	60 08 20 01 xx xx xx xx	T_SDO: OK
601	23 09 20 02 42 72 00 00	R_SDO: max. Wert Fahrbereich D_1: 7242h
581	60 09 20 02 xx xx xx xx	T_SDO: OK
601	23 09 20 01 CA 08 00 00	R_SDO: min. Wert Fahrbereich D_0: 8CAh
581	60 09 20 01 xx xx xx xx	T_SDO: OK
601	23 7D 60 02 AE 60 00 00	R_SDO: max. Wert Arbeitsbereich W_1: 60AEh
581	60 7D 60 02 xx xx xx xx	T_SDO: OK
601	23 7D 60 01 5E 1A 00 00	R_SDO: min. Wert Arbeitsbereich W_0: 1A5Eh
581	60 7D 60 01 xx xx xx xx	T_SDO: OK
601	23 10 10 03 73 61 76 65	R_SDO: Applikationsparameter speichern: "save"
581	60 10 10 03 xx xx xx xx	T_SDO: OK
601	2F 98 60 00 FF	R_SDO: Auswahl der Referenzierungsart
581	60 98 60 00 xx	T_SDO: OK
601	23 0B 20 00 BC 34 00 00	R_SDO: Maßsetzen, Istposition bis S_0: 34BCh
581	60 0B 20 00 xx xx xx xx	T_SDO: OK
601	2B 40 60 00 1F 00	R_SDO: Homing Operation Start (steigende Flanke, Bit 4)
581	60 40 60 00 xx xx	T_SDO: OK

Abb. 78: Listing der Referenzierungswerte

4.3.10.8.6 Lenze Frequenzumrichter am SSB

Abb. 79: Frequenzumrichter der Fa. Lenze

Notwendiges Material

- TwinCAT 2.9 Build 953 oder Größer
- BXxx00
- 1 x KL1xx4
- 1 x KL2xx4
- 1 x KL9010
- 1 x Lenze 8200 vector + Motor
- 1 x Lenze CANopen Interface 2175
- Verkabelungsmaterial sowie Spannungsversorgung

Für den Konfigurations-Download über ADS benötigen Sie eine BECKHOFF Feldbus-Masterkarte oder eine freie serielle Schnittstelle.

Lenze Beschreibung

Folgende Abschnitte sind Auszüge aus der Dokumentation des Lenze 2175-Handbuchs. Diese wurden uns von der Firma Lenze Drive Systems GmbH für die Beschreibung der grundlegenden Parameter zur Verfügung gestellt. Weitere Informationen finden Sie unter der Internet Adresse http://www.Lenze.com.

Erstinbetriebnahme

Stellen Sie die Spannungsversorgung für das Busmodul auf Interne Spannungsversorgung.

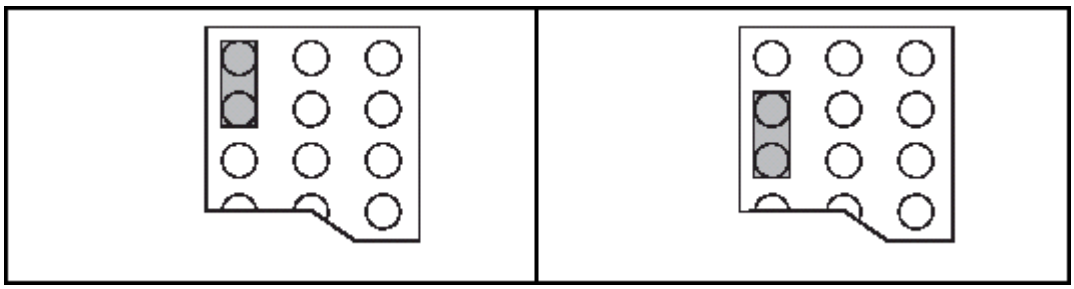


Abb. 80: Externe Spannungsversorgung - Interne Spannungsversorgung (Auslieferungszustand)

Für die CANopen-Kommunikation stellen Sie den DIP-Schalter 10 auf "ON".

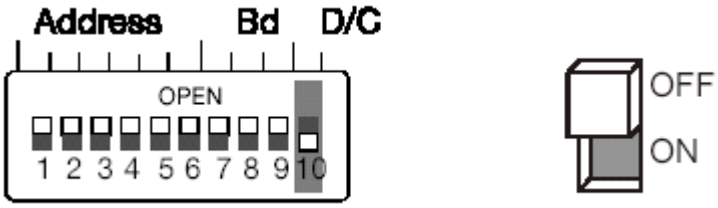


Abb. 81: DIP-Schalter

Baudrate DIP-Schalter 7-9

Übertragungrate [kBit/s]	S7	S8	S9
10	ON	ON	OFF
20	ON	OFF	ON
50	OFF	ON	ON
125	OFF	ON	OFF
250	OFF	OFF	ON
500 (default)	OFF	OFF	OFF
1000	ON	OFF	OFF

Wertigkeit der DIP-Schalter

DIP Schalter 6 hat die kleinste Wertigkeit.
Beispiel: Adresse 3 Schalter 5 und 6 auf "ON".

Freigabe des Kommunikationsmoduls

Um das Kommunikationsmodul freizuschalten muss die Bedienungsart auf 3 gestellt werden. Dies kann über den SSB erfolgen mit folgendem Eintrag:

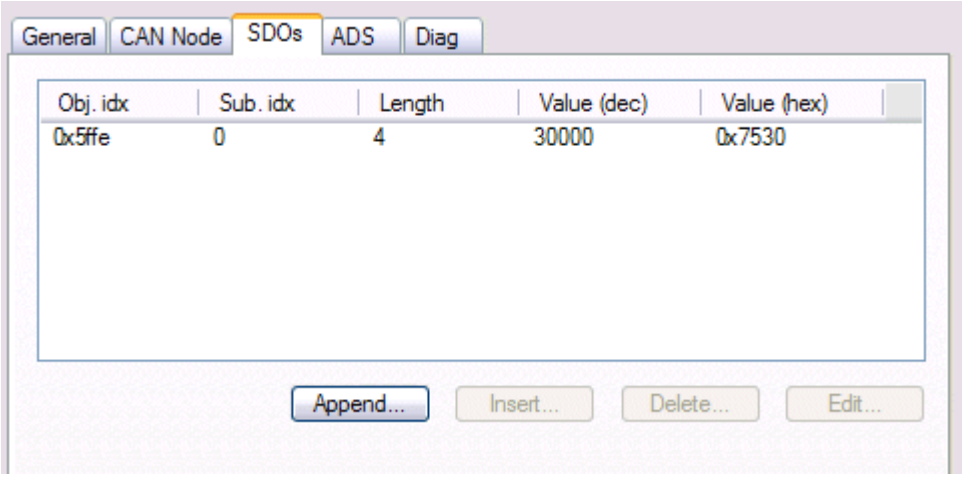
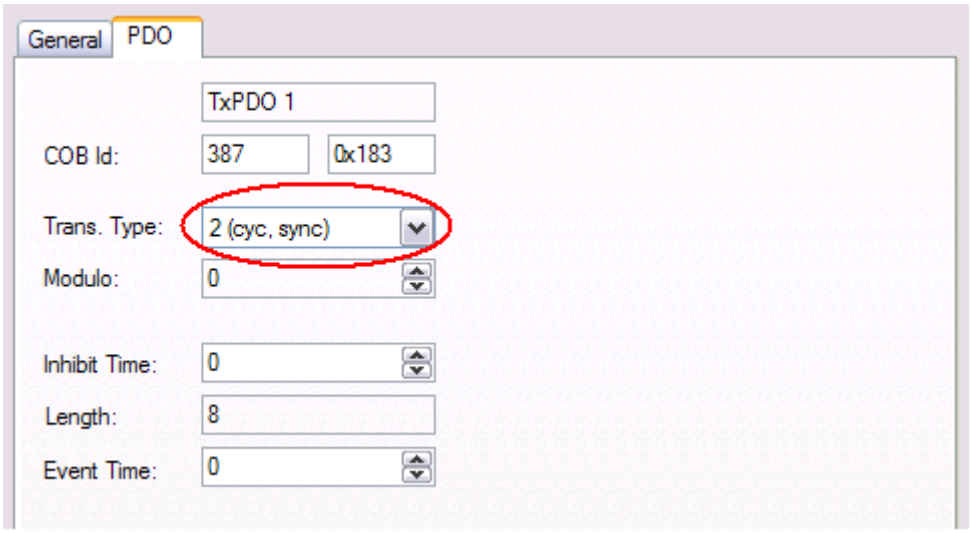


Abb. 82: Freigabe des Kommunikationsmoduls

Sync-Telegramm

In der Default-Einstellung sendet der Lenze-Drive seine Ausgangs-PDOs nur, nachdem er vom CAN-Master ein Sync-Telegramm empfangen hat. Wenn Sie den Trans. Type zum Beispiel auf 2 stellen, sendet der Lenze-Drive nach jedem 2. empfangenen Sync-Telegramm ein Ausgangs-PDO.



Beispiel Projekt

- TwinCAT-System-Manager-File: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207301131.zip)
- TwinCAT-PLC-File: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207303307.zip)

4.3.11 Echtzeit-Uhr (RTC)

Auf dem BX-Controller ist eine Echtzeit-Uhr (Real Time Clock - RTC) implementiert. Die Uhrzeit ist akkugepuffert.

Einstellen der Echtzeit-Uhr

Die einfachste Art die Uhr einzustellen ist über den System Manager. Sofern die ADS-Kommunikation steht wird die momentan aktuelle Uhrzeit auf dem BX-Controller angezeigt. Um die Uhrzeit einzustellen editieren Sie einfach die Zeit und mit der Drop Down Taste können Sie Tage, Monate und Jahre einstellen. Für die Einstellungen der Jahre klicken Sie auf die Jahresanzeige und stellen Sie das gewünschte Jahr ein. Ebenso verfahren Sie mit dem Monat. Nach dem Sie alle Einstellungen vorgenommen haben klicken Sie auf den Button *Update RTC on BX*.

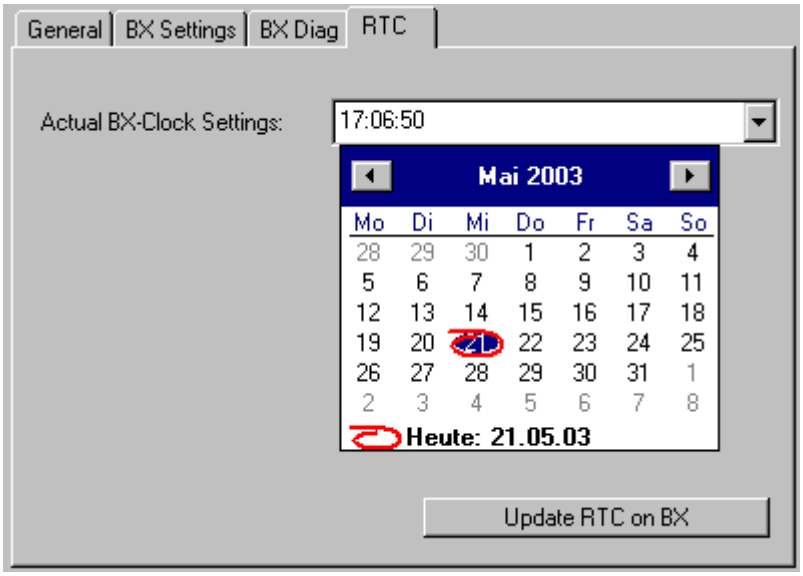


Abb. 83: Einstellen der Echtzeit-Uhr (RTC)

Lebensdauer des Akkus

Je nach Belastung des Akkus kann die Lebensdauer des Akkus variieren.

Lesen der RTC auf dem BX-Controller (siehe Beispiel [▶ 124] Programmierung\Bibliothek)

Die RTC kann per Funktionsbaustein ausgelesen werden. Notwendige Bibliotheken:

- TcSystemBX.lbx
- TCBaseBX.lbx

Schreiben der RTC auf dem BX-Controller

Die RTC kann per Funktionsbaustein eingestellt werden. Notwendige Bibliotheken:

- TcSystemBX.lbx
- TCBaseBX.lbx

Lesen der RTC per ADS

Beschreibung	Bedeutung	Wert
NETID	Ziel-Gerät	siehe System Manager "ADS"
Port	ADS Port Nummer	150_{dez}
IDXGRP	IDX Gruppe	$0x0000\_F100_{hex}$
IDXOFFS	IDX Offset	$0x0000\_0000_{hex}$
Länge	Länge der Daten	16 Byte
Variablen Typ	Art der Variabel	TIMESTRUCT

Schreiben der RTC per ADS

Beschreibung	Bedeutung	Wert
NETID	Ziel-Gerät	siehe System Manager "ADS"
Port	ADS Port Nummer	150_{dez}
IDXGRP	IDX Gruppe	$0x0000\_F100_{hex}$
IDXOFFS	IDX Offset	$0x0000\_0000_{hex}$
Länge	Länge der Daten	16 Byte
Variablen Typ	Art der Variabel	TIMESTRUCT

Einstellung über den Navigationsschalter

Siehe Menü. [▶ 87]

Technische Daten zur RTC

Genauigkeit: ca. 1 Sekunde/Tag
Speicherung der Uhrzeit: ca. 3 Monate bei aufgeladenem Akku
Lebensdauer des Akkus: ca. 10 Jahre bei 10 Zyklen pro Tag (1000 Zyklen bei einem kompletten Lade-/ und Entladezyklus)

4.3.12 COM-Port

Auf dem BX-Controller befinden sich zwei serielle Schnittstellen. Die PIN-Belegung entnehmen Sie bitte der Hardware-Beschreibung [▶ 27].

Einstellmöglichkeiten:

Beschreibung	Auswahl
Baudrate [▶ 116]	9600 Baud 19200 Baud 38400 Baud (beginnt bei der Autobaudratenerkennung) 57600 Baud 115200 Baud (nur COM 2)
Datenbits	7 8 (Default)
Parity	NONE ODD EVEN (Default)
Stoppbits	1 (Default) 2

COM 1

Die Schnittstelle COM 1 für die Kommunikation mit der KS2000 Software oder dem TwinCAT PLC Control (einloggen über die serielle Schnittstelle).

COM 2

Die Schnittstelle COM 2 (mit RS 232 oder RS 485 Physik) für die Nutzung eigener Protokolle oder der Protokoll-Bibliotheken (wie ModbusRTU, RK512, etc.) zum Anschluss andere serieller Geräte.

Bibliothek

Für die Kommunikation mit der seriellen Schnittstelle stehen Funktionsbausteine zur Verfügung.

- Dokumentation [▶ 116]
- Beispiel [▶ 121]
- Bibliothek [▶ 116]

4.4 Menü

4.4.1 BX-Menü-Settings

Um in das Menü zu wechseln, drei Sekunden den Navigationsschalter gedrückt halten. Als erstes erscheint das Verzeichnis *Menue*.

- Zwischen den Einstellungen eines Menüs, können Sie man mit den Tasten RIGHT/LEFT wechseln (es ist immer das in in der 1. Zeile angezeigte Menü aktiv).
- Drücken Sie zum Wechseln in ein Untermenü, die Taste DOWN.
- Um wieder in das Hauptmenü zu wechseln drücken Sie die Taste UP.

Im Untermenü wird in der ersten Zeile der Menüpunkt angezeigt und in der 2. Zeile die aktuelle Einstellung zu diesem Menüpunkt.

Einige Einstellungen sind nicht veränderbar *(read only)*. Diese Punkte sind rein für die Kontrolle und Information für den Anwender gedacht. Um das Menü zu beenden muss man sich im Hauptmenü befinden und dann den Navigationsschalter drei Sekunden gedrückt halten.

Bevor Einstellungen geändert werden können, muss Passwort gesetzt werden. Das Passwort bleibt auch beim Firmware-Update oder beim zurücksetzen auf die Herstellerkonfiguration gespeichert. Sollten Sie das Passwort vergessen haben, so müssen Sie den BX-Controller einschicken.

Abb. 84: Navigationsschalter der BX-Controller

Schalterbelegung

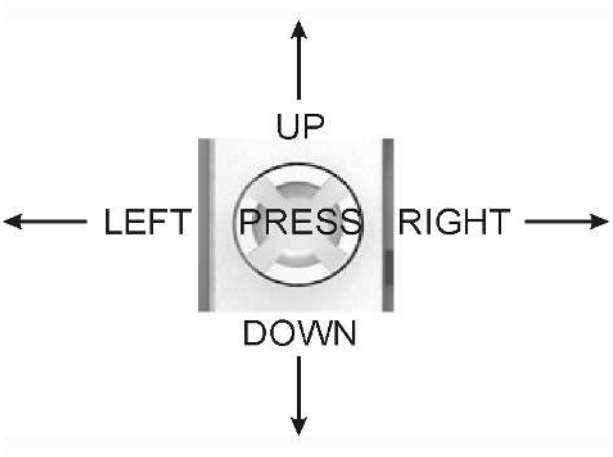


Abb. 85: Schalterbelegung

Hauptmenü	Untermenü 1.Zeile	Untermenü 2.Zeile	Read/Write
MENUE	Passwort	**** nicht gesetzt ???? gesetzt	siehe unten
	Factory Settings	Aktivieren?	wenn die Taste gedrückt wird, werden die Herstellereinstellungen gesetzt und der Controller automatisch rebootet
	Reboot	Aktivieren?	wenn die Taste gedrückt wird, rebootet der Controller neu
AMS	AMS	>AMS Net-ID<	read only
PLC	NAME	>aktueller NAME<	read only
	Curr. Exex. Time	>aktueller Wert<	[ms]
	Task Time	>aktueller Wert<	wenn die Taste gedrückt wird, kann die Zyklus Zeit eingestellt werden
	Status	>Boot-Prj< >PLC Status<	Bootprojekt vorhanden PLC Status
Config	NAME	>aktueller NAME<	read only
	Config löschen	Aktivieren?	wenn die rechte Taste gedrückt wird, löscht man die aktuelle Konfiguration
Real Time Clock	Date and Time	>aktuelle Zeit<	read only
	Year	Setting	2003-2xxx
	Month	Setting	1-12
	Day	Setting	1-31
	Day of week	Setting	Mon, ... Fri
	Houre	Setting	0-23
	Minute	Setting	0-59
	Second	Setting	0-59
COM 1 *read only*	Baudrate	>aktueller Wert<	9600/19200/38400/56800
COM 2 *read only*	Baudrate	>aktueller Wert<	9600/19200/38400/56800/115k
SSB *read only*	Baudrate	>aktueller Wert<	1MBaud, 500k, 250k, 125k, 100k, 50k
	Cycle Time	>aktueller Wert< [in µs]	read only
	Utilization	>aktueller Wert< [in %]	read only
K-Bus *read only*	Diagnose	>aktuelle Diagnose<	read only
	Anzahl der Busklemmen	>aktueller Wert<	read only

Busspezifische Menü Punkte

BX3100

F-Bus PROFIBUS *read only*	Adresse*	>aktueller Wert<	1-126
	Baudrate*	>aktueller Wert<	read only
	Status	>aktueller Wert<	read only
	Diagnose*	>aktueller Wert<	read only

BX5100

F-Bus CANopen *read only*	Adresse*	>aktueller Wert<	1-126
	Baudrate*	>aktueller Wert<	read only
	Status	>aktueller Wert<	read only
	Diagnose*	>aktueller Wert<	read only

*) in Vorbereitung

BX9000

Ethernet	MAC ID	>aktueller Wert<	000105-xx-xx-xx, read only
	ADDR.STATE	>aktueller Wert<	read only
	ADDRESSING MODE	FIXED IP (Default) DHCP BOOTP BOOTP & SAVE	read / write
	NAME	>aktueller Wert<	BX_xxxxxx (xxxxxxx last 3 Bytes from the MAC ID) read / write
	DEFAULT GATEWAY	0.0.0.0	read / write
	IP MASK	255.255.0.0	read / write
	IP ADRESS	172.16.21.20	read / write

Code

Die Default Einstellung ist "*****", das heißt das kein Passwort aktive ist. Um Einstellungen vorzunehmen ist ein Passwort zu setzen.

Menü Navigation

Wenn Sie den Navigationsschalter (Press) drei Sekunden drücken wechseln Sie in das Menü - Verzeichnis. Hier sind einige Menüpunkte beschrieben.

F-Bus (nur BX3100)

F-BUS **WAIT FOR SETPRM**	Feldbus Status (only read) WAIT FOR SETPRM - Wartet auf die Parameter Daten vom PROFIBUS

SSB

SSB	Smart System Bus

COM2

COM2	Serielle Schnittstelle COM2 (only read)

DOWN

Baudrate **xxx**	Aktuelle Baudrate (only read)

COM1

COM1	Serielle Schnittstelle COM2 (only read)

DOWN

Baudrate **xxx**	Aktuelle Baudrate (only read)

K-Bus

KBUS OK **10 TERMINALS**	K-Bus Diagnose (only read)

DOWN

KBUS RESET	Reset K-Bus

PRESS (short)

KBUS RESET EXCT?	PRESS 1sec - K-Bus Reset wird durchgeführt

PLC

PLC	PLC Status (only read)

DOWN

PROJECT **>NAME<**	PLC Projekt Name

RIGHT

CURR. EXEC. TIME **xxx ms**	Gesamte Bearbeitungszeit im [ms]

RIGHT

4.4.2 Erstellen eigener Menüs

Man kann den das Display und Navigations-Schalter auch für eigene Zwecke gebrauchen, z. B. um Diagnoseinformationen anzuzeigen und oder Parameter zu verändern. Für den einfachen Einstieg ist ein Beispiel angefügt, das Sie für Ihre ersten Schritte verwenden und anpassen können.

Download (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207305483.zip)

4.5 Konfigurations-Software KS2000

Eine Parametrierung und Konfigurierung von Busklemmen-Controllern der BCxx50-, BXxx20- und BXxx00-Serien ist mit der Konfigurations-Software KS2000 nicht möglich. Deren Konfiguration muss mit den TwinCAT System Manager vorgenommen werden.

Die Konfigurations-Software KS2000 kann Ihnen aber bei der Konfiguration oder Diagnose der an die Busklemmen-Controller angereihten Busklemmen helfen.
Es empfiehlt sich hierzu die Baudrate in Konfigurations-Software KS2000, BCxx50 BCxx20 und BXxx00 auf 38400 Baud zu stellen (8 Datenbits, Even, 1 Stoppbit).

COM1 - automatische Baudraten Erkennung

Die Schnittstelle COM 1 der BXxx00 hat eine automatische Baudraten Erkennung von 9,6 kBaud bis 56,4 kBaud.

Erforderliche KS2000-Version

Die Konfiguration oder Diagnose von Busklemmen an BXxx00 wird ab KS2000 Version 4.3.14 unterstützt.

Um bei einigen Busklemmen die Konfigurationsdialoge nutzen zu können (z. B. KL25xx, KL6811, KL6201, KL6401) müssen folgende Punkte eingestellt werden:

- Ein SPS-Projekt oder ein Boot-Projekt muss deaktiviert sein.
- Der BX-Controller muss in der Default Configuration sein. Setzen Sie die Herstellereinstellung oder schalten Sie im TwinCAT System Manager auf Config Mode (blaues TwinCAT Icon).
- Der BX-Controller muss im FreeRun Modus sein. Aktivieren Sie dies mit dem TwinCAT System Manager.

Nun können Sie sich mit der Konfigurations-Software KS2000 über ADS (Port 100) oder das serielle Kabel einloggen und die KS2000-Dialoge der Busklemmen nutzen.

5 Programmierung

5.1 PLC-Eigenschaften der BX-Controller

Beschreibung	Wert
Datenspeicher	256 kByte
Programmspeicher	256 kByte minus Task-Konfiguration minus POUs beim Online Change
Source Code Speicher	256 kByte
RETAIN	2 kByte
INPUT	2 kByte
OUTPUT	2 kByte
MERKER	4 kByte
max. größe einer Variable	16 kByte
max. POUs	Beschränkung durch Speicher

5.2 TwinCAT PLC

Das Beckhoff TwinCAT Software-System verwandelt jeden kompatiblen PC in eine Echtzeitsteuerung mit Multi-SPS-System, NC-Achsregelung, Programmierumgebung und Bedienstation. Die Programmierumgebung von TwinCAT wird auch für die Programmierung der BC/BX genutzt. Wenn Sie TwinCAT PLC (Windows NT4/2000/XP) installiert haben, können Sie die Feldbus-Verbindung oder die serielle Schnittstelle für Software-Download und Debugging verwenden.

TwinCAT I/O oder TwinCAT PLC können auch als Ethernet-Master (Host) genutzt werden, um Prozessdaten mit dem Busklemmen-Controller auszutauschen. TwinCAT stellt ihnen hierzu den System Manager als Konfigurationstool sowie die Treiber und das ADS-Protokoll zu Verfügung.

Busklemmen-Controller der Serien BCxx50, BCxx20 und BXxx00

Diese Busklemmen Controller der 2. Generation werden mit dem TwinCAT System Manager konfiguriert und mit TwinCAT PLC-Control programmiert. Sie müssen für diese Koppler TwinCAT PLC installieren (Windows NT4, Windows 2000, Windows XP).

Programmierung und Programmübertragung

- über die serielle Schnittstelle [▶ 174]
- über die Feldbus-Schnittstelle [▶ 173] (nur bei Busklemmen-Controllern für PROFIBUS, CANopen und Ethernet)

Online Change

Die Busklemmen-Controller Busklemmen-Controller der BX-Serie und die BCxx50 unterstützen Online Change. Das bedeutet, dass das PLC-Programm durch ein neues Programm ersetzt wird, ohne dass dadurch das Programm unterbrochen wird. Nach Beendigung der Task wird auf das neue Programm umgeschaltet. Damit verbunden ist die doppelte Haltung des SPS Programms. Es stehen 512 kByte zu Verfügung, die für die doppelte Haltung des PLC Programm durch zwei dividiert werden muss, also 256 kByte. Zusätzlich müssen noch für Task-Konfiguration etc. einige kByte abgezogen werden. Beim Online Change werden noch dynamische Daten in den Speicher abgelegt. Sollte ein Programm an die Speichergrenze stoßen (größer 240 kByte) kann es passieren das ein Online Change nicht mehr funktioniert auch wenn das Programm nach einem "Rebuild all" wieder in den BX geschrieben werden kann.

Wann geht kein Online Change?

Es gibt einige Punkte, nachdem ist ein Online Change nicht mehr möglich.

- einfügen einer neuen Bibliothek
- Änderung der Task-Einstellung
- ein "Rebuild all"
- Grenzbereich in der Speicherauslastung des Controllers (PLC Programm größer 90%)

5.3 TwinCAT PLC - Fehler-Codes

Fehlerart	Beschreibung
PLC-Kompilierfehler	Maximum number of POUs (...) exceeded
PLC-Kompilierfehler	Out of global data memory ...

Fehler POUs

Für jeden Baustein wird eine POU (Process Object Unit) angelegt. Per Default sind 256 Bausteine möglich.

Error 3612: Maximum number of POUs (100) exceeded! Compile is aborted.
Data allocation
1 Error(s), 0 Warning(s).

Abb. 86: Maximale Anzahl der POUs überschritten

Wenn man Bibliotheken einbindet, kann dieser Wert nicht mehr ausreichen. Erhöhen Sie dann die Anzahl an POUs.

Öffnen Sie dazu im PLC Control unter Projekte/Optionen...

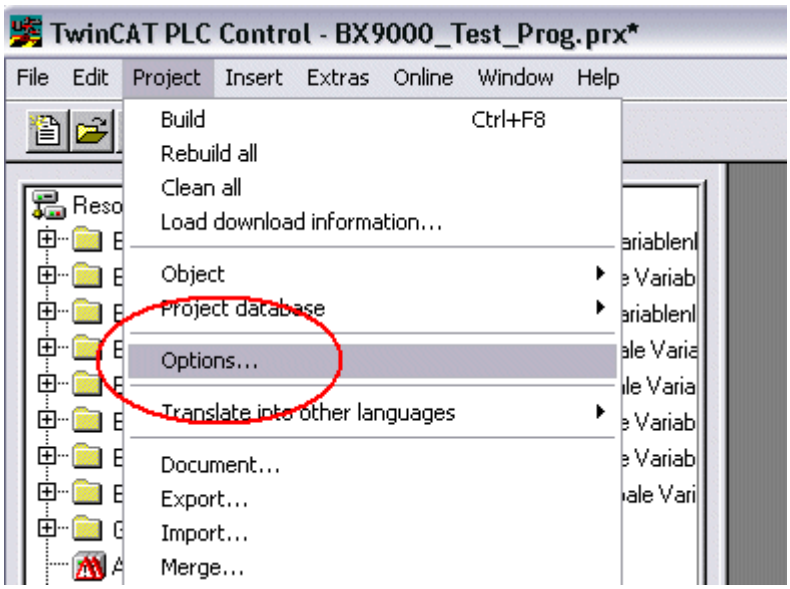


Abb. 87: Menüpfad Projekte / Optionen / Controller Settimgs

...die Controller Settings.

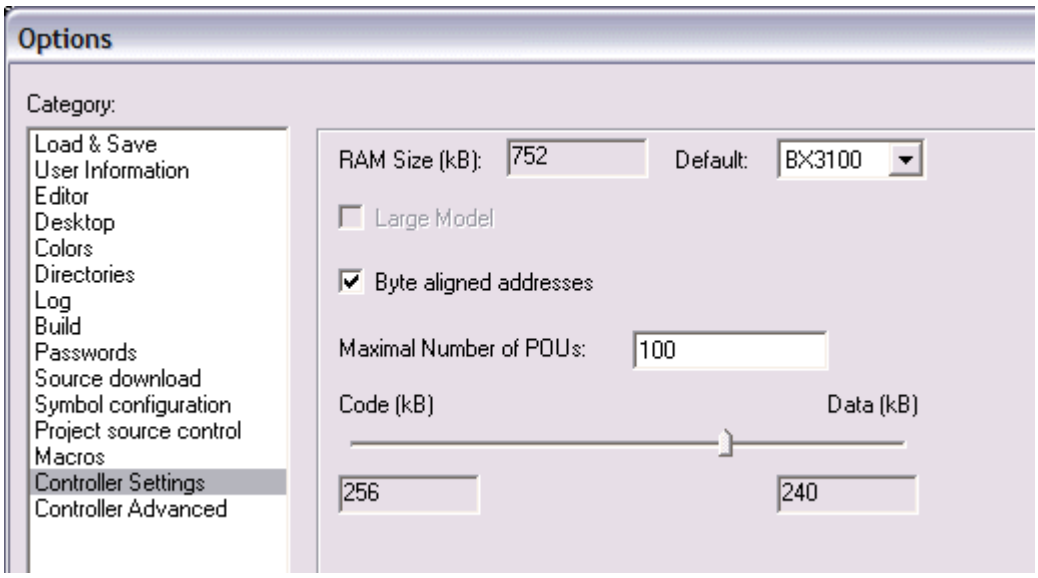


Abb. 88: Controller Settings

Eine Änderung dieser Einstellungen hat zufolge, das kein Online Change mehr geht.

Fehler Globaler Speicher

```
Interface of POU 'MAIN'
Data allocation
Error 3803: MAIN (7): Out of global data memory. Variable 'Test_', 16002 bytes.
1 Error(s), 0 Warning(s).
```

Abb. 89: Globaler Speicher nicht ausreichend

Per Default sind 2 x 16 kByte Daten angelegt. Wenn viele Daten benutzt werden sollen, müssen Sie diesen Bereich vergrößern. Beim BX sind maximal 14 Datensegmente möglich.

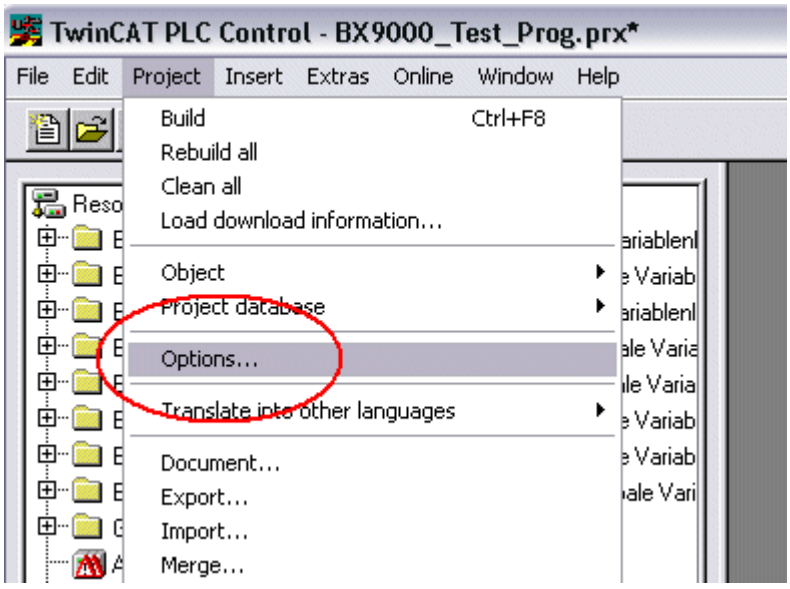


Abb. 90: Menüpfad Projekte / Optionen / Build

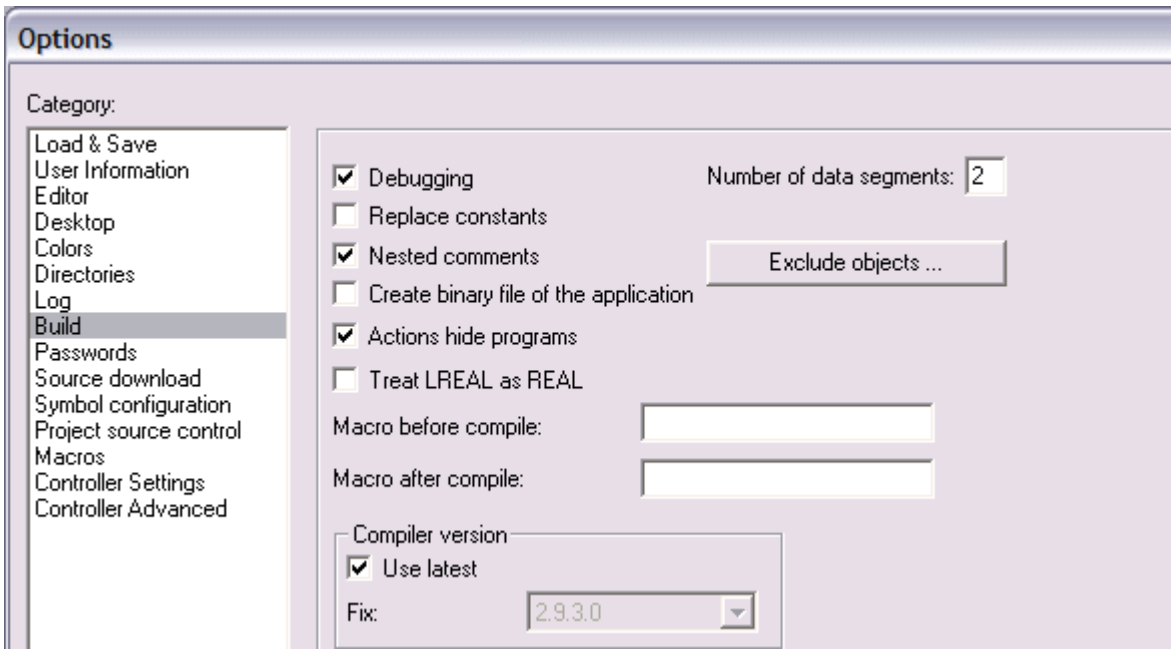


Abb. 91: Build

5.4 Remanente Daten

Auf dem BC9191 und den BX-Controller stehen 2000 Byte an remanenten Daten zur Verfügung. Diese Daten werden im PLC Control als VAR RETAIN deklariert.

Beispiel

```
VAR RETAIN
    Test    :BOOL;
    Count   :INT;
END_VAR
```

Zwischen VAR RETAIN und END_VAR stehen die Retain Daten. Diese Daten werden in einem NOVRAM gespeichert und sind über den ganzen 2 kByte großen Bereich konsistent. Die RETAIN Daten werden nach jedem Zyklus ins NOVRAM gespeichert. Für 2 kByte werden ca. 2 ms benötigt (für 1 kByte ca. 1 ms). Die Variablen können lokal oder global Konfiguriert sein. Variablen die lokiert sind (%MB, %QB, %IB) können nicht als Remanente Daten genutzt werden.

VAR_RETAIN nicht in Funktionsbausteinen benutzen

VAR_RETAIN sollte nicht in Funktionsbausteinen benutzt werden. Sämtliche Daten in einem FB werden in den Retain Speicher kopiert, damit erhöht sich unnötig die Zykluszeit und der Retain Speicher wird mit unnötigen Datenmengen gefüllt.

Variablen mit Adresse nicht als remanente Daten verwenden

Variablen die auf einer Adresse liegen (%MB, %QB, %IB) dürfen nicht als remanente Daten verwendet werden.

Beispiel für remanente Daten im Funktionsbaustein

Da immer alle Daten eines Funktionsbausteins, in dem auch nur ein remanentes Bit zu finden ist, gespeichert wird, sollte dies möglichst vermieden werden. Im Anschluss finden Sie ein Programmbeispiel.

Funktionsbaustein Test (Kein Programm Code notwendig - in ST reicht ein Semikolon)

```
FUNCTION_BLOCK Test
VAR_INPUT
END_VAR
VAR_OUTPUT
END_VAR
```

```
VAR
END_VAR
VAR_IN_OUT
    Counter   :INT;
END_VAR
```

Programm MAIN

```
PROGRAM MAIN
VAR
    fb_Test:Test;
END_VAR
VAR RETAIN
    iCounter1:INT;
END_VAR
```

```
fb_Test(Counter:=iCounter1);
```

5.5 Persistente Daten

Es stehen auf dem Busklemmen-Controller 1000 Byte an persistenten Daten zur Verfügung.
Im Unterschied zu den Retain-Daten werden diese auch bei einem neuen Projekt, bei einem Reset der SPS sowie bei einem neuen Download nicht gelöscht.

Um die persistenten Daten zu nutzen müssen diese erst einmal mit einem Funktionsbaustein aus der SPS heraus aktiviert werden.

Zweitens müssen die Variablen auf dem lokierten Merkerbereich liegen. Hier steht es Ihnen frei wo ihre persistenten Daten liegen.
Es stehen 4 kByte an lokierten Merkern zur Verfügung und sie können davon 1000 Byte als persistente Daten deklarieren.

Beispiel

```
VAR
    Test AT %MX1000   :BOOL;
    Count AT %MB1002  :INT;
END_VAR
```

Mit dem Baustein **Persistent_Data** legen sie die Anfangsadresse fest und die Länge in Byte ab der die Daten persistent sein sollen.

Mit der Eingangsvariable *WriteOffset* gibt mal den Byte Offset des Merkerbereichs an, mit *WriteSize* die Länge in Byte.

Den Baustein finden Sie in der TcSystemBX.lbx Bibliothek. Sollte diese nicht vorhanden sein, laden Sie sie sich aus dieser Dokumentation herunter (siehe Bibliotheken [▶ 108]).

Beispielwerte

WriteOffset 1000
WriteSize 10

Alle Daten die im Bereich %MB1000 - %MB1009 sind dann persistent. Es spielt keine Rolle um welchen Variablen Typ es sich handelt.

Die Daten werden wie bei den Retain Daten in NOVRAM kopiert und sind daher in jedem Zyklus beschreibbar.

Persistenten Daten ab Firmware 1.17

Die persistenten Daten werden bei allen BX-Controllern ab Firmware 1.17 oder höher unterstützt.

Parameter sind sofort gültig

Das Schreiben der Parameter muss nur einmal erfolgen und ist sofort gültig. Diese Daten werden dauerhaft gespeichert.
Das aktivieren der Herstellereinstellung löscht dies wieder und auch die persistenten Daten werden gelöscht.

Beispielprogramm

Klicken Sie auf den Link (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207307659.zip) um ein Beispielprogramm aus dieser Dokumentation herunterzuladen.

5.6 Lokierte Merker

Es stehen 4 kByte an lokierten Merkern zur Verfügung. Diese können genutzt werden, um unterschiedliche Variabelentypen auf die gleiche Adresse zu legen, zum Beispiel um aus Strings Bytes zu machen. Weiter können Daten hinterlegt werden, die per ADS von der Steuerung gelesen und/oder geschrieben werden können.

Lokierte Variablen sind keine remanenten Daten

Bei den Busklemmen-Controller der BX-Serie und den BCxx50 werden die lokierten Variablen **nicht** als remanente Daten gespeichert.

Read / Write lokierter Merker per ADS

Die Merker können auch über die Steuerung per ADS ausgelesen werden. Bei PROFIBUS werden dazu die DPV-1 Dienste genutzt, bei CANopen die SDO Kommunikation.
Die AmsNetId ist aus dem System Manager zu entnehmen oder man kann Sie sich in dem Menü des Busklemmen-Controllers Anzeigen lassen.
Die Port Nummer ist 800 für die PLC.

Index Group	Bedeutung	Index Offset (Wertebereich)
0x4020	Merker (nur BXxxx0)	0..4096

Beispiel

BX Programm

```
VAR
    Flag_01 AT %MB0: WORD;
END_VAR
```

TwinCAT PC/CX Master Programm

```
VAR
    fbADRSREAD: ADSREAD;
    Flag_M: WORD;
END_VAR
```

```
fbADRSREAD(
    NETID:='172.16.3.0.2.3' ,   (* AMSNetId BX *)
    PORT:=800 ,                 (* 800 - PLC *)
    IDXGRP:=16#4020 ,           (* 0x4020hex falgs *)
    IDXOFFS:=0 ,                (* byte offset *)
    LEN:=2 ,                    (* Lenght byte *)
    DESTADDR:=ADR(Merker) ,
    READ:=TRUE ,
    TMOUT:=t#1s );
IF NOT fbADRSREAD.BUSY THEN
    fbADRSREAD(READ:=FALSE);
END_IF
```

5.7 Lokales Prozessabbild im Auslieferungszustand (Default Config)

Das Prozessabbild der Busklemmen-Controller besteht aus Eingangs-, Ausgangs- und Merkerbereich. Daneben gibt es noch die unlokierten Daten. Diese Daten besitzen keine feste Adresse. Sie werden ohne Angabe einer Adresse angelegt. Für diese Art der Variablen stellt

- der BCxx50 48 kByte,
- der BC9x20, BC9191 128 kByte und
- der BXxx00 256 kByte Speicher zur Verfügung.

Eine Variable oder Struktur (Array) darf maximal 16 kByte groß sein. Für die lokierten Daten stehen 2048 Byte Ein- und 2048 Byte Ausgangsdaten bereit. Für den lokierten Merkerbereich verfügt der Busklemmen-Controller über 4 kByte Speicher.

Im Auslieferungszustand (Default Configuration) des BX/BCxx50 werden allen angeschlossenen Busklemmen feste Adressen zugewiesen. Die Daten für die Ethernet-Kommunikation beginnen ab dem Adress-Offset 1000_{dez}. Die Länge der Ethernet-Daten hängt von der Anzahl der konfigurierten Daten ab und beträgt maximal 1000 Byte beim BX9000.

Eingänge	Ausgänge
Busklemme %IB0 ...	Busklemmen %QB0 ...
Ethernet DATEN (SPS- Variablen) %IB1000 ... (ModbusTCP/ADS-TCP/ADS-UDP)	Ethernet DATEN (SPS- Variablen) %QB1000 ... (ModbusTCP/ADS-TCP/ADS-UDP)
... %IB2047 Maximal	... %QB2047 Maximal

Adressierung der Angeschlossenen Busklemmen

Alle angeschlossenen Busklemmen werden in der Default-Einstellung dem lokalen Prozessabbild zugewiesen. Das Mapping im Busklemmen Controller erfolgt nach folgender Gesetzmäßigkeit:
Erst alle komplexen Busklemmen, in der Reihenfolge wie diese gesteckt sind und anschließend die digitalen Busklemmen, die zu einem Byte aufgefüllt werden. Das Default-Mapping der komplexen Busklemmen ist:

- komplette Auswertung
- Intel-Format
- Word Alignment

Beispielaufbau

Busklemmen Controller: 1 x BCxx50, BCxx20 oder BXxx00
Position 1: 1 x KL1012
Position 2: 1 x KL1104
Position 3: 1 x KL2012
Position 4: 1 x KL2034
Position 5: 1 x KL1501
Position 6: 1 x KL3002
Position 7: 1 x KL4002
Position 8: 1 x KL6001
Position 9: 1 x KL9010

Tab. 1: Prozessabbild

Busklemme	Position	Eingangsabbild	Ausgangsabbild	Größe
KL1501	5	%IB0...%IB5	%QB0...%QB5	6 Byte
KL3002	6	%IB6...%IB13	%QB6...%QB13	8 Byte
KL4002	7	%IB14...%IB21	%QB14...%QB21	8 Byte
KL6001	8	%IB22...%IB29	%QB22...%QB29	6 Byte
KL1012	1	%IX30.0...%IX30.1	-	2 Bit
KL1104	2	%IX30.1...%IX30.5	-	4 Bit
KL2012	3	-	%QX30.0...%IX30.1	2 Bit
KL2034	4	-	%QX30.2...%IX30.5	4 Bit
KL9010	9	-	-	-

5.8 Mapping der Busklemmen

Die genaue Belegung der byteorientierten Busklemmen entnehmen Sie bitte der Konfigurations-Anleitung zur jeweiligen Busklemme. Diese Dokumentation finden Sie im Internet unter http://www.beckhoff.de.

byteorientierte Busklemmen	bitorientierte Busklemmen
KL15x1	KL10xx, KL11xx, KL12xx, KL17xx, KM1xxx
KL25xx	KL20xx, KL21xx, KL22xx, KL26xx, KL27xx, KM2xxx
KL3xxx	
KL4xxx	
KL5xxx	
KL6xxx	
KL7xxx	
KL8xxx	
	KL9110, KL9160, KL9210, KL9260

5.9 Lokales Prozessabbild in der TwinCAT-Konfiguration

Die TwinCAT-Konfiguration (TwinCAT-CONFIG) ermöglicht das freie Mapping zwischen Feldbus, K-Bus und SPS-Variablen. Variablen können unabhängig von ihrer Adresse mit Hilfe des System Managers verknüpft werden.

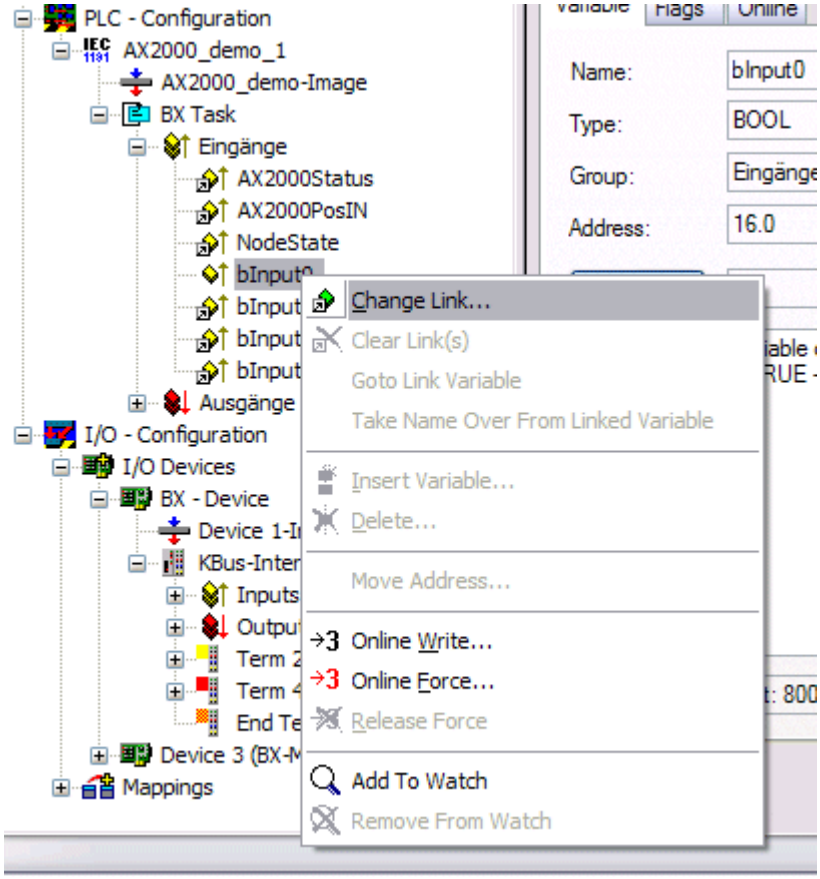


Abb. 92: Ändern der Verknüpfung von Variablen

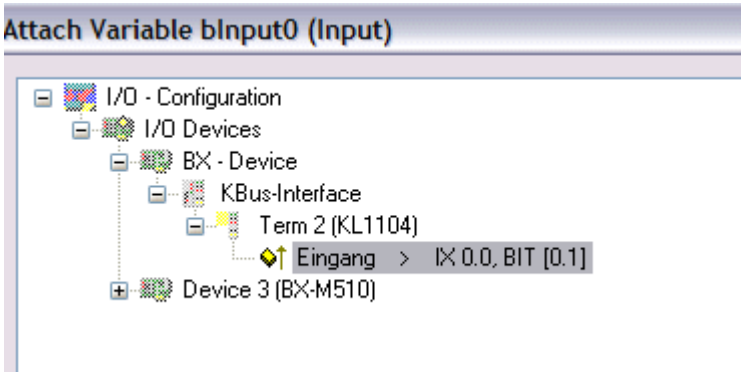


Abb. 93: Verknüpfen einer Variable mit einem Eingang

In der Default-Konfiguration liegen alle Busklemmen auf feste Adressen. Fügt man eine Busklemme ein, verschiebt sich eventuell der ganze Adressbereich. Mit der TwinCAT-Konfiguration hat man die Möglichkeit, seine lokierten Variablen, frei mit einer Busklemme zu verknüpfen. Man parametriert dies im System Manager und lädt dann diese erstellte Konfiguration auf den Busklemmen-Controller herunter (siehe TwinCAT-Konfiguration [▶ 34]). Ein Hochladen einer bestehenden TwinCAT-Konfiguration ist auch möglich.

5.10 Erzeugen eines Boot-Projekts

Für das Erzeugen des Boot-Projektes stehen

- auf den Busklemmen-Controllern der BX-Serie ca. 250 kByte Flash zu Verfügung.
- auf den Busklemmen-Controllern der BCxx50-Serie ca. 48 kByte Flash zu Verfügung.

PLC Control

Im TwinCAT PLC Control kann man, wenn man eingeloggt ist ein Boot-Projekt erzeugen.

- Öffnen eines PLC Projektes
- Auswahl des Zielsystem (oder Auswahl der seriellen Schnittstelle)
- Einloggen auf den BX/BCxx50
- Erzeugen des Boot-Projektes (Online\Create Bootproject)

Ist ein gültiges Boot-Projekt auf dem BX/BCxx50 leuchtet die LED PLC grün.

Bei den Busklemmen-Controllern der BX-Serie blinkt während der Erzeugung des Boot-Projektes die die PLC LED orange. Ist auf dem BX kein Boot-Projekt vorhanden, leuchtet die PLC LED orange.

Löschen eines Boot-Projektes

Sie können das Boot-Projekt auch vom Busklemmen-Controller löschen. Folgende Schritte sind einzuhalten:

- Öffnen des Projektes
- Einloggen auf den Busklemmen-Controller
- Löschen des Boot-Projektes (Online\Delete Boot Project)

Nach dem Löschen des Boot-Projektes ist die PLC LED orange.

Übernahme des aktuellen Projektes als Boot-Projekt

Nach einem Online Change ist als Boot-Projekt noch immer das alte Projekt eingetragen. Soll das aktuelle Projekt (nach dem Online-Change) als Boot-Projekt übernommen werden, muss dieses neu erzeugt werden.

Umgehen eines Starten des Boot-Projekt*

Bei den Busklemmen-Controllern der BX-Serie kann beim Booten das Starten des Boot-Projekts durch Drücken des Navi-Schalters verhindert werden. Das Boot-Projekt ist damit nicht gelöscht und wird beim erneuten Booten des Busklemmen-Controller wieder gestartet.

* ab Version 0.85

5.11 Kommunikation zwischen TwinCAT und BX/BCxx50

Um von TwinCAT-Daten zum Busklemmen-Controller zu transportieren liegt es nahe, die Daten in einer Struktur anzulegen. Da die Datenhaltung auf beiden Systemen unterschiedlich ist sind folgende Hinweise zu beachten.

- Wenn zwei unterschiedliche Datentypen aufeinander folgen (zum Beispiel Byte und INT) wird die folgende Variable auf den nächsten graden Adress-Offset gelegt
- Boolsche Variablen sollten nie einzeln in eine Struktur gelegt werden, da sie so immer 1 Byte belegen würden. Boolsche Ausdrücke sollten immer in ein Byte oder Wort maskiert sein.

Beispiel 1: Eine Struktur auf den BX/BCxx50 und auf dem PC

Variable	Speicher des BX/BCxx50	Speicher des PC (TwinCAT)
Byte	%..B0	%..B0
INT (1)	%..B2	%..B1
INT (2)	%..B4	%..B3

Dadurch das hinter dem ersten Byte eine anderer Variable-Typ (INT) folgt ist dieser im BX/BCxx50 auf die nächste freie grade Adresse gelegt worden. Will man beide auf beiden Systemen die gleiche Datenstruktur haben, muss im PC-Projekt ein Dummy-Byte eingefügt werden (siehe Beispiel 2).

Beispiel 2: Eine Struktur auf den BX/BCxx50 und auf dem PC mit gleicher Speicherbelegung

Variable	Speicher des BX/BCxx50	Speicher des PC (TwinCAT)
Byte	%..B0	%..B0
Byte (Dummy)	%..B1 (nicht unbedingt notwendig, da dies das System selber macht, wenn diese Variabel nicht vorhanden ist)	%..B1
INT (1)	%..B2	%..B2
INT (2)	%..B4	%..B4

Daten Struktur

```
Type PB_Data
STRUCT
    wVar_1:WORD;
    iValue_1:INT;
    iValue_2:INT;
    iValue_3:INT;
END_STRUCT
END_TYPE
```

Anlegen einer Struktur in den Variablen

```
VAR_Global
    strData_Out AT %QB1000:PB_Data; (*PLC Variables *)
    bInput_01 AT %IX0.0:BOOL; (* Input from a terminal *)
END_VAR
```

Kleines Programmbeispiel

```
strData_Out.wVar_1.0:=bInput_01;
```

Keine Real-Werte in gemischter Datenstruktur verwenden

In einer gemischten Datenstruktur sollten keine Real-Werte enthalten sein. Wenn dies doch der Fall ist, muss zusätzlich im BX/BCxx50 oder im TwinCAT-Masterprojekt das High und Low Word vertauscht werden. Verwenden Sie besser ein Array von Real-Werten oder übertragen Sie die Real-Werte einzeln.

Größere Feldbusdatenblöcke

Sie können auch größere Feldbusdatenblöcke übertragen, um eine Reserve für Ihre Struktur zu haben. Nachteil: Diese Reserven werden dann mit jedem Feldbustelegramm übertagen, was eine Mehrbelastung der Feldbuskommunikation verursacht.

5.12 Up- und Download von Programmen

Der Busklemmen-Controller verfügt über einen Speicher für den Quell-Code. Hier können das Programm, die Task-Konfiguration und die Bibliotheken abgespeichert werden. Sollte Speicher für den Quell-Code nicht ausreichen, kann man auch nur den Quell-Code ohne Task-Konfiguration und die Bibliotheken ablegen. Dies benötigt wesentlich weniger Speicherplatz!

Allgemeine Einstellungen

Unter Bearbeiten/Optionen kann eingestellt werden wann der Quell-Code zum Zielsystem heruntergeladen werden soll. Öffnen Sie das Optionsmenü.

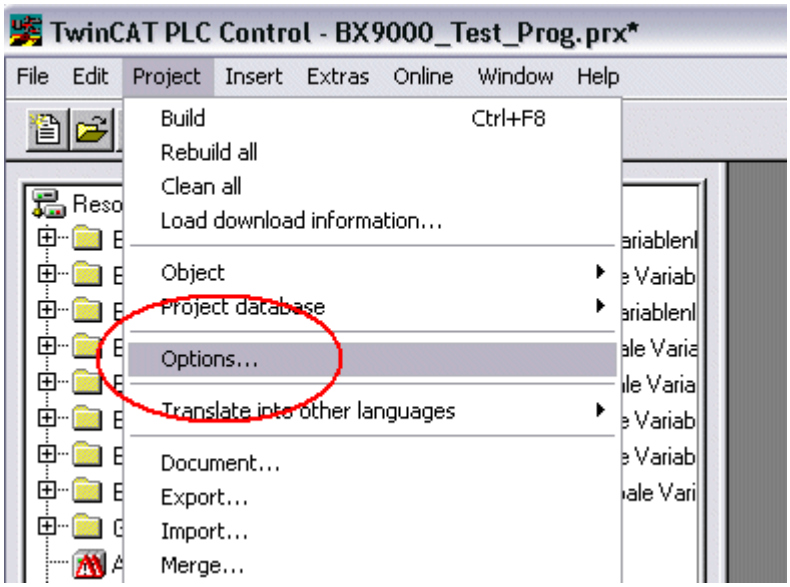


Abb. 94: Öffnen des Optionsmenüs

Wählen Sie nun den Source Download an.

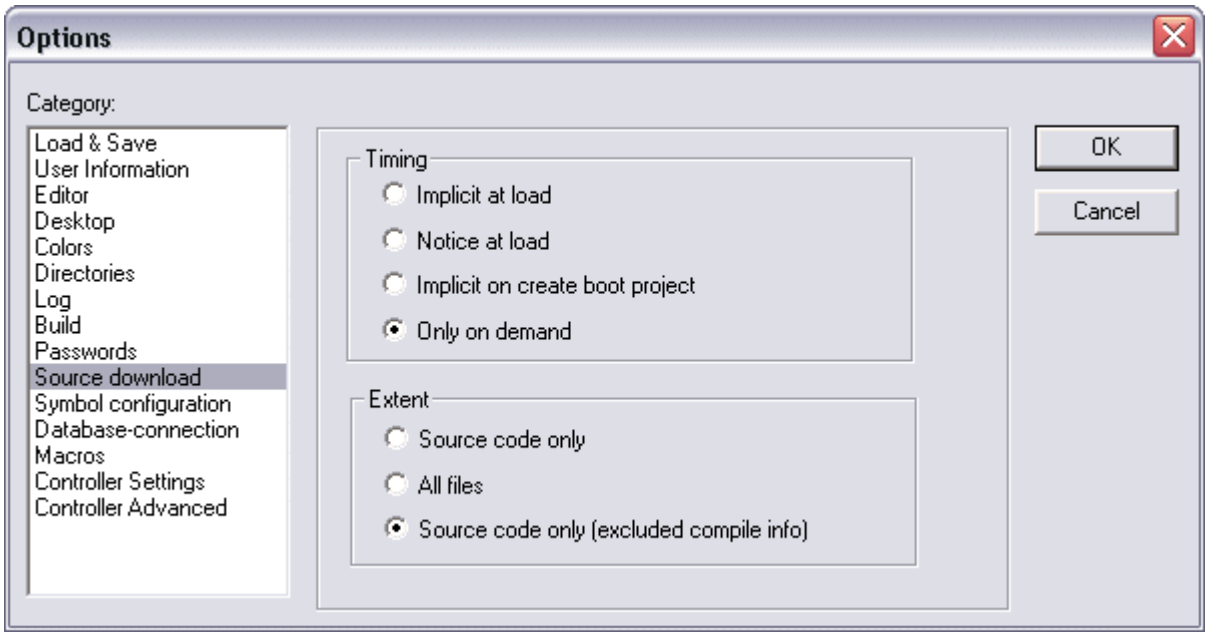


Abb. 95: Auswahl des Source Downloads

Hier könne Sie einstellen wann und was vom Source-Code zum Busklemmen-Controller runtergeladen werden soll.

Source code only: es wird das prx-File mit Informationen zum online Change übertragen. Damit ist ein einloggen per Online-Change möglich (die SPS stoppt nicht).
All files: Wie *Source code only* plus alle notwenigen Bibliotheken.
Source code only (excluded conpile info): es wird nur das prx File übertragen. Ein einloggen ist nur möglich, wenn die SPS stoppt.

Welche Option Sie verwenden können hängt von der Größe Ihrer Projekte ab.

Download eines Programms

Der Quell-Code kann man auf Anforderung zum Zielsystem übertragen. Dafür muss man mit seinem Programm eingeloggt sein. Unter Online/Quell-Code Download kann jetzt der Programm-Code zum Busklemmen-Controller übertragen werden.

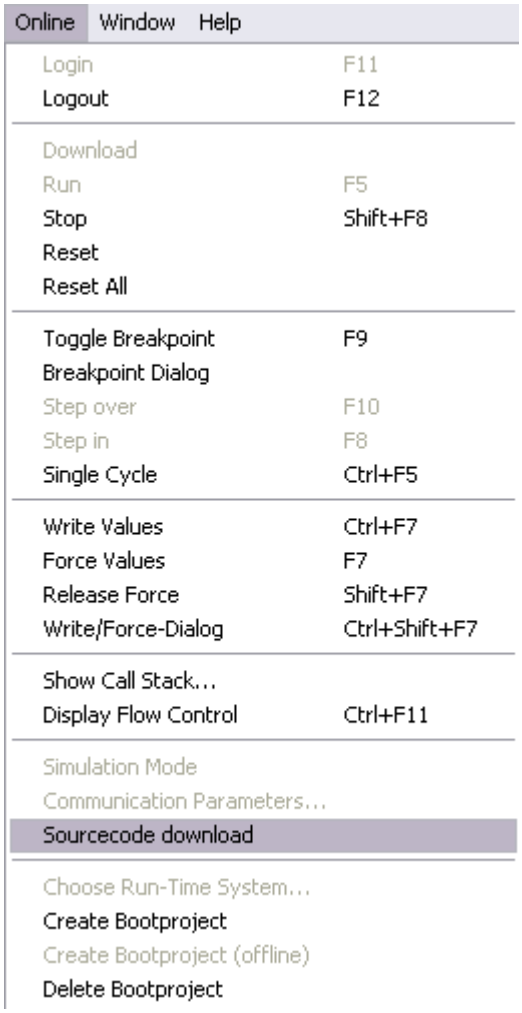


Abb. 96: Download des Programm Codes

Nach einer kurzen Zeit öffnet sich ein Fenster, das den Vorschritt des Downloads anzeigt.

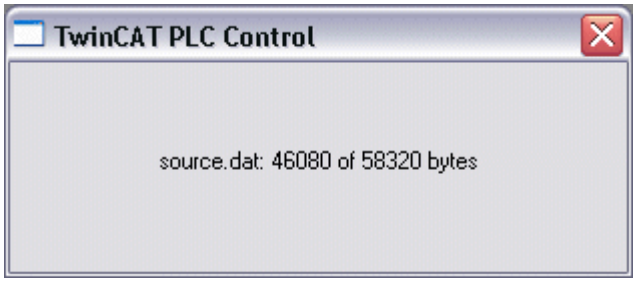


Abb. 97: Vorschritt des Downloads

Upload eines Programms

Um den Programm-Code wieder hoch zu laden, öffnen Sie im PLC Control ein neues File. Als nächstes klicken Sie auf die Schaltfläche PLC.

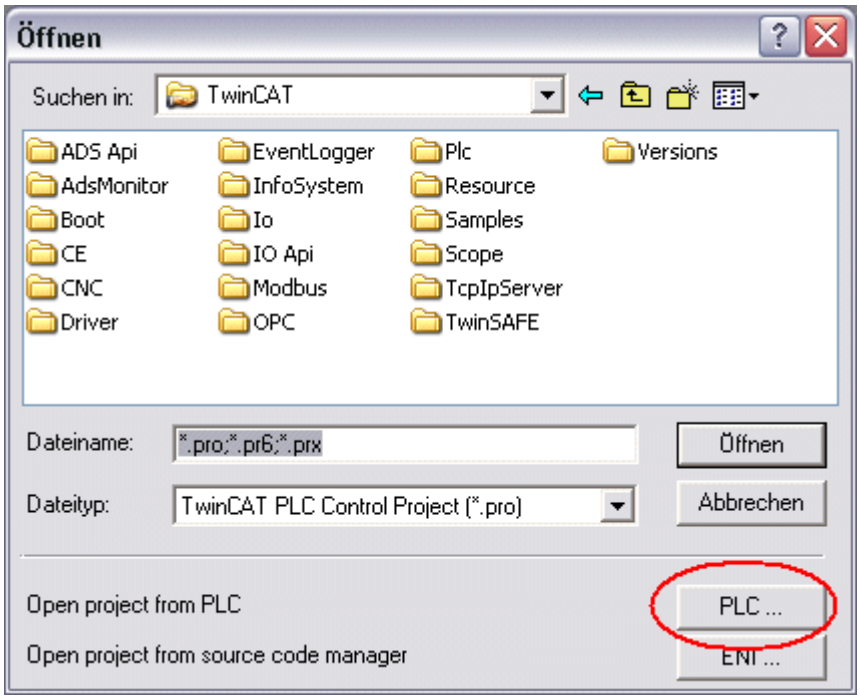


Abb. 98: Upload eines Programms

Wählen Sie den Datenübertragungsweg aus:

- *BCxx50 or BX via AMS*, wenn sie über den Feldbus mit dem Busklemmen-Controller verbunden sind oder
- *BCxx50 or BX via serial*, wenn sie über die serielle Schnittstelle mit dem Busklemmen-Controller verbunden sind.

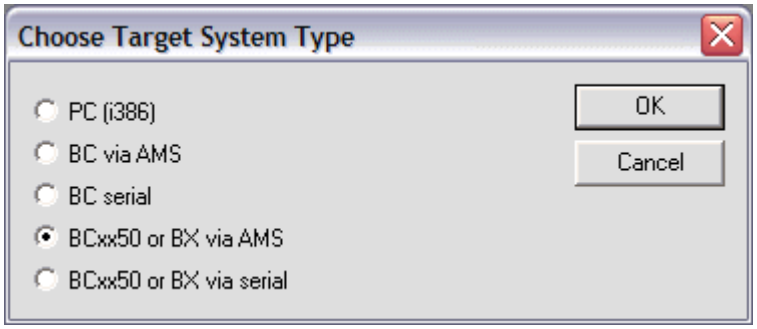


Abb. 99: Auswahl des Datenübertragungswegs

Als nächsten Schritt wählen Sie das Gerät aus und bestätigen Sie mit OK.

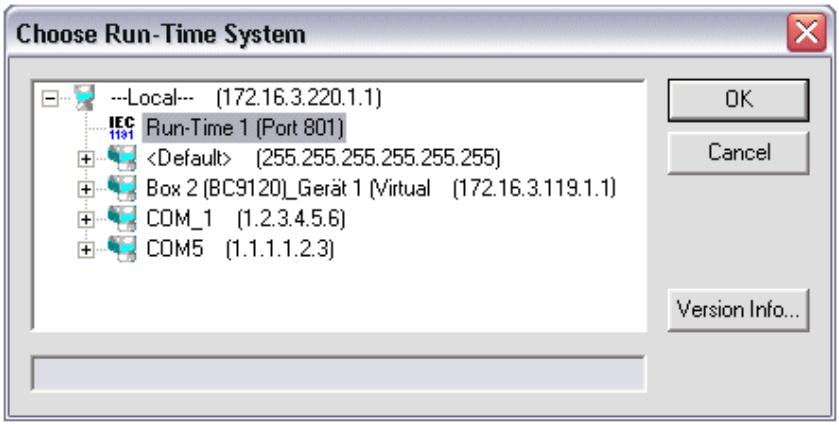


Abb. 100: Auswahl des Gerätes

Der Quell-Code wird nun hochgeladen.

Passwort

Mit einem Passwort können Sie Ihr Projekt schützen (im PLC Control Projekt/Optionen/Kennworte).

5.13 Bibliotheken

5.13.1 Bibliotheken - Übersicht

Für die Busklemmen Controller (Buskoppler mit SPS-Funktionalität, Bezeichnung BXxxxx) gibt es verschiedene Bibliotheken (siehe Beckhoff Information System).

Download

Zum Download der Bibliotheken bitte auf den Link mit der linken Maustaste klicken. Bitte die Bibliotheken in das Verzeichnis TwinCAT\PLC\LIB kopieren.

- Standard (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207309835.zip)
- TcSystemBX (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207312011.zip)
  (die TcSystemBX benötigt die TcBaseBX Bibliothek)
- TcBaseBX (Download)
  (die TcDisplayBX, TcNaciSwitchBX und TcDebugBX sind nun hier enthalten)
- TcComPortBX (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207314187.zip)
- ChrAscBX.lbx (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207316363.zip)

Verwenden Sie die zur Firmware passende Bibliothek

Für die neuste Firmware ist auch die neuste Bibliothek zu benutzen. Sollten Sie Ihren BX-Controller Updaten tauschen Sie auch die Bibliotheken.
Kopieren Sie diese Bibliotheken in das LIB-Verzeichnis, entfernen diese Bibliotheken aus Ihrem Projekt und fügen Sie sie erneut hinzu.

TcSystemBX

ADS	Version	Firmware				
		BX3100	BX5100	BX5200	BX8000	BX9000
ADSREAD	04.03.04	0.90	0.14	1.00	0.02	1.00
ADSWRITE	04.03.04	0.90	0.14	1.00	0.02	1.00
ADSRDWRT	04.03.04	0.90	0.14	1.00	0.02	1.00
ADSWRTCTL	04.03.04	0.90	0.14	1.00	0.02	1.00
ADSRDSTATE	04.03.04	0.90	0.14	1.00	0.02	1.00
ADSRDDEVINFO	04.03.04	0.90	0.14	1.00	0.02	1.00

Bit Functions	Version	Firmware				
		BX3100	BX5100	BX5200	BX8000	BX9000
CLEARBIT32	07.03.03	0.28	0.01	1.00	0.01	1.00
CSETBIT32	07.03.03	0.28	0.01	1.00	0.01	1.00
GETBIT32	07.03.03	0.28	0.01	1.00	0.01	1.00
SETBIT32	07.03.03	0.28	0.01	1.00	0.01	1.00

Display Function	Version	Firmware				
		BX3100	BX5100	BX5200	BX8000	BX9000
FB_DispWrite [▶ 123]	31.03.03	0.28	0.01	1.00	0.01	1.00

Diagnose	Version	Firmware				
		BX3100	BX5100	BX5200	BX8000	BX9000
BX_Security	15.08.06	1.12	1.14	-	1.12	1.14
DeviceTyp	15.08.06	1.12	1.14	-	1.12	1.14
FirmwareVersion	15.08.06	1.12	1.14	-	1.12	1.14
FirmwareVersionString	15.08.06	1.12	1.14	-	1.12	1.14
DeviceTyp	15.08.06	1.12	1.14	-	1.12	1.14
Read_Diagnose	15.08.06	1.12	1.14	-	1.12	1.14
CRCBootproject	15.08.06	1.14	1.14	-	1.14	1.14

Read Address	Version	Firmware				
		BX3100	BX5100	BX5200	BX8000	BX9000
ReadSlaveAddress	15.08.06	1.12	1.12	1.10	1.12	-

Controller	Version	Firmware				
		BX3100	BX5100	BX5200	BX8000	BX9000
FB_BasicPID	04.03.04	0.64	0.01	1.00	0.01	1.00

Event Logger Functions	Version	Firmware				
		BX3100	BX5100	BX5200	BX8000	BX9000
-	-	-	-	-	-	-

File Access	Version	Firmware				
		BX3100	BX5100	BX5200	BX8000	BX9000
FB_ReadFromFile	03.08.04	1.04	1.04	1.00	1.04	1.00
FB_WriteToFile	03.08.04	1.04	1.04	1.00	1.04	1.00
FB_ReadWriteFile	03.08.04	1.04	1.04	1.00	1.04	1.00

Memory Functions	Version	Firmware				
		BX3100	BX5100	BX5200	BX8000	BX9000
MEMCMP	07.03.03	0.41	0.01	1.00	0.01	1.00
MEMCYP	07.03.03	0.41	0.01	1.00	0.01	1.00
MEMMOVE	07.03.03	0.41	0.01	1.00	0.01	1.00
MEMSET	07.03.03	0.41	0.01	1.00	0.01	1.00

NOVRAM Functions	Version	Firmware				
		BX3100	BX5100	BX5200	BX8000	BX9000
-	-	-	-	-	-	-

Serial Communication Interface	Version	Firmware				
		BX3100	BX5100	BX5200	BX8000	BX9000
FB_COMPortClose	14.07.03	0.49	0.01	1.00	0.01	1.00
FB_COMPortOpen	14.07.03	0.49	0.01	1.00	0.01	1.00
F_COMPortRead	14.07.03	0.49	0.01	1.00	0.01	1.00
F_COMPortWrite	14.07.03	0.49	0.01	1.00	0.01	1.00

SFC	Version	Firmware				
		BX3100	BX5100	BX5200	BX8000	BX9000
AnalyzeExpression	07.03.03	0.28	0.01	1.00	0.01	1.00
AppendErrorString	07.03.03	0.28	0.01	1.00	0.01	1.00
SFCActionControl	07.03.03	0.28	0.01	1.00	0.01	1.00

System / Time / TBus	Version	Firmware				
		BX3100	BX5100	BX5200	BX8000	BX9000
DRAND	07.03.03	0.28	0.01	1.00	0.01	1.00
RTC	07.03.03	0.28	0.01	1.00	0.01	1.00
SYSTEMTIME_TO_DT	07.03.03	0.28	0.01	1.00	0.01	1.00
DT_TO_SYSTEMTIME	07.03.03	0.28	0.01	1.00	0.01	1.00
GetSysTick	14.07.03	0.49	0.01	1.00	0.01	1.00
PresetSysTick	14.07.03	0.49	0.01	1.00	0.01	1.00
Reboot	21.07.03	0.59	0.14	1.00	0.02	1.00
Persistent_Data	21.08.07	1.17	1.17	-	1.17	1.17

Debug	Version	Firmware				
		BX3100	BX5100	BX5200	BX8000	BX9000
F_ReadDebugTimer [▶ 122]	08.08.03	0.59	0.14	1.00	0.02	1.00
F_StartDebugTimer [▶ 122]	08.08.03	0.59	0.14	1.00	0.02	1.00

NaviSwitch	Version	Firmware				
		BX3100	BX5100	BX5200	BX8000	BX9000
Alle Bausteine [▶ 122]	10.10.03	0.64	0.14	1.00	0.02	1.00

TcComPortBX

Com FBs	Version	Firmware				
		BX3100	BX5100	BX5200	BX8000	BX9000
Alle Bausteine [▶ 128]	20.08.03	0.60	0.02	1.00	0.01	1.00

5.13.2 Bibliotheken für BX9000, BC9020, BC9050, BC9120 - Übersicht

Spezielle Funktionsbausteine für die Busklemmen-Controller BX9000, BC9050, BC9120 und BC9020.

Download

Zum Download der Bibliotheken klicken Sie bitte mit der linken Maustaste auf das Diskettensymbol.
Kopieren Sie die Bibliotheken dann in das Verzeichnis TwinCAT\PLC\LIB.

- TcBaseBX9000 (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207318539.zip)

Verwenden Sie die zur Firmware passende Bibliothek

Für die neuste Firmware ist auch die neuste Bibliothek zu benutzen. Sollten Sie Ihren BX-Controller Updaten, tauschen Sie auch die Bibliotheken.
Kopieren Sie die Bibliotheken in das Verzeichnis TwinCAT\PLC\LIB. Entfernen Sie Bibliotheken dann aus Ihrem Projekt und fügen Sie sie erneut wieder neu hinzu.

TcBaseBX9000

Services	Version	Firmware			
		BX9000	BC9020	BC9050	BC9120
FB_AddDnsServer [▶ 165]	30.05.06	1.12	B0	B0	B1
FB_GetHostByAddr [▶ 166]	30.05.06	1.12	B0	B0	B1
FB_GetHostByName [▶ 167]	30.05.06	1.12	B0	B0	B1
FB_GetNetworkConfig [▶ 168]	30.05.06	1.12	B0	B0	B1
FB_SetTargetName [▶ 169]	30.05.06	1.12	B0	B0	B1

SMTP	Version	Firmware			
		BX9000	BC9020	BC9050	BC9120
FB_SMTP [▶ 162]	30.05.06	1.12	B0	B0	B1

SNTP	Version	Firmware			
		BX9000	BC9020	BC9050	BC9120
FB_SNTP [▶ 164]	30.05.06	1.12	B0	B0	B1

IP-TCP/IP-UDP	Version	Firmware			
		BX9000	BC9020	BC9050	BC9120
FB_AddMultiRoute [▶ 147]	26.09.06	1.14	-	-	-
FB_DelMultiRoute [▶ 147]	26.09.06	1.14	-	-	-
FB_IpClose [▶ 147]	26.09.06	1.14	B0	B0	B1
FB_IpEndSession [▶ 147]	26.09.06	1.14	B0	B0	B1
FB_IpOpen [▶ 147]	26.09.06	1.14	B0	B0	B1
FB_IpReceive [▶ 147]	26.09.06	1.14	B0	B0	B1
FB_IpSend [▶ 147]	26.09.06	1.14	B0	B0	B1
FB_IpStartSession [▶ 147]	26.09.06	1.14	B0	B0	B1

ModbusTCP	Version	Firmware			
		BX9000	BC9020	BC9050	BC9120
FB_MBClose [▶ 158]	26.09.06	1.14	B0	B0	B1
FB_MBConnect [▶ 158]	26.09.06	1.14	B0	B0	B1
FB_MBGenericReq [▶ 158]	26.09.06	1.14	B0	B0	B1

5.13.3 TcBaseBX

5.13.3.1 System Task Info

```
VAR_GLOBAL
    SystemTaskInfoArr       : ARRAY[1..2] OF SYSTEMTASKINFOTYPE;
END_VAR
```

Systemflags sind implizit deklarierte Variablen. Mit der Eingabehilfe finden Sie unter Systemvariablen eine Variable SystemTaskInfoArr. Diese Variable ist ein Feld von vier Strukturen des Types SYTEMTASKINFOTYPE [▶ 112]. Die Strukturdefinition ist in der System-Bibliothek zu finden. Der Index in dieses Feld ist die Task-Id.

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.9.0	BX Controller	TcBaseBX.lbx

5.13.3.2 System Task Info Type

```
TYPE SYSTEMTASKINFOTYPE
STRUCT
    active            :       BOOL;
    taskName          :       STRING(16);
    firstCycle        :       BOOL;
    cycleTimeExceeded :       BOOL;
    cycleTime         :       UDINT;
    lastExecTime      :       UDINT;
    priority          :       BYTE;
    cycleCount        :       UDINT;
END_STRUCT
END_TYPE
```

Legende

active: Diese Variable zeigt an, ob die Task aktiv ist.
taskName: Der Taskname.
firstCycle: Diese Variable hat im ersten Zyklus der SPS-Task den Wert: TRUE.
cycleTimeExceeded: In dieser Variablen wird ein Überschreiten der eingestellten Taskzykluszeit gemeldet.
cycleTime :Eingestellte Taskzykluszeit in Vielfachen von 100 ns.
lastExecTime: Benötigte Zykluszeit für den letzten Zyklus in Vielfachen von 100 ns.
priority: Eingestellte Priorität der Task.
cycleCount: Zykluszähler.

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.9.0	BX Controller	TcBaseBX.lbx

5.13.3.3 System Info

```
VAR_GLOBAL
    SystemInfo     : SYSTEMINFOTYPE;
END_VAR
```

Systemflags sind implizit deklarierte Variablen. Mit der Eingabehilfe finden Sie unter Systemvariablen eine Variable Systeminfo. Der Typ SYSTEMINFOTYPE [▶ 113] ist in der System-Bibliothek deklariert. Um auf die Variable zugreifen zu können muss die System-Bibliothek in das Projekt eingebunden werden.

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.9.0	BX Controller	TcBaseBX.lbx

5.13.3.4 System Info Type

```
TYPE SYSTEMINFOTYPE
STRUCT
    runTimeNo           :       BYTE;
    projectName         :       STRING(32);
    numberOfTasks       :       BYTE;
    onlineChangeCount   :       UINT;
    bootDataFlags       :       BYTE;
    systemStateFlags    :       WORD;
END_STRUCT
END_TYPE
```

Legende

runTimeNo: Gibt die Nummer des Laufzeitsystems (1) an.
projectName: Name des Projekts als STRING.
numberOfTasks: Anzahl der im Laufzeitsystem befindlichen Tasks (max. 2).
onlineChangeCount: Anzahl der seit dem letzten Komplettdownload gemachten Online-Änderungen.
bootDataFlags: Reserviert
systemStateFlags: Reserviert.

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.9.0	BX Controller	TcBaseBX.lbx

5.13.3.5 ADS

5.13.3.5.1 Lokale ADS-Port Nummern - Übersicht

Port Nummer	Beschreibung
100 [▶ 114]$_{dez}$	Lesen und schreiben von Registern und Tabellen aus den Koppler und den komplexen Busklemmen
150 [▶ 84]$_{dez}$	Lesen und schreiben der RTC (Echt-Zeit-Uhr)
153 [▶ 65]$_{dez}$	SSB - auslesen der Emergency-Nachricht
800 [▶ 114]$_{dez}$	Lokales Prozessabbild der PLC, siehe auch Port 801
801 [▶ 114]$_{dez}$	Lokales Prozessabbild der PLC, siehe auch Port 800
0x1000 + Node ID [▶ 64]	SSB - SDO Kommunikation mit CANopen-Knoten (Slave Nummer)

5.13.3.5.2 ADS-Dienste

Lokales Prozessabbild PLC Task 1 Port 800/801

Im lokalen Prozessabbild können Daten gelesen und geschrieben werden. Sollten Ausgänge geschrieben werden muss darauf geachtet werden, das diese von der lokalen SPS nicht verwendet werden, da die lokale Steuerung diese Werte überschreibt. Die Daten sind nicht an einen Watchdog gebunden und müssen und dürfen daher nicht für Ausgänge verwendet werden, die im Fehlerfall ausgeschaltet werden müssen.

Index Group	Bedeutung	Index Offset (Wertebereich)
0xF020	Input - Eingänge	0...2047
0xF021	Input - Eingänge Bit	0...16376
0xF030	Output - Ausgänge	0...2047
0xF031	Output - Ausgänge Bit	0...16376
0x4020	Merker	0...4095
0x4021	Merker Bit	0...32760

Dienste des ADS

AdsServerAdsState

Datentyp (only Read)	Bedeutung
String	Start - die lokale PLC läuft Stop - die lokale PLC ist im Stop

AdsServerDeviceState

Datentyp (only Read)	Bedeutung
INT	0 - Start - die lokale PLC läuft 1 - Stop - die lokale PLC ist im Stop

AdsServerType

Datentyp (only Read)	Bedeutung
String	BX PLC Server

ADSWriteControl

Datentyp (write only)	Bedeutung
NetID	Net ID des Ethernet Controllers*
Port	800
ADSSTATE	5 - RUN / 6 - STOP
DEVSTATE	0
LEN	0
SRCADDR	0
WRITE	positive Flanke startet den Baustein
TMOUT	zum Beispiel: t#1000 ms

* BC9050, BC9020, BC9120, BX9000

Registerzugriff Port 100

Die ADS-Portnummer ist bei den Busklemmen-Controllern der BX-Serie und den BCxx50/xx20 für die Register-Kommunikation fest vorgegeben und beträgt 100.

Index Group	Index Offset (Wertebereich)		Bedeutung
	Hi-Word	Lo-Word	
0 [READ ONLY]	0...127	0...255	Register des Buskopplers Hi-Word Tabellennummer des Buskopplers Lo-Word Registernummer der Tabelle
1...255	0...3	1...255	Register der Busklemmen Hi-Word Kanalnummer Lo-Word Registernummer der Busklemme

Minimaler Time-Out

Beachten Sie beim Lesen der Register, dass der Time-Out beim ADS-Baustein auf eine Zeit größer eine Sekunde eingestellt wird.

Passwort setzen

Beachten Sie beim Schreiben auf die Register, dass das Passwort gesetzt wird (siehe Dokumentation zur entsprechenden Busklemme).

5.13.3.5.3 Deaktivieren der LED für Zykluszeit-Überschreitung

Der BX-Controller überwacht die eingestellte Task-Zykluszeit. Wenn diese Überschritten wird, wird das Bit cycleTimeExceeded [▶ 112] und die rote LED *PLC* gesetzt. In einigen Applikationen kann es zu kurzzeitigen, tolerierbaren Überschreitungen kommen. Zum Beispiel, wenn bei der seriellen Kommunikation, viele Daten empfangen werden. Um das Flackern der roten LED *PLC* zu verhindern, kann diese mit einem ADSWRITE abgeschaltet werden.

Aufbau des Befehls ADSWRITE

Mit dem Befehl ADSWRITE ist es möglich, die rote LED *PLC* des BX-Controllers zu deaktivieren.

Eingangsparameter	Beschreibung
NETID	lokale NetId des BX
Port Nummer	800
IDXGRP	16#0000_4080
IDXOFFS	0
LEN	1 Bytes
SRCADDR	Pointer auf 1 Byte 0: rote LED ON 1: rote LED OFF

5.13.3.6 COM Port

5.13.3.6.1 Com Port - Übersicht

Die Bibliothek beinhaltet Funktionsbausteine die einen Datenaustausch zwischen dem **BXxxxx**-Bus-Controller und einem Remote-Partner ermöglichen. Der maximale COM-Puffer beträgt 512 Byte für beide Richtungen.

Funktionsbausteine

Name	Beschreibung
FB_ComPortOpen [▶ 118]	Eine serielle Verbindung zu einem Partner öffnen.
FB_ComPortClose [▶ 118]	Eine serielle Verbindung zu einem Partner schließen.

Funktionen

Name	Beschreibung
F_ComPortRead [▶ 117]	Daten aus dem COM-Puffer lesen
F_ComPortWrite [▶ 117]	Daten in COM-Puffer schreiben

Unterstützte Baudraten

Baudrate [Baud]	COM 1	COM 2
300	NEIN	JA
600	NEIN	JA
1200	NEIN	JA
2400	NEIN	JA
4800	NEIN	JA
9600	JA	JA
19200	JA	JA
38400	JA	JA
57600	JA	JA
115200	NEIN	JA

Weitere hilfreiche Bausteine finden Sie in der TcComPortBX.lbx [▶ 127].

- Baustein für die Nutzung der ComLib, ModbusRTU etc. für die BX Com Ports
- Baustein für die Kommunikation mit den BK8x00 Buskopplern
- Baustein für die Emulation eines BK8x00 Slaves

5.13.3.6.2 COM Port - Funktionen

COM Port Read

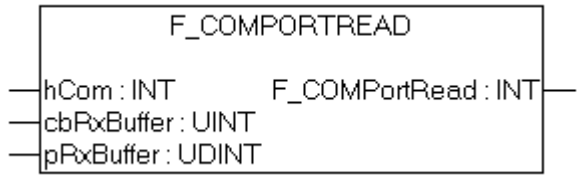


Abb. 101: Funktionsbaustein F_COMPORTREAD

FUNCTION F_COMPORTREAD

VAR_INPUT

```
hCom          :INT;
cbRxBuffer    :UINT;
pRxBuffer     :UDINT;
```

Legende

hCom: wir mit dem iHandle des FB_COMPORTOPEN verbunden

cbRxBuffer: Maximale Länge der Daten, die gelesen werden können.

pRxBuffer: Pointer auf die Daten, die mit dem COM Puffer Inhalt geschrieben werden sollen.

Rückgabewert	Bedeutung
> 0	Anzahl der Bytes, die vom COM Puffer in die PLC kopiert wurden
0x8000	Speicherüberlauf

COM Port Write

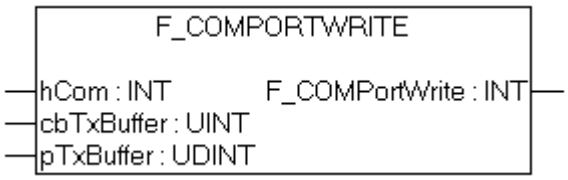


Abb. 102: Funktionsbaustein F_COMPORTWRITE

FUNCTION F_COMPORTWRITE

VAR_INPUT

```
hCom          :INT;
cbTxBuffer    :UINT;
pTxBuffer     :UDINT;
```

Legende

hCom: wir mit dem iHandle des FB_COMPORTOPEN verbunden

cbTxBuffer: Anzahl der Datenbytes, die in den COM Puffer kopiert wurden.

pTxBuffer: Pointer auf die Daten, aus denen der COM Puffer gefüllt werden soll.

Rückgabewert	Bedeutung
> 0	Anzahl der Bytes, die vom der PLC in den COM Puffer kopiert wurden
0x8000	Speicherüberlauf

5.13.3.6.3 COM Port - Funktionsbaustein

COM Port Open

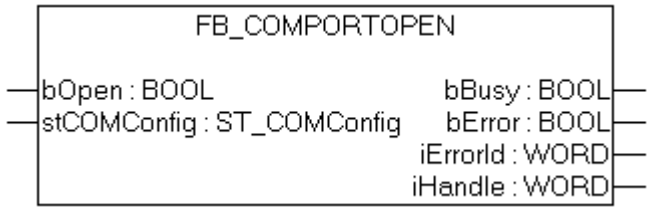


Abb. 103: Funktionsbaustein FB_COMPORTOPEN

FUNCTION_BLOCK FB_COMPORTOPEN

VAR_INPUT

```
bOpen         :BOOL;
stComConfig   :ST_COMCONFIG;
```

Legende

bOpen: Positive Flanke startet den Baustein
stComConfig [▶ 119]: Datenstruktur COM Schnittstelle.

VAR_OUTPUT

```
bBusy      :BOOL;
bErr       :BOOL;
iErrId     :WORD;
iHandle    :WORD;
```

Legende

bBusy: So lange der Baustein TRUE ist der Baustein aktiv.
bErr: Fehler Bit.
iErrId: Fehler Nummer.
iHandle: Pointerübergabe.

Rückgabeparameter iErrId	Bedeutung
0	kein Fehler
-1, 0xFFFF	Falscher COM-Port
-2, 0xFFFE	Fehlerhafte oder nicht unterstützte Baudrate. Überprüfen Sie den Parameter stComConfig.BaudRate.
-3, 0xFFFD	Fehlerhaftes oder falsches Datenformat. Überprüfen Sie den Parameter stComConfig.
-4, 0xFFFC	Fehlerhaft Initialisierung der COM-Schnittstelle
-5, 0xFFFB	Nicht unterstützter Instanz
-6, 0xFFFA	Falsche Größe des RX-Buffers
-7, 0xFFF9	Falsche Größe des TX-Buffers
-8, 0xFFF8	COM-Port ist gesperrt

COM Port Close

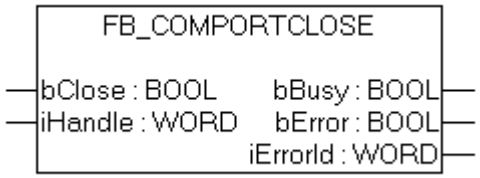


Abb. 104: Funktionsbaustein FB_COMPORTCLOSE

FUNCTION_BLOCK FB_COMPORTCLOSE

VAR_INPUT

```
bOpen         :BOOL;
iHandle       :WORD;
```

Legende

bClose: Positive Flanke startet den Baustein
iHandle: Pointerübergabe von FB_COMPORTOPEN.

VAR_OUTPUT

```
bBusy       :BOOL;
bErr        :BOOL;
iErrId      :WORD;
```

Legende

bBusy: So lange der Baustein TRUE ist der Baustein aktiv.
bErr: Fehler Bit.
iErrId: Fehler Nummer.

Rückgabeparameter iErrId	Bedeutung
0	kein Fehler
> 0	Fehler Nummer (#nicht dokumentiert#)

5.13.3.6.4 Datenstruktur ComConfig

Die Einstellungen für die seriellen Schnittstellen des BX werden mit der folgenden Datenstruktur übergeben.

```
TYPE ST_COMConfig:
STRUCT
    cbRxBufferLen   :WORD;
    cbTxBufferLen   :WORD;
   dwMode           :DWORD;
   BaudRate         :DWORD;
   eCommPort        :E_CommPort;
   eDataBits        :E_DataBits;
   eParity          :E_Parity;
   eStoppBits       :E_StoppBits;
END_STRUCT
END_TYPE
```

Legende

cbRxBufferLen: hat keine Bedeutung (ist aus kompatibilitäts- Gründen beibehalten worden)
cbTxBufferLen: hat keine Bedeutung (ist aus kompatibilitäts- Gründen beibehalten worden)
dwMode: Daten Mode COM 1 nur "0" - COM 2 RS232 "0" und RS485 "1"
BaudRate: Baudrate
eCommPort: Com Port COM1/COM2

eDataBits: Anzahl der Datenbits SEVEN_DATABITS/EIGHT_DATABITS
eParity: EVEN/ODD/NONE
eStoppBits: Anzahl der Stoppbits ONE_STOPPBIT/TWO_STOPPBITS

5.13.3.6.5 Beispiel

Beispiel Programm in ST

Download (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207320715.zip)

```
PROGRAM MAIN
VAR
(* EXAMPLE - BRIDGE between PIN 7 and 8 from X01 COM 2 Port*)
   fb_COMPortOpen_1               : FB_COMPortOpen;
   stCOMConfig_1                  : ST_COMConfig;
   hCOM                           : WORD;
   Result_R                       : INT;
   Result_W                       : INT;
   Var_M                          : ARRAY[0..9] OF BYTE:=11,22,0,33,0(6);
   Var_R                          : ARRAY[0..9] OF BYTE;
   Value                          : INT;
   Counter_V                      : BYTE; (* It is all OK, this value counts up *)
   i                              : INT;
   i_k                            : INT;
   fbTimer                        : TON;
END_VAR
```

```
stCOMConfig_1.cbRxBufferLen :=300;
stCOMConfig_1.cbTxBufferLen :=300;
stCOMConfig_1.dwMode :=0;
stCOMConfig_1.BaudRate :=19200;
stCOMConfig_1.eCommPort :=COM2;
stCOMConfig_1.eDataBits:=EIGHT_DATABITS;
stCOMConfig_1.eParity:=EVEN;
stCOMConfig_1.eStoppBits:=ONE_STOPPBIT;


CASE i OF
(* Open Port *)
0:  fb_COMPortOpen_1(bOpen:=TRUE , stCOMConfig:=stCOMConfig_1);
    IF NOT fb_COMPortOpen_1.bBusy THEN
        IF NOTfb_COMPortOpen_1.bError THEN
           hCOM:=fb_COMPortOpen_1.iHandle ;
           i:=i+1;
        ELSE

i:=100;
        END_IF
    END_IF
(* Write data*)
1:  fbTimer(IN:=FALSE);
    Result_W:=F_COMPortWrite(hCom, 4,ADR(Var_M[0]));
    IF Result_W>0 THEN
        i:=i+1;
       Var_M[2]:=Var_M[2]+1;
    ELSE
        i:=101;
   END_IF
(*Receive data*)
2:  Result_R:=F_COMPortRead(hCom, 100,ADR(Var_R[Value]));
    IF Result_R<>0 THEN
       Value:=Result_R+Value;
    END_IF
    IF Value>=4 THEN
        FOR i_k:=0 TO Value DO(*Check protocol*)
           IF Var_R[i_k-4]=11 AND Var_R[i_k-3]=22 AND Var_R[i_k-1]=33 THEN
               Counter_V:=Var_R[i_k-2];
               i:=1;
               Value:=0;
           END_IF
        END_FOR
    END_IF
    fbTimer(IN:=TRUE,PT:=t#1s); (*Watchdog receive*)
    IF fbTimer.Q THEN
       fbTimer(IN:=FALSE);
        i:=102;
    END_IF
100: ; (*ERROR open port*)
```

```
101: ; (*ERROR send data*)
102: i:=1; (*WD ERROR no data receive*)
END_CASE
```

5.13.3.7 BX Debug-Funktion

Diese Funktionen können benutzt werden um Befehls-Ausführungszeiten in einem PLC-Projekt zu messen. Die Einheit ist ein Tick. Ein Tick entspricht 5.12 µs.

Funktion Start Debug Timer

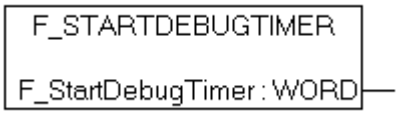


Abb. 105: Funktionsbaustein F_STARTDEBUGTIMER

Der Aufruf dieser Funktion startet den Timer. Der Rückgabe Wert ist "0".

Funktion Read Debug Timer

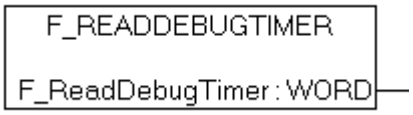


Abb. 106: Funktionsbaustein F_READDEBUGTIMER

Mit dieser Funktion wird der Timer-Wert gelesen. Der Rückgabewert muss mit 5.12 µs multipliziert werden.

Beispiel

```
VAR
    Timer_BX     :WORD;
    i            :INT;
END_VAR
```

Programm

```
F_STARTDEBUGTIMER();
For i:=0 to 1000 do
    ;
END_FOR
Timer_BX:=F_READDEBUGTIMER();
```

5.13.3.8 Navigations-Schalter

5.13.3.8.1 FUN GetNavSwitch

Dieser Funktionsbaustein ermöglicht es Ihnen das Auslesen des Navigationsschalters.

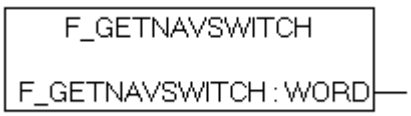


Abb. 107: Funktionsbaustein F_GETNAVSWITCH

VAR_Output

```
F_GETNAVSWITCH    :WORD;
```

Legende

F_GETNAVSWITCH: Daten des Schalters

Beschreibung des WORD

Bit	15	14	...	5	4	3	2	1	0
Name	LOCKED	-	...	-	PRESS	RIGHT	LEFT	DOWN	UP

Wenn Bit 15 gesetzt ist befindet man sich im Menü des BX-Controllers. Dieses Bit ist solange gesetzt, solange der Anwender sich im Menü des BX3100 befindet. Sobald das Menü verlassen wurde, wird der Navigationsschalter sofort wieder für die PLC freigegeben, d.h. dass das Drücken des Press Buttons noch in dem Programm sichtbar ist. Bitte berücksichtigen Sie das in Ihrer Anwendung. Werten Sie zum Beispiel den Schalter erst einer kurzen warte Zeit aus, in dem Sie mit der fallenden Flanke von Bit 15 einen Timer starten.

Download Beispiel Programm in ST https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207322891.zip

5.13.3.9 Display

5.13.3.9.1 FB DISPWRITE

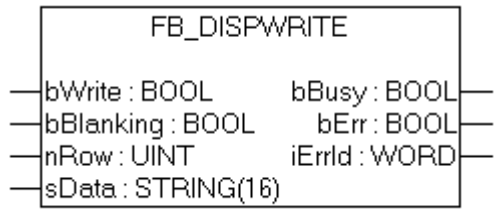


Abb. 108: Funktionsbaustein FB_DISPWRITE

VAR_INPUT

```
bWrite      :BOOL;
bBlanking   :BOOL;
nRow        :UINT;
sData       :STRING(16)
```

Legende

bWrite: Positive Flanke startet den Baustein

bBlanking: FALSE Hintergrundbeleuchtung an, TRUE Hintergrundbeleuchtung aus, Default ist diese an (wird FW 1.15 bei allen BX Controllern unterstützt).

nRow: Zeile im Display 1 oder 2.

sData: String der im Display angezeigt wird

VAR_OUTPUT

```
bBusy       :BOOL;
bErr        :BOOL;
iErrId      :WORD;
```

Legende

bBusy: So lange der Baustein TRUE ist der Baustein aktiv.

bErr: Fehler Bit.

iErrId: Fehler Nummer.

Rückgabeparameter	Bedeutung
0	kein Fehler
> 0	Fehler Nummer

Beispiel Programm in ST

Download https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207325067.zip

```
PROGRAM MAIN
VAR
    fb_DispWrite1: FB_DispWrite;

i:                 INT;
    udiCounter:    UDINT;
    strCounter:    STRING;
    strLine:       STRING;
   k:              INT;
END_VAR
```

```
CASE i OF
0:  strCounter:=CONCAT('Counter :',UDINT_TO_STRING(udiCounter));
    fb_DispWrite1(bWrite:=TRUE , nRow:=1 ,sData:=strCounter );
    IF NOT fb_DispWrite1.bBusy THEN
        IF NOTfb_DispWrite1.bErr THEN
           fb_DispWrite1(bWrite:=FALSE);
           udiCounter:=udiCounter+1;
           i:=1;
        END_IF
    END_IF
1:  fb_DispWrite1(bWrite:=TRUE , nRow:=2 , sData:=strLine);
    IF NOT fb_DispWrite1.bBusy THEN
        IF NOTfb_DispWrite1.bErr THEN
           fb_DispWrite1(bWrite:=FALSE);
           k:=k+1;
           strLine:=REPLACE(' ','#',1,k);
           IF k=16 THEN
               k:=0;
           END_IF
           i:=0;
        END_IF
    END_IF
END_CASE
```

Display ASCII Tabelle

Beispiel für das Zeichen "&" (siehe Zeile 1 Spalte 7): $00100110_{bin} = 38_{dez} = 26_{hex}$. Dies entspricht in dem SPS-Wert '$26' (String.)

https://infosys.beckhoff.com/content/1031/bx9000/Resources/pdf/3207327243.pdf

5.13.4 TcSystemBX

5.13.4.1 Echtzeit-Uhr - Beispiel

Auf dem BX-Controller ist eine Echtzeit-Uhr implementiert. Die aktuelle Uhrzeit kann per Funktionsbaustein ausgelesen werden. Das folgende Beispiel soll das verdeutlichen.

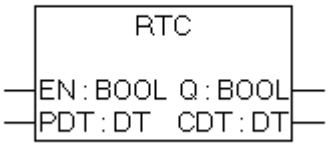


Abb. 109: Funktionsbaustein RTC

FUNCTION_BLOCK RTC

VAR_INPUT

```
EN          :BOOL;
PDT         :DT;
```

Legende

EN: Positive Flanke setzt die Uhrzeit auf den Wert der an den PDT-Eingang anliegt.

PDT: Datum und Uhrzeit die eingestellt werden sollen.

VAR_OUTPUT

```
Q           :BOOL;
CDT         :BOOL;
```

Legende

CDT: Aktuelle Uhrzeit.

Notwendige Bibliotheken:

- TcSystemBX.lb6
- TcBaseBX.lb6

Download Beispiel Programm in ST (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207329419.zip)

```
PROGRAM MAIN
VAR
    fbTimer: TON;
    fbRTC: RTC;
END_VAR
```

```
fbTimer(PT:=t#60s,IN:=NOT fbTimer.Q);

IF fbTimer.Q THEN
    fbRTC;
END_IF
```

RTC-Baustein nicht in jedem SPS-Zyklus aufrufen

Der Aufruf des RTC-Bausteins erhöht die Zykluszeit um ca. 5 ms, verursacht durch die Datenkonvertierung in eine TIME AND DATE Variable. Deshalb sollte der Baustein nicht in jedem SPS-Zyklus aufgerufen werden!
Alternativ können Sie die Uhrzeit auch per ADS-Baustein auslesen. Der ADS-Baustein liefert Datum und die Uhrzeit als WORD-Variablen zurück.
Beispiel 19:30 Uhr - Hour: 19 / Minute: 30

5.13.4.2 Laden und Speichern von Rezepturen

Der Funktionsbaustein fb_ReadWriteFile ermöglicht Daten (max. 16.000 Byte) dauerhaft im Flash-Speicher des BX-Controllers zu sichern. Ein neues Programm oder ein Reset des Projektes lässt diesen Speicherinhalt unberührt. Dieser Funktionsbaustein ist nicht für den Dauerhaften und ständigen Gebrauch geeignet. Es sind max. 10000 Schreibzyklen erlaubt. Es darf beliebig oft gelesen werden.

Anwendung: Sichern von Rezepten oder Einstellungen die sich selten oder gar nicht ändern, zum Beispiel Reglerparameter.

Beachten Sie beim Schreiben der Daten

- Während des Schreibens darf die Spannung nicht unterbrochen werden, es empfiehlt sich daher, das Schreiben von einem Bedien-Panel oder über die Navigationsschalter oder einfach über einen digitalen Eingang anzustoßen, um sicher zu gehen, dass während des Schreibens der BX-Controller nicht ausgeschaltet wird. Ein automatisches Schreiben empfiehlt sich nicht, da man nicht sicherstellen kann, ob nicht grade in diesem Moment das Schreiben unterbrochen wird.
- Das Schreiben der Daten benötigt ca. zwei Sekunden, unabhängig von der Anzahl der Daten, die geschrieben werden.
- Wird der BX-Controller während des Schreibvorgangs ausgeschaltet gehen die Daten verloren.
- Es ist nur eine Instanz dieses Bausteins erlaubt.

Funktionsbaustein fb_ReadWriteFile

Funktionsbaustein zum Lesen und Schreiben von Rezepturen

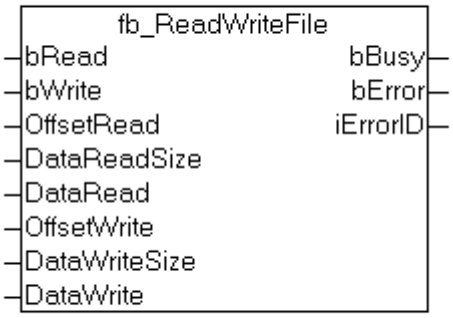


Abb. 110: Funktionsbaustein fb_ReadWriteFile

VAR_INPUT

```
bRead           :BOOL;
bWrite          :BOOL;
OffsetRead      :WORD;
DataReadSize    :WORD;
DataRead        :Pointer to Byte;
OffsetWrite     :WORD;
DataWriteSize   :WORD;
DataWrite       :Pointer to Byte;
```

Legende

bRead: Eine positive Flanke stößt das Lesen des Bausteins an (bWrite muss FALSE sein)
bWrite: Eine positive Flanke stößt das Schreiben des Bausteins an (bRead muss FALSE sein)
OffsetRead: Offset im Speicher max.16.000 Byte
DataReadSize: Größe der Daten in Byte, die gelesen werden sollen (max. 16.000 Byte)
DataRead: Pointer per ADR auf die Daten zeigen

OffsetWrite: Offset im Speicher max.16.000 Byte
DataWriteSize: Größe der Daten in Byte, die geschrieben werden sollen (max. 16.000 Byte)
DataWrite: Pointer per ADR auf die Daten zeigen

VAR_OUTPUT

```
bBusy           :BOOL;
bError          :BOOL;
bErrorId        :UDINT
```

Legende

bBusy: Zeigt an des der Baustein noch aktiv ist
bError: Baustein hat einen Fehler
bErrorId: Fehlernummer

Rückgabeparameter iErrorId	Bedeutung
0	kein Fehler
1_{dez}	READ: Daten-Offset und Daten-Länge über 16.000 Byte
2_{dez}	WRITE: Daten-Offset und Daten-Länge über 16.000 Byte
0x31440708	CRC-Fehler im Datenspeicher
0x31470708	Das Schreiben der Daten ist noch nicht abgeschlossen

Notwendige Bibliotheken:

- TcSystemBX.lb6
- TcBaseBX.lb6

Download Beispiel Programm in ST (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207331595.zip)

5.13.5 TcComPortBX

5.13.5.1 TcComPortBX - Übersicht

Erforderliche Bibliotheken:

- TcBaseBX
- TcSystemBX

Überblick

Name	Beschreibung
fb_BX_BK8x00 Master [▶ 128]	BK8x00 COM Port Funktionsbaustein, Kommunikation mit Buskoppler BK8x00 oder BC8x00
fb_BX_BK8x00 Slave [▶ 128]	BK8x00 COM Port Funktionsbaustein, Kommunikation mit PC. Es wird ein BK8x00 simuliert.

Name	Beschreibung
FB_BX_COM_5 [▶ 132]	Baustein zum Emulieren einer KL60x1(wenn COMlib.lb6 oder ModbusRTU.lb6 verwendet wird).
FB_BX_COM_64 [▶ 132]	Baustein zum Emulieren einer PC-Schnittstelle (wenn COMlib.lib oder ModbusRTU.lib verwendet wird).
FB_BX_COM_64ex [▶ 133]	Baustein zum Emulieren einer PC-Schnittstelle (wenn COMlib.lib oder ModbusRTU.lib verwendet wird). Hier kann die COM-Schnittstelle auch während des Betriebs geschlossen werden.

5.13.5.2 BK8x00 - FB COM-Port

Über diesen Funktionsbaustein kann über die serielle Schnittstelle des BXxxxx die seriellen Buskoppler BK8000 mit RS485 und BK8100 mit RS232-Anschluss verwendet werden. Die Maximale Baudrate beträgt 38400 Baud.

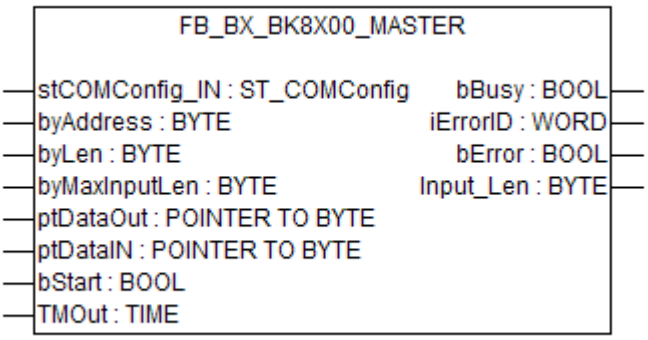


Abb. 111: Funktionsbaustein FB_BX_BK8X00_MASTER

VAR_INPUT

```
stCOMConfig        :ST_COMConfig;
byAddress          :BYTE;
byLen              :BYTE;
byMaxInputLen      :BYTE;
ptDataOut          :POINTER TO BYTE;
ptDataIN           :POINTER TO BYTE;
bStart             :BOOL;
TmOut              :TIME;
```

Legende

stComConfig: Struktur für die Auswahl der COM Parameter
byAddress: BX8x00 Adresse 1-98 (0 und 99 sind reserviert)
byLen: Daten Länge in [BYTE] (es sind nur grade Zahlen zugelassen 0, 2, 4, ...)
byMaxInputLen: wird mit SIZEOF und der Variable, die mit ptDataIN verbunden ist, angeschlossen
ptDataOut: wird mit ADR und den Daten Out verbunden
ptDataIn: wird mit ADR und den Daten In verbunden
bStart: positive Flanke startet den Baustein
TMOut: Wartezeit bis abgebrochen wird

VAR_OUTPUT

```
bBusy      :BOOL;
bError     :BOOL;
iErrorId   :WORD;
Input_len  :WORD;
```

Legende

bBusy: So lange der Baustein TRUE ist der Baustein aktiv
bError: Fehler Bit
iErrorId: Fehler Nummer
Input_Len: Anzahl der Daten die Empfangen wurden

Rückgabeparameter iErrorId	Bedeutung
0	kein Fehler
100_{dez}	Fehler beim Öffnen des COM-Ports
101_{dez}	Fehler beim Senden der Daten
102_{dez}	Watchdog Fehler, keine Antwort vom Slave innerhalb der WD-Zeit
105_{dez}	Der Input Puffer ist zu klein
200_{dez}	CRC-Fehler
$0x80xx_{hex}$	Buskoppler Fehler xx Status Byte des Buskopplers (siehe BX8x00 Dokumentation)

Hardware

RS 232 Kommunikation PIN Belegung

BX COM1 RS232	BX COM2 RS232	BK8100
2	8	2
3	7	3
5	5	5

RS485 Kommunikation PIN Belegung

Einstellungen im FB: Wichtig ist das bei der Nutzung der RS485-Verbindung die Variable stCOMConfig:=1 ist und die COM2-Schnittstelle angewählt ist.

BX COM2 RS485	BK8000
1	3
6	8

Beispiel Programm in ST

Download (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207333771.zip)

Notwendiges Material:

- BX3100 + Busklemme
- BK8100, KL1xx8, KL2xx8, KL9010
- Serielles Kabel, PIN Belegung: siehe im Beispielprogramm

5.13.5.3 BK8x00 - FB Slave COM-Port

Mit diesem Funktionsbaustein kann der PC (TwinCAT oder KS8000) über die serielle Schnittstelle mit dem BXxxxx verbunden werden. Dabei ist der PC der serielle Master und der BXxxxx emuliert mit Hilfe des Funktionsbausteins eines BK8x00.

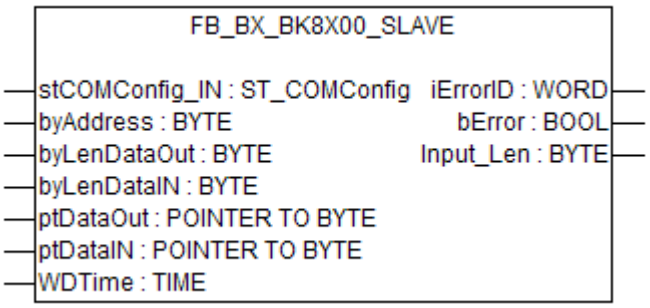


Abb. 112: Funktionsbaustein FB_BX_BK8X00_SLAVE

VAR_INPUT

```
stCOMConfig       :ST_COMConfig;
byAddress         :BYTE;
byLenDataOut      :BYTE;
byLenDataIN       :BYTE;
ptDataOut         :POINTER TO BYTE;
ptDataIN          :POINTER TO BYTE;
WDTime            :TIME;
```

Legende

stComConfig: Struktur für die Auswahl der COM Parameter
byAddress: BX8x00 Adresse 1-98 (0 und 99 sind reserviert)
byLenDataOut: Daten Länge in [BYTE] (es sind nur grade Zahlen zugelassen 0, 2, 4, ...)
byLenDataIn: Daten Länge in [BYTE] (es sind nur grade Zahlen zugelassen 0, 2, 4, ...)
ptDataOut: wird mit ADR und den Daten Out verbunden
ptDataIn: wird mit ADR und den Daten In verbunden
WDTime: Fehlermeldung wenn innerhalb der Watchdog - Zeit keine neuen Daten empfangen wurden (0 ms disable WD)

VAR_OUTPUT

```
bError        :BOOL;
iErrorId      :WORD;
Input_Len     :BYTE;
```

Legende

bError: Fehler Bit
iErrorId: Fehler Nummer
Input_Len: Anzahl der Daten die Empfangen wurden

Rückgabeparameter iErrorId	Bedeutung
0	kein Fehler
1	Watchdog Fehler wenn diese größer 0 ms ist (WD disable wenn 0 ms)
100_{dez}	Fehler beim Öffnen des COM Ports
101_{dez}	Fehler beim Senden der Daten
103_{dez}	Interner Recive Buffer übergelaufen
104_{dez}	Daten passen nicht in den PLC Buffer (größer 500 Byte)
105_{dez}	Daten können nicht in den PLC Buffer kopiert werden
200_{dez}	CRC - Fehler

Hardware

RS232-Kommunikation PIN-Belegung

BX COM 1 RS232	BX COM 2 RS232	PC COM Schnittstelle
2	8	2
3	7	3
5	5	5

RS485-Kommunikation PIN-Belegung

Einstellungen im FB: Wichtig ist das bei der Nutzung der RS485-Verbindung die Variable stCOMConfig:=1 ist und die COM2-Schnittstelle angewählt ist.

BX COM 2 RS485	PC COM Port (zum Beispiel: RS485 Karte W&T #13601, 2-Draht, ohne Echo, Automatik)
1	1 - 2 Brücken
6	6 - 7 Brücken

Beispiel Programm für den BXxxx in Strukturiertem Text

Download (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207335947.zip)

System Manager File für TwinCAT als Master. Wie man im Bild sieht, wird ein Buskoppler mit Busklemmen projektiert. Die Art und Anzahl der Busklemmen gibt dann die Datenlänge an. Welche Busklemmen das sind spielt im Prinzip keine Rolle. Zum Beispiel:

- 2 x KL3002 ergibt 4 Worte Eingänge
- 2 x KL4002 ergibt 4 Worte Ausgänge

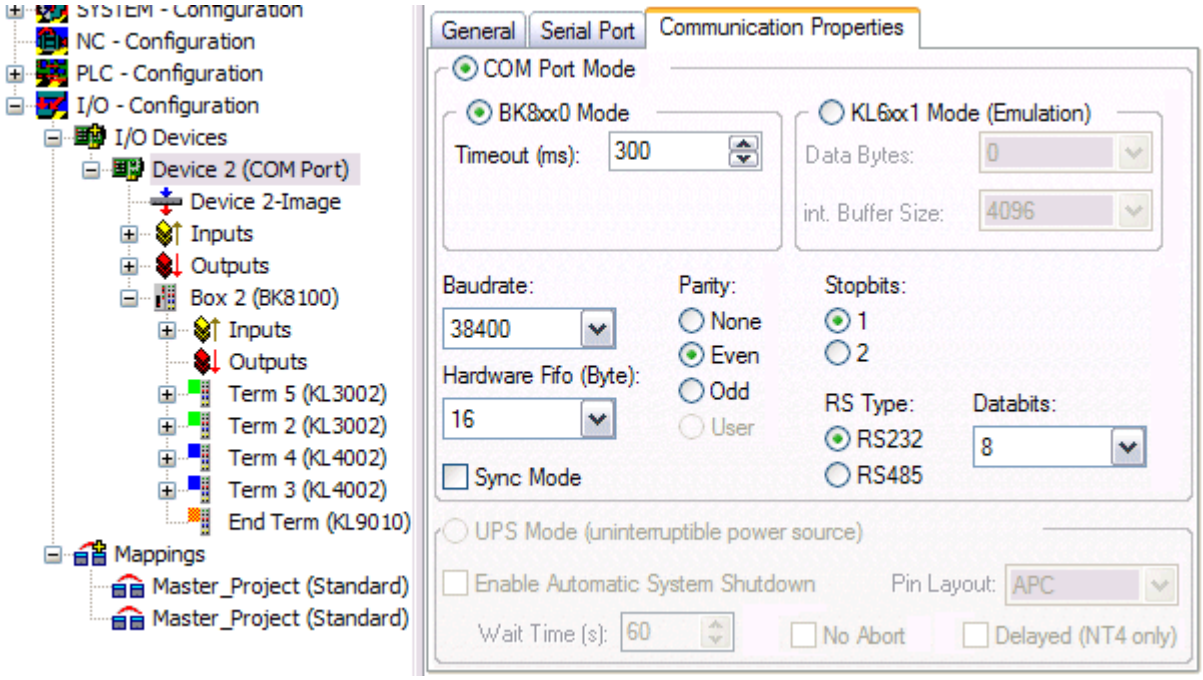


Abb. 113: Kommunikationseigenschaften

Notwendiges Material:

- BX3100 + Busklemme
- PC mit RS232-Schnittstelle und TwinCAT ab Version 2.9, serielles Kabel, PIN-Belegung: siehe oben

5.13.5.4 FB_BX_COM_5

Dieser Baustein verbindet die ModbusRTU.lb6, ModbusRTU.lib oder die ComLib.lb6 mit der seriellen Schnittstelle des BX-Controllers. Es wird eine KL60x1 Emuliert, die Daten werden nicht auf eine Busklemme ausgegeben, sonder über eine der beiden auf dem BX befindlichen seriellen Schnittstellen.

Abb. 114: Funktionsbaustein FB_BX_COM_5

VAR_INPUT

```
pstrEmo_IN          :POINTER TO BYTE;
pstrEmo_OUT         :POINTER TO BYTE;
ComConfig           :ST_COMConfig;
```

Legende

pstrEmo_IN: Pointer auf die KL6inData5B
pstrEmo_OUT: Pointer auf die KL6outData5B
ComConfig [▶ 119]: Parametrierung der COM Schnittstelle

Download Beispielprogramm in ST für die Verknüpfung COMLib zum BX: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207338123.zip)

5.13.5.5 FB_BX_COM_64

Dieser Baustein verbindet die ModbusRTU.lib oder ComLib.lib mit der seriellen Schnittstelle des BX-Controllers. Es wird eine PC-Schnittstelle emuliert. Die Daten werden nicht auf einer PC-Schnittstelle ausgegeben, sondern über eine der beiden auf dem BX befindlichen seriellen Schnittstellen (COM1 oder COM2).

Abb. 115: Funktionsbaustein FB_BX_COM_64

VAR_INPUT

```
pstrEmo_IN          :POINTER TO BYTE;
pstrEmo_OUT         :POINTER TO BYTE;
ComConfig           :ST_COMConfig;
```

Legende

pstrEmo_IN: Pointer auf die ModbusPCComInData
pstrEmo_OUT: Pointer auf die ModbusPCComInData
ComConfig [▶ 119]: Parametrierung der COM Schnittstelle

Download Beispielprogramm in ST für die Verknüpfung ModbusRTU zum BX: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207340299.zip)

Download Beispielprogramm in ST für die Verknüpfung ModbusRTU Version 2 zum BX: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207342475.zip)

Für das Beispiel ist die ModbusRTU-Bibliothek notwendig!

5.13.5.6 FB_BX_COM_64ex

Dieser Baustein verbindet die ModbusRTU.lib oder ComLib.lib mit der seriellen Schnittstelle des BX-Controllers. Es wird eine PC Schnittstelle mit 64 Byte Nutzdaten emuliert. Die Daten werden nicht auf eine PC-Schnittstelle ausgegeben, sondern über eine der beiden auf dem BX befindlichen seriellen Schnittstellen (COM1 oder COM2).

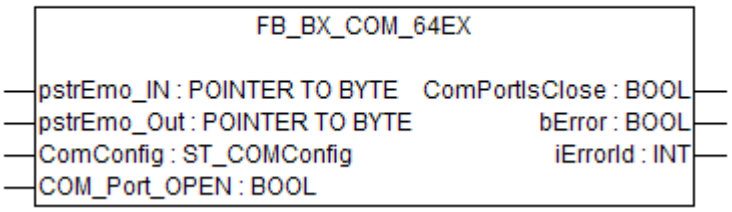


Abb. 116: Funktionsbaustein FB_BX_COM_64EX

VAR_INPUT

```
pstrEmo_IN          :POINTER TO BYTE;
pstrEmo_OUT         :POINTER TO BYTE;
ComConfig           :ST_COMConfig;
```

VAR_OUTPUT

```
ComPortIsClose      :BOOL;
bError              :BOOL;
iErrorId            :INT;
```

Legende

pstrEmo_IN:
Pointer auf die ModbusPCComInData
pstrEmo_OUT:
Pointer auf die ModbusPCComOutData
ComConfig [▶ 119]**:**
Parametrierung der COM Schnittstelle
COM_Port_Open:
Wird dieses Bit gesetzt, wird die Schnittstelle geöffnet. Wird dieses Bit zurückgesetzt wird die COM Schnittstelle geschlossen.
ComPortIsClose:
Wenn die Schnittstelle geschlossen ist, ist diese Bit gesetzt
bError:
Es liegt ein Fehler vor
iErrorId:
Fehlercode (siehe FB_COMPortOpen) [▶ 118]

Download Beispielprogramm in ST für die Verknüpfung ModbusRTU zum BX: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207344651.zip)

Für das Beispiel ist die ModbusRTU Bibliothek erforderlich!

5.13.5.7 Weitere Beispiele

5.13.5.7.1 BX COM-Port als ModbusRTU-Master

Die serielle Schnittstelle des BX kann auch dazu benutzt werden, diese als Modbus-Master zu verwenden.

Benötigte Komponenten

1 x BX3100
Busklemmen für den K-Bus (hier beliebig, da diese für das Beispiel nicht gebraucht werden)
1 x BK7300
2 x KL2xx4
2 x KL1xx4
1 x KL9010

Kabel RS 485*

BX3100 COM 2 / RS 485	BK7300 / RS 485
1	3
6	8

*) aktiver Abschlusswiderstand ist bei kurzen Leitungslängen (< 5 m) und kleinen Baudraten (<19200 Baud) nicht erforderlich

Download Beispiel Programm in ST für die Verknüpfung ModbusRTU Master zum BX: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207346827.zip**)**

Download Beispiel Programm in ST für die Verknüpfung ModbusRTU Master Version 2 zum BX: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207349003.zip**)**

Für das Beispiel ist die ModbusRTU, TcComPortBC, TcBaseBX Bibliothek notwendig.
Baudrate 9600, n, 8,1 Default BK7300,
BK7300 Adresse 11

Reaktionszeiten

Die Reaktionszeiten sind abhängig von der eingestellten Task-Zeit, die Anzahl an Slaves, die Länge der Modbus-Telegramme und der Antwortzeit der Slaves.
Für die Ermittlung folgender Tabelle wurden Beckhoff BK7300 Modbus Slaves verwendet. Da dies nicht auf jeden Slave übertragbar ist, sollte diese Tabelle nur als Richtwert verwendet werden.

Baudrate 38400 Baud (jeweils ein Read Reg. und ein Write Reg. Telegramm pro Slave)

Slave-Anzahl	Task-Zeit auf dem BX	Zeit für einen Durchlauf
1	5	100 ms* / 125 ms**
2	5	200 ms / 225 ms
1	10	180 ms / 220 ms
2	10	350 ms / 390 ms
1	20	350 ms / 350 ms
2	20	700 ms / 700 ms

*) 2 Worte Ein- und 2 Worte Ausgänge
**) 20 Worte Ein- und 20 Worte Ausgänge

5.13.5.7.2 BX COM-Port - ComLibV2

Beispiele für ComLibV2 senden von Strings über die interne COM Schnittstelle des BX Controllers. Um Daten zu empfangen kann man eine Brücke vom PIN 7 und 8 herstellen auf X01 (COM2).

Notwendiges Material

Hardware:

- BX-Controller

Software:

- TwinCAT ab 2.10
- COMlibV2.lib
- TcComPortBX.lbx
- Standard.lbx
- TcBase.lbx
- TcSystemBX.lbx

Download Beispielprogramm BX: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207351179.zip)

5.13.5.7.3 BX COM-Port - Cimrex-Panel

Die serielle Schnittstelle des BX-Controllers kann auch als Modbus-Slave benutzt werden. In diesem Beispiel wird ein Panel der Firma Beijers angeschlossen. Weitere Informationen zu dem Panel finden Sie unter http://www.beijerelectronics.de.

Abb. 117: Cimrex-Panel am COM-Port des BX-Controllers

Benötigte Komponeten

- 1 x BX3100
- 1 x Cimrex 12
- beliebige Busklemmen (beliebig, da diese im Beispiel nicht benutzt werden)

Kabel RS232

BX3100 COM 2 / RS485	Cimrex 12 RS232
7	2
8	3
9	5

Kabel RS485*

BX3100 COM 2 / RS 485	Cimrex 12 RS485
1	2 -3
6	15 -16

*) aktiver Abschlusswiderstand ist bei kurzen Leitungslängen (≤ 5 m) und kleinen Baudraten (≤19200 Baud) nicht erforderlich

Download Beispielprogramm in ST für den BX: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207340299.zip)

Download Beispiel mit Cimrex-Panel: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207353355.zip)

Für das Beispiel sind die Bibliotheken ModbusRTU, TcComPortBC und TcBaseBX erforderlich.
- Baudrate 9600,n,8,1 D
- Cimrex 12

5.13.5.7.4 BX COM-Port - RK512-Protokoll

Über die Schnittstelle COM1 oder COM2 des BX-Controllers kann das RK512-Protokoll Daten mit einer Gegenstelle austauschen. Dokumentation zum RK512-Baustein finden Sie im Beckhoff Information System. Über die 64 Byte Emulation des BX-Controllers wird die serielle Schnittstelle des PC simuliert.

Notwendiges Material

Hardware:

- PC mit RS232 Schittstelle und TwinCAT PLC ab 2.9
- BX-Controller
- Serielles Kabel für die Verbindung BX - PC

Software:

- TwinCAT ab 2.9
- COMlib.lib
- COMlib3964R.lib
- COMlibRK512.lib
- TcComPortBX.lbx
- Standard.lbx
- TcBase.lbx
- TcSystemBX.lbx
- ChrAscBx.lbx

 Download Beispielprogramm BX3100: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207355531.zip)

 Download Beispielprogramm PC: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207357707.zip)

 Download Beispiel System Manager File PC: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207359883.zip)

5.13.5.7.5 BX COM-Port - SMS über Mobiltelefon

Die serielle Schnittstelle kann auch dafür verwendet werden, vom BX-Controller aus eine SMS zu versenden. Das folgende Beispiel benutzt die SMS-Bibliothek mit einem Siemens S35 Mobiltelefon.

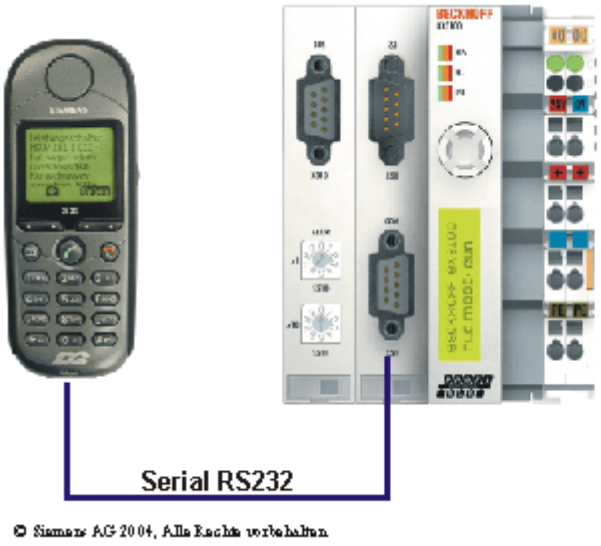


Abb. 118: Mobiltelefon am COM-Port des BX-Controllers

Download: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207362059.zip)

Pinbelegung (Siemens-Kabel S30880-S4501 A801-2)

S35	COM 1	COM 2
2	3	7
3	2	8
5	5	9

5.13.6 TcTwinSAFE

5.13.6.1 Übersicht

Die Busklemmen-Controller der BX-Serie unterstützen die TwinSAFE-Busklemmen wenn folgende Vorraussetzungen erfüllt sind:

- Am Busklemmen-Controller ist nur eine Logik-Klemme zugelassen und diese muss am K-Bus-Interface angeschossen sein, nicht am SSB.
- An dieser Logik-Klemme sind maximal 7 Verbindungen zugelassen.
- TwinSAFE-Ein- und Ausgangsklemmen können an den K-Bus wie auch an den SSB angeschossen werden, zum Beispiel über BK5120 oder BK515x.
- Falls Online-Changes verwendet werden soll, muss der Connection-Timeout auf 500 ms oder höher eingestellt werden.
- Für den Download der TwinSAFE-Projekte muss eine ADS-Verbindung bestehen. Diese kann seriell aber auch über den Feldbus hergestellt werden.
- Die Firmware-Version des Busklemmen-Controllers muss 1.17 oder höher sein.

TwinSAFE-Bibliothek

Die TwinSAFE-Bibliothek beinhaltet Funktionsbausteine, mit denen Dienste/Funktionen in Zusammenhang mit den TwinSAFE-Klemmen KL1904, KL2904 und KL6904 ausgeführt werden können.

Name	Beschreibung
F_GetVersionTcTwinSAFE [▶ 139]	Versionsnummer der Bibliothek
FB_TwinSAFE_KLx904_input [▶ 139]	Auswertung der TwinSAFE Daten, welche von einer KL1904 oder KL2904 zu einer KL6904 gesendet werden
FB_TwinSAFE_KLx904_output [▶ 142]	Auswertung der TwinSAFE Daten, welche von einer KL6904 zu einer KL1904 oder KL2904 gesendet werden

Download der TwinSAFE-Bibliothek: (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207364235.zip)

5.13.6.2 FUNCTION F_GetVersionTcTwinSAFE

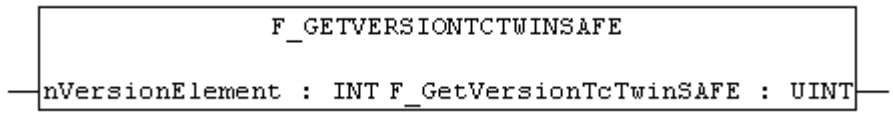


Abb. 119: Funktionsbaustein F_GETVERSIONTCTWINSAFE

Mit dieser Funktion können Versionsinformationen der SPS-Bibliothek ausgelesen werden.

FUNCTION F_GetVersionTcTwinSAFE : UINT

```
VAR_INPUT
    nVersionElement : INT;
END_VAR
```

nVersionElement: Versionselement, das gelesen werden soll. Mögliche Parameter:

- 1 : major number;
- 2 : minor number;
- 3 : revision number;

Entwicklungsumgebung	Zielplattform	IO-Hardware	Einzubindende SPS-Bibliotheken
TwinCAT v2.10.0 Build > 914	PC (i386)	-	TcTwinSAFE.Lib (Standard.Lib, TcBase.Lib und TcSystem.Lib werden automatisch eingebunden)
TwinCAT v2.10.0	BX-Serie	-	TcTwinSAFE.LBX (Standard.LBX; TcBaseBX.LBX; TcSystemBX.LBX werden automatisch eingebunden)

5.13.6.3 FUNCTION_BLOCK FB_TwinSAFE_KLx904_input

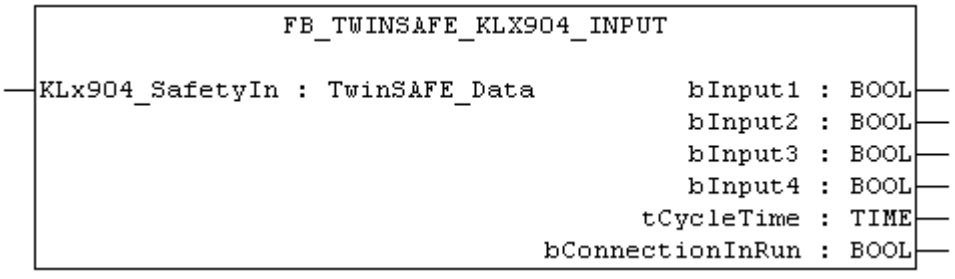


Abb. 120: Funktionsbaustein FB_TWINSAFE_KLX904_INPUT

Mit dem Funktionsbaustein *FB_TwinSAFE_KLx904_input* kann eine Auswertung der TwinSAFE Daten, welche von einer KL1904 oder KL2904 zu einer KL6904 gesendet werden durchgeführt werden. Der Eingangsparameter wird mit den SafetyIn Daten einer KL1904 oder KL2904 im System Manager doppelt verknüpft.

VAR_INPUT

```
VAR_INPUT
    KLx904_SafetyIn AT%I* : TwinSAFE_Data; (* Additional link to "SafetyIn" *)
END_VAR
```

KLx904_SafetyIn: TwinSAFE Telegramm, welches von einer KL1904 oder KL2904 zu einer KL6904 gesendet wird. Dieser Parameter wird im System Manager auf SafetyIn (Eingangsdaten der KLx904) doppelt verknüpft.

VAR_OUTPUT

```
VAR_OUTPUT
    bInput1             : BOOL;
    bInput2             : BOOL;
    bInput3             : BOOL;
    bInput4             : BOOL;
    tCycleTime          : TIME;
    bConnectionInRun    : BOOL;
END_VAR
```

bInput1: Liefert den Eingang 1 einer KL1904. Ist dieser Baustein für eine Verbindung zu einer KL2904 verwendet, ist der Wert immer 0.

bInput2: Liefert den Eingang 2 einer KL1904. Ist dieser Baustein für eine Verbindung zu einer KL2904 verwendet, ist der Wert immer 0.

bInput3: Liefert den Eingang 3 einer KL1904. Ist dieser Baustein für eine Verbindung zu einer KL2904 verwendet, ist der Wert immer 0.

bInput4: Liefert den Eingang 4 einer KL1904. Ist dieser Baustein für eine Verbindung zu einer KL2904 verwendet, ist der Wert immer 0.

tCycleTime: Liefert die Zykluszeit in ms, die benötigt wird um das TwinSAFE Telegramm zwischen den Teilnehmern auszutauschen.

bConnectionInRun: Liefert ein TRUE wenn kein Fehler in der Verbindung von der KLx904 zur KL6904 ansteht.

Beispiel für einen Aufruf im FUB:

```
PROGRAM MAIN
VAR
    fbTwinSAFE_KLx904_input         : FB_TwinSAFE_KLx904_input;
    bInput1_KL1904_S_Address_113    : BOOL;
    bInput2_KL1904_S_Address_113    : BOOL;
    bInput3_KL1904_S_Address_113    : BOOL;
    bInput4_KL1904_S_Address_113    : BOOL;
    tCycleTime_KL1904_KL6904        : TIME;
    bConnection3_In_Run             : BOOL;
END_VAR
```

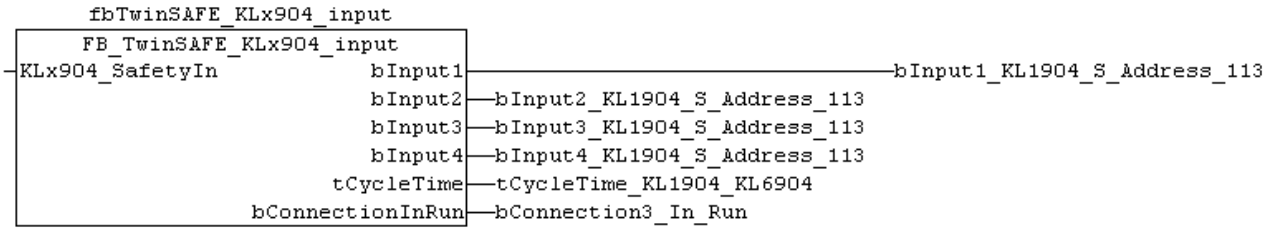


Abb. 121: Funktionsbaustein FB_TWINSAFE_KLX904_input

Im Beispiel werden die Werte der KL1904 Eingangsdaten auf die angeschlossenen Variablen geschrieben. Ist der Ausgang bConnectionInRun FALSE werden die Ausgänge generell auf FALSE gesetzt.

Zum Verknüpfen der Eingangsdaten markieren Sie den Parameter KLx904_SafetyIn und wählen aus dem Kontext Menü "Verknüpfung ändern..."

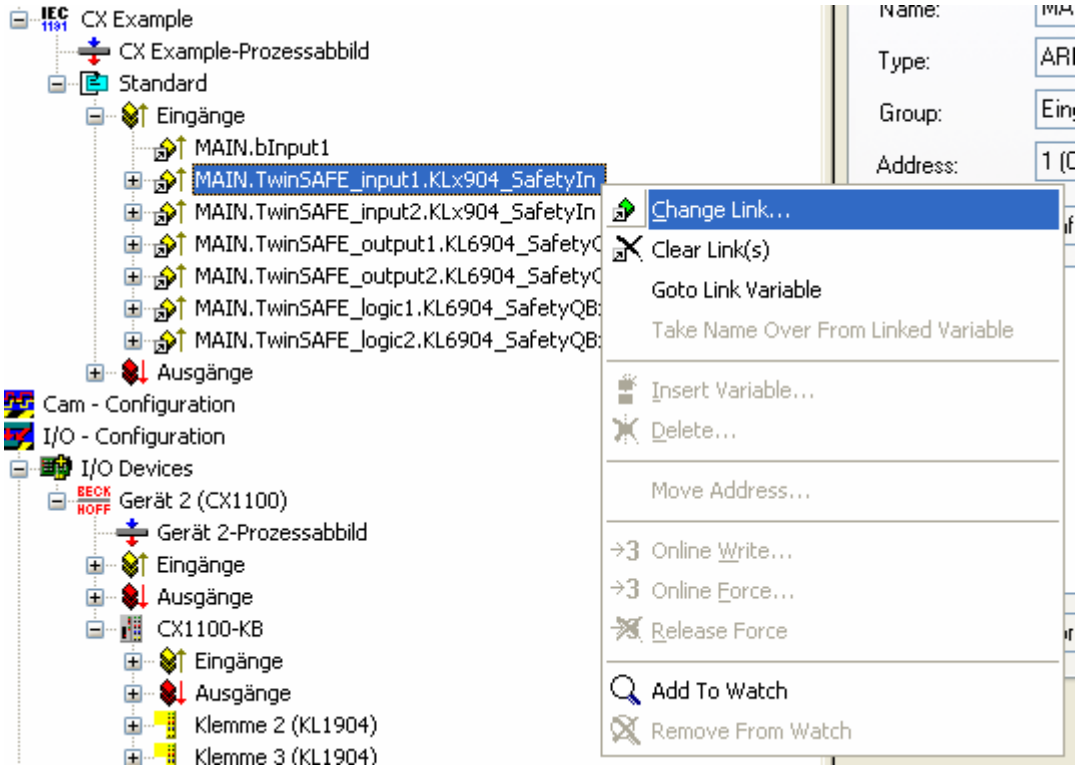


Abb. 122: Verknüpfen der Eingangsdaten

und wählen im folgenden Dialog die entsprechende SafetyIn-Variable

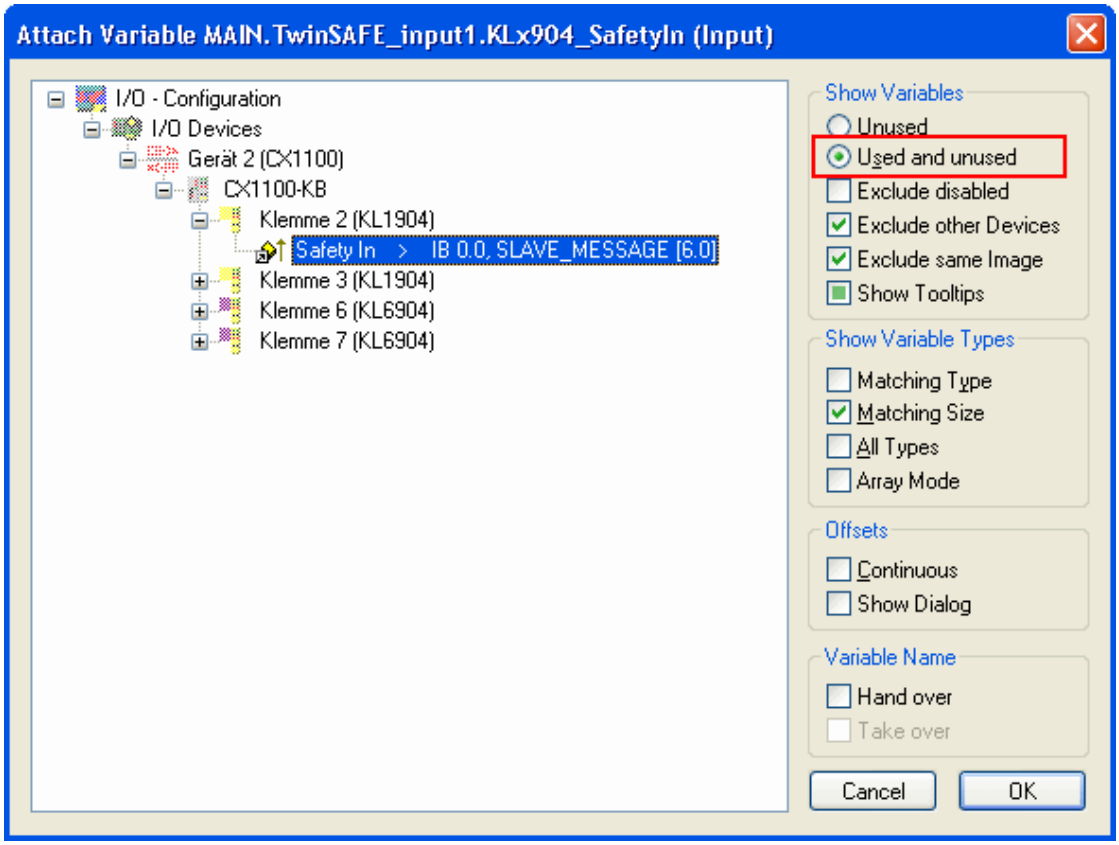


Abb. 123: Auswahl der SafetyIn-Variable

Entwicklungsumgebung	Zielplattform	IO-Hardware	Einzubindende SPS-Bibliotheken
TwinCAT v2.10.0 Build > 914	PC (i386)	KLx904	TcTwinSAFE.Lib (Standard.Lib, TcBase.Lib und TcSystem.Lib werden automatisch eingebunden)
TwinCAT v2.10.0 Build > 914	BX-Serie	KLx904	TcTwinSAFE.LBX (Standard.LBX, TcBaseBX.LBX und TcSystemBX.LBX werden automatisch eingebunden)

5.13.6.4 FUNCTION_BLOCK FB_TwinSAFE_KLx904_output

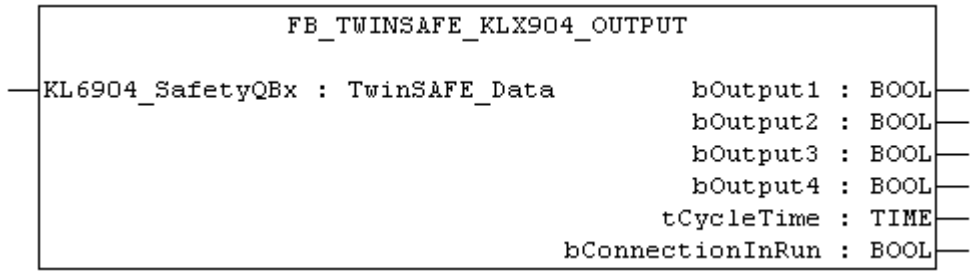


Abb. 124: Funktionsbaustein FB_TWINSAFE_KLX904_OUTPUT

Mit dem Funktionsbaustein *FB_TwinSAFE_KLx904_output* kann eine Auswertung der TwinSAFE Daten, welche von einer KL6904 zu einer KL1904 oder KL2904 gesendet werden durchgeführt werden. Der Eingangsparameter wird mit den SafetyQBx Daten einer KL6904 im System Manager doppelt verknüpft.

VAR_INPUT

```
VAR_INPUT
    KL6904_SafetyQBx AT%I* : TwinSAFE_Data; (* Additional link to "SafetyQBx" *)
END_VAR
```

KL6904_SafetyQBx: TwinSAFE Telegramm, welches von einer KL6904 zu einer KL1904 oder KL2904 gesendet wird. Dieser Parameter wird im System Manager auf SafetyQBx (Eingangsdaten der KL6904) doppelt verknüpft, wobei das x für Ziffern zwischen 1 und 15 steht entsprechend der verwendeten TwinSAFE Connection.

VAR_OUTPUT

```
VAR_OUTPUT
    bOutput1            : BOOL;
    bOutput2            : BOOL;
    bOutput3            : BOOL;
    bOutput4            : BOOL;
    tCycleTime          : TIME;
    bConnectionInRun    : BOOL;
END_VAR
```

bOutput1: Liefert den Ausgang 1 einer KL2904. Wird der Baustein für eine Verbindung zur KL1904 verwendet, ist dieser Wert grundsätzlich 0.

bOutput2: Liefert den Ausgang 2 einer KL2904. Wird der Baustein für eine Verbindung zur KL1904 verwendet, ist dieser Wert grundsätzlich 0.

bOutput3: Liefert den Ausgang 3 einer KL2904. Wird der Baustein für eine Verbindung zur KL1904 verwendet, ist dieser Wert grundsätzlich 0.

bOutput4: Liefert den Ausgang 4 einer KL2904. Wird der Baustein für eine Verbindung zur KL1904 verwendet, ist dieser Wert grundsätzlich 0.

tCycleTime: Liefert die Zykluszeit in ms, die benötigt wird um das TwinSAFE Telegramm zwischen den Teilnehmern auszutauschen.

bConnectionInRun: Liefert ein TRUE wenn kein Fehler in der Verbindung von der KL6904 zur KLx904 ansteht.

Beispiel für einen Aufruf im FUB

```
PROGRAM MAIN
VAR
    fbTwinSAFE_KLx904_output                : FB_TwinSAFE_KLx904_output;
    bOutput1_KL6904_Connection_to_113       : BOOL;
    bOutput2_KL6904_Connection_to_113       : BOOL;
    bOutput3_KL6904_Connection_to_113       : BOOL;
    bOutput4_KL6904_Connection_to_113       : BOOL;
    tCycleTime_KL6904_KL1904                : TIME;
    bConnection3_In_Run_2                   : BOOL;
END_VAR
```

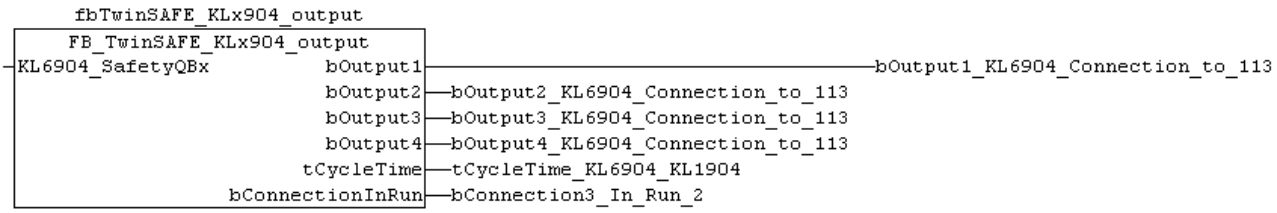


Abb. 125: Aufruf des Funktionsbausteins FB_TWINSAFE_KLX904_OUTPUT

Im Beispiel werden die Werte der TwinSAFE Klemme KL6904 zur KL1904 ausgewertet. Da in dieser Verbindung keine Ausgangssignale verwendet werden, sind die Ausgänge grundsätzlich FALSE. Es kann nur tCycleTime und bConnectionInRun ausgewertet werden.

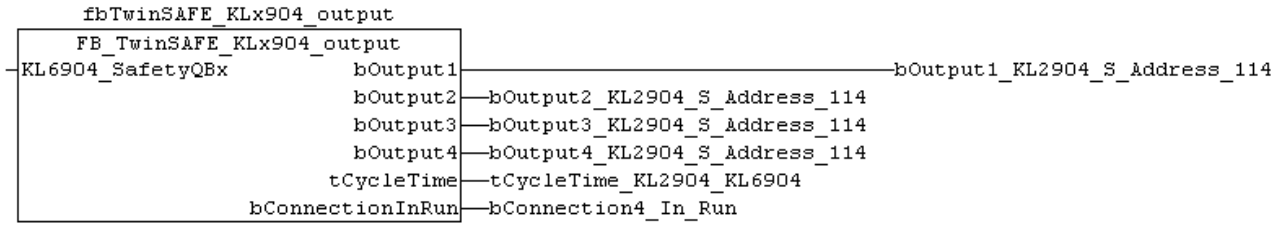


Abb. 126: Aufruf des Funktionsbausteins FB_TWINSAFE_KLX904_OUTPUT

Im Beispiel werden die Werte der TwinSAFE Klemme KL6904 zur KL2904 ausgewertet. In dieser Verbindung werden die Ausgangssignale zur KL2904 geschrieben und vom Baustein auf die angeschlossenen Variablen kopiert. Ist der Ausgang bConnectionInRun auf FALSE werden die Ausgänge auf FALSE gesetzt.

Zum Verknüpfen der Eingangsdaten markieren Sie den Parameter KL6904_SafetyQBx und wählen aus dem Kontext Menü "Verknüpfung ändern..."

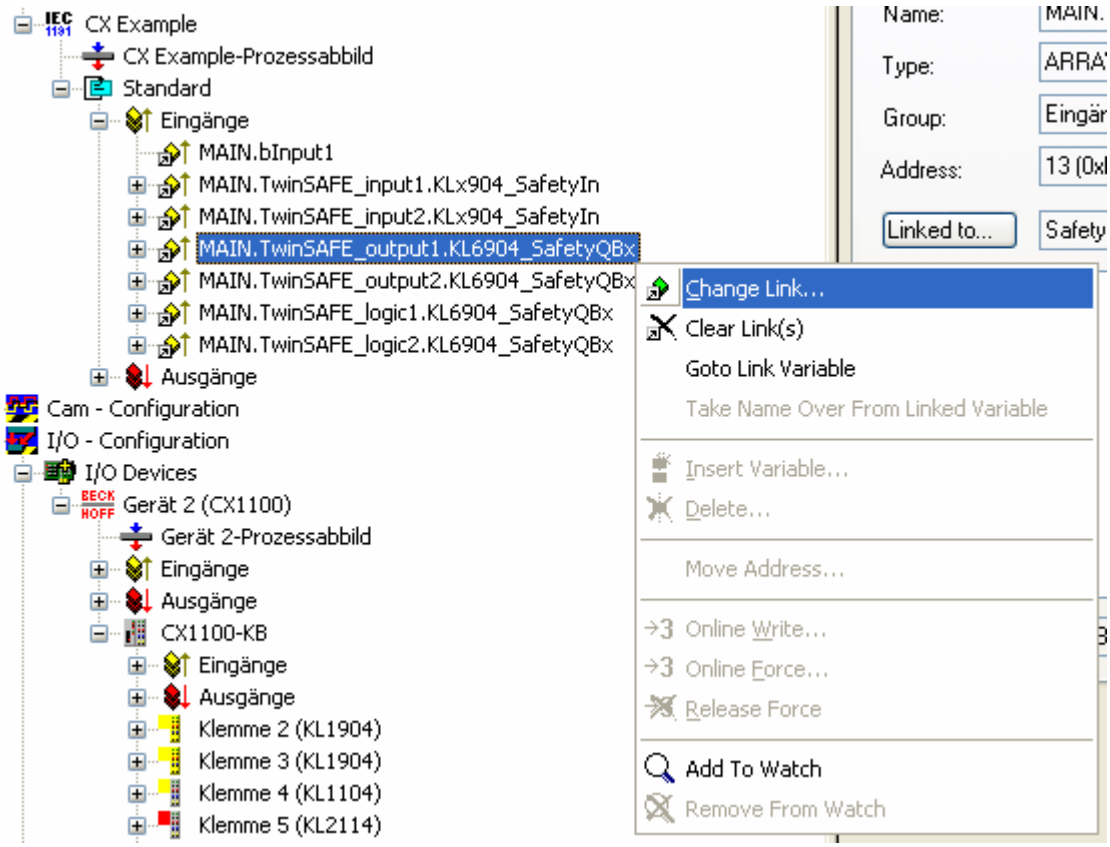


Abb. 127: Verknüpfen der Eingangsdaten

und wählen im folgenden Dialog die entsprechende SafetyQBx-Variable

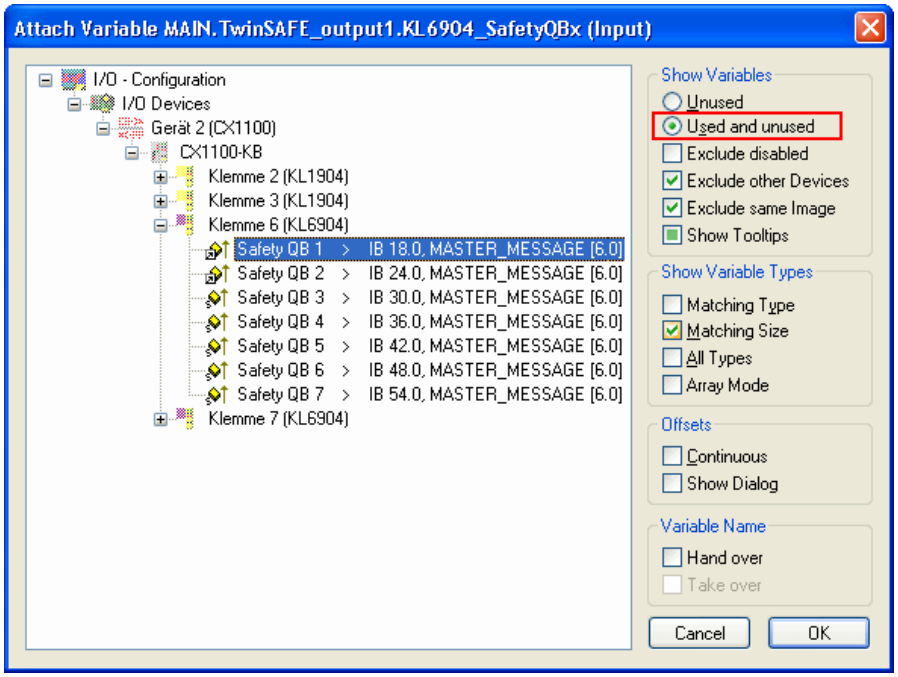


Abb. 128: Auswahl der entsprechenden SafetyQBx-Variable

Entwicklungsumgebung	Zielplattform	IO-Hardware	Einzubindende SPS-Bibliotheken
TwinCAT v2.10.0 Build > 914	PC (i386)	KLx904	TcTwinSAFE.Lib (Standard.Lib, TcBase.Lib und TcSystem.Lib werden automatisch eingebunden)
TwinCAT v2.10.0 Build > 914	BX-Serie	KLx904	TcTwinSAFE.LBX (Standard.LBX, TcBaseBX.LBX und TcSystemBX.LBX werden automatisch eingebunden)

5.13.7 TcBaseBX9000

5.13.7.1 Übersicht: TcBaseBX9000

Download

Zum Download der Bibliotheken bitte auf den Link mit der linken Maustasten klicken. Bitte die Bibliotheken in das Verzeichnis TwinCAT\PLC\LIB kopieren.

- Download TcBaseBX9000 (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207318539.zip)

TcBaseBX9000

IP-TCP/IP-UDP	Version	Firmware
		BX9000
FB_IpClose [▶ 149]	12.10.06	1.15
FB_IpEndSession [▶ 148]	12.10.06	1.15
FB_IpOpen [▶ 149]	12.10.06	1.15
FB_IpReceive [▶ 151]	12.10.06	1.15
FB_IpSend [▶ 152]	12.10.06	1.15
FB_IpStartSession [▶ 147]	12.10.06	1.15

ModbusTCP	Version	Firmware
		BX9000
FB_MBClose [▶ 158]	12.10.06	1.15
FB_MBConnect [▶ 158]	12.10.06	1.15
FB_MBGenericReq [▶ 158]	12.10.06	1.15

Services	Version	Firmware
		BX9000
FB_AddDnsServer [▶ 165]	12.10.06	1.15
FB_GetHostByAddr [▶ 166]	12.10.06	1.15
FB_GetHostByName [▶ 167]	12.10.06	1.15
FB_GetNetworkConfig [▶ 168]	12.10.06	1.15
FB_SetTargetName [▶ 169]	12.10.06	1.15

SMTP	Version	Firmware
		BX9000
FB_Smtp [▶ 162]	12.10.06	1.15

SNTP	Version	Firmware
		BX9000
FB_Sntp [▶ 164]	12.10.06	1.15

5.13.7.2 Socket Interface

5.13.7.2.1 Übersicht: Socket-Interface

Das Socket-Interface dient dazu, beliebige Ethernet Telegramme zu empfangen oder auch von der Steuerung zu senden. So können beliebige Protokolle auf SPS Ebene in der IEC 61131-3 programmiert werden.

Unterstützte Ethernet Protokolle

Type	STREAM	DGRAM	RAW
IP	n.i.	n.i.	n.i.
ICMP	n.i.	n.i.	n.i.
IGMP	n.i.	n.i.	n.i.
TCP	Implementiert*	n.i.	n.i.
UDP	n.i.	Implementiert*	n.i.
RAW	n.i.	n.i.	n.i.

Tabelle 1
n.i. nicht implementiert

Arbeitsweise einer Socket-Verbindung

Bevor man ein Socket öffnen kann, muss der Steuerung für diese Art von Verbindung Ressourcen zur Verfügung gestellt werden. Dies geschieht durch das Starten einer Session. Hier gibt man an wie und was man tun möchte. Danach kann die Session genutzt werden, um Ethernet Telegramme zu verschicken oder zu empfangen. Die implementierten Protokolle sind aus Tabelle 1 zu entnehmen.

Client/Server-Beziehung

Unter Client versteht man den Teilnehmer, der eine Verbindung aufbauen möchte und den aktiven Teil eines Verbindungsaufbaus initialisiert. Der Server ist erst einmal passiv und wartet auf eine Client Anfrage. Erst wenn ein Client eine Verbindung aufbaut, wird auch der Server aktiv. Ist eine Verbindung zwischen Client und Server aufgebaut, können beide Teilnehmer beliebig Daten senden und auch empfangen.

* zu Tab1

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.10.0 und höher	BC9050 (165) Firmware-Version >=B0	TcBaseBX9000.lbx
TwinCAT v2.10.0 und höher	BC9020 (165) Firmware-Version >=B0	TcBaseBX9000.lbx
TwinCAT v2.10.0 und höher	BC9120 (165) Firmware-Version >=B1	TcBaseBX9000.lbx
TwinCAT v2.10.0 und höher	BC9191 Firmware-Version >= 3.1	TcBaseBX9000.lbx
TwinCAT v2.10.0 und höher	BX9000 (165) Firmware-Version >=1.14	TcBaseBX9000.lbx

5.13.7.2.2 IP-Bausteinübersicht

IP-TCP/IP-UDP	Firmware	Beschreibung
FB_IpStartSession [▶ 147]	1.14	Öffnen einer Session
FB_IpEndSession [▶ 148]	1.14	Schließen einer Session
FB_IpOpen [▶ 149]	1.14	Öffnen eine TCP/IP-Verbindung (wird für eine UDP-Kommunikation nicht gebraucht)
FB_IpClose [▶ 149]	1.14	Schließen einer TCP/IP-Verbindung (wird für eine UDP-Kommunikation nicht gebraucht)
FB_IpReceive [▶ 151]	1.14	Empfang von TCP- oder UDP-Telegrammen
FB_IpSend [▶ 152]	1.14	Senden von TCP- oder UDP-Telegrammen

Multicast-Bausteine	Firmware	Beschreibung
FB_AddMultiRoute [▶ 164]	1.14	Eine Muticast-Adresse anlegen
FB_DelMultiRoute [▶ 164]	1.14	Eine Muticast-Adresse löschen

FB_IpStartSession

Der Baustein reserviert Ressourcen auf der Steuerung für die Ethernet Kommunikation. Mit *bStart* wird der Baustein aktiv. Solange der Baustein arbeitet, ist das **bBusy** gesetzt. **iDevice** ist immer mit Null zu belegen. **iPort** ist für die lokale TCP oder UDP Port Nummer. **iMaxConnection** gibt die Anzahl der maximal möglichen Verbindungen an (maximal 3).

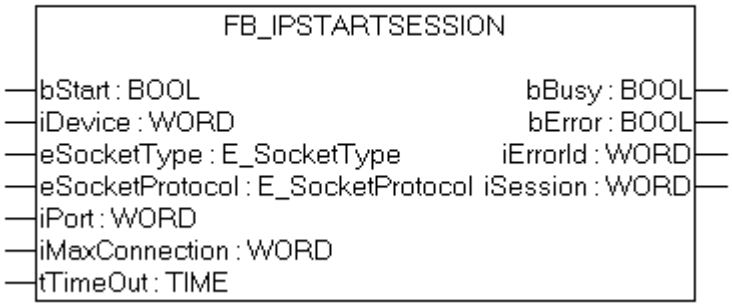


Abb. 129: Funktionsbaustein FB_IPSTARTSESSION

INPUT

```
VAR_INPUT
   bStart           : BOOL;
   iDevice      : WORD;
   eSocketType  : E_SocketType;
   eSocketProtocol      : E_SocketProtocol;
   iPort        : WORD;
   iMayConnection       : WORD;
   tTimeout     : TIME;
END_VAR
```

bStart: Eine steigende Flanke aktiviert den Baustein.

iDevice: immer "0"

eSocketType: bei TCP/IP ist "SOCK_STREAM" zu benutzen. Beachten Sie, dass die Daten als Byte Stream abgelegt werden. Sie sollten möglichst die Länge der empfangenden Daten kennen oder ein Protokoll mit einer Start- und Endkennung verwenden, damit Sie im Daten-Stream eindeutig Anfang und Ende erkennen können. Bei UDP/IP ist "SOCK_DGRAM" einzustellen. Dabei wird ein UDP-Frame immer komplett in einen Speicher eingetragen. Es stehen 4 Speicher zur Verfügung. Liest das Anwenderprogramm die Daten nicht schnell genug aus dem Speicher der SPS aus, gehen weitere UDP-Frames verloren.

eSocketProtocol: für TCP/IP ist "IPPROTO_TCP" zu benutzen und für UDP/IP "IPPROTO_UDP".

iPort: Absender Port Nummer

iMaxConnection: Anzahl der maximal möglichen Verbindungen (max. ist 3)

tTimeout: Zeit nach der abgebrochen werden soll.

OUTPUT

```
VAR_OUTPUT
   bBusy        : BOOL;
   bError          : BOOL;
   iErrorId     : WORD;
   iSession     : WORD;
END_VAR
```

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis die Befehlsausführung abgeschlossen wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehler-Code des zuletzt ausgeführten Befehls (siehe Tabelle).

iSession: Gibt die Session-Nummer an alle IP-Bausteine weiter, für die diese Verbindung angelegt wurde.

FB_IpEndSession

Der Baustein schließt eine geöffnete Session. Positive Flanke von *bStart* schließt die Session und gibt Resourcen frei.

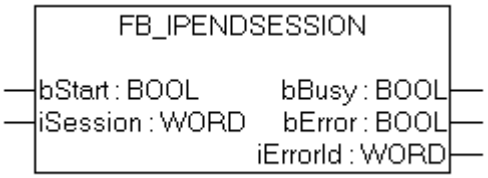


Abb. 130: Funktionsbaustein FB_IPENDSESSION

INPUT

```
VAR_INPUT
   bStart          : BOOL;
   iSession     : WORD;
END_VAR
```

bStart: Eine steigende Flanke aktiviert den Baustein.

iSession: wird mit der Session Nummer aus dem Baustein FB_IpStartSession.

OUTPUT

```
VAR_OUTPUT
   bBusy        : BOOL;
   bError          : BOOL;
   iErrorId     : WORD;
END_VAR
```

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis die Befehlsausführung abgeschlossen wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle).

FB_IpOpen

Der Baustein ist notwendig für das Öffnen einer TCP/IP Verbindung aus der SPS Steuerung heraus. In diesem Fall ist die Steuerung der Client, der aktive eine Verbindung zu einem TCP/IP Server aufbaut. Mit einer positiven Flanke von **bStart** wird eine Verbindung zu einem Server mit der IP-Adresse aus **sRemoteIpAddr** aufgebaut. Die Empfänger Port Nummer wurde durch das Starten der Session vorgegeben (siehe FB_IpStartSession [▶ 147]). Die Absender Port Nummer wird von der Steuerung vergeben und kann aus **iPortNo** ausgelesen werden. Die Absender Port Nummer wird für das Senden (siehe FB_IpSend [▶ 152]) gebraucht. Das **bBusy** wird solange gesetzt wie der Baustein aktiv ist. Wird das **bBusy** zurückgesetzt und ist das **bError** FALSE ist der Aufbau der TCP/IP Verbindung erfolgreich abgeschlossen und es können Daten gesendet oder empfangen werden.

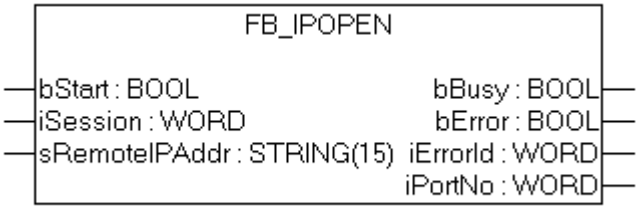


Abb. 131: Funktionsbaustein FB_IPOPEN

INPUT

```
VAR_INPUT
   bStart           : BOOL;
   iSession     : WORD;
   sRemoteIPAddr    : STRING(15);
END_VAR
```

bStart: Eine steigende Flanke aktiviert den Baustein.

iSession: wird mit der Session Nummer aus dem Baustein FB_IpStartSession verbunden.

OUTPUT

```
VAR_OUTPUT
   bBusy        : BOOL;
   bError           : BOOL;
   iErrorId     : WORD;
   iPortNo      : WORD;
END_VAR
```

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis die Befehlsausführung abgeschlossen wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle).

iPortNo: TCP Port Nummer, die beim Öffnen der TCP/IP Verbindung vergeben wurde (Lokale Port Nummer).

FB_IpClose

Der Baustein ist notwendig für das Schließen einer TCP/IP Verbindung aus der SPS Steuerung heraus. Mit einer positiven Flanke von **bStart** wird eine Verbindung mit der IP-Adresse aus **sRemoteIpAddr** abgebaut. Der Baustein sendet ein FIN und wartet auf die Bestätigung des Verbindungsabbaus. Hierfür müssen die Gegenstelle und die Verbindung existieren und funktionieren. Ist das **bResetConnection** gesetzt sendet der Baustein, dass die Verbindung beendet wurde, wartet aber nicht auf eine Bestätigung des anderen Teilnehmers. Das **bBusy** wird solange gesetzt wie der Baustein aktiv ist. Wird das **bBusy** zurückgesetzt und ist das **bError** FALSE ist der Abbau der TCP/IP Verbindung erfolgreich abgeschlossen.

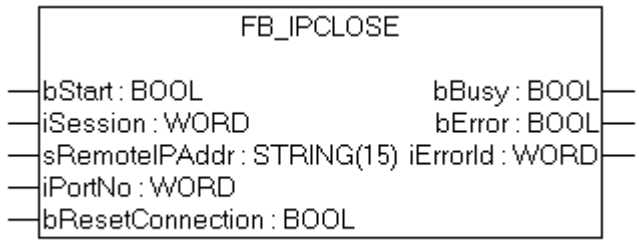


Abb. 132: Funktionsbaustein FB_IPCLOSE

INPUT

```
VAR_INPUT
   bStart              : BOOL;
   iSession         : WORD;
   sRemoteIPAddr       : STRING(15);
   iPortNo          : WORD;
   bResetConnection    : BOOL;
END_VAR
```

bStart: Eine steigende Flanke aktiviert den Baustein.

iSession: wird mit der Session Nummer aus dem Baustein FB_IpStartSession verbunden.

sRemoteIPAddr: IP-Adresse des Teilnehmers, mit dem die Verbindung abgebaut werden soll.

iPortNo: Port Nummer des Teilnehmers, mit dem die Verbindung abgebaut werden soll.

bResetConnection: FALSE es wird ein FIN gesendet - TRUE die TCP/IP Verbindung wird geschlossen ohne das auf ein Quittung der Gegenstelle gewartet wird, es ist in beiden Fällen ein erneutes Open notwendig um die Verbindung wieder aufzubauen.

OUTPUT

```
VAR_OUTPUT
   bBusy        : BOOL;
   bError           : BOOL;
   iErrorId     : WORD;
END_VAR
```

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis die Befehlsausführung abgeschlossen wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle).

FB_IpReceive

Der Baustein *FB_IpReceice* ermöglicht das Empfangen von UDP- oder TCP-Telegrammen. Welche der beiden Verbindungen verwendet wird, wurde im FB_IpStartSession [▶ 147] festgelegt.

Daten die Empfangen werden, müssen in einer SPS-Variabel abgelegt werden. Dafür müssen Sie einen Pointer auf **pBuffAddr** legen und die Größe der Variable in *cbBuffLen* eintragen. Mit einer positiven Flanke von **bValid** wird angezeigt, dass Daten im Speicher liegen bzw. die Daten ihrer Variabel jetzt gültig sind. Mit einer positiven Flanke von **bClear** ist der Baustein wieder bereit Daten zu empfangen, bzw. liegen noch Daten im Puffer werden diese als nächstes in die Variabel kopiert.

Achten Sie besonders bei TCP darauf, wie viele Daten empfangen wurden bzw. wie viele Daten noch im Puffer liegen. Bei TCP/IP werden die Daten als "Stream" abgelegt, d.h. es gibt hier kein Anfang und kein Ende. Im Gegensatz dazu wird bei UDP immer der Inhalt eines UDP Frames in einen eigenen Puffer abgelegt.

Es können 4 UDP-Telegramme gepuffert werden, weitere UDP Telegramme gehen verloren. **cbReceive** zeigt Ihnen die Anzahl der Daten in Bytes an, die in Ihre Variabel kopiert wurden. Sind mehr Daten in Puffer als ausgelesen wurden, steht die Anzahl, der noch verbleibenden Daten in **cbBytesInStream**.

sReceiveIPAddr gibt Ihnen die IP-Adresse des Teilnehmers an, der Daten zur Beckhoff-Steuerung gesendet hat und die entsprechende Port Nummer **iReceivePortNo**. Beide Variablen werden mit **bClear** wieder gelöscht. Sie können **sReceiveIPAddr** und *iReceivePortNo* dafür nutzen um Daten an den Teilnehmer wieder zurück zu senden mit dem Baustein FB_IpSend [▶ 152].

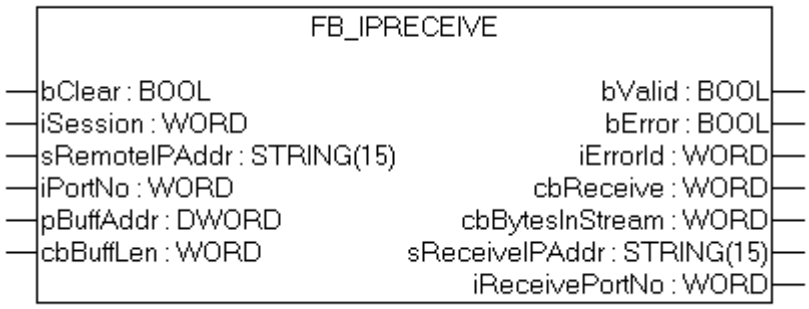


Abb. 133: Funktionsbaustein FB_IPRECEIVE

INPUT

```
VAR_INPUT
   bClear           : BOOL;
   iSession     : WORD;
   sRemoteIPAddr    : STRING(15);
   iPortNo      : WORD;
   pBuffAddr    : DWORD;
   cbBuffLen    : WORD;
END_VAR
```

bClear: Eine steigende Flanke löscht den Speicher und der Baustein ist wieder bereit Daten zu empfangen.

iSession: wird mit *iSession* aus dem Baustein FB_StartSession [▶ 147] verbunden.

sRemoteIPAddr: kann als Filter verwendet werden um nur eine spezielle IP Adresse zuzulassen.

iPortNo: kann als Filter verwendet werden um nur einen speziellen Port zuzulassen.

pBuffAddr: Mit **ADR** wird hier der Pointer auf die Variable übergeben, wohin die Daten kopiert werden sollen, die Empfangen wurden.

cbBuffLen: Größe der Variabel, kann mit **SIZEOF** ermittelt werden.

OUTPUT

```
VAR_OUTPUT
   bValid           : BOOL;
   bError           : BOOL;
   iErrorId     : WORD;
   cbReceive    : WORD;
   cbBytesInStream      : WORD;
   sReceiveIPAddr       : STRING(15)
   iReceivePortNo       : WORD;
END_VAR
```

bValid: Ein positiver Flankenwechsel zeigt an das neue Daten Empfangen wurden. Diese Daten stehen nun in der Variable zur Verfügung, die über den Pointer **pBuffAddr** verbunden wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle).

cbReceive: Gibt die Anzahl an Bytes an die kopiert worden sind.

cbBytesInStream: Gibt die noch im Speicher befindlichen Daten an. Dies sollte immer Null sein. Ist der Wert >0, haben Sie die Variable die mit **pBuffAddr** verbunden wurde zu klein gewählt. Es muss dann noch einmal ausgelesen werden und den Rest der Daten zu erhalten.

sReceiveIPAddr: Zeigt Ihnen die IP-Adresse an von dem Teilnehmer, der die Daten geschickt hat. Wird mit positiver Flanke von **bClear** gelöscht.

iReceivePortNo: Zeigt Ihnen die Port-Nummer an von dem Teilnehmer, der die Daten geschickt hat. Wird mit positiver Flanke von **bClear** gelöscht.

FB_IpSend

Der Baustein sendet Daten über TCP oder UDP. Welche der Verbindung verwendet wird, ist in der FB_IpStartSession [▶ 147] festgelegt worden. An den **pBuffAddr** wird per "ADR" der Pointer auf die Variabel festgelegt, die Daten für das Senden beinhaltet. **cbBuffLen** gibt die Länge der Daten an. Mit **sRemoteIPAddr** wird die IP Adresse angegeben, an die die Daten gesendet werden. Mit **iPortNo** wird bei TCP dies mit der Port Nummer des FB_IpOpen [▶ 149] verknüpft. Bei UDP kann eine beliebige Portnummer verwendet werden. **iPortNo** ist die Absender Port Nummer. Mit einer positiven Flanke von **bStart** wird der Baustein aktiviert und die Daten werden versendet. Solange der Baustein aktiv ist, ist das **bBusy** auf TRUE. Sind die Daten versendet worden geht das **bBusy** auf FALSE und das **bError** bleibt ebenfalls auf FALSE. Bei einem Fehler wird das **bError** gesetzt.

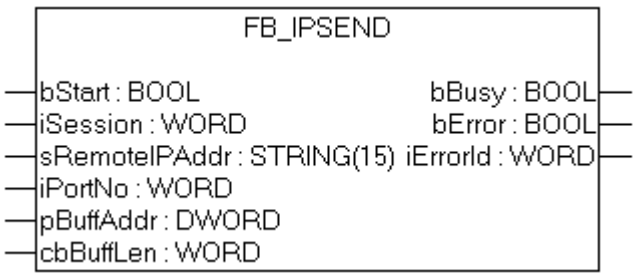


Abb. 134: Funktionsbaustein FB_IPSEND

INPUT

```
VAR_INPUT
   bStart               : BOOL;
   iSession         : WORD;
   sRemoteIPAddr        : STRING(15);
   pBuffAddr        : DWORD;
   cbBuffLen        : WORD;
END_VAR
```

bStart: Eine steigende Flanke aktiviert den Baustein.

iSession: wird mit der Session Nummer aus dem Baustein FB_IpStartSession [▶ 147] verbunden.

sRemoteIPAddr: IP-Adresse des Teilnehmers, an den die Daten gesendet werden sollen.

iPortNo: Absender Port Nummer. Bei TCP ist die Port Nummer aus den FB_IpOpen [▶ 149] Baustein zu verwenden, bei UDP kann eine beliebige Port Nummer verwendet werden.

pBuffAddr: Pointer auf die Daten die gesendet werden sollen (Befehl: **ADR**).

cbBuffLen: Länge der Daten, die gesendet werden sollen. Die Länge sollte immer kleiner/gleich der Variabel sein auf die der Pointer von *pBuffAddr* zeigt (Befehl: **SIZEOF**)

OUTPUT

```
VAR_OUTPUT
   bBusy        : BOOL;
   bError           : BOOL;
   iErrorId     : WORD;
END_VAR
```

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis die Befehlsausführung abgeschlossen wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle).

5.13.7.2.3 TCP/IP

TCP/IP Client

TCP/IP ist eine verbindungsorientierte Kommunikationsart, eigentlich eine peer-to-peer Verbindung. In diesem Beispiel wird ihnen gezeigt wie eine Verbindung von der Beckhoff Steuerung zu einem Server aufgebaut wird. Die benötigten Bausteine sind FB_IpStartSession, FB_IpOpen, FB_IpSend und optional FB_IpReceive, FB_IpClose und FB_IpEndSession.

Es empfiehlt sich eine Schrittkette zu programmieren, wie in dem Beispiel (siehe unten) angegeben ist.

Die Beckhoff Steuerung ist hier der Client und die Gegenstelle ein TCP Server. Der Server ist ein VB6 Programm.

Schritt 1

FB_IpStartSession

Der Baustein reserviert Resourcen auf der Steuerung für die TCP/IP Kommunikation. Mit *bStart* wird der Baustein aktiv. Solange der Baustein arbeitet ist das **bBusy** gesetzt. **iDevice** ist immer mit null zu belegen. **iPort** ist für die lokale TCP/IP Port Nummer. die **iMaxConnection** gibt die Anzahl der maximal möglichen Verbindungen an, maximal ist dieser Wert 3.

INPUT

bStart: Eine steigende Flanke aktiviert den Baustein.

iDevice: immer "0"

eSocketType: bei TCP/IP ist "SOCK_STREAM" zu benutzen. Beachten Sie das die Daten als Byte-Stream abgelegt werden.
Sie sollten die Länge der zu empfangenden Daten wissen oder ein Protokoll mit Start- und Endkennung verwenden, damit Sie im Daten-Stream eindeutig Anfang und Ende erkennen können.

eSocketProtocol: für TCP/IP ist "IPPROTO_TCP" zu benutzen

iPort: Absender Port Nummer

iMaxConnection: Anzahl der maximal möglichen Verbindungen (max. ist 3)

tTimeout: Zeit nach der abgebrochen werden soll.

OUTPUT

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis die Befehlsausführung abgeschlossen wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle).

iSession: Gibt die Session Nummer an alle IP Bausteine weiter, für die diese Verbindung angelegt wurde.

Schritt 2

FB_IpOpen

Daten die Empfangen werden, müssen in eine PLC Variable abgelegt werden. Dafür müssen Sie einen Pointer auf **pBuffAddr** legen und die Größe der Variable in **cbBuffLen** eintragen. Mit einer positiven Flanke von bValid wird angezeigt das Daten im Speicher liegen bzw. die Daten ihrer Variable jetzt gültig sind.

INPUT

bClear: Eine steigende Flanke löscht den Speicher und der Baustein ist wieder bereit Daten zu empfangen.

iSession: wird mit dem iSession aus dem Baustein FB_StartSession verbunden

sRemoteIPAddr: kann als Filter verwendet werden um nur eine spezielle IP-Adresse zuzulassen

eSocketProtocol: für TCP/IP ist "IPPROTO_TCP" zu benutzen

iPort: Locale Port Nummer

iMaxConnection: Anzahl der maximal möglichen Verbindungen (max. ist 3)

tTimeout: Zeit nach der abgebrochen werden soll.

OUTPUT

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis die Befehlsausführung abgeschlossen wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle).

iSession: Gibt die Session Nummer an alle IP Bausteine weiter, für die diese Verbindung angelegt wurde.

Sobald die ersten Daten empfangen wurden können auch Daten wieder zurückgeschickt werden. Dies ist optional und wird in dem Beispiel verwendet.

Beispiel

 Download VB6 Programm als TCP/IP Server Zip File (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207366411.zip)

 Download TwinCAT Projekt als TCP/IP Client prx File (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207368587.zip)

TCP/IP Server

TCP/IP ist eine verbindungsorientierte Kommunikationsart, eigentlich eine peer-to-peer Verbindung. In diesem Beispiel wird ihnen gezeigt wie eine Verbindung von außen auf die Beckhoff Steuerung aufgebaut wird. Die benötigten Bausteine sind FB_IpStartSession, FB_IpReceive und optional FB_IpSend, FB_IpClose und FB_IpEndSession.

Es empfiehlt sich eine Schrittkette zu programmieren, wie in dem Beispiel (siehe unten) angegeben ist.

Die Beckhoff Steuerung ist hier Server und die Gegenstelle ein TCP Client. Der Client ist ein VB6 Programm.

Schritt 1

FB_IpStartSession

Der Baustein reserviert Resourcen auf der Steuerung für die TCP/IP Kommunikation. Mit *bStart* wird der Baustein aktive. Solange der Baustein arbeitet ist das **bBusy** gesetzt. **iDevice** ist immer mit null zu belegen. **iPort** ist für die lokale TCP/IP Port Nummer. die **iMaxConnection** gibt die Anzahl der maximal möglichen Verbindungen an, maximal ist dieser Wert 3.

INPUT

bStart: Eine steigende Flanke aktiviert den Baustein.

iDevice: immer "0"

eSocketType: bei TCP/IP ist "SOCK_STREAM" zu benutzen. Beachten Sie das die Daten als Byte Stream abgelegt werden. Sie sollten möglichst die Länge der empfangenden Daten wissen oder ein Protokoll mit einer Start und Endkennung damit Sie aus den Stream von Daten eindeutig den Anfang und das Ende erkennen können.

eSocketProtocol: für TCP/IP ist "IPPROTO_TCP" zu benutzen

iPort: Empfangsport Nummer

iMaxConnection: Anzahl der maximal möglichen Verbindungen (max. ist 3)

tTimeout: Zeit nach der abgebrochen werden soll.

OUTPUT

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis die Befehlsausführung abgeschlossen wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle).

iSession: Gibt die Session Nummer an alle IP Bausteine weiter, für die diese Verbindung angelegt wurde.

Schritt 2

FB_IpReceive

Daten die Empfangen werden, müssen in eine PLC Variable abgelegt werden. Dafür müssen Sie einen Pointer auf pBuffAddr legen und die Größe der Variable in cbBuffLen. Mit einer positiven Flanke von bValid wird angezeigt das Daten im Speicher liegen bzw. die Daten ihrer Variable jetzt gültig sind.

INPUT

bClear: Eine steigende Flanke löscht den Speicher und der Baustein ist wieder bereit Daten zu empfangen.

iSession: wird mit dem iSession aus dem Baustein FB_StartSession verbunden

sRemoteIPAddr: kann als Filter verwendet werden um nur eine spezielle IP Adresse

eSocketProtocol: für TCP/IP ist "IPPROTO_TCP" zu benutzen

iPort: Locale Port Nummer

iMaxConnection: Anzahl der maximal möglichen Verbindungen (max. ist 3)

tTimeout: Zeit nach der abgebrochen werden soll.

OUTPUT

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis die Befehlsausführung abgeschlossen wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle).

iSession: Gibt die Session Nummer an alle IP Bausteine weiter, für die diese Verbindung angelegt wurde.

Beispiel

Download VB6 Programm als TCP/IP Client Zip File (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207370763.zip)

Download TwinCAT Projekt als TCP/IP Server prx File (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207372939.zip)

5.13.7.2.4 UDP/IP

UDP/IP-Verbindung

UDP ist eine sehr einfache Ethernet-Verbindung. UDP-Daten werden versendet, ohne dass es einen Mechanismus gibt ob das Telegramm angekommen ist oder nicht. Für eine funktionierende UDP-Kommunikation muss die Port-Nummer auf beiden Seiten bekannt sein.

Der BX9000 sendet Daten an ein VB6-Programm, das diese Daten wieder zurück schickt.

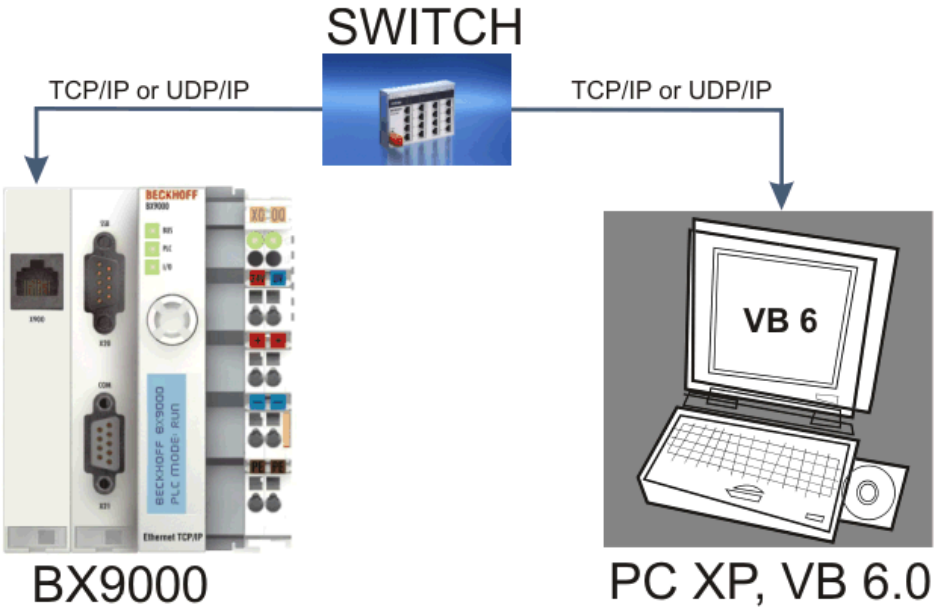


Abb. 135: UDP/IP-Verbindung

Schritt 1: Vorbereitung der UDP Kommunikation

FB_IpStartSession

Der Baustein reserviert Resourcen auf der Steuerung für die UDP/IP Kommunikation. Mit *bStart* wird der Baustein aktiv. Solange der Baustein arbeitet ist das **bBusy** gesetzt. **iDevice** ist immer mit null zu belegen. **iPort** ist für die lokale UDP Port Nummer des BX9000. Die **iMaxConnection** gibt die Anzahl der maximal möglichen Verbindungen an, maximal ist dieser Wert 3. Der *eSocketTyte* ist auf "SOCK_DGRAM" zu stellen. **eSocketProtocol** muss auf "IPPROTO_UDP" stehen für die UDP Kommunikation. Der *tTimeOut* wird für die UDP Kommunikation nicht verwendet. Die **iSession** muss mit den folgenden Bausteinen FB_IpSend und FB_ipReceive verknüpft sein.

Schritt 2: Senden von UDP-Frames

FB_IpSend

Mit einer positiven Flanke von **bStart** wird ein UDP Frame abgeschickt. Die IP-Adresse wird mit **sRemoteIPAddr** beschrieben und die Ziel UDP Port Nummer mit **iPortNo**. Die Absender UDP Port Nummer wurde ja schon in den **FB_IpStartSession** konfiguriert. Wird das **bBusy** vom Baustein zurückgesetzt ist der Befehl ausgeführt worden.

Schritt 3: Empfangen von UDP-Frames

FB_IpReceive

Mit diesem Baustein empfängt man die Daten. Sobald der Baustein aufgerufen wird hört der Baustein ob UDP-Frames angekommen sind. Es werden bis zu 4 UDP-Frames gepuffert, weitere UDP-Frames werden verworfen. Mit **sRemoteIPAddr** kann man einen IP-Adressfilter parametrieren um nur von einem

bestimmten Teilnehmer Daten zu empfangen. Will man alle UDP-Frames empfangen gibt man ein leer String an oder lässt diese Variabel offen. Hat man einen IP-Adressfilter parametriert kann man zusätzlich noch die Port Nummer rausfiltern. Dafür setzt man einfach in die Variabel **iPortNo** die entsprechende Portnummer ein. Will man auch hier alle UDP Daten empfangen dann lässt man die Variabel offen.
Werden Daten empfangen wird die Variabel *bValid* auf TRUE gesetzt. Die Daten sind jetzt gültig. Ist der Wert der Variabel **cbBytesInStream** ungleich Null, so ist die Variabel, die an den Baustein verknüpft worden ist zu klein gewählt worden und es befinden sich noch Daten im Puffer.

Ablauf des Beispielprogramms

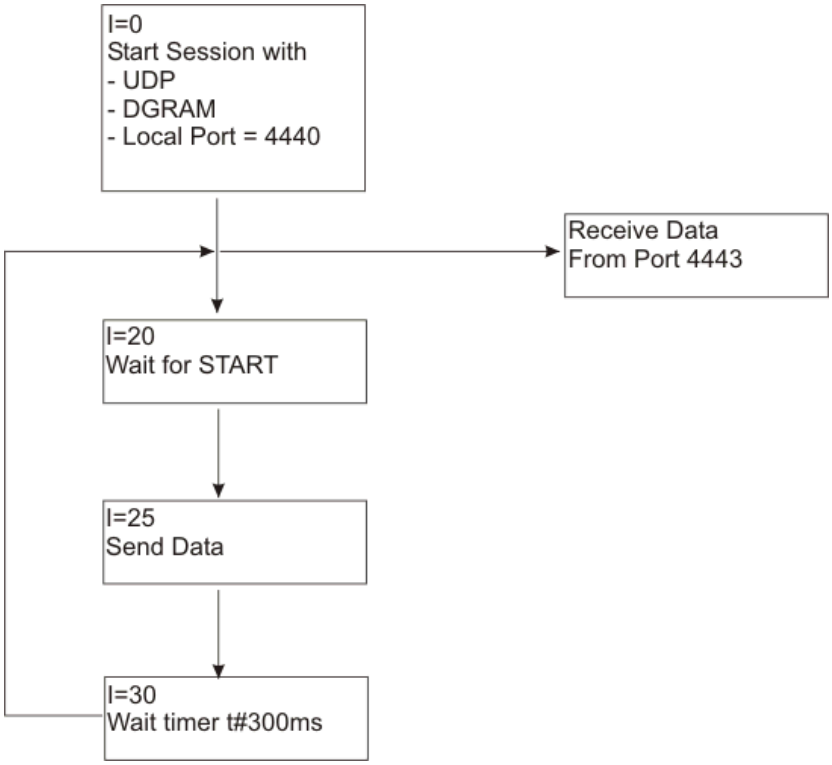


Abb. 136: Ablauf des Beispielprogramms

Beispiel

Download VB6 Programm als TCP/IP Server, Zip File (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207375115.zip)

Download BX9000 TwinCAT Projekt als TCP/IP Client, prx File (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207377291.zip)

5.13.7.3 ModbusTCP Client

5.13.7.3.1 ModbusTCP Client

Mit dem ModbusTCP Client (Modbus Master) kann der BX9000 eine Verbindung zu einem ModbusTCP-Server (Modbus Slave) aufnehmen. Es stehen 3 Bausteine zur Verfügung. Fb_MBConnect zum Aufbauen einer TCP/IP Verbindung, FB_MBGenericReq für das Versenden und Empfangen von beliebigen Modbus Funktionen und den FB_MBClose zum Schließen der TCP/IP Verbindung. Da es sich beim FB_MBGenericReq um einen allgemeinen Baustein handelt, müssen die entsprechenden Modbus Funktionen noch implementiert werden, dies erlaubt daher auch nicht Modbus konforme Protokolle zu versenden. Es sind maximal 3 Client Verbindungen gleichzeitig möglich. Durch die Möglichkeit eine TCP/IP Verbindung zu schließen und neu öffnen zu können, sind damit nahezu beliebig viele ModbusTCP-Server ansprechbar.

FB_MBConnect

Der FB_MBConnect verbindet den BX9000 über ModbusTCP mit einem anderen Teilnehmer. Es wird eine Verbindung aufgebaut mit steigender Flanke von *bExecute*. Die IP-Adresse wird über *sIPAddr* eingestellt. Die zu verwendete TCP-Portnummer über *nTCPPort* und ist normalerweise bei ModbusTCP auf 502 spezifiziert. Solange der Baustein aktiv ist, ist das **bBusy** gesetzt. Wurde die ModbusTCP Verbindung erfolgreich initialisiert, wird das *bBusy* auf FALSE gesetzt und das *bError* Flag ist FALSE. Sollte das *bError* Flag TRUE sein, ist der Verbindungsaufbau fehlgeschlagen und in der Variable *nErrId* ist der entsprechende Fehlercode enthalten. **nHandle***muss* mit den Bausteinen **FB_MBGernericReq** und **FB_MBClose** verbunden werden.

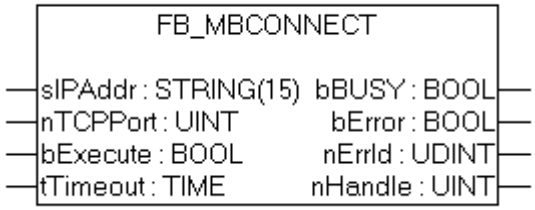


Abb. 137: Funktionsbaustein FB_MBCONNECT

INPUT

```
VAR_INPUT
   sIPAddr      : STRING(15);
   nTCPPort     : UINT;
   bExecute     : BOOL;
   tTimeout     : TIME;
END_VAR
```

sIPAddr: IP-Adresse des ModbusTCP-Servers (Slave Gerät) mit dem eine Modbus Verbindung hergestellt werden soll.

nTCPPort: ModbusTCP Port Nummer, normalerweise ist diese bei ModbusTCP 502*dez*.

bExecute: Eine steigende Flanke aktiviert den Baustein.

tTimeout: Zeit nach der abgebrochen werden soll.

OUTPUT

```
VAR_OUTPUT
   bBusy        : BOOL;
   bError          : BOOL;
   iErrId          : WORD;
   nHandle      : UINT;
END_VAR
```

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis die Befehlsausführung abgeschlossen wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle [▶ 162]).

nHandle: Handle, der mit den Bausteinen FB_MBGernericReq und FB_MBClose verbunden werden muss.

FB_MBGENERICREQ

Der Baustein FB_MBGENERICREQ ermöglicht das Senden und Empfangen beliebiger ModbusTCP Funktionen. Mit steigender Flanke von *bExecute* wird der Baustein aktive, das **bBusy** wird gesetzt und er sendet seine Daten, die im **pReqBuff** (Pointer auf die Daten die verschickt werden sollen) enthalten sind. Die Länge der zu senden Daten entnimmt der Baustein der Variable **cbReqLen**. Die Variable **nHandle** muss mit der Variable aus dem Baustein Fb_MBConnect **nHandle** verbunden werden. Die Antwort des ModbusTCP-Servers (Slave) wird in dem **pResBuff** (Pointer auf die Empfangsdaten) abgelegt. Die Größe der Variable sollte groß genug gewählt werden, damit alle Daten eines ModbusTCP Telegramms empfangen

werden. Empfängt der ModbusClient innerhalb der **tTimeout** Zeit keine Antwort, so wird das **bBusy** auf FALSE gesetzt und das **bError** gesetzt. In der Variable **nErrId** findet man den Fehlercode. In **cbResponse** ist die Anzahl der empfangenden Bytes enthalten. Diese können mit der Puffergröße der **cdResLen** verglichen werden. Ist **cbResponse** größer als **dResLen** gehen Daten verloren und Buffer muss größer gewählt werden.

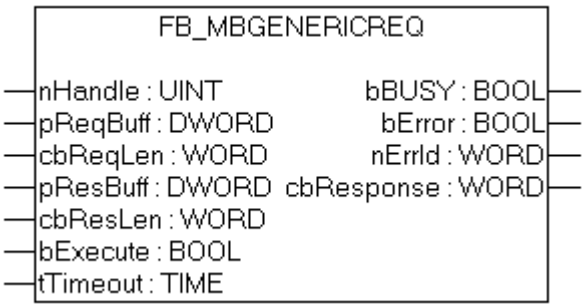


Abb. 138: Funktionsbaustein FB_MBGENERICREQ

INPUT

```
VAR_INPUT
   nHandle      : UINT;
   pRecBuff     : DWORD;
   cbReqLen     : WORD;
   pResBuff     : DWORD;
   cbResLen     : WORD;
   bExecute     : BOOL;
   tTimeout     : TIME;
END_VAR
```

nHandle: Handle, wird mit den Bausteinen Fb_MBConnect verbunden.

pReqBuff: Pointer auf die Daten, die versendet werden sollen.

cbReqLen: Länge der Daten die versendet werden sollen.

pResBuff: Pointer auf die Daten, die empfangen werden sollen. Die Variable muss groß genug angelegt werden.

cbResLen: Länge der Daten die empfangen werden sollen. Die Länge muss groß genug sein.

bExecute: Eine steigende Flanke aktiviert den Baustein.

tTimeout: Zeit nach der abgebrochen werden soll.

OUTPUT

```
VAR_OUTPUT
   bBusy        : BOOL;
   bError          : BOOL;
   iErrId          : WORD;
   cbResponse   : WORD;
END_VAR
```

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis die Befehlsausführung abgeschlossen wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle [▶ 162]).

cbResponse: Anzahl an Bytes die empfangen wurden.

FB_MBCLOSE

Der Baustein FB_MBCLOSE ermöglicht das Schließen einer TCP/IP Verbindung. Mit einer steigenden Flanke von **bExecute** wird der Baustein aktiviert. Solange der Baustein arbeitet ist das Bit **bBusy** gesetzt. **nHandle** muss mit dem Baustein Fb_MBConnect **nHandle** verbunden sein. Tritt ein Fehler auf wird **bError** auf TRUE gesetzt und in **nErrId** findet man den Fehlercode. Nach fehlerfreien Aufruf des Bausteins kann mit Fb_MBConnect ein neuer Socket geöffnet werden.

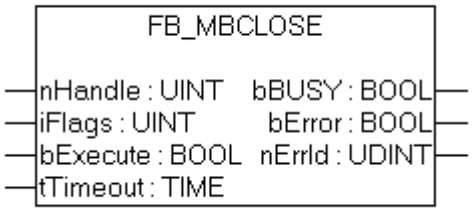


Abb. 139: Funktionsbaustein FB_MBCLOSE

INPUT

```
VAR_INPUT
   nHandle      : UINT;
   iFlags          : UINT;
   bExecute     : BOOL;
   tTimeout     : TIME;
END_VAR
```

nHandle: Handle, wird mit den Bausteinen Fb_MBConnect verbunden.

iFlags: 0 - TCP/IP Verbindung wird geschlossen auch wenn die Gegenstelle nicht erreichbar ist, 1 - TCP/IP Verbindung wird mit "FIN" beendet, damit der Baustein ohne Fehler die TCP/IP Verbindung schließt muss dieser vorhanden sein und die Kommunikation funktionieren. Es empfiehlt sich die 0 zu nehmen da hier auf jedenfall unabhängig vom Zustand der Gegenstelle die TCP/IP Verbindung geschlossen wird.

bExecute: Eine steigende Flanke aktiviert den Baustein.

tTimeout: Zeit nach der abgebrochen werden soll.

OUTPUT

```
VAR_OUTPUT
   bBusy        : BOOL;
   bError          : BOOL;
   iErrId          : WORD;
END_VAR
```

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis die Befehlsausführung abgeschlossen wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle [▶ 162]).

Beispiel

Für das Beispiel brauchen Sie zwei BX9000 oder statt eines BX9000 einen BC9x00.

Download Erster BX9000 als ModbusTCP Client (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207379467.zip)

Download Zweiter BX9000 als ModbusTCP-Server (oder einen BC9x00) (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207381643.zip)

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.10.0 und höher	BX9000 (165) firmware version >=1.14	TcBaseBX9000.lbx

Error Codes

Fehlercode	Beschreibung
0xFB00	keine gültige IP-Adresse
0xFC00	Response Länge größer als Speicher
0xFD00	Request konnte nicht abgesetzt werden
0xFF00	TimeOut während der Kommunikation

Modbus Close

Fehlercode	Beschreibung
0xFA00	Verbindung bereits geschlossen
0xFF00	TimeOut beim Schließen der Verbindung. Es wird RST verwendet und der Remote Teilnehmer ist nicht vorhanden.

Modbus Client Connect

Fehlercode	Beschreibung
0xFA00	Interne Verbindung wird verweigert
0xFA01	Connection Timeout Event
0xFA02	Connect wurde vom Remote nicht akzeptiert
0xFA03	ungültige IP Adresse
0xFA04	Alle Modbus Verbindungen belegt
0xFA05	kein interner Socket mehr vorhanden
0xFA06	interner Socket Fehler
0xFA07	Fehler beim Connect Aufruf

5.13.7.4 SMTP

5.13.7.4.1 FB_Smtp

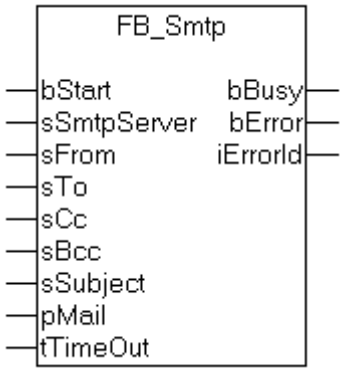


Abb. 140: Funktionsbaustein FB_Smtp

Der Funktionsbaustein nutzt das SMTP-Protokoll (Simple Mail Transfer Protocol) um E-Mails zu verschicken. Der Funktionsbaustein kann z. B. dazu benutzt werden Fehler, Diagnoseinformationen, Warnungen als E-Mail zu verschicken. Die Empfängeradressen werden als Strings an die *sTo*-, *sCc*-, *sBcc*- und *sSubject*-Eingangsvariablen übergeben. Die maximale Stringlänge der Empfängeradressen wurde auf 80 Zeichen begrenzt um die Ressourcen zu schonen. Der String mit dem eigentlichen Mail-Text darf auch länger sein.

VAR_INPUT

```
VAR_INPUT
    bStart      : BOOL;
    sSmtpServer     : STRING(15);
    sFrom       : STRING;
    sTo         : STRING;
    sCc         : STRING;
    sBcc        : STRING;
    sSubject        : STRING;
    pMail       : DWORD;
    tTimeOut        : TIME;
END_VAR
```

bStart: Eine steigende Flanke an diesem Eingang aktiviert den Funktionsbaustein.

sSmtpServer: IP-Adresse des SMTP-Servers als String.

sFrom: Ein String mit der E-Mail Adresse des Absenders. Wird ein Leerstring angegeben, dann erzeugt der Bus-Controller eine E-Mail Adresse aus dem Namen des Bus-Controllers und der MAC ID. Die maximale Stringlänge ist auf 80 Zeichen begrenzt. Es dürfen auch mehrere Empfängeradressen getrennt durch Semikolons angegeben werden.

sTo: Ein String mit der E-Mail Adresse des Empfängers. Es muss eine gültige E-Mail Adresse angegeben werden. Die maximale Stringlänge ist auf 80 Zeichen begrenzt. Es dürfen auch mehrere Empfängeradressen getrennt durch Semikolons angegeben werden.

sCc: Ein String mit der E-Mail Adresse eines weiteren Empfängers (Cc=Carbon Copy). Es kann auch ein Leerstring angegeben werden. Eine Kopie der E-Mail wird an diesen Empfänger gesendet. Die E-Mail Adresse dieses Empfängers wird bei anderen Empfängern **sichtbar**. Die maximale Stringlänge ist auf 80 Zeichen begrenzt. Es dürfen auch mehrere Empfängeradressen getrennt durch Semikolons angegeben werden.

sBcc: Ein String mit der E-Mail Adresse eines weiteren Empfängers (Bcc=Blind Carbon Copy). Es kann auch ein Leerstring angegeben werden. Eine Kopie der E-Mail wird an diesen Empfänger gesendet. Die E-Mail Adresse dieses Empfängers wird bei anderen Empfängern **nicht sichtbar**. Die maximale Stringlänge ist auf 80 Zeichen begrenzt. Es dürfen auch mehrere Empfängeradressen getrennt durch Semikolons angegeben werden.

sSubject: Ein String mit der Betreff-Zeile zu der E-Mail. Es kann auch ein Leerstring angegeben werden. Die maximale Stringlänge ist auf 80 Zeichen begrenzt.

pMail: Die Adresse (Pointer) eines nullterminierten Strings mit dem E-Mail Text. Es kann auch ein Leerstring angegeben werden. Die Adresse des Strings kann mit dem ADR-Operator ermittelt werden.

tTimeOut: Maximale Zeit, die bei der Ausführung des Befehls nicht überschritten werden darf.

VAR_OUTPUT

```
VAR_OUTPUT
    bBusy       : BOOL;
    bError      : BOOL;
    iErrorId        : WORD;
END_VAR
```

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis der Baustein eine Befehlsanforderung ausführt, längstens aber für die Dauer der, an dem *tTimeOut*-Eingang angelegten Zeit.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (Tabelle).

Fehlercode (hex)	Beschreibung
0x8000	SMTP server not found.
0x8001	Resource error.
0x8002	Socket resource error.
0x8003	Connection fault.
0x8004	Communication fault.
0x8005	Rx error. Communication time exceeded.
0x8006	Rx error. Communication fault.
0x8007	Rx error. Frame error.
0x8008	Communication error. Wrong response.
0x8009	Tx error. Communication fault.
0x800A	Communication shutdown error.
0x800B	Communication timeout.
0x8010	Invalid parameter.

Beispiel für einen Aufruf in FUP

```
PROGRAM MAIN
VAR
    fbSMTP   : FB_Smtp;
    bSend    : BOOL;
    sMsg     : STRING(100):='Test';
    bBusy    : BOOL;
    bError   : BOOL;
    nErrId   : UDINT;
END_VAR
```

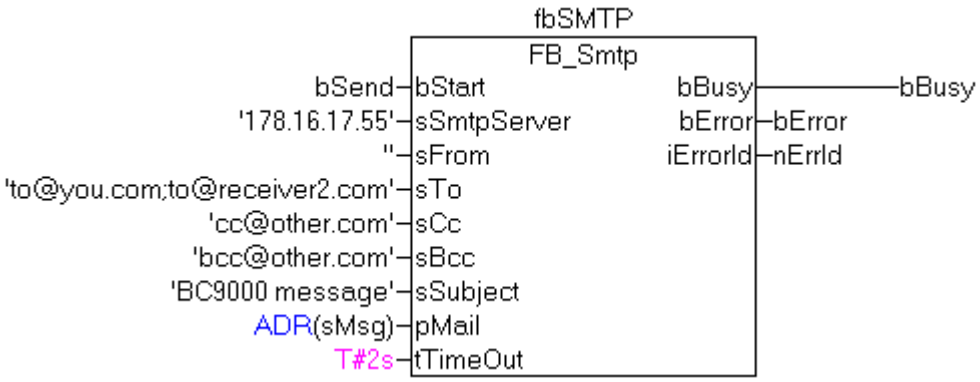


Abb. 141: Funktionsbaustein fbSMTP

Im Beispiel wird bei einer steigenden Flanke am *bStart*-Eingang eine Mail an 4 Empfänger verschickt.

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.10.0 und höher	BX9000 (165) firmware version >=1.12	TcBaseBX9000.lbx

5.13.7.5 SNTP

5.13.7.5.1 Time Protokoll (SNTP)

(BX9000 ab Firmware-Version 1.12, BC9050, BC9x20)

Das Simple Network Time Protokoll dient zu synchronisieren von Uhren über das Internet. Sie können den BX9000 zu einem Time-Server synchronisieren.

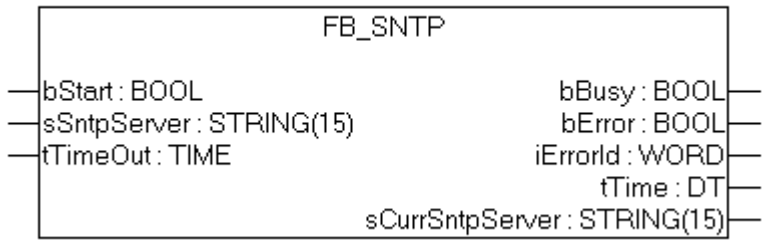


Abb. 142: Funktionsbaustein FB_SNTP

FUNCTION_BLOCK FB_SNTP

Wird eine IPAdresse eingetragen benutzt der Busklemmen Controller das SNTP-Protokoll. Wird eine Leer String übergeben wird das Time-Protokoll (UDP Port 37) verwendet.

VAR_INPUT

```
bStart              :BOOL;
sSntpServer         :STRING(15);
tTimeOut            :TIME;
```

bOpen: Positive Flanke startet den Baustein
sSntpServer: Eintrag des SNTP Servers. Wenn ein Leerstring eingegeben wird, wird das Time-Protokoll verwendet (UDP Port 37)*.
tTimeOut: TMOut nach dem abgebrochen werden soll

VAR_OUTPUT

```
bBusy               :BOOL;
bError              :BOOL;
iErrorId            :WORD;
tTime               :DT;
cCurrSntpServer     :STRING(15);
```

bBusy: So lange der Baustein TRUE ist der Baustein aktiv.
bError: Fehler Bit.
iErrorId: Fehler Nummer.
tTime: Zeit und Datum.
sCurrSnptServer: IP-Adresse des SNTP-Servers

Rückgabeparameter iErrId	Bedeutung
0	kein Fehler
<> 0	Fehler Nummer [▶ 170]

Download (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207383819.zip)

5.13.7.6 Services

5.13.7.6.1 FB_AddDnsServer

Abb. 143: Funktionsbaustein FB_AddDnsServer

Mit dem Funktionsbaustein können dem Bus-Controller maximal bis zu drei DNS-Server übergeben werden.

INPUT

```
VAR_INPUT
   bStart        : BOOL;
   sDNSServer     : STRING(15);
END_VAR
```

bStart: Eine steigende Flanke an diesem Eingang aktiviert den Funktionsbaustein.

sDNSServer: String mit der IP-Adresse des DNS-Servers.

OUTPUT

```
VAR_OUTPUT
   bBusy      : BOOL;
   bError     : BOOL;
   iErrorId       : WORD;
END_VAR
```

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis die Befehlsausführung abgeschlossen wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle [▶ 170]).

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.10.0 und höher	BX9000 (165) firmware version >=1.12 BC9191 firmware version >= 3.1	TcBaseBX9000.lbx

5.13.7.6.2 FB_GetHostByAddr

Abb. 144: Funktionsbaustein FB_GetHostByAddr

Mit dem Funktionsbaustein kann zu einer bestimmten IP-Adresse der zugehörige Hostname ermittelt werden.

INPUT

```
VAR_INPUT
   bStart     : BOOL;
   sIPAddr        : STRING(15);
   pHostName      : DWORD;
   cbMaxNameLen   : WORD;
END_VAR
```

bStart: Eine steigende Flanke an diesem Eingang aktiviert den Funktionsbaustein.

sIPAddr: Ein String mit der IP-Adresse des Hosts.

pHostName: Enthält die Adresse eines String-Puffers, in den der ermittelte Hostname hineingeschrieben wird. Der Programmierer ist selbst dafür verantwortlich, den Puffer in der Größe so zu dimensionieren, dass *cbMaxNameLen*-Bytes daraus entnommen werden können. Die Adresse kann man mit dem ADR - Operator ermitteln.

cbMaxNameLen: Enthält die Bytelänge des Puffers in den der ermittelte Hostname hineingeschrieben werden soll.

OUTPUT

```
VAR_OUTPUT
   bBusy     : BOOL;
   bError    : BOOL;
   iErrorId      : WORD;
END_VAR
```

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis die Befehlsausführung abgeschlossen wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle [▶ 170]).

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.10.0 und höher	BX9000 (165) firmware version >=1.12 BC9191 firmware version >= 3.1	TcBaseBX9000.lbx

5.13.7.6.3 FB_GetHostByName

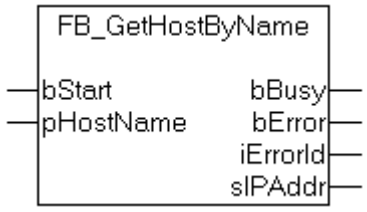


Abb. 145: Funktionsbaustein FB_GetHostByName

Mit dem Funktionsbaustein kann zu einem bestimmten Hostnamen die zugehörige IP-Adresse ermittelt werden.

INPUT

```
VAR_INPUT
   bStart       : BOOL;
   pHostName    : DWORD;
END_VAR
```

bStart: Eine steigende Flanke an diesem Eingang aktiviert den Funktionsbaustein.

pHostName: Enthält die Adresse einer Stringvariablen, mit dem Hostnamen. Die Adresse einer Stringvariablen kann mit dem ADR-Operator ermittelt werden.

OUTPUT

```
VAR_OUTPUT
   bBusy    : BOOL;
   bError       : BOOL;
   iErrorId     : WORD;
   sIPAddr      : STRING(15);
END_VAR
```

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis die Befehlsausführung abgeschlossen wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle [▶ 170]).

sIPAddr: Beim Erfolg enthält die Variable die IP-Adresse des Hosts.

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.10.0 und höher	BX9000 (165) firmware version >=1.12 BC9191 firmware version >= 3.1	TcBaseBX9000.lbx

5.13.7.6.4 FB_GetNetworkConfig

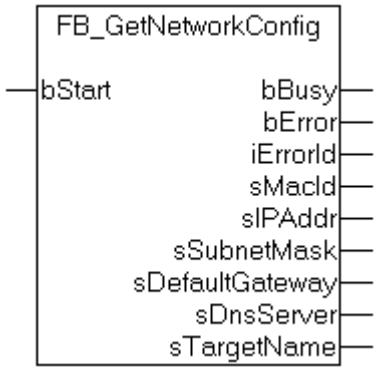


Abb. 146: Funktionsbaustein FB_GetNetworkConfig

Mit dem Funktionsbaustein können Informationen zu der aktuellen Netzwerkkonfiguration ermittelt werden.

INPUT

```
VAR_INPUT
   bStart          : BOOL;
END_VAR
```

bStart: Eine steigende Flanke an diesem Eingang aktiviert den Funktionsbaustein.

OUTPUT

```
VAR_OUTPUT
   bBusy           : BOOL;
   bError          : BOOL;
   iErrorId        : WORD;
   sMacId          : STRING(17);
   sIPAddr         : STRING(15);
   sSubnetMask     : STRING(15);
   sDefaultGateway     : STRING(15);
   sDnsServer      : STRING(15);
   sTargetName     : STRING(20);
END_VAR
```

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis die Befehlsausführung abgeschlossen wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle [▶ 170]).

sMacId: Liefert die MAC ID des Bus-Controllers zurück, z. B. '00-01-05-13-45-63'.

sIPAddr: Liefert die IP-Adresse des Bus-Controllers zurück.

sSubnetMask: Liefert die SubNet Mask.

sDefaultGateway: Liefert den default Geteway.

sDnsServer: Liefert den default DNS-Server (vom DHCP-Server zugewiesen) zurück.

sTargetName: Liefert den aktuellen Target-Namen des Bus-Controllers zurück.

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.10.0 und höher	BX9000 (165) firmware version >=1.12 BC9191 firmware version >= 3.1	TcBaseBX9000.lbx

5.13.7.6.5 FB_SetTargetName

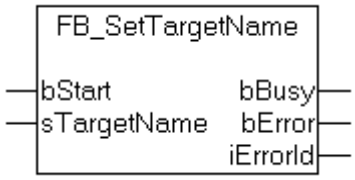


Abb. 147: Funktionsbaustein FB_SetTargetName

Mit dem Funktionsbaustein kann dem Bus-Controller ein Target-Name zugewiesen werden.

INPUT

```
VAR_INPUT
   bStart     : BOOL;
   sTargetName    : STRING(20);
END_VAR
```

bStart: Eine steigende Flanke an diesem Eingang aktiviert den Funktionsbaustein.

sTargetName: Ein String, der den neuen Target-Namen des Bus-Controllers enthält.

OUTPUT

```
VAR_OUTPUT
   bBusy     : BOOL;
   bError    : BOOL;
   iErrorId      : WORD;
END_VAR
```

bBusy: Dieser Ausgang bleibt solange auf TRUE, bis die Befehlsausführung abgeschlossen wurde.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls (siehe Tabelle).

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.10.0 und höher	BX9000 (165) firmware version >=1.12 BC9191 firmware version >= 3.1	TcBaseBX9000.lbx

5.13.7.6.6 Fehlernummern

Hex	Beschreibung
0x8010	falsche Parameter
0x8011	DNS-Server Liste voll
0x801F	Resourcen Fehler (intern)
0x8020	ungültige Hostlänge
0x8021	kein Hostname gefunden
0x802E	Heap Fehler (intern)
0x802F	Resourcen Fehler (intern)
0x8030	falsche Parameter
0x8031	ungültige Hostname
0x8032	Hostname zu lang
0x803F	Resourcen Fehler (intern)
0x804F	Resourcen Fehler (intern)
0x8050	keine gültige IP-Adresse
0x8052	keine gültige Klasse-D IP-Adresse
0x8055	Fehler im Stack (intern)
0x805F	Resourcen Fehler (intern)
0x8060	ERR_SNTP_INVALID_SERVER
0x8061	ERR_SNTP_NO_MORE_SOCKETS
0x8064	ERR_SNTP_RX_ERR
0x8065	**ERR_SNTP_CONN_SHUT_DOWN**
0x8066	ERR_SNTP_TIMEOUT
0x8100	falsche Parameter
0x8102	keine offene Verbindung
0x8104	keine Daten
0x8106	Pufferüberlauf (nur bei UDP)
0x8200	Verbindung wurde schon geöffnet
0x8201	Verbindungsfehler
0x8202	Zeitüberschreitung
0x8203	Resourcen Fehler (intern)
0x8300	Zeitüberschreitung
0x8301	Fernverbindung beendet
0x8302	Iinterner Fehler
0x8303	Resourcen Fehler (intern)
0x8304	keine Fernverbindung
0x8400	Verbindung icht geöffnet
0x8401	Zeitüberschreitung
0x8800	keine gültige Netzwerkkonfiguration (z.B. DHCP läuft noch)
0x8F09	ADS Socket steht nicht zur Verfügung (Zu viele offene ADS Verbindungen)
0x8F10	Session nicht gestartet
0x8F11	Session nicht bereit
0x8F12	ungültige Session ID
0x8F13	ungültiger Zustand (intern)
0x8F20	keine freien Slots. Es sind bereits 2 Sessions gestartet
0x8F21	Fehler bei der Taskinitialisierung
0x8F22	Session ist bereits geschlossen

5.13.8 TcSystemBX9000

5.13.8.1 Übersicht: TcSystemBX9000

Download

Zum Download der Bibliotheken bitte auf den Link mit der linken Maustasten klicken. Bitte die Bibliotheken in das Verzeichnis TwinCAT\PLC\LIB kopieren.

- TcSystemBX9000 (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207385995.zip)

TcSystemBX9000

Ethernet Diagnose	Version	Firmware
		BX9000
FreeAdsTcpConnections	12.10.06	1.15
FreeModbusConnections	12.10.06	1.15
ModbusTCP_Diag	12.10.06	1.15
ModbusTCP_Prm	12.10.06	1.15
EthernetDiag	12.10.06	1.15

Ethernet Utilities	Version	Firmware
		BX9000
NT_GetTime	12.10.06	1.15

HTML Page	Version	Firmware
		BX9000
HTTP [▶ 171]	12.10.06	1.15

5.13.8.2 HTML-Seite auf dem BX9000

Für Diagnosezwecke können Sie auf dem BX9000 eine einfache HTML-Seite anlegen. Hier können Strings aus der SPS heraus zur Anzeige gebracht werden. Ist die Seite aktiv, kann sie durch einfache Eingabe der IP-Adresse von einem Webbrowser angezeigt werden.

Anzeige Ihrer individuellen HTML-Seite

Im Auslieferungszustand ist auf dem BX9000 eine allgemeine HTML-Oberfläche hinterlegt. Diese muss deaktiviert werden um Ihre individuelle HTML-Seite anzuzeigen. Das übernimmt der Funktionsbaustein HTTP. Beim ersten Aufruf deaktiviert er die Default-Seite und meldet "bRebootNecessary:=TRUE". Das bedeutet, dass der Koppler neu gestartet werden muss. Beim erneuten Starten des BX9000 ist dann Ihre individuelle HTML-Seite abrufbar.

HTTP

Eine positive Flanke von **bActive** aktiviert die Web Seite. Steht die HTML Seite zur Verfügung ist das Bit **bIsActive** auf TRUE.

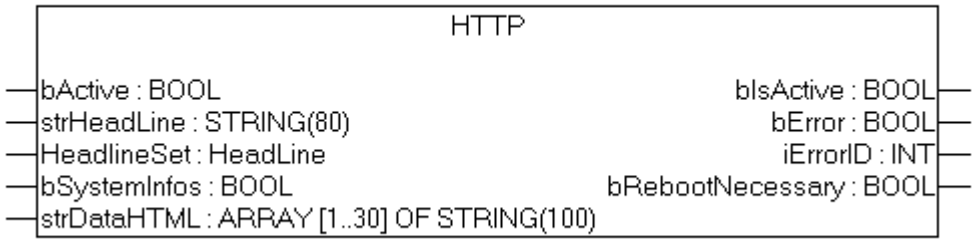


Abb. 148: Funktionsbaustein HTTP

INPUT

```
VAR_INPUT
   bActive      : BOOL;
   strHeadLine      : STRING(80);
   HeadlineSet  : HEADLINE;
   bSystemInfos : BOOL;
   strDataHTML  : ARRAY [1..30] OF STRING(100);
END_VAR
```

bActive: Positive Flanke aktiviert den Baustein.

strHeadLine: Überschrift der HTML Seite max. 80 Zeichen.

HeadLineSet: Einstellungen für die Schriftgröße und Schriftfarbe der Überschrift.

bSystemInfos: Systeminfos werden Angezeigt (TRUE), werden nicht mit Angezeigt (FALSE).

strDataHTML: max. 30 Zeilen mit max. 100 Zeichen pro Zeile. Ein leerer String beendet die HTML Seite, weitere Zeilen werden nicht angezeigt.

OUTPUT

```
VAR_OUTPUT
   bIsActive        : BOOL;
   bError               : BOOL;
   iErrorId         : INT;
   bRebootNecessary     : BOOL;
END_VAR
```

bIsActive: Die HTML Seite steht einem Webbrowser zur Verfügung.

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

iErrorId: Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls.

bRebootNecessary: Um die HTML Seite zu starten ist ein Reboot des Controllers notwendig.

Beispiel

Für das Beispiel brauchen Sie einen BX9000.

Download BX9000 mit HTML Seite (https://infosys.beckhoff.com/content/1031/bx9000/Resources/zip/3207388171.zip)

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.10.0 und höher	BX9000 (165) firmware version >=1.15	Standard.lbx, TcBaseBX.lbx, TcBaseBX9000.lbx, TcSystemBX9000.lbx, TcSystem.lbx

5.14 Programmübertragung

5.14.1 Programmübertragung über Ethernet

TwinCAT bietet die Möglichkeit, das Anwenderprogramm über den Feldbus auf den Busklemmen-Controller zu übertragen. Im PLC Control kann nach dem Sichern in der Registry und einem Restart des TwinCAT-Systems, als Zielsystem der BC/BX angewählt werden. Erforderlich ist der TwinCAT-Level TwinCAT PLC.

Mindestanforderungen:

- TwinCAT 2.10 Build 1251

Initialisieren des Busklemmen-Controllers

Möglichkeit 1: Wenn Sie TwinCAT als pollende PLC nutzen.
Um den Koppler im PLC Control auswählen zu können, muss er dem System erst einmal bekannt gemacht werden.
Tragen Sie den Busklemmen Controller in dem System Manager ein, legen Sie Art, Anzahl und Größe der Feldbusvariablen an und verknüpfen sie diese mit einer Task. Sichern Sie die Einstellungen und aktivieren Sie die Konfiguration. Danach starten Sie das TwinCAT System und die zyklische Task.

Möglichkeit 2: Wenn Sie TwinCAT nur zum Programmieren oder Konfigurieren nutzen:
Klicken Sie auf das TwinCAT-Icon und starten Sie die Eigenschaften. Unter AMS Router können Sie den BX9000 eintragen.
Name: beliebig
AMS Net Id: IP Adresse plus ".1.1"
IP Adresse: IP Adresse des BX9000
Transport Typ: TCP/IP

Nun starten Sie TwinCAT in den Config- (blaues TwinCAT Icon) oder RUN-Modus (Grünes TwinCAT Icon)

TwinCAT System Manager

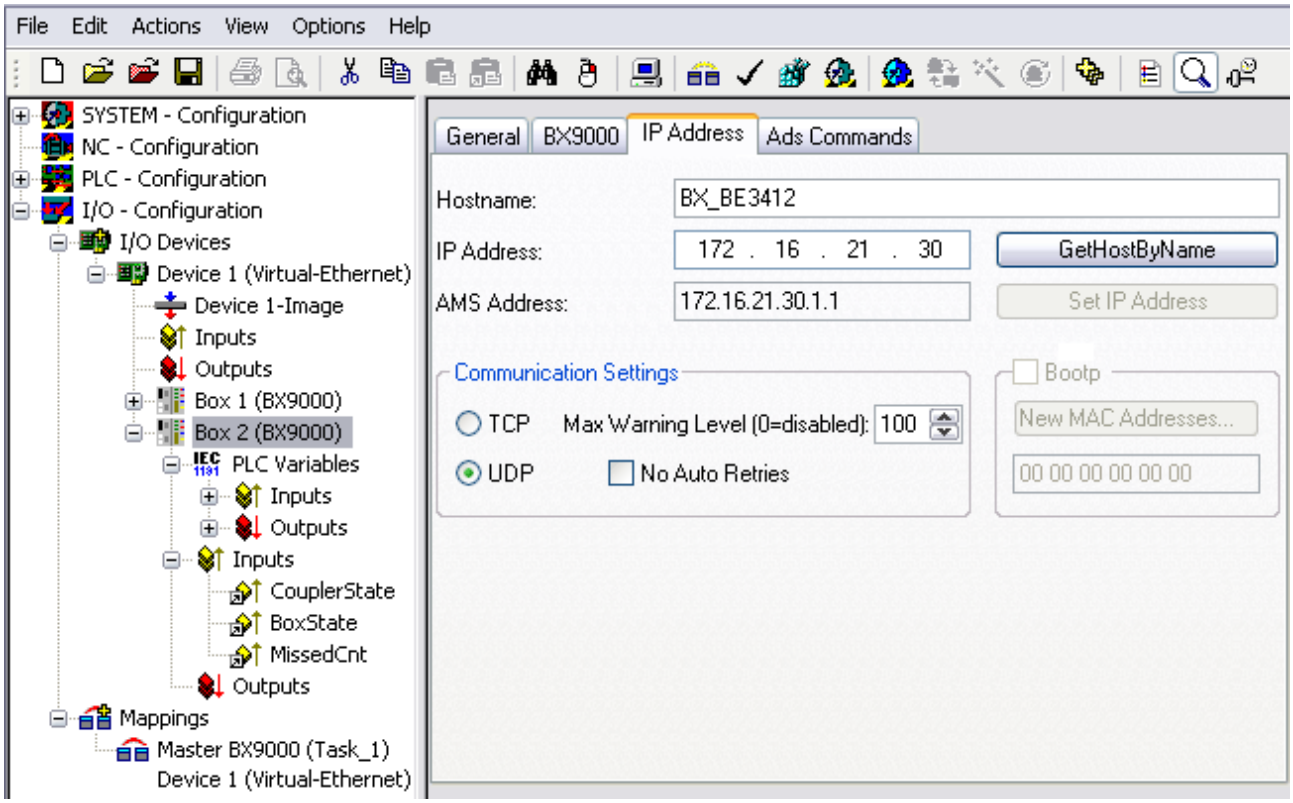


Abb. 149: IP-Adresse des BX9000 im TwinCAT System manager

PLC Control

Beim Neustart des TwinCAT PLC Control fragt TwinCAT nach der Zielplattform, d.h. auf welchem Gerät später das Anwender-Programm laufen soll. TwinCAT bietet als Steuerung zwei Zielplattformen, den PC oder den Busklemmen-Controller.

Für die Übertragung zum Busklemmen-Controller stehen Ihnen zwei Wege zur Verfügung:

- AMS für BCxx00 (Busklemmen Controller ohne Online Change)
- AMS für BCxx50 und BX (Busklemmen Controller mit Online Change)
- BC seriell, das serielle Kabel für die Kommunikation über die RS232-Schnittstelle [▶ 174] des PCs und die Programmierschnittstelle des Busklemmen-Controllers
- Für den BC9191 wählen Sie “BCxx50 or BX via AMS“ aus

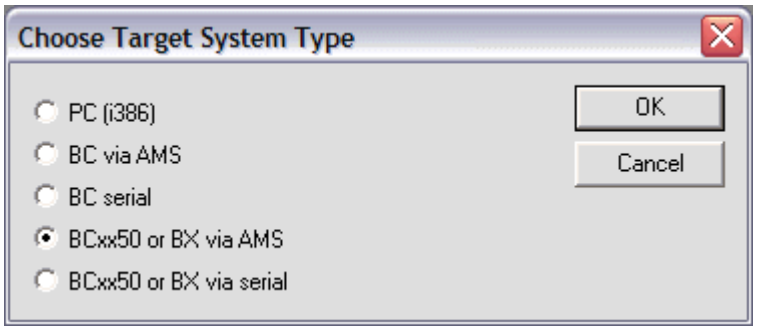


Abb. 150: Auswahl des Datenübertragungswegs - AMS

Wählen Sie nach Erstellung Ihres Programms unter der Symbolleiste *Online* das Zielsystem aus. Hierzu muss TwinCAT gestartet sein. Im Beispiel ist dies die Ethernet-Karte mit der Box 1 und die Run-Time 1 des Busklemmen-Controllers.

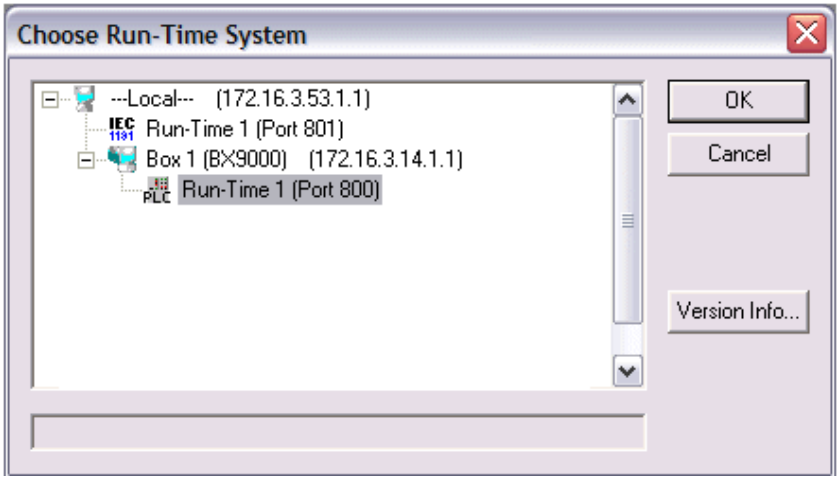


Abb. 151: Auswahl des Zielsystems

5.14.2 Programmübertragung über die serielle Schnittstelle

Jeder Busklemmen-Controller kann über die RS232-Schnittstelle des PCs programmiert werden.

Wählen Sie im TwinCAT PLC Control die serielle Schnittstelle an.

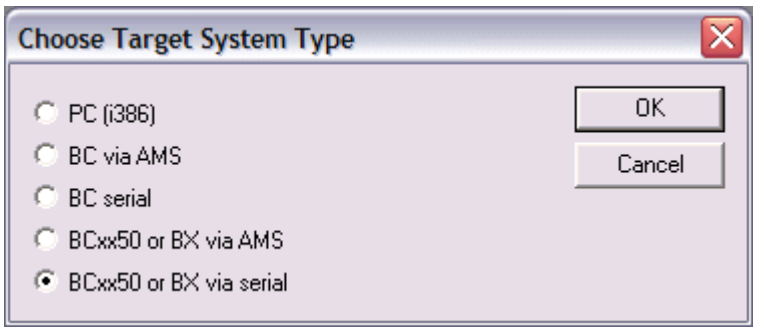


Abb. 152: *Auswahl des Datenübertragungswegs - Serielle Schnittstelle*

Unter Online/Kommunikationsparameter finden Sie im PLC Control die Einstellungen zur seriellen Schnittstelle, Portnummer, Baud-Rate usw.

Der Busklemmen Controller benötigt folgende Einstellung:

- Baud-Rate: 9600/19200/38400/57600 Baud (automatische Baudratenerkennung)
- Stop Bits: 1
- Parity: Gerade

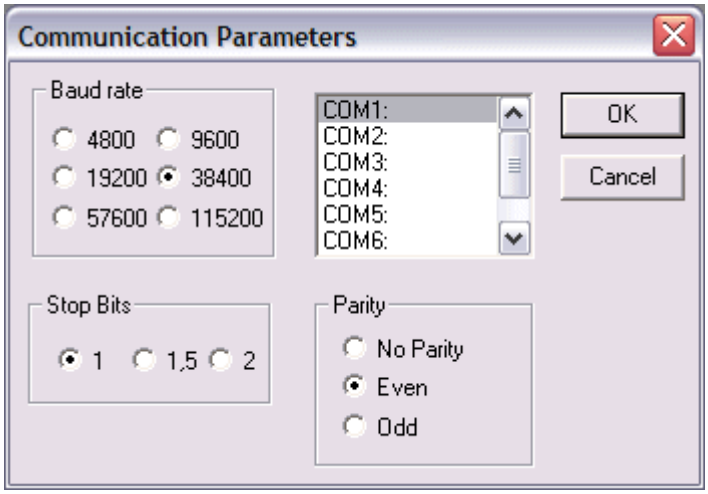


Abb. 153: Parametrierung der seriellen Schnittstelle

Programmübertragung über die serielle Schnittstelle per ADS

Der Busklemmen-Controller kann über die RS232-Schnittstelle des PC programmiert werden. Bevor Sie damit arbeiten können muss der Busklemmen-Controller dem TwinCAT bekannt gemacht werden (siehe serielles ADS [▶ 42]).

Wählen Sie im TwinCAT PLC-Control die ADS-Verbindung an.

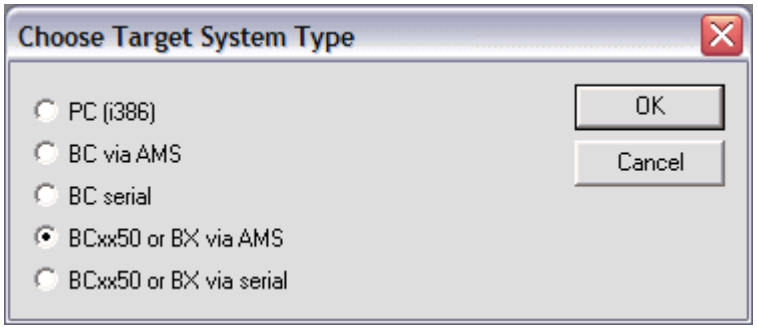


Abb. 154: *Auswahl des Datenübertragungswegs - AMS*

Im PLC Control kann man sich unter *Online/Communication Parameters...* einwählen.

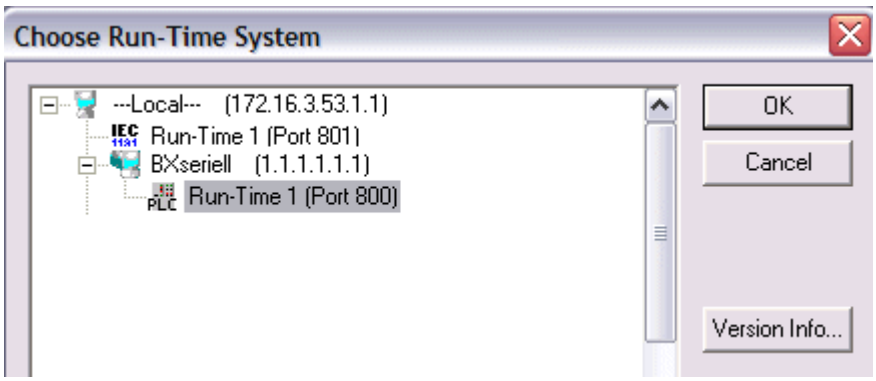


Abb. 155: Auswahl des Gerätes

5.15 Prozessabbild

5.15.1 Feldbusprozessabbild

Es gibt zwei Arten von Konfiguration auf den BX-Controller. Welche von beiden aktiv ist können sie auf den Display des BX auslesen nachdem der BX gestartet ist.

Defaut Config

In der Default Config liegen die Feldbusdaten ab der Adresse 1000. (%IB1000... und %QB1000).

TwinCAT Config

In der TwinCAT Config sind die Feldbusdaten nicht auf bestimmte Adressen gebunden sondern in Abhängigkeit wie diese Verknüpft worden sind.

Das Feldbusprozessabbild ist unabhängig vom verwendeten Ethernet Protokoll.

6 Ethernet

6.1 Systemvorstellung

6.1.1 Ethernet

Ethernet wurde ursprünglich von DEC, Intel und Xerox (als DIX-Standard) für die Datenübertragung zwischen Bürogeräten entwickelt. Heute versteht man darunter meist die Spezifikation *IEEE 802.3 CSMA/CD*, die 1985 veröffentlicht wurde. Diese Technologie ist durch ihren weltweiten Einsatz und die hohen Stückzahlen überall erhältlich und sehr preiswert. Eine Anbindung an vorhandene Netze kann so problemlos realisiert werden.

Mittlerweile gibt es die verschiedensten Übertragungsmedien: Koaxialkabel (10Base5), Lichtwellenleiter (10BaseF) oder verdrillte Zweidrahtleitung (10BaseT) mit Schirmung (STP) oder ohne Schirmung (UTP). Mit Ethernet lassen sich verschieden Topologien aufbauen wie Ring, Linie oder Stern.

Ethernet transportiert Ethernet-Pakete von einem Sender zu einem oder mehreren Empfängern. Diese Übertragung verläuft ohne Quittung und ohne Wiederholung von verlorenen Paketen. Für die sichere Daten-Kommunikation stehen Protokolle wie TCP/IP zu Verfügung, die auf Ethernet aufsetzen.

MAC-ID

Sender und Empfänger von Ethernet-Paketen werden über die MAC-ID adressiert. Die MAC-ID ist ein 6 Byte großer Identifikations-Code, der eineindeutig, d.h. für jedes Ethernet-Gerät weltweit unterschiedlich ist. Die MAC-ID besteht aus zwei Teilen. Der erste Teil (d.h. die ersten 3 Byte) ist eine Herstellerkennung. Die Firma Beckhoff hat die Kennung 00 01 05. Die nächsten 3 Byte werden durch den Hersteller vergeben und entsprechen einer eindeutigen Seriennummer. Die MAC-ID kann zum Beispiel beim BootP-Protokoll zum Einstellen der TCP/IP-Nummer verwendet werden. Dafür wird ein Telegramm zum entsprechenden Knoten geschickt, das die Informationen wie Name oder TCP/IP-Nummer beinhaltet. Sie können die MAC-ID mit der Konfigurationssoftware KS2000 auslesen.

Internet-Protokoll (IP)

Die Grundlage der Datenkommunikation ist das Internet-Protokoll (IP). IP transportiert Datenpakete von einem Teilnehmer zu einem anderen, der sich im gleichen oder in einem anderen Netz befinden kann. IP kümmert sich dabei um das Adress-Management (Finden und Zuordnen der MAC-IDs), die Segmentierung und die Pfadsuche (Routing). Wie das Ethernet-Protokoll gewährleistet auch IP keinen gesicherten Transport der Daten; Datenpakete können verloren gehen oder in ihrer Reihenfolge vertauscht werden.

Für einen standardisierten, gesicherten Informationsaustausch zwischen beliebig vielen verschiedenen Netzwerken wurde TCP/IP entwickelt. Dabei ist TCP/IP weitgehend unabhängig von der verwendeten Hard- und Software. Oftmals als ein Begriff verwendet, handelt es sich hierbei um mehrere aufeinander aufgesetzte Protokolle: z. B. IP, TCP, UDP, ARP und ICMP.

Transmission Control Protocol (TCP)

Das auf IP aufsetzende Transmission Control Protocol (TCP) ist ein verbindungsorientiertes Transport-Protokoll. Es umfasst Fehlererkennungs- und Behandlungsmechanismen. Verlorengegangene Telegramme werden wiederholt.

User Datagram Protocol (UDP)

UDP ist ein verbindungsloses Transport-Protokoll. Es gibt keine Kontrollmechanismen beim Datenaustausch zwischen Sender und Empfänger. Dadurch resultiert eine schneller Verarbeitungsgeschwindigkeit als zum Beispiel bei TCP. Eine Prüfung ob das Telegramm angekommen ist muss vom übergeordneten Protokoll durchgeführt werden.

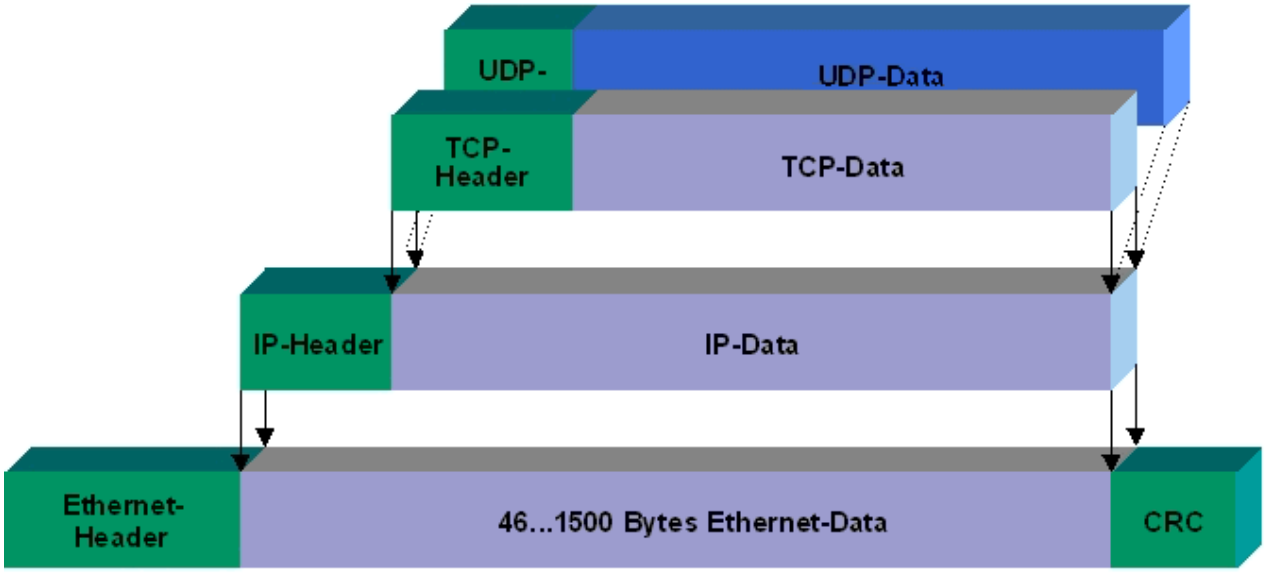


Abb. 156: User Datagram Protocol (UDP)

Auf TCP/IP und UDP/IP aufsetzende Protokolle

Auf TCP/IP bzw. UDP können folgende Protokolle aufsetzen:

- ADS
- ModbusTCP

Beide Protokolle sind parallel auf dem Buskoppler implementiert, so dass für die Aktivierung der Protokolle keine Konfiguration nötig ist.

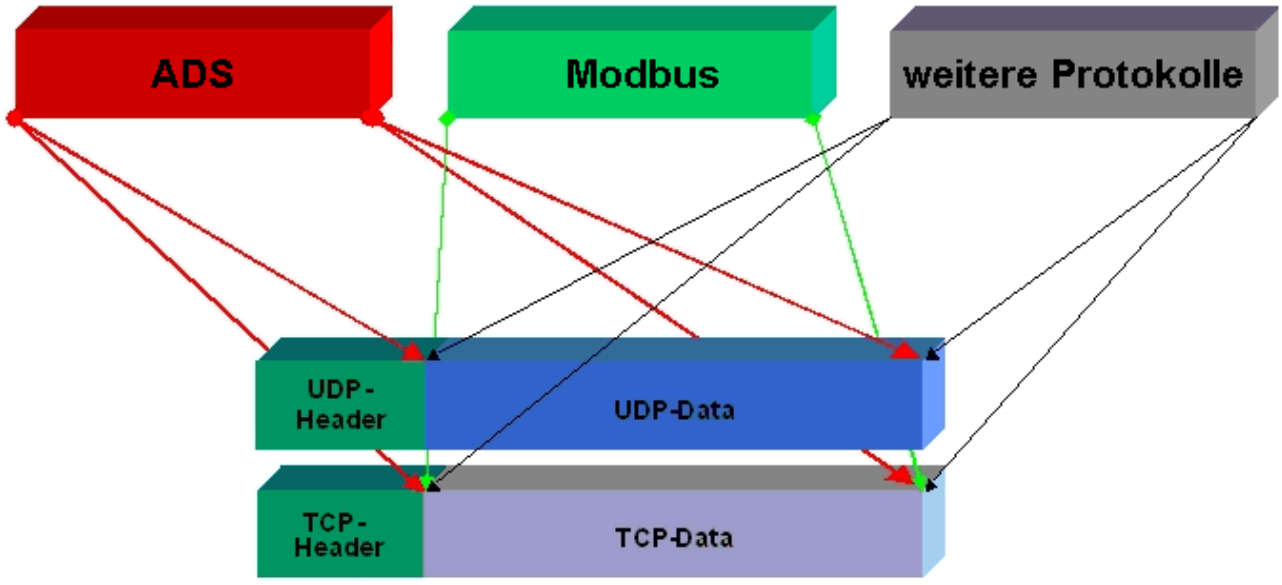


Abb. 157: Auf TCP/IP und UDP/IP aufsetzende Protokolle

ADS setzt wahlweise auf TCP oder UDP auf, während ModbusTCP stets auf TCP/IP basiert.

6.1.2 Topologie

Mit 10BaseT und 100BaseT werden mehrere Stationen im Ethernet-Standard sternförmig verbunden.

Stern-Topologie

Ein Stern-LAN besteht in der einfachsten Netzform aus einzelnen Punkt-zu-Punkt-Verbindungen. Alle Nachrichten laufen über einen zentralen Knoten (Hub oder Switch), der je nach Zieladresse die Informationen an den gewünschten Empfänger weitergibt.

Baum-Topologie

Eine Baum Topologie besteht aus mehreren verbundenen Stern-Topologien. Sobald mehrere Hubs oder Switches im Netz vorhanden sind, ist eine Baumtopologie vorhanden. Ideal ist es, die Verbindungen zwischen den Sternkopplern besonders breitbandig auszuführen, da diese den meisten Datenverkehr transportieren. Beim Aufbau von Baum-Topologien ist die Repeater-Regel zu beachten, die auch als 5-4-3-Repeater-Regel bezeichnet wird. Maximal zwei Repeater-Paare (bzw. Hub-Paare) dürfen im Übertragungsweg zwischen zwei beliebigen Stationen sein, sofern sie nicht durch Bridges, Switches oder Router getrennt sind. Ein Übertragungsweg kann aus maximal fünf Segmenten und vier Repeater-Sets (zwei Repeater-Paaren) bestehen. Dabei können bis zu drei Segmente Koax-Segmente sein, an denen die Stationen angeschlossen sind, die restlichen Segmente müssen Punkt-zu-Punkt-Verbindungen sein, die auch als IRL-Verbindung (Inter Repeater Link) bezeichnet werden.

Verkabelungsrichtlinien

Allgemeine Richtlinien für den Netzwerkaufbau eines LAN gibt die *Strukturierte Verkabelung* vor. Darin sind maximal zulässige Kabellängen für die Gelände-, Gebäude- und Etagenverkabelung festgelegt. In den Standards EN 50173, ISO 11801 und TIA 568-A normiert, bildet die *Strukturierte Verkabelung* die Grundlage für eine zukunftsweisende, anwendungsunabhängige und wirtschaftliche Netzwerk-Infrastruktur. Die Verkabelungsstandards definieren einen Geltungsbereich mit einer geographischen Ausdehnung von bis zu 3 km und für eine Bürofläche von bis zu 1 Mio. Quadratmeter mit 50 bis 50.000 Endgeräten. Darüber beschreiben sie Empfehlungen für den Aufbau eines Verkabelungssystems. Abhängig von der gewählten Topologie, den unter Industriebedingungen eingesetzten Übertragungsmedien und Koppelmodulen sowie von dem Einsatz von Komponenten verschiedener Hersteller in einem Netz können sich abweichende Angaben ergeben. Die Angaben verstehen sich hier deshalb lediglich als Empfehlungen.

6.1.3 Ethernet-Kabel

Übertragungsstandards

10Base5

Das Übertragungsmedium für 10Base5 ist ein dickes Koaxialkabel (Yellow Cable) mit einer max. Übertragungsgeschwindigkeit von 10 MBaud und einer Linien-Topologie mit Abzweigen (Drops), an die jeweils ein Teilnehmer angeschlossen wird. Da hier alle Teilnehmer an einem gemeinsamen Übertragungsmedium angeschlossen sind, kommt es bei 10Base5 zwangsläufig häufig zu Kollisionen.

10Base2

10Base2 (Cheaper net) ist eine Weiterentwicklung von 10Base5 und hat den Vorteil dass dieses Koaxialkabel billiger und durch eine höhere Flexibilität einfacher zu verlegen ist. Es können mehrere Geräte an eine 10Base2-Leitung angeschlossen werden. Häufig werden die Abzweige eines 10Base5-Backbones als 10Base2 ausgeführt.

10BaseT

Beschreibt ein Twisted-Pair-Kabel für 10 MBaud. Hierbei wird das Netz sternförmig aufgebaut, so dass nun nicht mehr jeder Teilnehmer am gleichem Medium hängt. Dadurch führt ein Kabelbruch nicht mehr zum Ausfall des gesamten Netzes. Durch den Einsatz von Switches als Sternkoppler können Kollisionen vermindert oder bei Voll-Duplex Verbindungen auch vollständig vermieden werden.

100BaseT

Twisted-Pair-Kabel für 100 MBaud. Für die höhere Datengeschwindigkeit ist eine bessere Kabelqualität und die Verwendung entsprechender Hubs oder Switches erforderlich.

10BaseF

Der Standard 10BaseF beschreibt mehrere Lichtwellenleiter-Varianten.

Kurzbezeichnung der Kabeltypen für 10BaseT und 100BaseT

Twisted-Pair Kupferkabel für sternförmige Topologie, wobei der Abstand zwischen zwei Geräten 100 Meter nicht überschreiten darf.

UTP

Unshielded Twisted-Pair (nicht abgeschirmte, verdrillte Leitung)
Dieser Kabeltyp gehört zur Kategorie 3 und sind für industrielle Umgebungen nicht empfehlenswert.

S/UTP

Screened/Unshielded Twisted-Pair (mit Kupfergeflecht abgeschirmte, verdrillte Leitung)
Besitzen einen Gesamtschirm aus einem Kupfergeflecht zur Reduktion der äußeren Störeinflüsse. Dieses Kabel wird zum Einsatz mit dem Buskopplern empfohlen.

FTP

Foilesshielded Twisted-Pair (mit Alufolie abgeschirmte, verdrillte Leitung)
Dieses Kabel hat eine alukaschierten Kunststoff-Folie-Gesamtschirm.

S/FTP

Screened/Foilesshielded Twisted-Pair (mit Kupfergeflecht und Alufolie abgeschirmte, verdrillte Leitung)
Besitzt einen alukaschierten Gesamtschirm mit einem darüber liegenden Kupfergeflecht. Solche Kabel können eine Störleistungsunterdrückung bis zu 70 dB erreichen.

STP

Shielded Twisted-Pair (abgeschirmte, verdrillte Leitung)
Beschreibt ein Kabel mit Gesamtschirm ohne weitere Angabe der Art der Schirmung.

S/STP

Screened/Shielded Twisted-Pair (einzeln abgeschirmte, verdrillte Leitung)
Ein solche Bezeichnung kennzeichnet ein Kabel mit einer Abschirmung für jedes Leitungspaar sowie einen Gesamtschirm.

ITP

Industrial Twisted-Pair
Ist von Aufbau dem S/STP ähnlich, besitzt allerdings im Gegensatz zum S/STP nur 2 Leitungspaare.

6.2 ModbusTCP

6.2.1 ModbusTCP-Protokoll

Das Ethernet-Protokoll wird über die MAC-ID adressiert. Der Anwender braucht sich meist um diese Adresse nicht zu kümmern. Die IP-Nummer ist 4 Byte groß und muss vom Anwender auf dem Buskoppler und in der Anwendung parametriert werden. Der TCP-Port ist bei ModbusTCP auf 502 festgelegt. Die UNIT ist bei ModbusTCP frei wählbar und braucht vom Anwender nicht konfiguriert werden.

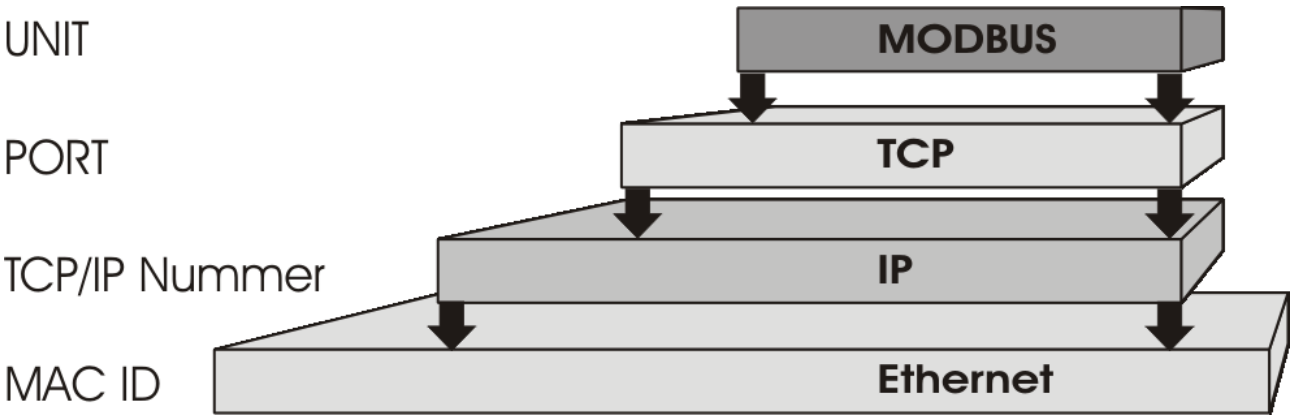


Abb. 158: ModbusTCP-Protokoll

TCP-Port-Nummer

Die TCP-Port-Nummer von ModbusTCP wurde auf den Wert 502 standardisiert.

Modbus-Unit

Die Unit wird vom Slave zurückgegeben.

ModbusTCP-Protokoll

Byte	Name	Beschreibung
0	Transaction identifier	wird vom Slave zurückgesendet
1	Transaction identifier	wird vom Slave zurückgesendet
2	Protocol identifier	immer 0
3	Protocol identifier	immer 0
4	Length field	0 (wenn die Nachricht kleiner 256 Byte ist)
5	Length field	Anzahl der folgenden Bytes
6	UNIT identtifier	wird vom Slave zurückgegeben
7	Modbus	es folgt das Modbus-Protokoll beginnend mit der Funktion

6.2.2 Modbus TCP-Interface

Adresse		Beschreibung		
0x0000 0x00FF		Prozessdaten-Inferface Eingänge		
0x0800 0x08FF		Prozessdaten-Inferface Ausgänge		
0x1000 0x1006	Read only	Buskopplerkennung		
0x100A		2 Byte SPS-Interface		
0x100B		Busklemmendiagnose		
0x100C		Buskoppler Status		
0x1010		Prozessabbildlänge in Bit, analoge Ausgänge (ohne SPS Variablen)		
0x1011		Prozessabbildlänge in Bit, analoge Eingänge (ohne SPS Variablen)		
0x1012		Prozessabbildlänge in Bit, digitale Ausgänge		
0x1013		Prozessabbildlänge in Bit, digitale Eingänge		
0x1020		Watchdog, aktuelle Zeit in [ms]		
0x110A	Read/Write	2 Byte SPS Interface		
0x110B		Busklemmendiagnose		
0x1120		Watchdog vordefinierte Zeit in [ms] (Default: 1000)		
0x1121		Watchdog Reset Register		
0x1122		Art des Watchdogs	1	Telegramm Watchdog (Default)
			0	Schreibtelegramm Watchdog
0x1123**		ModbusTCP Mode**	1	Fast Modbus
			0	Normal Modbus (Default)
0x4000* 0x47FF		Merkerbereich (%MB..)*		

* alle Busklemmen Controller BC9xx0 und BX9000
** für BC9x00 ab Firmware B7 und BK9000 ab Firmware B5 und alle nicht aufgeführten BK9xxx und BC/BX9xxx

Watchdog

Der Watchdog ist im Auslieferungszustand aktiviert. Nach dem ersten Schreibtelegramm wird der Watchdog scharf geschaltet und bei jedem empfangenden Telegramm dieses Teilnehmers getriggert. Andere Teilnehmer haben auf den Watchdog keinen Einfluss. Eine zweite Möglichkeit, die eine schärfere Bedingung des Watchdogs darstellt, ist, dass der Watchdog nur nach jedem Schreibtelegramm getriggert wird. Dafür Schreiben Sie in das Register 0x1122 eine Null (Default "1").
Der Watchdog kann deaktiviert werden in dem im Offset 0x1120 eine Null geschrieben wird. Das Watchdog Register darf nur dann beschrieben werden, wenn der Watchdog noch nicht aktiv ist. Die Daten in diesem Register bleiben gespeichert.

Watchdog-Register

Sollte der Watchdog auf Ihren Slave abgelaufen sein können sie diesen durch ein zweimaliges beschreiben des Registers 0x1121 zurücksetzen. Dazu muss folgendes in das Register geschrieben werden: 0xBECF 0xAFFE. Dies kann mit der Funktion 6 oder der Funktion 16 geschehen.

Statusregister des Buskopplers

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	FB	-	-	-	-	-	-	-	-	-	-	-	-	-	CNF	KB

Legende

Bit	Name	Wert	Beschreibung
15	FB	1_{bin}	Feldbusfehler, Watchdog abgelaufen
14...2	-	-	reserviert
1	CNF	1_{bin}	Buskoppler Konfigurationsfehler
0	KB	1_{bin}	Busklemmenfehler

ModbusTCP Mode

Der Fast Modbus Mode sollte nur in kleinen lokalen Netzwerken eingesetzt werden. In der Default Einstellung ist der Fast ModbusTCP nicht aktiviert. Werden Probleme mit dieser Kommunikationsart festgestellt, ist der Buskoppler auf die "normale" ModbusTCP Kommunikation umzustellen. Per Modbus Interface Offset 0x1123 ist dieser Mode einzustellen. Nach der Änderung ist ein Reset (zum Beispiel mit der ModbusTCP Funktion 8) des Buskopplers notwendig. Im Fast Modbus Mode ist es nicht gestattet, mehr als ein Modbus Dienst in einem Ethernet Frame zu versenden.

2 Byte SPS-Interface

Mit dem 2 Byte SPS-Interface können Register der komplexen Klemmen sowie Register des Busklemmen-Controllers gelesen bzw. beschrieben werden. Die Register der komplexen Klemmen sind in der Dokumentation zur jeweiligen Klemme beschreiben. Über die Register des Buskopplers können z. B. K-Busdiagnosedaten, der Klemmenaufbau oder Zykluszeiten gelesen sowie die programmierte Konfiguration beschrieben werden. Weiterhin kann darüber auch ein manueller K-Bus-Reset durchgeführt werden. Das 2 Byte SPS-Interface benötigt je zwei Bytes in den Ein- und Ausgangsdaten, über die ein spezielles Protokoll abgewickelt wird. Eine Beschreibung des 2 Byte SPS-Interface, der verfügbaren Register im Buskopplers sowie Funktionsbausteine für verschiedene SPS-Systeme, die das 2 Byte SPS-Interface unterstützen, kann auf Anfrage geliefert werden.

2 Byte Diagnose-Interface

Die Fehlermeldungen der Klemmen können mit dem 2 Byte Diagnose-Interface gesendet werden. Dazu ist aber die K-Busdiagnose zu aktivieren. Das 2 Byte Diagnose-Interface belegt je zwei Bytes in den Ein- und Ausgangsdaten, über die ein spezielles Protokoll durchgeführt wird. Eine Beschreibung des 2 Byte Diagnose-Interfaces kann auf Anfrage geliefert werden.

6.2.3 Fehlerantwort des ModbusTCP Slaves (BK9000, BX/BC9xx0, BC9191, IP/ILxxxx-B/C900, EK9000)

Wenn der Anwender dem Slave eine Anforderung oder Mitteilung sendet, die der Koppler nicht versteht, antwortet der Slave mit einer Fehlermitteilung. Diese Antwort enthält die Funktion und den Fehler-Code. Der Funktionsrückgabewert wird mit 0x80 addiert.

Code	Name	Bedeutung
1	ILLEGAL FUNKTION	Nicht implementierte Modbus-Funktion
2	ILLEGAL DATA ADDRESS	Ungültige Adresse oder Länge
3	ILLEGAL DATA VALUE	Ungültige Parameter - Diagnose-Funktionen - falsches Register
4	SLAVE DEVICE ERROR	Watchdog- oder K-Bus-Fehler EK9000: E-Bus Fehler
6	SLAVE DEVICE BUSY	Es wird schon von einem anderen IP-Teilnehmer Ausgangs-Daten empfangen

6.2.4 ModbusTCP-Funktionen

6.2.4.1 Read holding register (Funktion 3)

Mit der Funktion *Read holding register* können die Ein- und Ausgangsworte und die Register gelesen werden. Eingänge ab dem Offset 0 - 0xFF und Ausgänge ab den Offset 0x800 - 0x8FF und bei den Steuerungen (BC, BX) der Merker Bereich ab den Offset 0x4000.

In diesem Beispiel werden die ersten zwei analogen Ausgänge (oder 2 Ausgangsworte) gelesen. Die analogen Ausgänge (oder Ausgangsworte) beginnen beim Offset 0x800. Die Länge bezeichnet die Anzahl, der zu lesenden Kanäle (oder der zu lesenden Worte).

Anfrage (Query)

Byte Name	Beispiel
Funktions-Code	3
Start-Adresse high	8
Start-Adresse low	0
Anzahl high	0
Anzahl low	2

Der Feldbus-Koppler antwortet mit dem Byte Count 4, d.h. es kommen 4 Byte Daten zurück. Die Anfrage waren zwei Analogkanäle, die auf zwei Worte aufgeteilt sind. Im analogen Ausgangsprozessabbild hat der erste Kanal einen Wert von 0x3FFF und der zweite Kanal einen Wert von 0x0.

Antwort (Response)

Byte Name	Beispiel
Funktions-Code	3
Byte Count	4
Daten 1 High-Byte	63
Daten 1 Low-Byte	255
Daten 2 High-Byte	0
Daten 2 Low-Byte	0

6.2.4.2 Read input register (Funktion 4)

Die Funktion *Read input register* liest wort-orientiert die Eingänge aus.

In diesem Beispiel werden die ersten zwei analogen Eingänge (oder die erste 2 Eingangsworte) gelesen. Die analogen Eingänge (oder Eingangsworte) beginnen bei einem Offset von 0x0000. Die Länge bezeichnet die Anzahl der zu lesenden Worte. Eine KL3002 z. B. hat zwei Worte Eingangsdaten, daher ist die einzugebende Länge bei *Anzahl low* zwei.

Anfrage (Query)

Byte Name	Beispiel
Funktions-Code	4
Start-Adresse high	0
Start-Adresse low	0
Anzahl high	0
Anzahl low	2

Der Feldbus-Koppler antwortet mit dem Byte Count 4, d.h. es kommen vier Byte Daten zurück. Die Anfrage waren zwei analog Kanäle, die jetzt auf zwei Worte aufgeteilt werden. Im analogen Eingangsprozessabbild hat der erste Kanal einen Wert von 0x0038 und der zweite Kanal einen Wert von 0x3F1B.

Antwort (Response)

Byte Name	Beispiel
Funktions-Code	4
Byte Count	4
Daten 1 High-Byte	0
Daten 1 Low-Byte	56
Daten 2 High-Byte	63
Daten 2 Low-Byte	11

6.2.4.3 Preset single register (Funktion 6)

Mit der Funktion *Preset single register* kann auf das Ausgangs- oder Merkerprozessabbild (nur bei den Steuerungen) und das Modbus-TCP-Interface [▶ 182] zugegriffen werden.

Bei der Funktion 6 wird das erste Ausgangswort beschrieben. Die Ausgänge beginnen bei einem Offset von 0x0800. Auch hier beschreibt der Offset immer ein Wort. Das heißt, dass der Offset 0x0803, das 4. Wort auf dem Ausgangsprozessabbild ist.

Anfrage (Query)

Byte Name	Beispiel
Funktions-Code	6
Start-Adresse high	8
Start-Adresse low	0
Daten high	63
Daten low	255

Der Feldbus-Koppler antwortet mit dem gleichen Telegramm und der Bestätigung der empfangenen Werte.

Antwort (Response)

Byte Name	Beispiel
Funktions-Code	6
Start-Adresse high	8
Start-Adresse low	0
Daten high	63
Daten low	255

6.2.4.4 Preset multiple register (Funktion 16)

Mit der Funktion *Preset multiple register* können mehrere Ausgänge beschrieben werden. In diesem Beispiel werden die ersten zwei analogen Ausgangsworte beschrieben. Die Ausgänge beginnen bei einem Offset von 0x0800. Hier beschreibt der Offset immer ein Wort. Der Offset 0x0003 schreibt ab dem vierten Wort auf das Ausgangsprozessabbild. Die Länge gibt die Anzahl der Worte an und der *Byte Count* setzt sich aus den zu schreibenden Bytes zusammen.

Beispiel: 4 Worte - entsprechen 8 Byte Count

Die Datenbytes enthalten die Werte für die analogen Ausgänge. In diesem Beispiel sind es zwei Worte, die zu beschreiben sind. Das erste Wort mit dem Wert 0x7FFF und das zweite Wort mit dem Wert 0x3FFF.

Anfrage (Query)

Byte Name	Beispiel
Funktions-Code	16
Start-Adresse high	8
Start-Adresse low	0
Länge high	0
Länge low	2
Byte Count	4
Daten 1 Byte 1	127
Daten 1 Byte 2	255
Daten 2 Byte 1	63
Daten 2 Byte 2	255

Antwort (Response)

Der Koppler antwortet mit der Start-Adresse und der Länge der gesendeten Worte.

Byte Name	Beispiel
Funktions-Code	16
Start-Adresse high	8
Start-Adresse low	0
Länge high	0
Länge low	2

6.2.4.5 Read / write registers (Funktion 23)

Mit der Funktion *Read / write registers* können mehrere analoge Ausgänge beschrieben und in einem Telegramm mehrere analoge Eingänge gelesen werden. In diesem Beispiel werden die ersten zwei analogen Ausgangsworte beschrieben und die ersten zwei analogen Eingänge gelesen. Die analogen Ausgänge beginnen beim Offset 0x0800 und die Eingänge ab dem Offset 0x0000. Hier beschreibt der Offset immer ein Wort. Der Offset 0x0003 schreibt ab dem 4. Wort auf das Ausgangsprozessabbild. Die Länge gibt die Anzahl der Worte an und der *Byte Count* setzt sich aus den zu schreibenden Bytes zusammen. Beispiel: 4 Worte - entsprechen 8 Byte Count

Die Datenbytes enthalten die Werte für die analogen Ausgänge. In diesem Beispiel sind es zwei Worte, die zu beschreiben sind. Das erste Wort mit dem Wert 0x3FFF und das zweite Wort mit dem Wert 0x7FFF.

Anfrage (Query)

Byte Name	Beispiel
Funktions-Code	23
Lesen Start-Adresse high	0
Lesen Start-Adresse low	0
Lesen Länge high	0
Lesen Länge low	2
Schreiben Start-Adresse high	8
Schreiben Start-Adresse low	0
Schreiben Länge high	0
Schreiben Länge low	2
Byte Count	4
Daten 1 high	63
Daten 1 low	255
Daten 2 high	127
Daten 2 low	255

Antwort (Response)

Der Koppler antwortet mit der Start-Adresse und der Länge der übertragenen Bytes im *Byte Count*. Es folgen die Dateninformationen. In diesem Beispiel steht im ersten Wort eine 0x0038 und im zweiten Wort eine 0x3F0B.

Byte Name	Beispiel
Funktions-Code	23
Byte Count	4
Daten 1 high	0
Daten 1 low	56
Daten 2 high	63
Daten 2 low	11

6.3 ADS-Kommunikation

6.3.1 ADS-Kommunikation

Das ADS-Protokoll (ADS: Automation Device Specification) ist eine Transportschicht innerhalb des TwinCAT-Systems. Es ist für den Datenaustausch der verschiedenen Software-Module entwickelt worden, zum Beispiel für die Kommunikation zwischen der NC und der PLC. Mit diesem Protokoll haben Sie die Freiheit von jedem Punkt im TwinCAT mit anderen Tools kommunizieren zu können. Wird die Kommunikation zu anderen PCs oder Geräten benötigt, setzt das ADS-Protokoll auf TCP/IP auf. Somit ist es in einem vernetzten System möglich, alle Daten von einem beliebigen Punkt aus zu erreichen.

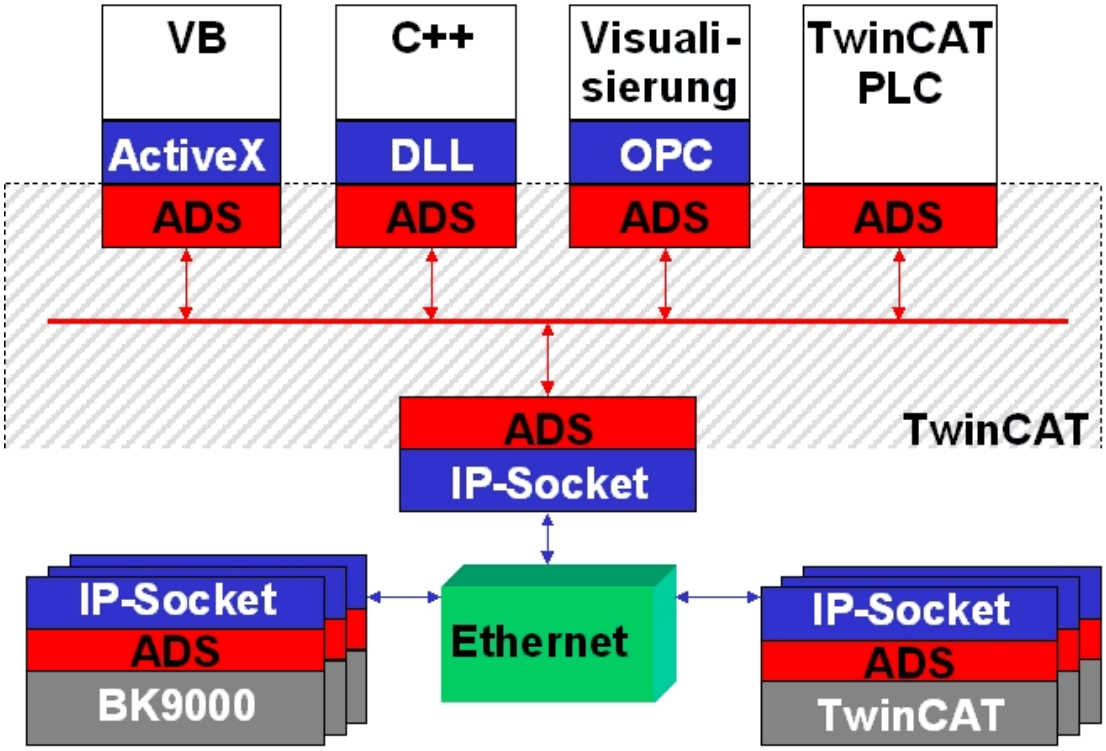


Abb. 159: Das ADS-Protokoll als Transportschicht innerhalb des TwinCAT-Systems

Das ADS-Protokoll wird auf das TCP/IP- oder UDP/IP-Protokoll aufgesetzt. Es ermöglicht dem Benutzer innerhalb des Beckhoff-Systems über nahezu beliebige Verbindungswege mit allen angeschlossenen Geräten zu kommunizieren und diese zu parametrieren. Außerhalb des Beckhoff-Systems stehen verschiedene Wege offen, um mit anderen Software-Tools Daten auszutauschen.

Software-Schnittstellen

ADS-OCX

Das ADS-OCX ist eine Active-X-Komponente und bietet eine Standardschnittstelle zum Beispiel zu Visual Basic, Delphi, usw.

ADS-DLL

Sie können die ADS-DLL (DLL: Dynamic Link Library) in Ihr C-Programm einbinden.

OPC

Die OPC-Schnittstelle ist eine genormte Standardschnittstelle für die Kommunikation in der Automatisierungstechnik. Beckhoff bietet hierfür einen OPC-Server an.

6.3.2 ADS-Protokoll

Die ADS-Funktionen bieten die Möglichkeit, direkt vom PC auf Informationen des Buskopplers zuzugreifen. Dafür können ADS-Funktionsbausteine im TwinCAT PLC Control verwendet werden. Die Funktionsbausteine sind in der Bibliothek *TcSystem.lib* enthalten. Genauso ist es möglich, die ADS-Funktionen von AdsOCX, ADSDLL oder OPC aufzurufen. Über die ADS-Portnummer 300 sind alle Daten und über die ADS-Portnummer 100 Zugriffe auf die Register des Buskopplers und der Klemmen möglich.

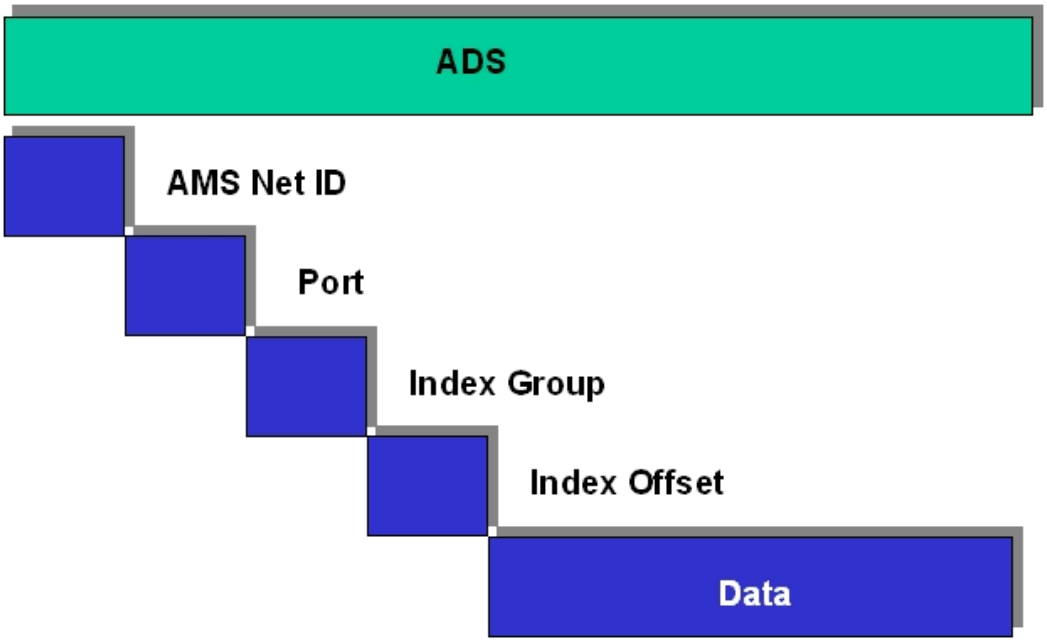


Abb. 160: Aufbau des ADS-Protokolls

AMSNetID

Die AMSNetID beschreibt das anzusprechende Gerät. Diese wird aus der eingestellten TCP/IP-Adresse und zusätzlichen 2 Byte erstellt. Diese zusätzlichen 2 Byte bestehen aus "1.1" und sind nicht veränderbar.
Beispiel:
IP-Adresse 172.16.17.128
AMSNetID 172.16.17.128.1.1

Port-Nummer

Die Portnummer unterscheidet im angeschlossenen Gerät Unterelemente.
Port 100: Registerzugriff
Port 300: Prozessdaten Feldbus
Port 800: lokale Prozessdaten (nur BC90x0, C900)

Index Group

Die Index Group unterscheidet innerhalb eines Ports verschiedene Daten.

Index Offset

Gibt den Offset an, ab welchem Byte gelesen oder geschrieben werden soll.

Len

Gibt die Länge der Daten in Byte an, die gelesen bzw. geschrieben werden sollen.

TCP-Port-Nummer

Die TCP-Port-Nummer beträgt für das ADS-Protokoll 48898 oder 0xBF02.

6.3.3 ADS-Dienste

Lokales Prozessabbild PLC Task 1 Port 800/801

Im lokalen Prozessabbild können Daten gelesen und geschrieben werden. Sollten Ausgänge geschrieben werden muss darauf geachtet werden, das diese von der lokalen SPS nicht verwendet werden, da die lokale Steuerung diese Werte überschreibt. Die Daten sind nicht an einen Watchdog gebunden und dürfen daher nicht für Ausgänge verwendet werden, die im Fehlerfall ausgeschaltet werden müssen.

Index Group	Bedeutung	Index Offset (Wertebereich)
0xF020	Input - Eingänge	0...2047
0xF021	Input - Eingänge Bit	0...16376
0xF030	Output - Ausgänge	0...2047
0xF031	Output - Ausgänge Bit	0...16376
0x4020	Merker	0...4095
0x4021	Merker Bit	0...32760

Dienste des ADS

AdsServerAdsState

Datentyp (only Read)	Bedeutung
String	Start - die lokale PLC läuft Stop - die lokale PLC ist im Stop

AdsServerDeviceState

Datentyp (only Read)	Bedeutung
INT	0 - Start - die lokale PLC läuft 1 - Stop - die lokale PLC ist im Stop

AdsServerType

Datentyp (only Read)	Bedeutung
String	BX PLC Server

ADSWriteControl

Datentyp (write only)	Bedeutung
NetID	Net ID des Ethernet Controllers*
Port	800
ADSSTATE	5 - RUN / 6 - STOP
DEVSTATE	0
LEN	0
SRCADDR	0
WRITE	positive Flanke startet den Baustein
TMOUT	zum Beispiel: t#1000ms

* BC9050, BC9020, BC9120, BX9000

Register Zugriff Port 100

Die ADS-Portnummer ist bei den Busklemmen-Controllern der BX-Serie und den BCxx50/xx20 für die Register-Kommunikation fest vorgegeben und beträgt 100.

Index Group	Index Offset (Wertebereich)		Bedeutung
	Hi-Word	Lo-Word	
0 [READ ONLY]	0...127	0...255	Register des Buskopplers Hi-Word Tabellennummer des Buskopplers Lo-Word Registernummer der Tabelle
1...255	0...3	1...255	Register der Busklemmen Hi-Word Kanalnummer Lo-Word Registernummer der Busklemme

Registerkommunikation mit K-Bus Busklemmen

Beachten Sie beim Lesen der Register, dass der Time-Out beim ADS-Baustein auf eine Zeit größer eine Sekunde eingestellt wird.

Schreiben von Registern in die K-Bus Busklemmen

Beachten Sie beim Schreiben auf die Register, dass das Passwort gesetzt wird (siehe Dokumentation zur entsprechenden Busklemme).

7 Fehlerbehandlung und Diagose

7.1 Diagnose

Zustand des Ethernet (FieldbusState)

In vielen Fällen ist es wichtig zu wissen, ob die Kommunikation mit dem übergeordneten Master noch funktioniert. Verknüpfen Sie hierfür die Variable *FieldbusState* mit Ihrem SPS-Programm.

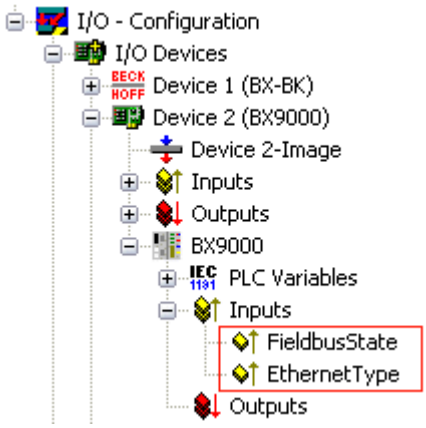


Abb. 161: Ethernet-Feldbus-Status

FieldbusState

Fehlernummer	Beschreibung	Ursache
0	kein Fehler	-
1	Watchdog Fehler	Kommunikation unterbrochen

EthernetType

Hier kann man erkennen, welches Ethernet-Protokoll auf die SPS-Variablen zugreift und damit den Watchdog aktiviert (zum Beispiel die Daten in der Default Config ab der Adresse %IB1000 und %QB1000).

Diagnosenummer	Beschreibung	Protokoll
0x0000	kein Protokoll greift auf die SPS Variablen zu	-
0x0001	ADS TCP	Kommunikation über ADS TCP/IP
0x0002	ADS UDP	Kommunikation über ADS UDP/IP
0x0010	ModbusTCP	Kommunikation über Modbus TCP/IP
0x0011	ModbusUDP	Kommunikation über Modbus UDP

Auslesen des Feldbusstatus per ADS

In der Default-Konfiguration oder in TwinCAT-Konfiguration kann der Feldbusstatus über ADSREAD ausgelesen werden.

Parameter ADSREAD Baustein	Beschreibung
NetID	lokal - Leerstring
Port	1
IndexGroup	6
IndexOffset	0x000C_A200
LEN	4

1. Wort (FieldbusState)

Fehlernummer	Beschreibung	Ursache
0x0000	No error	-
0x0001	Watchdog error	Kommunikation unterbrochen

2. Wort (EthernetType)

Diagnosenummer	Beschreibung	Protokoll
0x0001	ADS TCP	Kommunikation über ADS TCP/IP
0x0002	ADS UDP	Kommunikation über ADS UDP/IP
0x0010	ModbusTCP	Kommunikation über Modbus TCP/IP
0x0011	ModbusUDP	Kommunikation über Modbus UDP

Zustand des K-Bus

Sollte der interne Bus oder einer seiner Busklemmen ein Problem haben, wird dies im K-Bus-Status angezeigt. Eine genauere Fehlerursache kann mit einem Funktionsbaustein ausgelesen werden (in Vorbereitung). Verknüpfen Sie hierfür die Variable *K-Bus-State* mit Ihrem SPS-Programm.

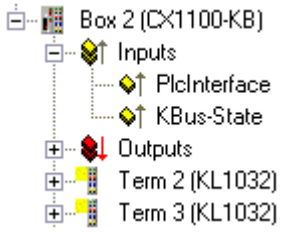


Abb. 162: K-Bus-Status

Fehlerbit	Beschreibung	Fehlerart
0	kein Fehler	Kein FEHLER
Bit 0	K-Bus Fehler	FEHLER
Bit 2	K-Bus wird Nachgetriggert	HINWEIS

Auslesen des K-Bus-Status per ADS

In der Default-Konfiguration oder in der TwinCAT-Konfiguration kann der K-Bus-Status über ADSREAD ausgelesen werden.

Parameter ADSREAD Baustein	Beschreibung
NetID	lokal - Leerstring
Port	1
IndexGroup	16#0006
IndexOffset	16#000C_9000
LEN	1

7.2 Diagnose-LEDs

Zur Statusanzeige besitzt der Buskoppler zwei Gruppen von LEDs. Die linken Gruppe "DIAG LEDs" geben einen Zustand des Feldbusses, der PLC und des K-Busses an. Auf der rechten oberen Seite des Buskopplers befinden sich zwei grüne LEDs (Power LEDs). Diese Dienen der Anzeige der Versorgungsspannung des BX-Contollers und der 24 V_{DC} der Powerkontakte.

LEDs zur Diagnose der Spannungsversorgung

LED (Power LEDs)	Bedeutung
Linke LED aus	Buskoppler hat keine Spannung 24 V_{DC}
Rechte LED aus	Keine Spannungsversorgung 24 V_{DC} an den Powerkontakten angeschlossen

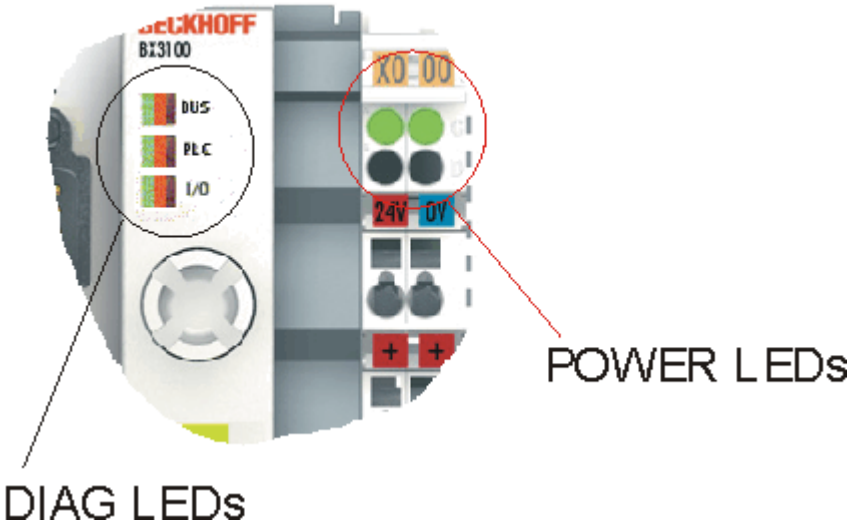


Abb. 163: Diagnose LEDs für den Feldbus, die SPS, den K-Bus und die Spannungsversorgungen

Die DIAG LEDs sind unterteilt in:

- Bus: Feldbus-Diagnose
- PLC: SPS-Diagnose
- I/O: K-Bus-Diagnose

Die LEDs können die Zustände aus, grün, orange und rot annehmen.

Abb. 164: Diagnose LEDs für den Feldbus, die SPS und den K-Bus

Nach dem Einschalten überprüft der Buskoppler sofort die angeschlossene Konfiguration. Der fehlerfreie Hochlauf wird durch das Verlöschen der LED *I/O* signalisiert. Das rote Leuchten der LED *I/O* zeigt einen Fehler im Bereich der Busklemmen an. Die Fehlerart wird im Display angezeigt. Das ermöglicht eine schnelle Fehlerbeseitigung.

LED BUS - Feldbus-Diagnose

LED	Bedeutung
LED aus	kein Feldbusangeschlossen, Buskoppler sucht die Baudrate
LED rot	Fehler Blinken - Fehlerart - Anzeige Display
LED orange	Buskoppler hat Baudrate gefunden, wartet auf Config- und Parameterdaten
LED grün	Buskommunikation ist in Ordnung, BX-Controller ist im Datenaustausch

LED PLC - SPS-Diagnose

LED	Bedeutung
LED aus	PLC Stop oder kein Programm vorhanden
LED rot	blinken - die eingestellte Taskzeit wird ab und zu überschritten an - die eingestellte Taskzeit wird immer überschritten Disable der roten LED
LED orange	PLC ohne Bootprojekt läuft (nur während des Zyklus an), beim Erzeugen des Bootprojektes blinkt die LED orange
LED grün	Bootprojekt - PLC läuft (nur während des Zyklus an)

LED I/O - K-Bus-Diagnose

LED	Bedeutung
LED aus	über den K-Bus werden keine Daten ausgetauscht
LED rot	Fehler Blinken - Fehlerart - Anzeige Display
LED orange	Register- bzw. KS2000-Online-Zugriff
LED grün	K-Bus ist in Ordnung und läuft

Fehler-Codes zur K-Bus-Diagnose

Fehler-Code	Fehler-Argument	Beschreibung	Abhilfe
0	-	EMV-Probleme	• Spannungsversorgung auf Unter- oder Überspannungsspitzen kontrollieren • EMV-Maßnahmen ergreifen • Liegt ein K-Bus-Fehler vor, kann durch erneutes Starten (Aus- und Wiedereinschalten des BX-Controllers) der Fehler lokalisiert werden
1	0	EEPROM-Prüfsummenfehler	Herstellereinstellung mit der Konfigurationssoftware KS2000 setzen
	1	Überlauf im Code Buffer	Weniger Busklemmen stecken. Bei prog. Konfiguration sind zu viele Einträge in der Tabelle
	2	Unbekannter Datentyp	Software-Update des BX-Controllers notwendig
2	-	Reserve	-
3	0	K-Bus-Kommandofehler	• Keine Busklemme gesteckt • Eine der Busklemmen ist defekt, angehängte Busklemmen halbieren und prüfen ob der Fehler bei den übrigen Busklemmen noch vorhanden ist. Dies weiter durchführen, bis die defekte Busklemme lokalisiert ist.
4	0	K-Bus-Datenfehler, Bruchstelle hinter dem BX-Controllers	Prüfen ob die n+1 Busklemme richtig gesteckt ist, gegebenenfalls tauschen
	n	Bruchstelle hinter Busklemme n	Kontrollieren ob die Busendklemme KL9010 gesteckt ist
5	n	K-Bus-Fehler bei Register-Kommunikation mit Busklemme n	n-te Busklemme tauschen
6	0	Fehler bei der Initialisierung	BX-Controllers austauschen
	1	Interner Datenfehler	Hardware-Reset des BX-Controllers (Aus - und Wiedereinschalten)
	2	DIP-Schalter nach einem Software-Reset verändert	Hardware-Reset des BX-Controllers (Aus - und Wiedereinschalten)
	3	IP Adressen Kollision	Prüfen Sie ob die IP-Adresse schon im Netzwerk vorhanden ist.
7	0	Hinweis: Zykluszeit wurde überschritten	Warning: Die eingestellte Zykluszeit wurde überschritten. Dieser Hinweis (blinken der LEDs) kann nur durch erneutes booten des BX-Controllers gelöscht werden. Abhilfe: Zykluszeit erhöhen
9	0	Checksummenfehler im Programm-Flash	Programm erneut zum BX-Controllers übertragen
	1	Falsche oder fehlerhaft Bibliothek implementiert	Entfernen Sie die fehlerhaft Bibliothek
10	n	Die Busklemme n stimmt nicht mit der Konfiguration, die beim Erstellen des Boot-Projektes existiert überein	Die n-te Busklemme überprüfen. Sollte eine n-te Busklemme gewollt eingefügt worden sein, muss das Bootprojekt gelöscht werden.
14	n	n-te Busklemme hat das falsche Format	BX-Controller erneut starten, falls der Fehler erneut auftritt, die Busklemme tauschen.
15	n	Anzahl der Busklemmen stimmt nicht mehr	BX-Controller erneut starten. Falls der Fehler erneut auftritt, BX-Controller mit der Konfigurationssoftware auf Auslieferungszustand zurücksetzen.
16	n	Länge der K-Bus-Daten stimmt nicht mehr	

7.3 Diagnose-Display

Während des Hochstarten zeigt das Display für ca. drei Sekunden die aktuelle Firmware-Version an.

Sollte beim Starten ein Fehler vorliegen wird ihnen dieser per Blink-Sequenz der entsprechenden LED angezeigt.

Konfigurationsfehler werden im Display mit TC-Config und einer Fehlernummer angegeben. Bitte überprüfen Sie in diesem Fall Ihre Hardwarekonfiguration im System Manager oder setzen Sie sich mit dem Support in Verbindung.

Anzeige Display	Bedeutung
TC-Config 0xE02E	Eine komplexe Busklemme liegt auf einer Bitadresse. Überprüfen Sie die TwinCAT Konfiguration.
TC-Config 0xF0nn	Die nn' te Busklemme stimmt nicht mit der Konfiguration überein. Vergleichen Sie für die nn'te Busklemme den Busaufbau mit der Konfiguration.
TC-Config 0xC0nn	Die nn' te Busklemme stimmt nicht mit der Konfiguration überein. Vergleichen Sie für die nn'te Busklemme den Busaufbau mit der Konfiguration.

Firmware Fehler werden im Display mit FW-Error und einer Fehlernummer angegeben. Bitte setzen Sie sich mit dem Support in Verbindung.

Anzeige Display	Bedeutung
FW-Error 0xnnnn	Wenden Sie sich an den Support

8 Anhang

8.1 BX9000 - Erste Schritte

Ändern Sie die IP-Adresse

Default IP-Adresse: 172.16.21.20
SubNetMask: 255.255.0.0
Default gateway: 0.0.0.0

Abb. 165: Navigationsschalter

Um in das Menü zu wechseln drücken Sie den Navigationsschalter drei Sekunden lang. Zuerst erscheint das Menü-Directory.

- Sie können mit den RIGHT/LEFT-Tasten zwischen den Menüpunkten wählen (Reihe 1 des Displays zeigt die aktive Menüebene).
- Drücken Sie die DOWN-Taste um in ein Untermenü zu wechseln.
- Drücken Sie die UP-Taste um zum Hauptmenü zurück zu gelangen.

Reihe 1 des Untermenüs zeigt den Menüpunkt, Reihe 2 die aktuelle Einstellung des Menüpunkts.

Einige Einträge können nicht verändert werden *(read only)*. Diese Einträge ermöglichen Überprüfungen oder geben dem Anwender Informationen. Um die Menüanzeige zu beenden müssen Sie im Hauptmenü sein und den Navigationsschalter drei Sekunden lang drücken.

Bevor Änderungen gespeichert werden können muss ein Passwort festgelegt werden. Das Passwort bleibt auch bei einem Firmware-Update oder dem Wiederherstellen des Auslieferungszustands erhalten. Falls Sie das Passwort vergessen sollten, müssen Sie den BX-Controller einschicken.

Schalterbelegung

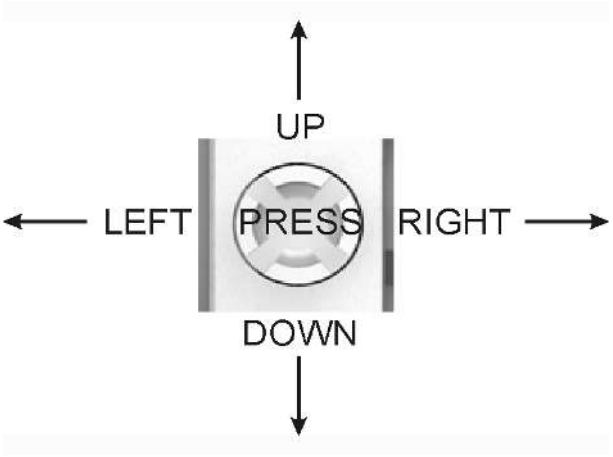


Abb. 166: Schalterbelegung

Code

Die Werkseinstellung ist "****", z. B. kein Passwort aktiviert. Um Parameter einstellen zu können ist ein Passwort erforderlich.

Menü-Navigation

Sie öffnen das Menü-Directory indem Sie den Navigationsschalter drei Sekunden lang drücken. Einige Menüpunkte sind im Folgenden erläutert.

Tab. 2: MENU

MENU	Hauptmenü	
	DOWN (kurz drücken)	
	PASSWORD **????**	???? - Passwort gesetzt **** - kein Passwort gesetzt
	PRESS (kurz drücken)	
	PASSWORD ENTER? **????**	Möchten Sie das Passwort setzen? Yes - (Drücken Sie für ca. zwei Sekunden, dann geben Sie das Passwort ein) / No - drücken Sie UP
	PRESS (Drücken Sie für ca. zwei Sekunden, dann geben Sie das Passwort ein)	
	PASSWORD **???** *?*	Passwort eingeben Drücken Sie <OK>
	PRESS	
	PASSWORD **1234**	Passwort eingeben Drücken Sie OK und drücken noch mal OK zum Bestätigen (Drücken Sie für ca. eine Sekunde bis OK auf dem Display erscheint)
	PRESS UP	
MENU	Haupt-Menü	
Drücken Sie RIGHT		
AMS	AMS-Menü	
Drücken Sie RIGHT		
ETHERNET	ETHERNET-Menü	
Drücken Sie DOWN		
	MAC ID 000105-xx-xx-xx	ETHERNET Menu / MAC ID xx-xx-xx is your MAC ID
	Drücken Sie LEFT	
	IP ADDRESS 172.16.21.20	ETHERNET Menu / IP ADDRESS Default IP Address
	Drücken Sie DOWN um die IP-Adresse einzugeben	
	MODIFY *1*72.16.21.20	ETHERNET Menu / IP ADDRESS / Edit the IP Address Drücken Sie DOWN um eins runter zu zählen (-1), Drücken Sie UP um eins hoch zu zählen (+1), Drücken Sie RIGHT für die nächste Ziffer
	Drücken Sie den Taster zwei Sekunden Sie ACCEPTED? sehen	
	ACCEPT ? 172.16.44.44	ETHERNET Menu / IP ADDRESS / Edit the IP Address Drücken Sie DOWN um eins runter zu zählen (-1), Drücken Sie UP um eins hoch zu zählen (+1), Drücken Sie RIGHT für die nächste Ziffer
	Drücken Sie den Taster wenn Sie ACCEPTED sehen	
	ACCEPTED 172.16.44.44	ETHERNET Menu / IP ADDRESS / Edit the IP Address Drücken Sie DOWN um eins runter zu zählen (-1), Drücken Sie UP um eins hoch zu zählen (+1), Drücken Sie RIGHT für die nächste Ziffer
	Drücken Sie UP um IP-Adresse zu verlassen	
	ETHERNET	ETHERNET-Menü

Drücken Sie drei Sekunden um das Menü zu verlassen

Nächster Schritt

Installieren Sie TwinCAT 2.10 (finden Sie auf der BX-CD)

Fügen Sie den BX-Controller im AMS-Router hinzu

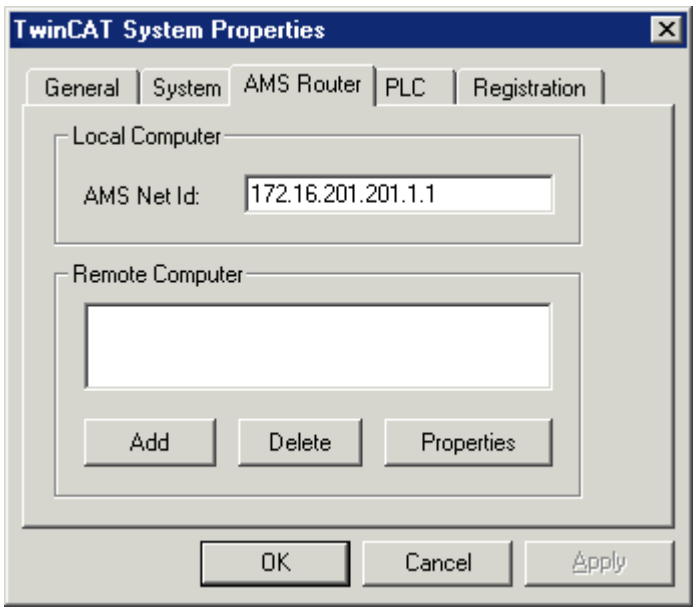


Abb. 167: Hinzufügen des BX-Controller im AMS-Router

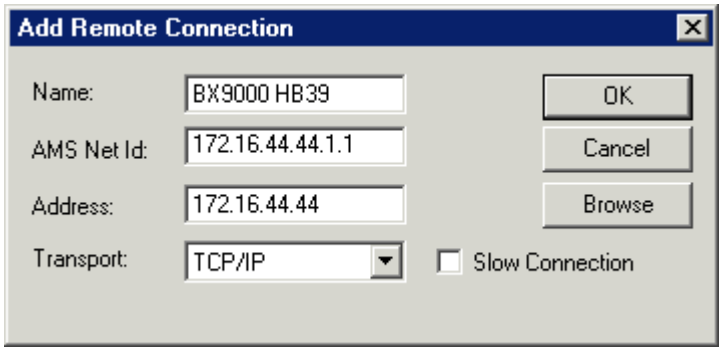


Abb. 168: Hinzufügen der Remote Connection

Starten Sie TwinCAT neu - nun können Sie den BX9000 programmieren.

Starten Sie das Programm PLC-Control und erzeugen Sie ein neues Projekt.

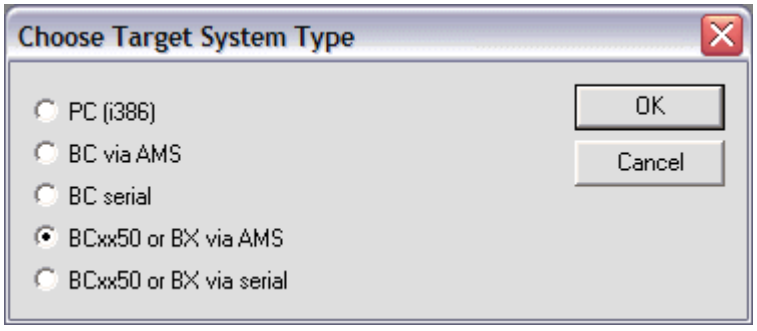


Abb. 169: Auswahl des Zielsystems

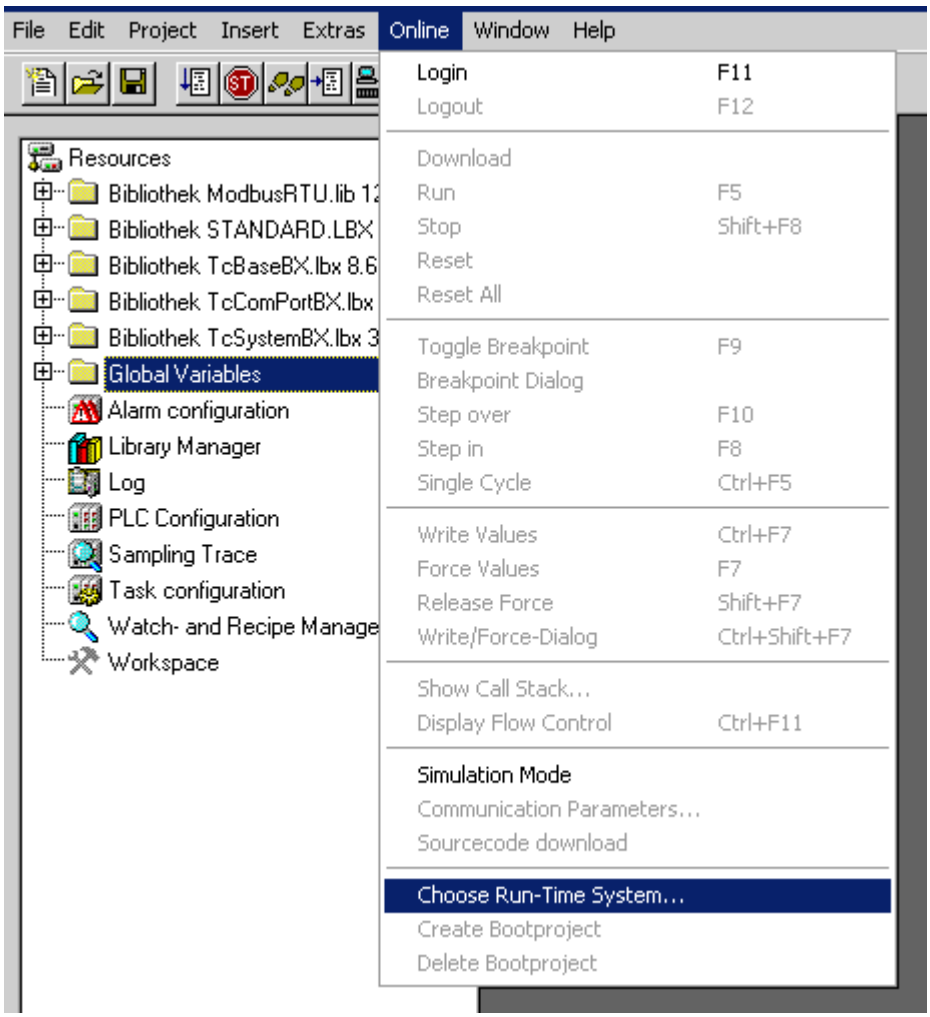


Abb. 170: Auswahl des Run-Time-Systems (Schritt 1)

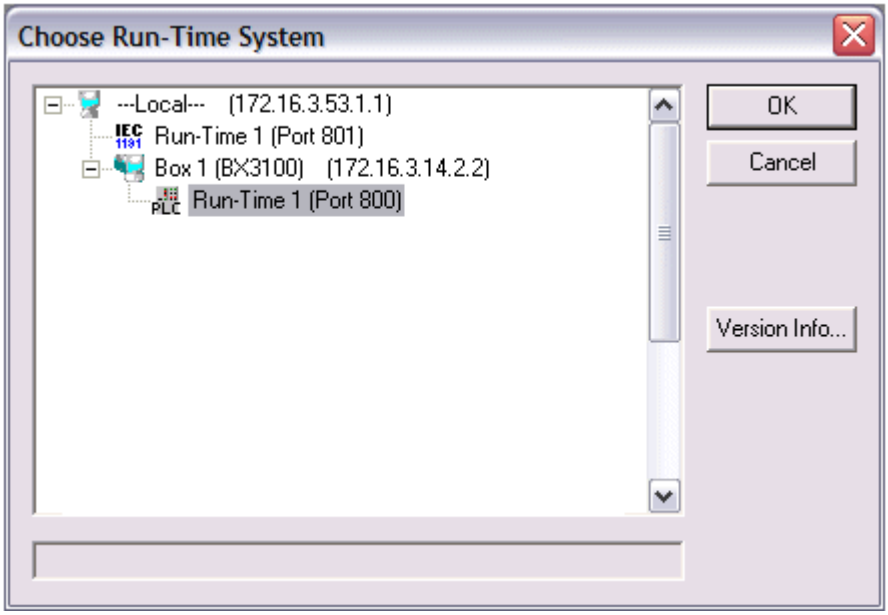


Abb. 171: Auswahl des Run-Time-Systems (Schritt 2)

8.2 Umstieg zwischen den Controllern

Umstieg vom BCxx00 zum BCxx50/BCxx20

Dateinamen

Bei Busklemmen-Controllern der Serie BCxx50 und BCxx20 heißen Bibliotheken *.lbx und Programme werden als *.prx gespeichert.

Merker Variablen

Die lokierten Merker Variablen

- der BCxx00 belegen %MB0...%MB511 (außer BC9000/BC9100: %MB0...%B4095).
- der BCxx20 belegen %MB0...%MB4095
- der BCxx50 belegen %MB0...%MB4095

Zustandsinformationen wie K-Bus-/Feldbus-Status, Zyklustick werden nicht in den BCxx50/BCxx20 kopiert. Diese Informationen stehen in der TcSystemBCxx50.lbx für den BCxx50/BCxx20 als Funktion zur Verfügung.

Die lokierten Merker arbeiten **nicht** als Retain-Variablen.

Retain-Daten

Die Retain-Daten müssen als VAR RETAIN [▶ 96] deklariert werden. Es stehen bis zu 2 kByte zur Verfügung.

SPS-Variablen

In der Default-Config fangen die SPS-Variablen ab %IB1000 und %QB1000 an.

Large Model

Gibt es nicht bei BCxx50 und BCxx20.

Max. Speicher:

- BCxx50: 48 kByte
- BCxx20: 128 kByte

Task Time

Die Task-Zeit wird im PLC-Control festgelegt, diese sollte auf eine realistisch Größe eingestellt sein (messen der PLC Zykluszeit und des K-Busses). Die Background-Zeit entfällt.

Task-Konfiguration

Es steht maximal eine Task zur Verfügung. Diese Task muss konfiguriert werden.

PLC und Feldbusklemmen

Bei den Standard Busklemmen Controllern (BCxx00) gab es die Möglichkeit zu wählen, ob eine Busklemme dem Feldbus oder der lokalen PLC zugewiesen wird.

In der Default-Konfiguration des BCxx50/BCxx20 sind alle Busklemmen der lokalen PLC zugewiesen. Eine Zuordnung zum Feldbus ist hier nicht möglich.

Umstieg vom BCxx00 zum BXxx00

Dateinamen

Bei Busklemmen-Controllern der Serie BXxx00 heißen Bibliotheken *.lbx und Programme werden als *.prx gespeichert.

Merker-Variablen

Die lokierten Merker-Variablen

- der BCxx00 belegen %MB0...%MB511 (außer BC9000/BC9100: %MB0...%B4095).
- der BXxx00 belegen %MB0...%MB4095

Zustandsinformationen wie K-Bus-/Feldbus-Status, Zyklustick werden nicht in den BXxx00 kopiert. Diese Informationen stehen in der TcSystemBX.lbx für den BXxx00 als Funktion zur Verfügung.

Die lokierten Merker arbeiten **nicht** als Retain-Variablen.

Retain-Daten

Die Retain-Daten müssen als VAR_RETAIN [▶ 96] deklariert werden. Es stehen bis zu 2 kByte zur Verfügung.

SPS-Variablen

In der Default-Config fangen die SPS-Variablen ab %IB1000 und %QB1000 an.

Large Model

Gibt es nicht bei BXxx00. Max. Speicher: 256 kByte.

Task Time

Die Task-Zeit wird im PLC-Control festgelegt, diese sollte auf eine realistisch Größe eingestellt sein (messen der PLC Zykluszeit und des K-Buses). Die Background-Zeit entfällt.

Task-Konfiguration

Es steht maximal eine Task zur Verfügung. Diese Task muss konfiguriert werden.

PLC und Feldbusklemmen

Bei den Standard Busklemmen Controller (BCxx00) gab es die Möglichkeit zu wählen, ob eine Busklemme dem Feldbus oder der lokalen PLC zugewiesen wird.

In der Default-Konfiguration des BXxx00 sind alle Busklemmen der lokalen PLC zugewiesen. Eine Zuordnung zum Feldbus ist hier nicht möglich.

Umstieg vom PC zum BCxx50/BCxx20/BXxx00

Dateinamen

Bei den Busklemmen-Controllern der Serien BCxx50/BCxx20 und BXxx00 heißen Bibliotheken *.lbx und Programme werden als *.prx gespeichert.

Lokierte Variablen

Es stehen bei den Busklemmen-Controllern der Serien BCxx50/BCxx20 und BXxx00 eine begrenzte Anzahl von lokierten Daten zur Verfügung:

- Inputs 2 kByte, %IB0…2048
- Outputs 2 kByte, %QB0…2048

- Merker 4 kByte, %MB0…4095

Task-Konfiguration

Maximal steht eine Task zur Verfügung. Es ist eine sinnvolle Task-Zeit zu wählen. Passen Sie die Task-Zeit Ihrer Anwendung an, messen Sie dazu die benötige Systemzeit (PLC + K-Bus + Feldbus + sonstiges).

Retain-Daten

Bei den Busklemmen-Controllern der Serien BCxx50, BCxx20 und BXxx00 stehen bis zu 2 kByte Retain-Daten zur Verfügung. Achten Sie daher darauf das Sie keine (oder nur sehr beschränkt) Retain-Daten in Funktionsblöcken verwenden (siehe RETAIN Daten [▶ 96]).

8.3 Firmware-Update BX9000

Firmware Update über Ethernet

Sie können mit Hilfe des TwinCAT System Manager über Ethernet eine neue Firmware auf den BX9000 laden. Anforderungen:

TwinCAT 2.10 Build 1251
BX9000 Firmware ab 1.07
Funktionierende Ethernet Verbindung (vom PC zum BX9000).

Bevor Sie den BX9000 updaten, löschen Sie das Bootprojekt und starten Sie den BX9000 ohne PLC Programm auf.

Schalten Sie TwinCAT in den Config Mode  und fügen Sie eine virtuelle Ethernet-Schnittstelle als Gerät an.

Fügen als nächstes den BX9000 an und stellen Sie die IP-Adresse ein.

Als nächstes klicken sie mit der rechten Maustaste auf das Gerät BX9000 und wählen Sie die Funktion *Firmware-Update via ADS* aus.

Wählen Sie als nächstes die entsprechende Firmware aus. Das Update erfolgt automatisch. Nach Beendigung des Updates startet der BX9000 automatisch neu.

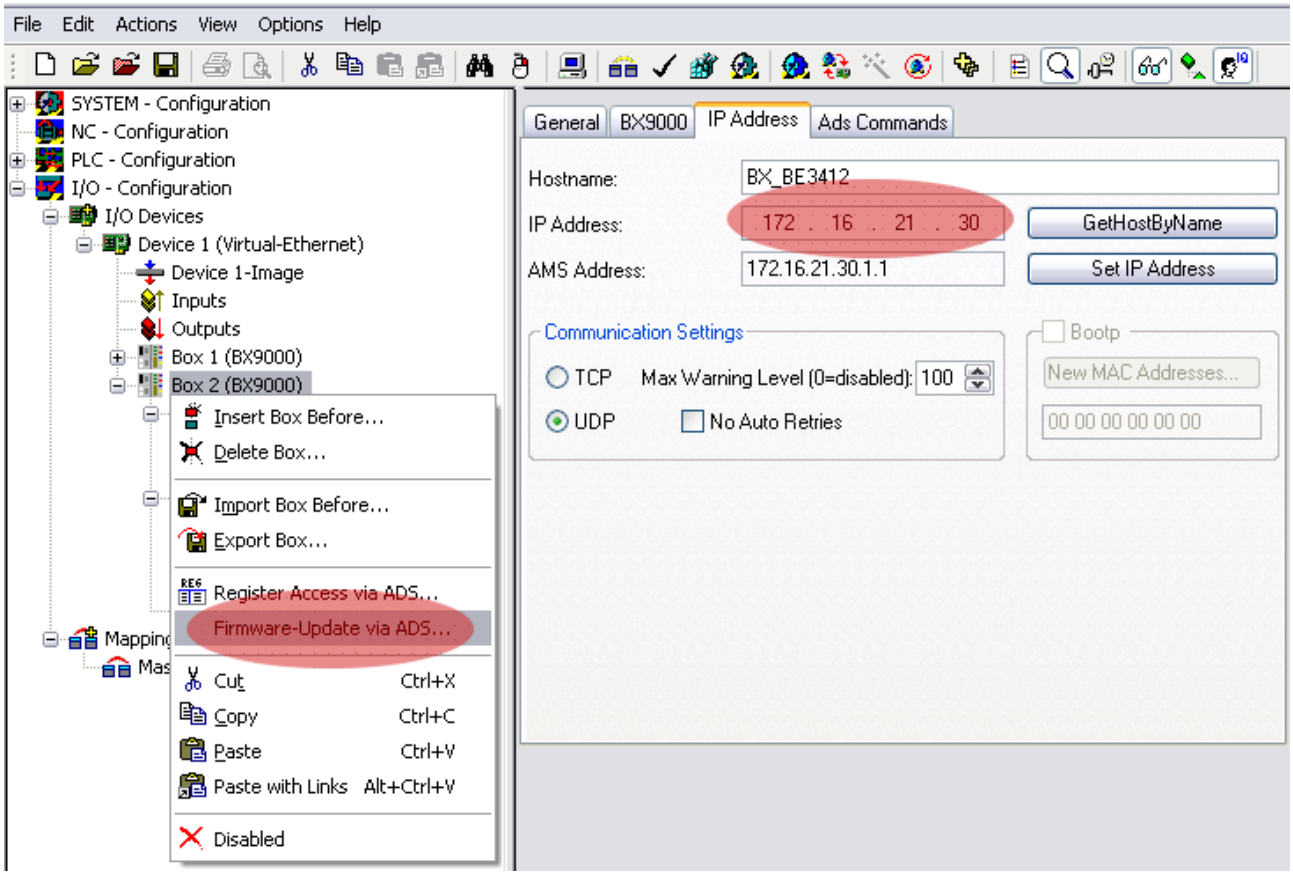


Abb. 172: Firmware Update über Ethernet

8.4 Beispielprogramme - Übersicht

Benennung	Beschreibung
Display [▶ 123]	Beispiel zum Ansteuern des Displays
Navigationsschalter [▶ 122]	Auslesen des Navigationsschalter aus der PLC heraus
Menü [▶ 92]	Beispiel für ein eignes Menü mit Navigations-Schalter und Display
RTC [▶ 84]	Beispiel zum Auslesen der Real Time Clock (RTC) über Funktionsbausteine
COM Port - BK/BC8x00 Master Interface [▶ 128]	COM1- oder COM2-Schittstelle als Master mit dem BK8x00 Protokoll
COM Port - BK8x00 Slave Interface [▶ 129]	COM1- oder COM2-Schittstelle als Slave mit dem BK8x00 Protokoll
COM Port - Cimrex 12 [▶ 135]	Beispiel zum Ansteuern eines Cimrex 12 Displays über ModbusRTU
COM Port - ModbusRTU Slave [▶ 132]	Verknüpfung der ModbusRTU Lib mit der COM 1 oder COM 2 Schnittstelle des BXes
COM Port - ModbusRTU Master [▶ 134]	Verknüpfung der ModbusRTU Lib mit der COM 1 oder COM 2 Schnittstelle des BXes
COM Port - RK512 Protokoll [▶ 137]	RK512-Protokoll über die COM 1 oder COM 2
COM Port - SMS über COM Port [▶ 138]	Anschluss eines Siemens S35 Mobiltelefons an COM-Schnittstelle zum Versenden von SMS-Nachrichten
COM Port - COMlibV2 [▶ 135]	String verschicken und empfangen mit der COMlibV2
SSB - Display [▶ 75]	Cimrex Panel am SSB
SSB - AX2000 [▶ 73]	AX2000 am SSB
SSB - BK51x0 [▶ 71]	BK5120 am SSB
SSB - BX / BX Kommunikation [▶ 71]	Kommunikation von BX zu BX (über SSB)
SSB - IclA Drive [▶ 76]	IclA Drive am SSB
SSB - Lenze Drive [▶ 81]	Lenze Frequenzumrichter am SSB

8.5 Beispielprogramme für BX9000 - Übersicht

Benennung	Beschreibung
E-Mail versenden [▶ 162]	Versenden einer SMTP zu einem E-Mail-Server
HTML-Seite bereitstellen [▶ 171]	Anzeigen von Werten in auf einer HTML-Seite
IP-Konfiguration auslesen [▶ 168]	Auslesen der Ethernet-Einstellungen
ModbusTCP-Client [▶ 158]	BX9000 als ModbusTCP-Client (Master)
UDP-Kommunikation [▶ 157]	Kommunikation zu einem Ethernetgerät über UDP
Zeit auslesen über SNTP [▶ 164]	Auslesen der Zeit über SNTP

8.6 Allgemeine Betriebsbedingungen

Um einen fehlerfreien Betrieb der Feldbuskomponenten zu erreichen, müssen die nachfolgenden Bedingungen eingehalten werden.

Bedingungen an die Umgebung

Betrieb

An folgenden Orten dürfen die Komponenten nicht ohne Zusatzmaßnahmen eingesetzt werden:

- unter erschwerten Betriebsbedingungen, wie z. B. ätzende Dämpfe oder Gase, Staubbildung
- bei hoher ionisierender Strahlung

Bedingung	zulässiger Bereich
zulässige Umgebungstemperatur im Betrieb	Siehe Technische Daten
zulässige Umgebungstemperatur im Betrieb	-25°C ... +85°C
Einbaulage	beliebig
Vibrationsfestigkeit	gemäß EN 60068-2-6
Schockfestigkeit	gemäß EN 60068-2-27, EN 60068-2-29
EMV-Festigkeit	gemäß EN 61000-6-2
Aussendung	gemäß EN 61000-6-4

Transport und Lagerung

Bedingung	zulässiger Bereich
zulässige Umgebungstemperatur bei Lagerung	-25°C... +85°C
Relative Feuchte	95 %, keine Betauung
Freier Fall	originalverpackt bis 1 m

Schutzklasse und Schutzart

Bedingung	zulässiger Bereich
Schutzklasse nach IEC 536 (VDE 0106, Teil 1)	An der Profilschiene ist ein Schutzleiteranschluss erforderlich!
Schutzart nach IEC 529	IP20 (Schutz gegen Berührung mit Standard Prüffinger)
Schutz gegen Fremdkörper	kleiner 12 mm im Durchmesser
Schutz gegen Wasser	kein Schutz

Kennzeichnung der Komponenten

Jede ausgelieferte Komponente enthält einen Aufkleber, mit Informationen über die Zulassung des Produkts.

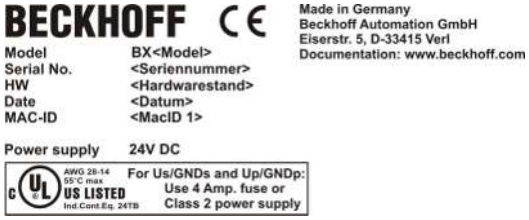


Abb. 173: Typenschild eines BX-Controllers

Auf dem Aufkleber sind folgende Informationen abzulesen:

Aufdruck	Bedeutung für diesen Aufkleber
genaue Produktbezeichnung	Model BX
Versorgungsspannung Us	24 V_{DC} (Benutzen Sie eine 4 A Sicherung oder eine der Class 2 entsprechende Spannungsversorgung um die UL-Anforderungen zu erfüllen!)
Hersteller	Beckhoff Automation GmbH
CE-Zeichen	Konformitätskennzeichnung
UL-Zeichen 	Kennzeichen für UL-Zulassung. UL steht für Underwriters Laboratories Inc., die führende Zertifizierungsorganisation für Nordamerika mit Sitz in den USA. C = Kanada, US = USA, UL File Nummer: E172151
Produktionsbezeichnung	Serial No.: Seriennummer HW: Hardware Stand Date: Herstelldatum optional nur beim BX9000 MAC-ID

8.7 Prüfnormen für Geräteprüfung

EMV

EMV-Festigkeit

EN 61000-6-2

EMV-Aussendung

EN 61000-6-4

Vibrations-/ Schockfestigkeit

Vibrationsfestigkeit

EN 60068-2-6

Schockfestigkeit

EN 60068-2-27

8.8 Literaturverzeichnis

TCP/IP

TCP/IP (deutsch)

- Aufbau und Betrieb eines TCP/IP Netzes
- von Kevin Washburn, Jim Evans
- Verlag: ADDISON-WESLEY Longmann Verlag

TCP/IP (englisch)

- Illustrated, Volume1 The Protocols

- von W. Richard Stevens
- Verlag: ADDISON-WESLEY Longmann Verlag

Modbus/TCP

http://www.modicon.com/
http://www.modbus.org

TwinCAT

Beckhoff Information System
http://infosys.beckhoff.com

8.9 Abkürzungsverzeichnis

ADS

Automation Device Specification

IP (20)

Schutzart der Busklemmen

IPC

Industrie-PC

E/A

Ein- und Ausgänge

K-Bus

Klemmen-Bus

KS2000

Konfigurationssoftware für Busklemmen, Bus Koppler, Busklemmen Controller, Feldbus Box Module usw.

PE

Der PE-Powerkontakt kann als Schutzerde verwendet werden.

TwinCAT

The Windows Control and Automation Technology

8.10 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

Abbildungsverzeichnis

Mehr Informationen:
www.beckhoff.de/BX9000

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