Modicon M340

Serielle Verbindung
Benutzerhandbuch

(Übersetzung des englischen Originaldokuments)

12/2018

35012431.14

www.schneider-electric.com

Schneider Electric

Die Informationen in der vorliegenden Dokumentation enthalten allgemeine Beschreibungen und/oder technische Leistungsmerkmale der hier erwähnten Produkte. Diese Dokumentation dient keinesfalls als Ersatz für die Ermittlung der Eignung oder Verlässlichkeit dieser Produkte für bestimmte Verwendungsbereiche des Benutzers und darf nicht zu diesem Zweck verwendet werden. Jeder Benutzer oder Integrator ist verpflichtet, angemessene und vollständige Risikoanalysen, Bewertungen und Tests der Produkte im Hinblick auf deren jeweils spezifischen Verwendungszweck vorzunehmen. Weder Schneider Electric noch deren Tochtergesellschaften oder verbundene Unternehmen sind für einen Missbrauch der Informationen in der vorliegenden Dokumentation verantwortlich oder können diesbezüglich haftbar gemacht werden. Verbesserungs- und Änderungsvorschlage sowie Hinweise auf angetroffene Fehler werden jederzeit gern entgegengenommen.

Sie erklären, dass Sie ohne schriftliche Genehmigung von Schneider Electric dieses Dokument weder ganz noch teilweise auf beliebigen Medien reproduzieren werden, ausgenommen zur Verwendung für persönliche nichtkommerzielle Zwecke. Darüber hinaus erklären Sie, dass Sie keine Hypertext-Links zu diesem Dokument oder seinem Inhalt einrichten werden. Schneider Electric gewährt keine Berechtigung oder Lizenz für die persönliche und nichtkommerzielle Verwendung dieses Dokument oder seines Inhalts, ausgenommen die nichtexklusive Lizenz zur Nutzung als Referenz. Das Handbuch wird hierfür „wie besehen“ bereitgestellt, die Nutzung erfolgt auf eigene Gefahr. Alle weiteren Rechte sind vorbehalten.

Bei der Montage und Verwendung dieses Produkts sind alle zutreffenden staatlichen, landesspezifischen, regionalen und lokalen Sicherheitsbestimmungen zu beachten. Aus Sicherheitsgründen und um die Übereinstimmung mit dokumentierten Systemdaten besser zu gewährleisten, sollten Reparaturen an Komponenten nur vom Hersteller vorgenommen werden.

Beim Einsatz von Geräten für Anwendungen mit technischen Sicherheitsanforderungen sind die relevanten Anweisungen zu beachten.

Die Verwendung anderer Software als der Schneider Electric-eigenen bzw. einer von Schneider Electric genehmigten Software in Verbindung mit den Hardwareprodukten von Schneider Electric kann Körperverletzung, Schäden oder einen fehlerhaften Betrieb zur Folge haben.

Die Nichtbeachtung dieser Informationen kann Verletzungen oder Materialschäden zur Folge haben!

Inhaltsverzeichnis

Sicherheitshinweise

Wichtige Informationen

HINWEISE

Lesen Sie sich diese Anweisungen sorgfältig durch und machen Sie sich vor Installation, Betrieb, Bedienung und Wartung mit dem Gerät vertraut. Die nachstehend aufgeführten Warnhinweise sind in der gesamten Dokumentation sowie auf dem Gerät selbst zu finden und weisen auf potenzielle Risiken und Gefahren oder bestimmte Informationen hin, die eine Vorgehensweise verdeutlichen oder vereinfachen.

Wird dieses Symbol zusätzlich zu einem Sicherheitshinweis des Typs „Gefahr“ oder „Warnung“ angezeigt, bedeutet das, dass die Gefahr eines elektrischen Schlags besteht und die Nichtbeachtung der Anweisungen unweigerlich Verletzung zur Folge hat.

Dies ist ein allgemeines Warnsymbol. Es macht Sie auf mögliche Verletzungsgefahren aufmerksam. Beachten Sie alle unter diesem Symbol aufgeführten Hinweise, um Verletzungen oder Unfälle mit Todesfälle zu vermeiden.

⚠ GEFAHR

GEFAHR macht auf eine gefährliche Situation aufmerksam, die, wenn sie nicht vermieden wird, Tod oder schwere Verletzungen **zur Folge hat.**

⚠ WARNUNG

WARNUNG macht auf eine gefährliche Situation aufmerksam, die, wenn sie nicht vermieden wird, Tod oder schwere Verletzungen **zur Folge haben kann.**

⚠ VORSICHT

VORSICHT macht auf eine gefährliche Situation aufmerksam, die, wenn sie nicht vermieden wird, leichte Verletzungen **zur Folge haben kann.**

HINWEIS

HINWEIS gibt Auskunft über Vorgehensweisen, bei denen keine Verletzungen drohen.

BITTE BEACHTEN

Elektrische Geräte dürfen nur von Fachpersonal installiert, betrieben, bedient und gewartet werden. Schneider Electric haftet nicht für Schäden, die durch die Verwendung dieses Materials entstehen.

Als qualifiziertes Fachpersonal gelten Mitarbeiter, die über Fähigkeiten und Kenntnisse hinsichtlich der Konstruktion und des Betriebs elektrischer Geräte und deren Installation verfügen und eine Schulung zur Erkennung und Vermeidung möglicher Gefahren absolviert haben.

BEVOR SIE BEGINNEN

Dieses Produkt nicht mit Maschinen ohne effektive Sicherheitseinrichtungen im Arbeitsraum verwenden. Das Fehlen effektiver Sicherheitseinrichtungen im Arbeitsraum einer Maschine kann schwere Verletzungen des Bedienpersonals zur Folge haben.

⚠ WARNUNG

UNBEAUFSICHTIGTE GERÄTE

- Diese Software und zugehörige Automatisierungsgeräte nicht an Maschinen verwenden, die nicht über Sicherheitseinrichtungen im Arbeitsraum verfügen.
- Greifen Sie bei laufendem Betrieb nicht in das Gerät.

Die Nichtbeachtung dieser Anweisungen kann Tod, schwere Verletzungen oder Sachschäden zur Folge haben.

Dieses Automatisierungsgerät und die zugehörige Software dienen zur Steuerung verschiedener industrieller Prozesse. Der Typ bzw. das Modell des für die jeweilige Anwendung geeigneten Automatisierungsgeräts ist von mehreren Faktoren abhängig, z. B. von der benötigten Steuerungsfunktion, der erforderlichen Schutzklasse, den Produktionsverfahren, außergewöhnlichen Bedingungen, behördlichen Vorschriften usw. Für einige Anwendungen werden möglicherweise mehrere Prozessoren benötigt, z. B. für ein Backup-/Redundanzsystem.

Nur Sie als Benutzer, Maschinenbauer oder -integrator sind mit allen Bedingungen und Faktoren vertraut, die bei der Installation, der Einrichtung, dem Betrieb und der Wartung der Maschine bzw. des Prozesses zum Tragen kommen. Demzufolge sind allein Sie in der Lage, die Automatisierungskomponenten und zugehörigen Sicherheitsvorkehrungen und Verriegelungen zu identifizieren, die einen ordnungsgemäßen Betrieb gewährleisten. Bei der Auswahl der Automatisierungs- und Steuerungsgeräte sowie der zugehörigen Software für eine bestimmte Anwendung sind die einschlägigen örtlichen und landesspezifischen Richtlinien und Vorschriften zu beachten. Das National Safety Council's Accident Prevention Manual (Handbuch zur Unfallverhütung; in den USA landesweit anerkannt) enthält ebenfalls zahlreiche nützliche Hinweise.

Für einige Anwendungen, z. B. Verpackungsmaschinen, sind zusätzliche Vorrichtungen zum Schutz des Bedienpersonals wie beispielsweise Sicherheitseinrichtungen im Arbeitsraum erforderlich. Diese Vorrichtungen werden benötigt, wenn das Bedienpersonal mit den Händen oder anderen Körperteilen in den Quetschbereich oder andere Gefahrenbereiche gelangen kann und somit einer potenziellen schweren Verletzungsgefahr ausgesetzt ist. Software-Produkte allein können das Bedienpersonal nicht vor Verletzungen schützen. Die Software kann daher nicht als Ersatz für Sicherheitseinrichtungen im Arbeitsraum verwendet werden.

Vor Inbetriebnahme der Anlage sicherstellen, dass alle zum Schutz des Arbeitsraums vorgesehenen mechanischen/elektronischen Sicherheitseinrichtungen und Verriegelungen installiert und funktionsfähig sind. Alle zum Schutz des Arbeitsraums vorgesehenen Sicherheitseinrichtungen und Verriegelungen müssen mit dem zugehörigen Automatisierungsgerät und der Softwareprogrammierung koordiniert werden.

HINWEIS: Die Koordinierung der zum Schutz des Arbeitsraums vorgesehenen mechanischen/elektronischen Sicherheitseinrichtungen und Verriegelungen geht über den Umfang der Funktionsbaustein-Bibliothek, des System-Benutzerhandbuchs oder andere in dieser Dokumentation genannten Implementierungen hinaus.

START UND TEST

Vor der Verwendung elektrischer Steuerungs- und Automatisierungsgeräte ist das System zur Überprüfung der einwandfreien Funktionsbereitschaft einem Anlauftest zu unterziehen. Dieser Test muss von qualifiziertem Personal durchgeführt werden. Um einen vollständigen und erfolgreichen Test zu gewährleisten, müssen die entsprechenden Vorkehrungen getroffen und genügend Zeit eingeplant werden.

⚠ WARNUNG

GEFAHR BEIM GERÄTEBETRIEB

- Überprüfen Sie, ob alle Installations- und Einrichtungsverfahren vollständig durchgeführt wurden.
- Vor der Durchführung von Funktionstests sämtliche Blöcke oder andere vorübergehende Transportsicherungen von den Anlagekomponenten entfernen.
- Entfernen Sie Werkzeuge, Messgeräte und Verschmutzungen vom Gerät.

Die Nichtbeachtung dieser Anweisungen kann Tod, schwere Verletzungen oder Sachschäden zur Folge haben.

Führen Sie alle in der Dokumentation des Geräts empfohlenen Anlauftests durch. Die gesamte Dokumentation zur späteren Verwendung aufbewahren.

Softwaretests müssen sowohl in simulierten als auch in realen Umgebungen stattfinden.

Sicherstellen, dass in dem komplett installierten System keine Kurzschlüsse anliegen und nur solche Erdungen installiert sind, die den örtlichen Vorschriften entsprechen (z. B. gemäß dem National Electrical Code in den USA). Wenn Hochspannungsprüfungen erforderlich sind, beachten Sie die Empfehlungen in der Gerätedokumentation, um eine versehentliche Beschädigung zu verhindern.

Vor dem Einschalten der Anlage:

- Entfernen Sie Werkzeuge, Messgeräte und Verschmutzungen vom Gerät.
- Schließen Sie die Gehäusetür des Geräts.
- Alle temporären Erdungen der eingehenden Stromleitungen entfernen.
- Führen Sie alle vom Hersteller empfohlenen Anlauftests durch.

BETRIEB UND EINSTELLUNGEN

Die folgenden Sicherheitshinweise sind der NEMA Standards Publication ICS 7.1-1995 entnommen (die Englische Version ist maßgebend):

- Ungeachtet der bei der Entwicklung und Fabrikation von Anlagen oder bei der Auswahl und Bemessung von Komponenten angewandten Sorgfalt, kann der unsachgemäße Betrieb solcher Anlagen Gefahren mit sich bringen.
- Gelegentlich kann es zu fehlerhaften Einstellungen kommen, die zu einem unbefriedigenden oder unsicheren Betrieb führen. Für Funktionseinstellungen stets die Herstelleranweisungen zu Rate ziehen. Das Personal, das Zugang zu diesen Einstellungen hat, muss mit den Anweisungen des Anlagenherstellers und den mit der elektrischen Anlage verwendeten Maschinen vertraut sein.
- Bediener sollten nur über Zugang zu den Einstellungen verfügen, die tatsächlich für ihre Arbeit erforderlich sind. Der Zugriff auf andere Steuerungsfunktionen sollte eingeschränkt sein, um unbefugte Änderungen der Betriebskenngrößen zu vermeiden.

Über dieses Buch

Auf einen Blick

Ziel dieses Dokuments

In diesem Handbuch werden die Grundlagen der Hardware- und Softwareimplementierung für die Zeichenmodus- und Modbus-Kommunikation für SPS der Baureihe Modicon M340 beschrieben.

Gültigkeitsbereich

Diese Dokumentation ist gültig ab EcoStruxure™ Control Expert 14.0.

Die technischen Merkmale der hier beschriebenen Geräte sind auch online abrufbar. So greifen Sie auf diese Informationen online zu:

Schritt	Aktion
1	Gehen Sie zur Homepage von Schneider Electric *www.schneider-electric.com*.
2	Geben Sie im Feld **Search** die Referenz eines Produkts oder den Namen einer Produktreihe ein. ● Die Referenz bzw. der Name der Produktreihe darf keine Leerstellen enthalten. ● Wenn Sie nach Informationen zu verschiedenen vergleichbaren Modulen suchen, können Sie Sternchen (*) verwenden.
3	Wenn Sie eine Referenz eingegeben haben, gehen Sie zu den Suchergebnissen für technische Produktdatenblätter (**Product Datasheets**) und klicken Sie auf die Referenz, über die Sie mehr erfahren möchten. Wenn Sie den Namen einer Produktreihe eingegeben haben, gehen Sie zu den Suchergebnissen **Product Ranges** und klicken Sie auf die Reihe, über die Sie mehr erfahren möchten.
4	Wenn mehrere Referenzen in den Suchergebnissen unter **Products** angezeigt werden, klicken Sie auf die gewünschte Referenz.
5	Je nach der Größe der Anzeige müssen Sie ggf. durch die technischen Daten scrollen, um sie vollständig einzusehen.
6	Um ein Datenblatt als PDF-Datei zu speichern oder zu drucken, klicken Sie auf **Download XXX product datasheet**.

Die in diesem Dokument vorgestellten Merkmale sollten denen entsprechen, die online angezeigt werden. Im Rahmen unserer Bemühungen um eine ständige Verbesserung werden Inhalte im Laufe der Zeit möglicherweise überarbeitet, um deren Verständlichkeit und Genauigkeit zu verbessern. Sollten Sie einen Unterschied zwischen den Informationen im Dokument und denen online feststellen, nutzen Sie die Online-Informationen als Referenz.

Verwandte Dokumente

Titel der Dokumentation	Referenznummer
Modicon X80, BMXNOM0200 Serielles Leitungsmodul, Benutzerhandbuch	EIO0000002696 (Englisch), EIO0000002697 (Französisch), EIO0000002698 (Deutsch), EIO0000002699 (Italienisch), EIO0000002700 (Spanisch), EIO0000002701 (Chinesisch)
Modicon M580, M340 und X80 I/O-Plattformen, Normen und Zertifizierungen	EIO0000002726 (Englisch), EIO0000002727 (Französisch), EIO0000002728 (Deutsch), EIO0000002730 (Italienisch), EIO0000002729 (Spanisch), EIO0000002731 (Chinesisch)
EcoStruxure™ Control Expert, Betriebsarten	33003101 (Englisch), 33003102 (Französisch), 33003103 (Deutsch), 33003104 (Spanisch), 33003696 (Italienisch), 33003697 (Chinesisch)
EcoStruxure™ Control Expert Kommunikation, Bausteinbibliothek	33002527 (Englisch), 33002528 (Französisch), 33002529 (Deutsch), 33003682 (Italienisch), 33002530 (Spanisch), 33003683 (Chinesisch)
EcoStruxure™ Control Expert, E/A-Verwaltung, Block-Bibliothek	33002531 (Englisch), 33002532 (Französisch), 33002533 (Deutsch), 33003684 (Italienisch), 33002534 (Spanisch), 33003685 (Chinesisch)

Sie können diese technischen Veröffentlichungen sowie andere technische Informationen von unserer Website herunterladen: *www.schneider-electric.com/en/download*.

Produktbezogene Informationen

⚠ WARNUNG

UNBEABSICHTIGTER GERÄTEBETRIEB

Die Anwendung dieses Produkts erfordert Fachkenntnisse bezüglich der Entwicklung und Programmierung von Steuerungssystemen. Nur Personen mit solchen Fachkenntnissen sollten dieses Produkt programmieren, installieren, ändern und anwenden.

Befolgen Sie alle landesspezifischen und örtlichen Sicherheitsnormen und -vorschriften.

Die Nichtbeachtung dieser Anweisungen kann Tod, schwere Verletzungen oder Sachschäden zur Folge haben.

Teil I

Hardwaretechnische Installation für die serielle Modbus- und Zeichenmoduskommunikation

Inhalt dieses Teils

Dieser Abschnitt enthält eine Beschreibung der hardwaretechnischen Installation für die serielle Modbus- und Zeichenmoduskommunikation.

Inhalt dieses Teils

Dieser Teil enthält die folgenden Kapitel:

Kapitel 1
Einführung in die serielle Kommunikation

Inhalt dieses Kapitels

Dieses Kapitel enthält eine Einführung in die serielle Kommunikation auf der Modicon M340-Plattform.

Die nachstehende Tabelle bietet eine Kurzübersicht über die zwei Möglichkeiten zur Implementierung einer Kommunikation über eine serielle Verbindung:

Verwendung des integrierten Ports der M340-CPU	**Verwendung des Kommunikationsmoduls BMX NOM 0200** ***(siehe Modicon X80, Serielles Verbindungsmodul BMXNOM0200, Benutzerhandbuch)***
- Begrenzte Übertragungsgeschwindigkeit - Nicht-potentialgetrennte serielle Leitungen - Bereitstellung einer Spannungsversorgung für die Endgeräte	- Erhöhte Anzahl an verfügbaren Kommunikationskanälen - Verwaltung der modemspezifischen RS232-Signale - Höherer Übertragungsgeschwindigkeit - Zwei potentialgetrennte serielle RS458-Leitungen

Inhalt dieses Kapitels

Dieses Kapitel enthält die folgenden Themen:

Kommunikation im seriellen Modbus- und im Zeichenmodus

Allgemeines

Serielle Verbindungen unterstützen zwei Kommunikationsprotokolle:

- Modbus seriell
- Zeichenmodus

Modbus-Protokoll

Modbus ist ein Standardprotokoll und weist folgende Eigenschaften auf:

- Das Protokoll baut eine Client/Server-Kommunikation zwischen verschiedenen Modulen über eine Bus- oder serielle Verbindung auf. Der Client wird vom Master identifiziert und die Slave-Module fungieren als Server.
- Das Protokoll basiert auf einem Datenaustauschmodus, bestehend aus Requests und Antworten zur Bereitstellung von Diensten anhand unterschiedlicher Funktionscodes.
- Das Protokoll ermöglicht den Austausch der Frames Modbus-fähiger Anwendungen in zwei Arten von Code:
  - RTU-Modus
  - ASCII-Modus

Die Verwaltung des Austauschs verläuft folgendermaßen:

- Nur jeweils ein Gerät kann Daten auf dem Bus senden.
- Der Datenaustausch wird vom Master verwaltet. Nur der Master kann den Austausch initiieren. Slaves können keine Nachrichten senden, wenn sie nicht zuvor eine entsprechende Aufforderung erhalten.
- Bei einem ungültigen Austausch wiederholt der Master seinen Request. Der Slave, an den der Request gerichtet ist, wird vom Master als abwesend eingestuft, wenn dieser nicht innerhalb eines vorgegebenen Zeitraums antwortet.
- Wenn der Slave den Request nicht versteht bzw. nicht verarbeiten kann, sendet er eine Ausnahmeantwort an den Master. In diesem Fall kann der Master den Request erneut senden oder nicht.

Zwischen Master und Slave(s) sind zwei Typen von Dialog möglich:

- Der Master sendet einen Request an eine bestimmte Slave-Nummer und wartet auf die Antwort des betreffenden Slaves.
- Der Master sendet einen Request an alle Slaves, ohne auf eine Antwort zu warten (das entspricht dem allgemeinen Broadcast-Prinzip).

Kommunikation im Zeichenmodus

Bei der Kommunikation im Zeichenmodus handelt es sich um einen Datenaustausch im Punkt-zu-Punkt-Modus zwischen zwei Einheiten. Im Gegensatz zum Modbus-Protokoll wird in diesem Modus keine hierarchisch strukturierte Kommunikation über eine serielle Verbindung aufgebaut bzw. es werden keine Dienste über Funktionscodes bereitgestellt.

Der Zeichenmodus ist asynchron. Jedes Element der Textinformationen wird Zeichen für Zeichen in unregelmäßigen Zeitintervallen gesendet oder empfangen. Die Dauer des jeweiligen Austauschs wird durch Folgendes bestimmt:

- Ein oder zwei Frame-Endzeichen
- Timeout
- Anzahl der Zeichen

Beschreibung der seriellen Verbindung von Modicon M340-Prozessoren

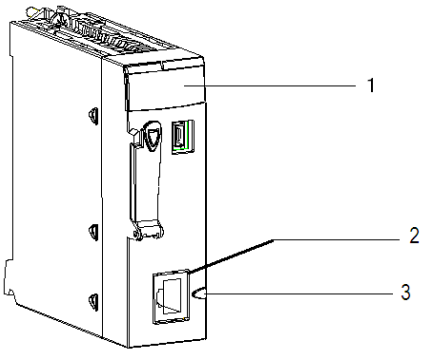
Allgemeines

Folgende Prozessoren verfügen über einen integrierten Kommunikationskanal für die serielle Kommunikation und und ermöglichen folglich eine Kommunikation über eine serielle Verbindung:

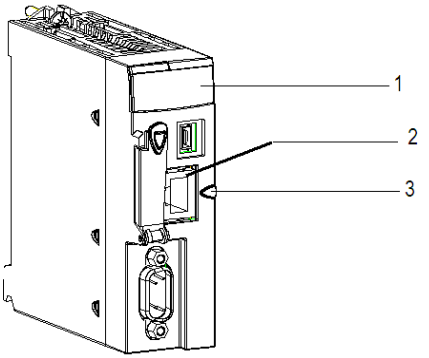
- BMX P34 1000
- BMX P34 2000
- BMX P34 2010
- BMX P34 20102
- BMX P34 2020

Position des seriellen Ports

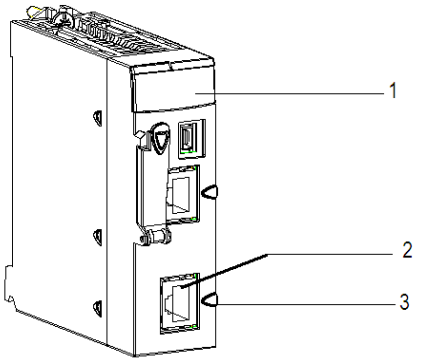
Die nachstehende Abbildung zeigt die Position des seriellen Ports von Modicon M340-Prozessoren:



BMX P34 1000/2000



BMX P34 2010/20102



BMX P34 2020

Diese Prozessoren umfassen folgende Elemente:

Adresse	Beschreibung
1	Prozessorstatus-LEDs an der Vorderseite
2	Integrierter Kanal (Kanal 0) für die serielle Verbindung
3	Identifikationsring des seriellen Ports (schwarz)

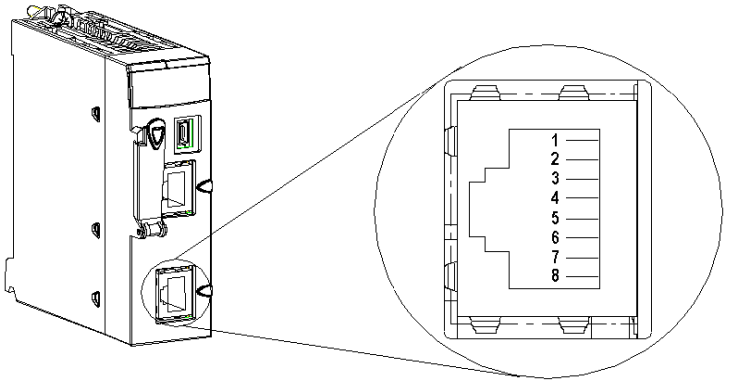
Sichtdiagnose der seriellen Kommunikation

Der Status der seriellen Kommunikation wird über die gelbe LED SER COM an der Vorderseite dieser Prozessoren ausgewiesen:

- LED blinkend: Serielle Kommunikation aktiv
- LED aus: Keine serielle Kommunikation aktiv

Beschreibung des seriellen Portanschlusses

Die nachstehende Abbildung zeigt den seriellen RJ45-Port:



Der RJ45-Anschluss verfügt über acht Pins. Welche Pins verwendet werden, ist von der verwendeten physischen Verbindung abhängig.

Im Folgenden werden die von der seriellen RS232-Verbindung verwendeten Pins aufgeführt:

- Pin 1: RXD-Signal
- Pin 2: TXD-Signal
- Pin 3: RTS-Signal
- Pin 6: CTS-Signal
- Pin 8: Potenzielle Erdung der seriellen Verbindung (0 V)

Im Folgenden werden die von der seriellen RS485-Verbindung verwendeten Pins aufgeführt:

- Pin 4: D1-Signal
- Pin 5: D0-Signal

Pin 7 wird ausschließlich zur Speisung der Mensch-Maschine-Schnittstellen (HMI) oder kleinerer Geräte über das serielle Verbindungskabel verwendet.

- Pin 7: Spannungsversorgung über die serielle Verbindung: 5 VDC / 190 mA

Detaillierte Merkmale

Merkmale der Gleichstromversorgung:

- Maximale stabilisierte Stromaufnahme: 190 mA
- Mindestspannung am CPU-Anschluss für 190 mA: 4,9 V
- Höchstspannung am CPU-Anschluss für 190 mA: 5,25 V
- Höchstspannung am CPU-Anschluss ohne Last: 5,5 V

Merkmale der Wechselstromversorgung:

- Kondensatorlast: (bei 5 V)
  - Keramikkondensator 1 µF max.
  - 10 µF Tantal
- Pumplast Start: (bei 5 V)
  - 4 x Keramikkondensator 1 µF
  - 2 x 10 µF Tantal

HINWEIS: Der 4-Draht-Anschluss RS232, der 2-Draht-Anschluss RS485 und der 2-Draht-Anschluss RS485 mit Spannungsversorgung verwenden alle dieselbe RJ45-Anschlussbuchse. Lediglich die Signalverdrahtung ist unterschiedlich.

Merkmale der Stromleitung

Die RS232- und die RS485-Leitung sind nicht potentialgetrennt.

Bei einer nicht-äquipotentialen Erde zwischen den angeschlossenen Geräten (Kabel mit einer Länge von mindestens 30 m) muss ein Trenner-Modul TWDXCAISO im RS485-Modus zugeschaltet werden.

Die Polarisierung der RS485-Leitung ist in die SPS integriert und wird automatisch vom System in Übereinstimmung mit der im -Fenster ausgewählten KonfigurationControl Expert aktiviert oder deaktiviert:

- Modbus-Master: Die Leitungspolarisierung ist aktiviert.
- Modbus-Slave: Die Leitungspolarisierung ist deaktiviert.
- Zeichenmodus: Die Leitungspolarisierung ist deaktiviert.

Die Polarisierung wird nicht durch die dynamische Protokollumschaltung beeinflusst. Der Polarisationswiderstand beträgt 560 Ohm.

Im RS232-Modus ist keine Polarisierung erforderlich.

Es ist kein integrierter Leitungsabschluss vorhanden.

Technische Kenndaten des Kanals

Der Kanal dieser Prozessoren umfasst folgende Elemente:

- Nicht-potentialgetrennte physische RS485-Schnittstelle
- Nicht-potentialgetrennte physische RS232-Schnittstelle
- Kommunikationstypen Modbus seriell (ASCII und RTU) und Zeichenmodus

Technische Kenndaten der Verbindung für die zwei Protokolle:

	Modbus seriell / RS485	**Modbus seriell / RS232**	**Zeichenmodus / RS485**	**Zeichenmodus / RS232**
Typ	Master/Slave	Master/Slave	Half Duplex	Full Duplex
Datenfluss	19200 Baud. Die Parameter können auf einen Wert zwischen 300 und 38400 Baud eingestellt werden.	19200 Baud. Die Parameter können auf einen Wert zwischen 300 und 38400 Baud eingestellt werden.	9600 Baud. Die Parameter können auf einen Wert zwischen 300 und 38400 Baud eingestellt werden.	9600 Baud. Die Parameter können auf einen Wert zwischen 300 und 38400 Baud eingestellt werden.
Anzahl Geräte	32	32	_	_
Zulässige Slave-Adressen	1 bis 247	1 bis 247	_	_
Max. Buslänge ohne Abzweige	1000 m (15 m mit Abzweig)	15 m	1000 m (15 m mit Abzweig)	15 m
Nachrichtengröße	Modbus seriell: ● RTU: 256 Byte (252 Byte an Daten) ● ASCII: 513 Byte (2x252 Byte an Daten)	Modbus seriell: ● RTU: 256 Byte (252 Byte an Daten) ● ASCII: 513 Byte (2x252 Byte an Daten)	1024 Byte	1024 Byte
Dienstprogramme	Lesen von Wörtern/Bits Schreiben von Wörtern/Bits Diagnose	Lesen von Wörtern/Bits Schreiben von Wörtern/Bits Diagnose	Senden von Zeichenfolgen Empfangen von Zeichenfolgen	Senden von Zeichenfolgen Empfangen von Zeichenfolgen

Normen und Zertifizierungen

Online-Hilfe

Über die Online-Hilfe von Control Expert können Sie die für die Module dieser Produktfamilie geltenden Normen und Zertifizierungen abrufen. Diese sind im Handbuch *Modicon M580, M340 und X80 I/O-Plattformen, Normen und Zertifizierungen* enthalten.

Download

Klicken Sie auf die Verknüpfung für Ihre bevorzugte Sprache, um die Normen und Zertifizierungen für die Module dieser Produktfamilie (im PDF-Format) herunterzuladen:

Sprache	
Englisch	*Modicon M580, M340 und X80, Normen und Zertifizierungen*
Französisch	*Modicon M580, M340 und X80, Normen und Zertifizierungen*
Deutsch	*Modicon M580, M340 und X80, Normen und Zertifizierungen*
Italienisch	*Modicon M580, M340 und X80, Normen und Zertifizierungen*
Spanisch	*Modicon M580, M340 und X80, Normen und Zertifizierungen*
Chinesisch	*Modicon M580, M340 und X80, Normen und Zertifizierungen*

Hinweise zur Verdrahtung

Richtlinien für den Betrieb

⚠ WARNUNG

UNBEABSICHTIGTER BETRIEB VON GERÄTEN

Obwohl die CPUs des Typs BMX P34 20x0 auch bei unter Spannung stehender Station angeschlossen und von den Anschlüssen getrennt werden können, kann die jeweils gerade ausgeführte Anwendung dadurch unterbrochen werden.

Die Nichtbeachtung dieser Anweisungen kann Tod, schwere Verletzungen oder Sachschäden zur Folge haben.

Verbindung

Die folgenden Situationen können eine temporäre Unterbrechung der Anwendung oder Kommunikation bewirken:

- Der Steckverbinder der RJ45 wird angeschlossen oder getrennt, während die Stromversorgung eingeschaltet ist.
- Die Module werden neu initialisiert, wenn die Stromversorgung wieder eingeschaltet wird.

Kapitel 2
Serielle Kommunikationsarchitekturen

Inhalt dieses Kapitels

Dieses Kapitel bietet eine Einführung in Architekturen, die auf eine serielle Kommunikation zurückgreifen, und enthält die Anforderungen an die Verdrahtung.

Inhalt dieses Kapitels

Dieses Kapitel enthält die folgenden Themen:

Modbus-Leitungsabschluss und Polarisierung (RS485)

Übersicht

Ein Mehrpunkt-Modbus-Netzwerk erfordert einen Leitungsabschluss und eine Polarisierung.

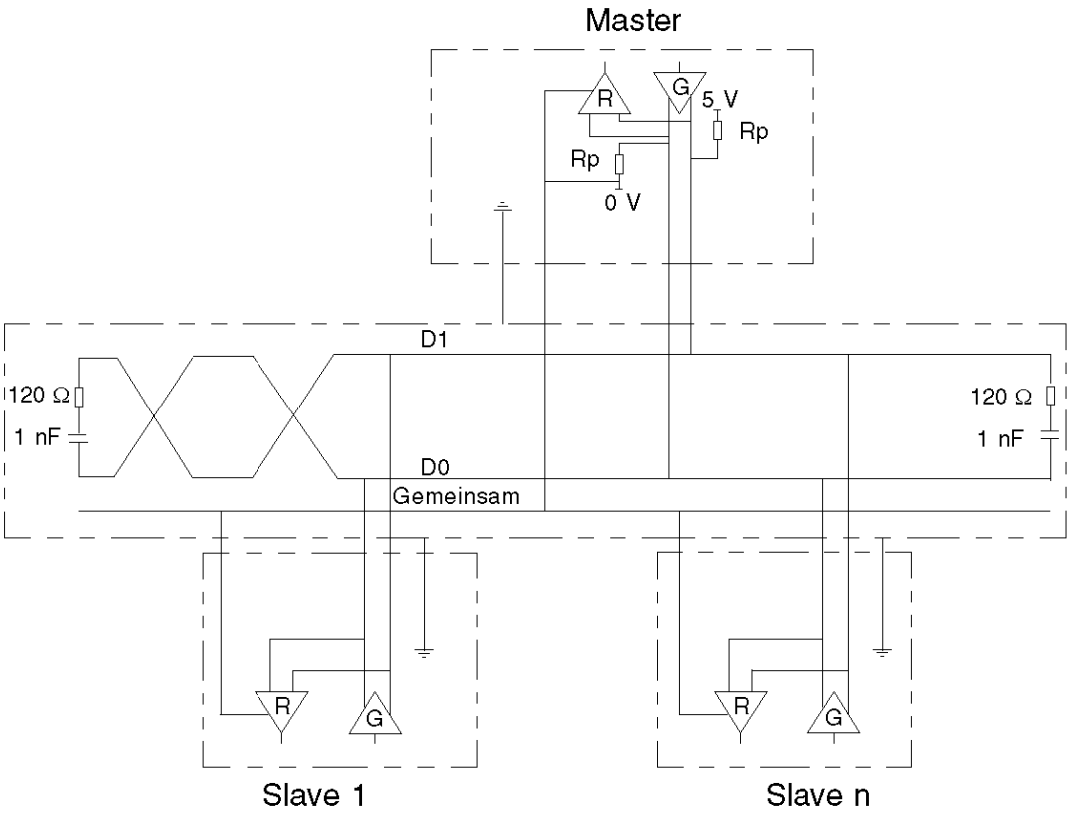
An diesen Bus sind folgende Geräte anschließbar:

- Andere SPS wie M340, Premium, Quantum, Twido oder Nano
- Geräte von Schneider Automation wie Altivar, Sicherheitsmodule XPS, SEPAM, XBT oder Momentum
- Andere Modbus-kompatible Geräte
- Modem, Hub

Ein Beispiel für ein **Mehrpunkt-Modbus-Netzwerk** *(siehe Seite 31)* mit einem BMX P34 2010-Prozessor wird in diesem Handbuch vorgestellt.

HINWEIS: Modbus-Netzwerke können auch als Punkt-zu-Punkt-Netzwerk realisiert werden.

Elektrische Grundlagen zu einem Leitungsabschluss und einer Polarisierung:



Leitungsabschluss

Der Leitungsabschluss erfolgt extern: Sie besteht aus zwei 120-Ω-Widerständen und einen 1-nF-Kondensator an jedem Ende des Netzwerks (VW3 A8 306 RC oder VW3 A8 306 DRC).

Platzieren Sie den Leitungsabschluss nicht am Ende von Abzweigkabeln.

Leitungspolarisierung

Bei einem Modbus ist eine Polarisierung des RS485-Netzwerks erforderlich.

- Wenn die M340-CPU als Master eingesetzt wird, **erfolgt diese automatisch durch das System** *(siehe Seite 22)*, so dass keine externe Polarisierung erforderlich ist.
- Wenn die M340-CPU als Slave eingesetzt wird, muss die Polarisierung in Form von zwei Widerständen (Rp) von 450 bis 650 Ω, je einem pro RS485-Kreis erfolgen:
  - ein Pull-Up-Widerstand gegen 5 V im D1-Stromkreis,
  - ein Pull-Down-Widerstand gegen Masse im D0-Stromkreis.

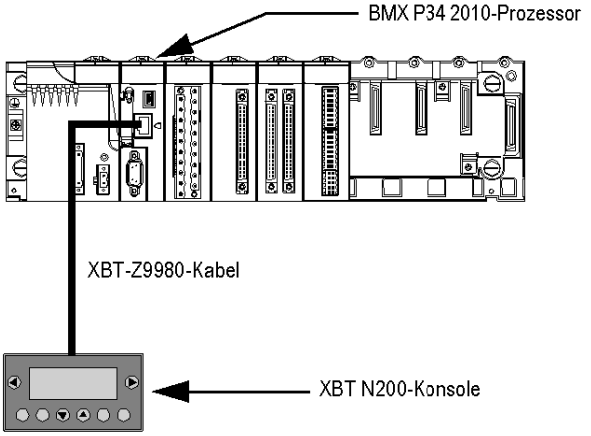
Verbindung von Modbus-Geräten (RS4859

Allgemeines

Auf den folgenden Seiten werden zwei Beispiele für die Verbindung von Modbus-Geräte und eine serielle Modbus-Verbindungsarchitektur vorgestellt.

Verbinden von über die serielle Verbindung gespeisten Modbus-Geräten

Die nachstehende Abbildung zeigt die Verbindung eines Prozessors BMX P34 2010 mit einer Konsole XBT N200, die über die serielle Modbus-Verbindung gespeist wird:



Die Geräte sind folgendermaßen konfiguriert:

- Der Prozessor BMX P34 2010 ist als Slave konfiguriert.
- Das HMI XBT N200 (Mensch-Maschine-Schnittstelle) ist als Master konfiguriert.

Die Kabel XBT-Z9980 weisen folgende Eigenschaften auf:

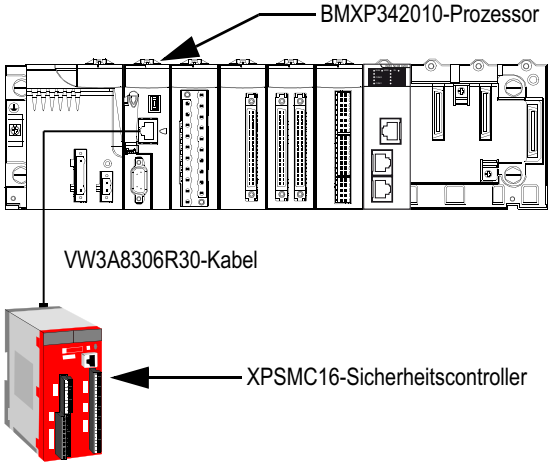
- Anschluss: 2 RTJ45-Steckverbinder
- Verdrahtung: 2 Drähte für die physische RS485-Leitung und 2 für die Spannungsversorgung über die serielle Verbindung

Verbinden von nicht über die serielle Verbindung gespeisten Modbus-Geräten

Diese Architektur besteht aus folgenden Elementen:

- Prozessor BMX P34 2010
- Sicherheitssteuerung XPSMC16

Die nachstehende Abbildung zeigt die Verbindung eines Prozessors BMX P34 2010 mit einer Sicherheitssteuerung XPSMC16:



Die Geräte sind folgendermaßen konfiguriert:

- Der Prozessor BMX P34 2010 ist als Master konfiguriert.
- Die Sicherheitssteuerung XPSMC16 ist als Slave konfiguriert.

Das Kabel VW3 A8 306 R30 weist folgende Eigenschaften auf:

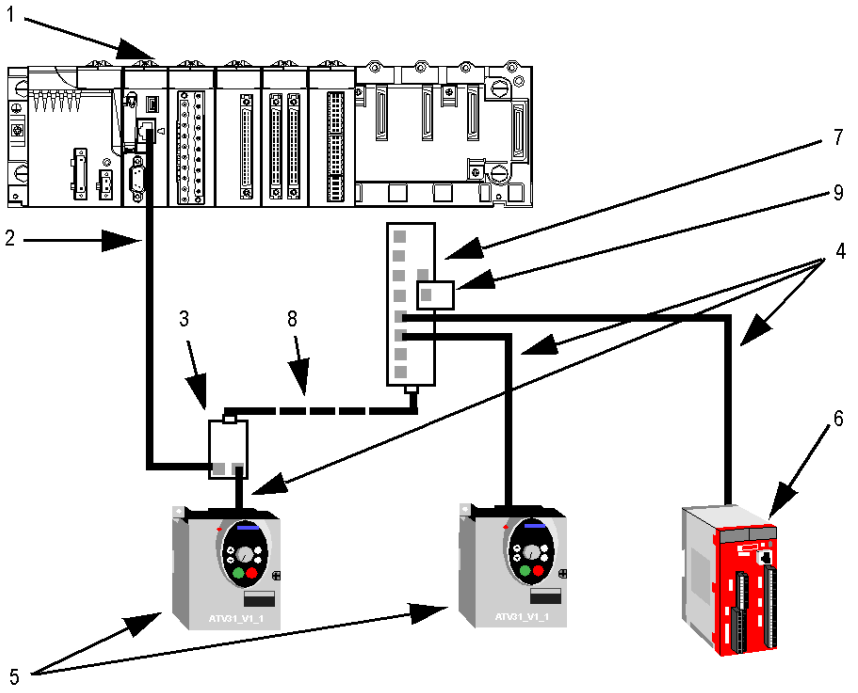
- Anschluss: 2 RTJ45-Steckverbinder
- Verdrahtung: 2 Drähte für die physische RS485-Leitung

Serielle Modbus-Verbindungsarchitektur

Die Architektur einer seriellen Modbus-Verbindung umfasst folgende Elemente:

- Als Master konfigurierter Prozessor BMX P34 2010/20102
- Als Slave konfigurierte Sicherheitssteuerung XPSMC16
- Isolierte Splitterbox TWDXCAISO
- Splitterblock LU9 GC3
- Zwei als Slaves konfigurierte ATV32-Antriebe

Die nachstehende Grafik illustriert die oben beschriebene serielle Verbindungsarchitektur:



1 Prozessor BMX P34 2010
2 Kabel XBT-Z9980
3 Isolierte Splitterbox TWDXCAISO
4 Kabel VW3 A8 306 R30
5 ATV31-Antrieb
6 Sicherheitssteuerung XPSMC16
7 Splitterblock LU9 GC3
8 Kabel TSXCSAx00
9 Modbus-RC-Leitungsabschluss VW3 A8 306

Anschließen von Datenendeinrichtungen (DEE) (RS232)

Allgemein

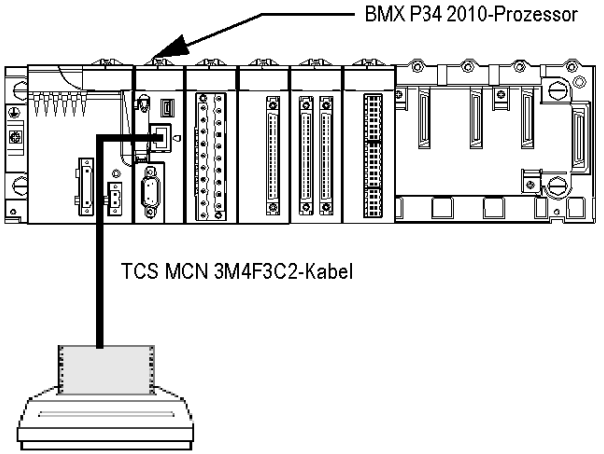
Der Begriff Datenendeinrichtung beschreibt beispielsweise folgende Geräte:

- Herkömmliche Peripheriegeräte (Drucker, Tastaturbildschirm, Endgeräte in Werkstätten usw.),
- Spezielle Peripheriegeräte (Strichcodelesegeräte usw.),
- PCs.

Jede Datenübertragungseinrichtung wird über ein serielles, gekreuztes Kabel mittels der physischen RS232-Verbindung mit BMX P34 1000/2000/2010/20102/2020-Prozessoren verbunden.

Anschließen von Datenendeinrichtungen

Die folgende Abbildung verdeutlicht, wie ein Drucker mit einem BMX P34 2010-Prozessor verbunden wird:



Als Kommunikationsprotokoll wird der Zeichenmodus verwendet.

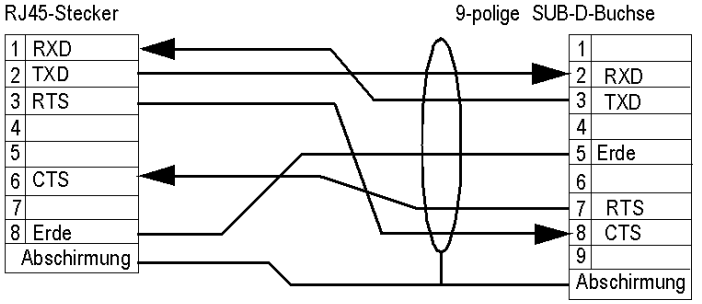
HINWEIS: Mit einem BMX P34 1000/2000/2010/20102/2020-Prozessor kann jeweils nur eine Datenendeinrichtung verbunden werden.

Serielles, gekreuztes RS232-Kabel

Das serielle, gekreuzte TCS MCN 3M4FC2-Kabel weist zwei Anschlüsse auf:

- RJ45-Stecker
- 9-polige SUB-D-Buchse

Die folgende Abbildung zeigt die Anschlussbelegung eines seriellen, gekreuzten Kabels TCS MCN 3M4F3C2:



Verbinden von Kabeln und Zubehör

Die folgende Tabelle führt die Referenznummern der Kabel und Adapter auf, die gemäß dem seriellen Anschluss der Datenendeinrichtung verwendet werden müssen:

Serieller Anschluss für Datenendeinrichtung	Verdrahtung
9-poliger SUB-D-Stecker	TCS MCN 3M4F3C2-Kabel
25-poliger SUB-D-Stecker	• TCS MCN 3M4F3C2-Kabel • TSX CTC 07-Adapter
25-polige SUB-D-Buchse	• TCS MCN 3M4F3C2-Kabel • TSX CTC 10-Adapter

Anschließen von Datenübertragungseinrichtungen (DÜE) (RS232)

Allgemein

Der Begriff Datenübertragungseinrichtung (DÜE) beschreibt Geräte wie beispielsweise Modems.

Bei einem DÜE-Gerät werden die RTS- und CTS-Leitungen direkt angeschlossen (nicht gekreuzt).

Jede Datenendeinrichtung wird über ein serielles, ungekreuztes Kabel mittels der physischen RS232-Verbindung mit BMX P34 1000/2000/2010/20102/2020-Prozessoren verbunden.

HINWEIS: Die Unterschiede beim Anschluss von DÜE und DEE liegen im Wesentlichen in der Steckerbelegung und den Signalrichtungen (Eingang oder Ausgang). Zum Beispiel ist ein PC in der Regel eine Datenendeinrichtung (DEE), während ein Modem eine Datenübertragungseinrichtung (DÜE) ist.

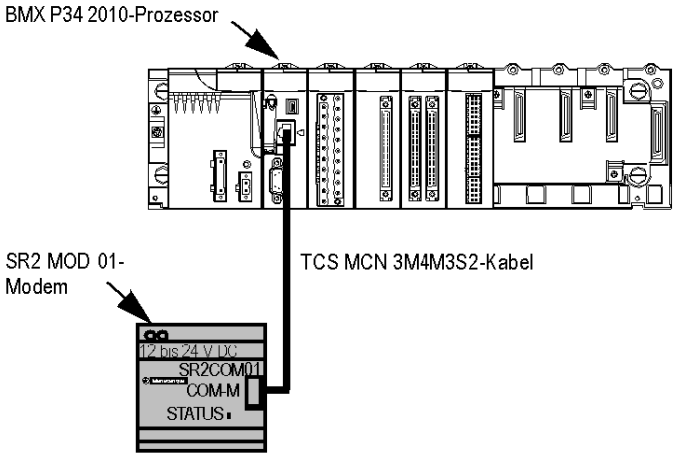
Technische Daten der Modems

M340-CPUs unterstützen die meisten handelsüblichen Modems. Um ein Modem an den seriellen Port eines BMX P34 1000/2000/2010/20102/2020-Prozessors anzuschließen, muss das Modem folgende Merkmale aufweisen:

- 10 bzw. 11 Bits pro Zeichen unterstützen, falls der Terminal-Port im seriellen Modbus verwendet wird:
  - 7 bzw. 8 Datenbits
  - 1 bzw. 2 Stoppbits
  - Ungerade, gerade oder keine Parität
- Ohne Datenträgerprüfung arbeiten.

Anschließen von Datenendeinrichtungen

Die folgende Abbildung verdeutlicht, wie ein Modem mit einem BMX P34 2010-Prozessor verbunden wird:



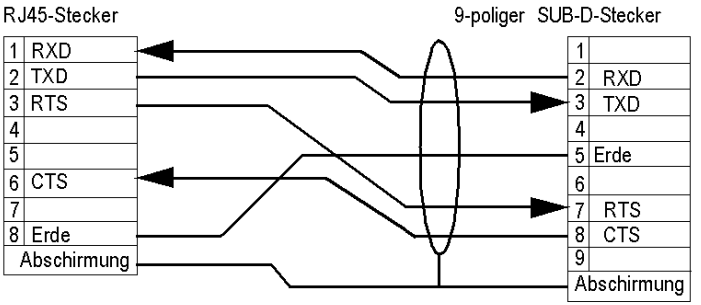
HINWEIS: Beim seriellen Modbus muss die Wartezeit zwischen 100 und 250 ms liegen.

Serielles, ungekreuztes RS232-Kabel

Das serielle, ungekreuzte Kabel TUS MCN 3M4M3S2 weist zwei Anschlüsse auf:

- RJ45-Stecker,
- 9-poliger SUB-D-Stecker.

Die folgende Abbildung zeigt die Anschlussbelegung eines seriellen, ungekreuzten Kabels TCS MCN 3M4M3S2:



Verbinden von Kabeln und Zubehör

Die folgende Tabelle führt die Referenznummern der Kabel und Adapter auf, die gemäß dem seriellen Anschluss der Datenendeinrichtung verwendet werden müssen:

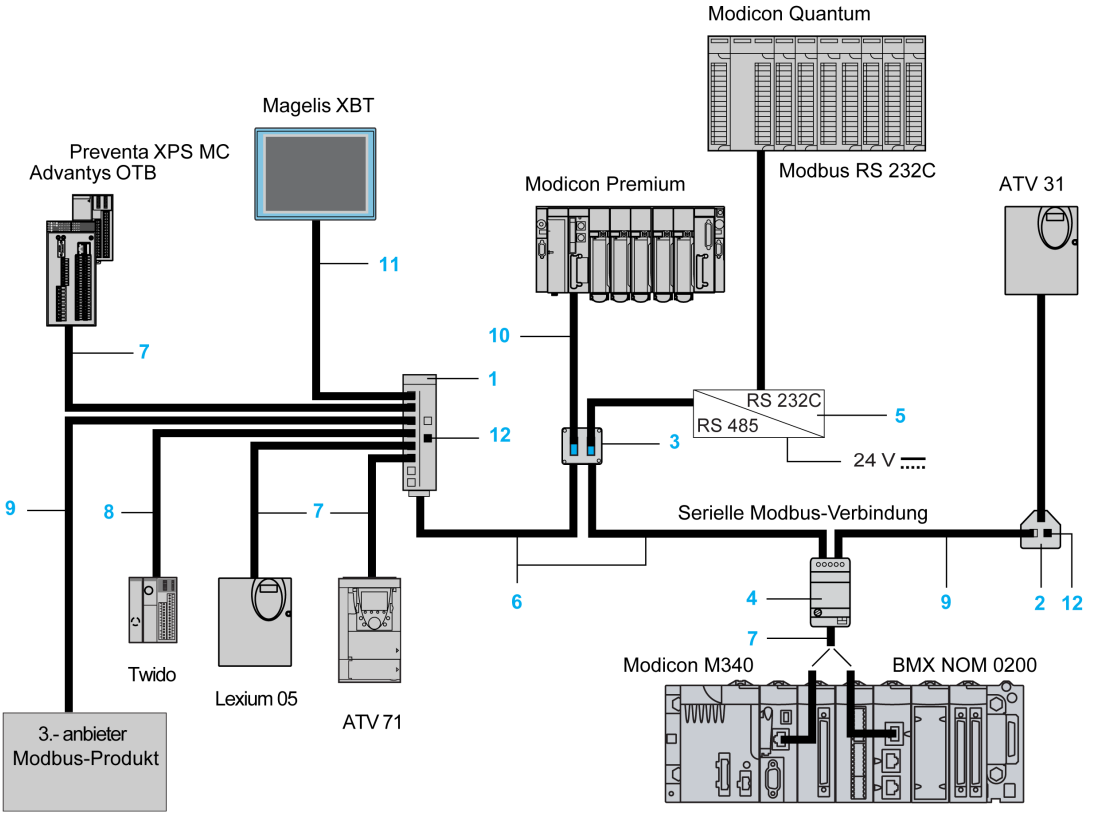
Serieller Anschluss für Datenendeinrichtung	Verdrahtung
9-polige SUB-D-Buchse	TCS MCN 3M4M3S2-Kabel
25-polige SUB-D-Buchse	• TCS MCN 3M4M3S2-Kabel • TSX CTC 09-Adapter

Verkabelung

Verkabelungssystem

Zur Einrichtung einer seriellen Verbindung sind mehrere Kabel und Zubehörteile erforderlich.

Die nachstehende Abbildung zeigt ein Beispiel für das Verkabelungssystem einer seriellen Modbus-Verbindung im Zeichenmodus. Die in der Abbildung ausgewiesenen **Kabel** *(siehe Seite 39)* und **Anschlusszubehörteile** *(siehe Seite 40)* werden in den folgenden Tabellen beschrieben:



Kabel

Die folgende Tabelle zeigt die verfügbaren Kabel, die mit einer seriellen Kommunikation mit diesen Prozessoren und Modulen kompatibel sind:

Abbildungs-referenz	Bezeichnung	Eigenschaften	Länge	Produktreferenz
6	Doppelt geschirmtes, verdrilltes RS485-Twisted-Pair-Trunkkabel	Zwei freie Enden	100 m	TSX CSA 100
			200 m	TSX CSA 200
			500 m	TSX CSA 500
7	Modbus-RS485-Kabel	Zwei RJ45-Stecker	0,3 m	VW3 A8 306 R03
			1 m	VW3 A8 306 R10
			3 m	VW3 A8 306 R30
-	Modbus-RS485-Kabel	• Ein RJ45-Stecker • Ein 15-poliger SUB-D-Steckverbinder	3 m	VW3 A8 306
8	Modbus-RS485-Kabel	• Ein RJ45-Stecker • Ein mini-DIN-Steckverbinder	0,3 m	TWD XCA RJ003
			1 m	TWD XCA RJ010
			3 m	TWD XCA RJ030
9	Modbus-RS485-Kabel	• Ein RJ45-Stecker • Ein freies Ende	3 m	VW3 A8 306 D30
10	Modbus-RS485-Kabel	• Ein Miniaturstecker • Ein 15-poliger SUB-D-Steckverbinder	3 m	TSX SCP CM 4630
11	RS485-Kabel für Maglis XBT-Display und -Bedienpult	• Ein RJ45-Stecker • Eine 25-polige SUB-D-Steckbuchse **Hinweis:** Dieses Kabel ist nicht mit dem Modul BMX NOM 0200 kompatibel.	2,5 m	XBT-Z938
-	RS485-Kabel für über die serielle Verbindung gespeiste Geräte	Zwei RJ45-Stecker **Hinweis:** Dieses Kabel ist nicht mit dem Modul BMX NOM 0200 kompatibel.	3 m	XBT-Z9980
-	4-drahtiges RS232-Kabel für Datenendgeräte (DTE: Data Terminal Equipment)	• Ein RJ45-Stecker • Eine 9-polige SUB-D-Steckbuchse	3 m	TCS MCN 3M4F3C2
-	4-drahtiges RS232-Kabel für Datenübertragungsgeräte (DCE: Data Circuit-terminating Equipment)	• Ein RJ45-Stecker • Ein 9-poliger SUB-D-Steckverbinder	3 m	TCS MCN 3M4M3S2

Abbildungs-referenz	Bezeichnung	Eigenschaften	Länge	Produktreferenz
-	7-drahtiges RS232-Kabel für Datenübertragungsgeräte (DCE: Data Circuit-terminating Equipment)	• Ein RJ45-Stecker • Ein 9-poliger SUB-D-Steckverbinder	3 m	TCS XCN 3M4F3S4

Anschlussszubehör

Die folgende Tabelle zeigt die verfügbaren Anschlusszubehörteile, die mit einer seriellen Kommunikation mit diesen Prozessoren und Modulen kompatibel sind:

Abbildungs-referenz	Bezeichnung	Eigenschaften	Produktreferenz
1	Modbus-Splitterbox	• Zehn RJ45-Stecker • Eine Schraubklemmenleiste	LU9 GC3
2	T-Anschlusskasten	• Zwei RJ45-Stecker • Integriertes Kabel 0,3 m mit RJ45-Stecker	VW3 A8 306 TF03
		• Zwei RJ45-Stecker • Integriertes Kabel 1 m mit RJ45-Stecker	VW3 A8 306 TF10
-	Passiver T-Anschlusskasten	• Drei Schraubklemmenleisten • RC-Leitungsendadapter	TSX SCA 50
3	Passives 2-Kanal-Teilnehmer-Socket	• Zwei 15-polige SUB-D-Steckbuchsen • Zwei Schraubklemmenleisten • RC-Leitungsendadapter	TSX SCA 62
4	Isolierter RS485-T-Anschlusskasten	• Ein RJ45-Steckverbinder • Eine Schraubklemmenleiste	TWD XCA ISO
-	T-Anschlusskasten	Drei RJ45-Steckverbinder	TWD XCA T3RJ
-	Modbus-/Bluetooth-Adapter	• Ein Bluetooth-Adapter mit einem RJ45-Stecker • Ein Kabelsatz für PowerSuite mit zwei RJ45-Steckern • Ein Kabelsatz für TwidoSuite mit einem RJ45- und einem mini-DIN-Stecker • Ein 9-poliger Adpater RJ45/SUB-D für ATV-Regelantriebe	VW3 A8 114
5	RS232C/RS485-Leitungsadapter ohne Modemsignale	19,2 KBit/s	XGS Z24

Abbildungs-referenz	Bezeichnung	Eigenschaften	Produktreferenz
12	Leitungsabschluss für RJ45-Stecker	• Widerstand 120 Ω • Kapazität 1 nF	VW3 A8 306 RC
-	Leitungsabschluss für Schraubklemmenleiste	• Widerstand 120 Ω • Kapazität 1 nF	VW3 A8 306 DRC
-	Adapter für Nicht-Standardgeräte	• Zwei 25-polige SUB-D-Steckverbinder	XBT ZG999
-	Adapter für Nicht-Standardgeräte	• Ein 25-poliger SUB-D-Steckverbinder • Ein 9-poliger SUB-D-Steckverbinder	XBT ZG909
-	Adapter für Datenendgeräte	• Ein 9-poliger SUB-D-Steckverbinder • Eine 25-polige SUB-D-Steckbuchse	TSX CTC 07
-	Adapter für Datenendgeräte	• Ein 9-poliger SUB-D-Steckverbinder • Ein 25-poliger SUB-D-Steckverbinder	TSX CTC 10
-	Adapter für Datenübertragungsgeräte (DCE)	• Eine 9-polige SUB-D-Steckbuchse • Ein 25-poliger SUB-D-Steckverbinder	TSX CTC 09

HINWEIS: Diese Liste mit Kabeln und Zubehörteilen ist nicht vollständig.

Teil II
Softwaretechnische Implementierung der seriellen Modbus- und Zeichenmoduskommunikation

Inhalt dieses Teils

Dieser Abschnitt enthält eine Beschreibung der softwaretechnischen Implementierung der seriellen Modbus- und Zeichenmoduskommunikation mit Control Expert-Software.

Inhalt dieses Teils

Dieser Teil enthält die folgenden Kapitel:

Kapitel 3
Installationsmethodik

Einführung in die Installationsphase

Einführung

Die Softwareinstallation der anwendungsspezifischen Module wird in den verschiedenen Control Expert-Editoren durchgeführt:

- im Offline-Modus
- im Online-Modus

Wenn Sie keinen Prozessor für die Verbindung haben, können Sie in Control Expert einen ersten Test mit einem Simulator durchführen. In diesem Fall verläuft die Installation unterschiedlich.

Installationsphasen bei Verwendung eines Prozessors

Die folgende Tabelle verdeutlicht die verschiedenen Installationsphasen bei Verwendung eines Prozessors:

Phase	Beschreibung	Modus
Konfiguration des Prozessors	Deklaration des Prozessors	Offline
	Konfiguration des seriellen Ports des Prozessors	
Konfiguration des Moduls (sofern zutreffend)	Deklaration des Moduls	Offline
	Konfiguration der Modulkanäle	
	Eingabe der Konfigurationsparameter	
Deklaration der Variablen	Deklaration der prozessor-/modulspezifischen Variablen vom Typ IODDT sowie der projektspezifischen Variablen	Offline(1)
Zuordnung	Zuordnung der IODDT-Variablen zu den konfigurierten Kanälen (Variableneditor)	Offline(1)
Programmierung	Programmierung des Projekts	Offline(1)
Generierung	Generierung des Projekts (Analyse und Bearbeitung der Verbindungen)	Offline
Übertragung	Übertragungs des Projekts in die SPS	Online
Debug	Debugging des Projekts in den Debug-Fenstern und Animationstabellen	Online
(1) Diese Phasen können auch online durchgeführt werden.		

Phase	Beschreibung	Modus
Dokumentation	Erstellung einer Dokumentationsdatei und Druck verschiedener Informationen mit Bezug auf das Projekt	Online
Funktionsweise	Anzeige der verschiedenen, zur Überwachung des Projekts erforderlichen Informationen	Online
(1) Diese Phasen können auch online durchgeführt werden.		

Installationsphasen bei Verwendung eines Simulators

Die folgende Tabelle verdeutlicht die verschiedenen Installationsphasen bei Verwendung eines Simulators:

Phase	Beschreibung	Modus
Konfiguration des Prozessors	Deklaration des Prozessors	Offline
	Konfiguration des seriellen Ports des Prozessors	
Konfiguration des Moduls (sofern zutreffend)	Deklaration des Moduls	Offline
	Konfiguration der Modulkanäle	
	Eingabe der Konfigurationsparameter	
Deklaration der Variablen	Deklaration der prozessor-/modulspezifischen Variablen vom Typ IODDT sowie der projektspezifischen Variablen	Offline(1)
Zuordnung	Zuordnung der IODDT-Variablen zu den konfigurierten Kanälen (Variableneditor)	Offline(1)
Programmierung	Programmierung des Projekts	Offline(1)
Generierung	Generierung des Projekts (Analyse und Bearbeitung der Verbindungen)	Offline
Übertragung	Übertragungs des Projekts in den Simulator	Online
Simulation	Simulation des Programms ohne Ein-/Ausgänge	Online
Einstellung/Debugging	Debugging des Projekts ausgehend von Animationstabellen	Online
	Änderung des Programms und Anpassung der Parameter	
(1) Diese Phasen können auch online durchgeführt werden.		

Konfiguration von Prozessor und Modul

Der Zugriff auf die Konfigurationsparameters ist nur über die Control Expert-Software möglich.

Erstellung einer technischen Dokumentation

Control Expert ermöglicht Ihnen die Erstellung einer **projekttechnischen Dokumentation** *(siehe EcoStruxure™ Control Expert, Betriebsarten)*.

Das allgemeine Format des Ausdrucks umfasst folgende Elemente:

- Den Titel: Teilenummer und Position des Moduls
- Einen Abschnitt mit der Modulidentifikation
- Einen Abschnitt pro Kanal mit allen kanalspezifischen Parametern

Der Ausdruck entspricht der Konfiguration: Unbedeutende, grau abgeblendete Informationen werden nicht gedruckt.

Kapitel 4
Serielle Modbus-Kommunikation für Modicon M340-Prozessoren

Inhalt dieses Kapitels

In diesem Kapitel wird das Verfahren der Softwareimplementierung für die serielle Modbus-Kommunikation für Modicon M340-Prozessoren beschrieben.

Inhalt dieses Kapitels

Dieses Kapitel enthält die folgenden Abschnitte:

Abschnitt 4.1
Allgemeine Informationen

Inhalt dieses Abschnitts

Dieser Abschnitt enthält die allgemeine Beschreibung der seriellen Modbus-Kommunikation und ihrer Dienste.

Inhalt dieses Abschnitts

Dieser Abschnitt enthält die folgenden Themen:

Über Modbus Serial

Einführung

Die Kommunikation über Modbus ermöglicht den Datenaustausch zwischen allen mit dem Bus verbundenen Geräten. Modbus seriell ist ein Protokoll, dass eine hierarchische Struktur (ein Master und mehrere Slaves) erstellt.

Der Master verwaltet den Austausch auf zwei Arten:

- Der Master tauscht Daten mit dem Slave aus und wartet auf die Antwort.
- Der Master tauscht Daten mit allen Slaves aus, ohne auf eine Antwort zu warten (Broadcast).

HINWEIS: Achten Sie darauf, dass zwei Master (auf demselben Bus) nicht gleichzeitig einen Request senden, denn dann gehen die Requests verloren und jede Rückmeldung enthält ein negatives Ergebnis, z. B. 16#0100 (Request konnte nicht verarbeitet werden) oder 16#ODFF (Slave nicht vorhanden).

⚠ WARNUNG

KRITISCHER DATENVERLUST

Verwenden Sie die Kommunikationsports ausschließlich für die nicht-kritische Datenübertragung.

Die Nichtbeachtung dieser Anweisungen kann Tod, schwere Verletzungen oder Sachschäden zur Folge haben.

Leistung

Einführung

Die folgenden Tabellen können zur Beurteilung der typischen Austauschzeiten bei der Modbus-Kommunikation nach verschiedenen Kriterien herangezogen werden.

Die angezeigten Ergebnisse entsprechen der durchschnittlichen Ausführungszeit der `READ_VAR`-Funktion in Millisekunden.

Definition der Austauschzeit

Die Austauschzeit ist die Zeit, die zwischen der Einrichtung eines Austauschs und dessen Ende verstreicht. Sie umfasst die Zeit der Kommunikation über die serielle Verbindung.

Der Austausch wird eingerichtet, sobald die Kommunikationsfunktion aufgerufen wird.

Er endet, wenn eines der folgenden Ereignisse auftritt:
- Daten werden empfangen.
- Eine Anomalität tritt auf.
- Der Timeout-Zeitraum läuft ab.

Austauschzeit für ein Wort

Die nachstehende Tabelle zeigt die Austauschzeiten für ein Wort bei der Modbus-Kommunikation mit einem Prozessor BMX P34 2020:

Austauschzeit in ms (der Modbus-Slave ist ein zyklischer BMX P34 1000)		**Zykluszeit in ms**		
		Zyklisch	**10**	**50**
Baudrate der Kommunikation in Bits pro Sekunde	**4800**	68	72	100
	9600	35	40	50
	19200	20	27	50
	38400	13	20	50

Die Austauschzeiten des Prozessors BMX P34 2000/2010/20102 entsprechen denjenigen des Prozessors BMX P34 2020. Die Austauschzeiten von BMX P34 1000 fallen um 10 % kürzer aus.

HINWEIS: Alle oben aufgeführten Austauschzeiten basieren auf Messwerten mit einer Genauigkeitsmarge von +/- 10 ms.

Austauschzeit für 100 Wörter

Die nachstehende Tabelle zeigt die Austauschzeiten für 100 Wörter bei der Modbus-Kommunikation mit einem Prozessor BMX P34 2020:

Austauschzeit in ms (der Modbus-Slave ist ein zyklischer BMX P34 1000)		**Zykluszeit in ms**		
		Zyklisch	**10**	**50**
Baudrate der Kommunikation in Bits pro Sekunde	**4800**	500	540	595
	9600	280	288	300
	19200	142	149	150
	38400	76	80	100

Die Austauschzeiten des Prozessors BMX P34 2000/2010/20102 entsprechen denjenigen des Prozessors BMX P34 2020. Die Austauschzeiten von BMX P34 1000 fallen um 10 % kürzer aus.

HINWEIS: Alle oben aufgeführten Austauschzeiten basieren auf Messwerten mit einer Genauigkeitsmarge von +/- 10 ms.

Zugriff auf die seriellen Verbindungsparameter

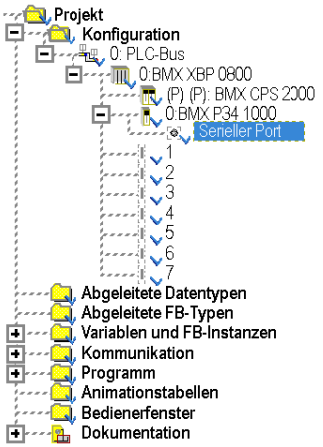
Einführung

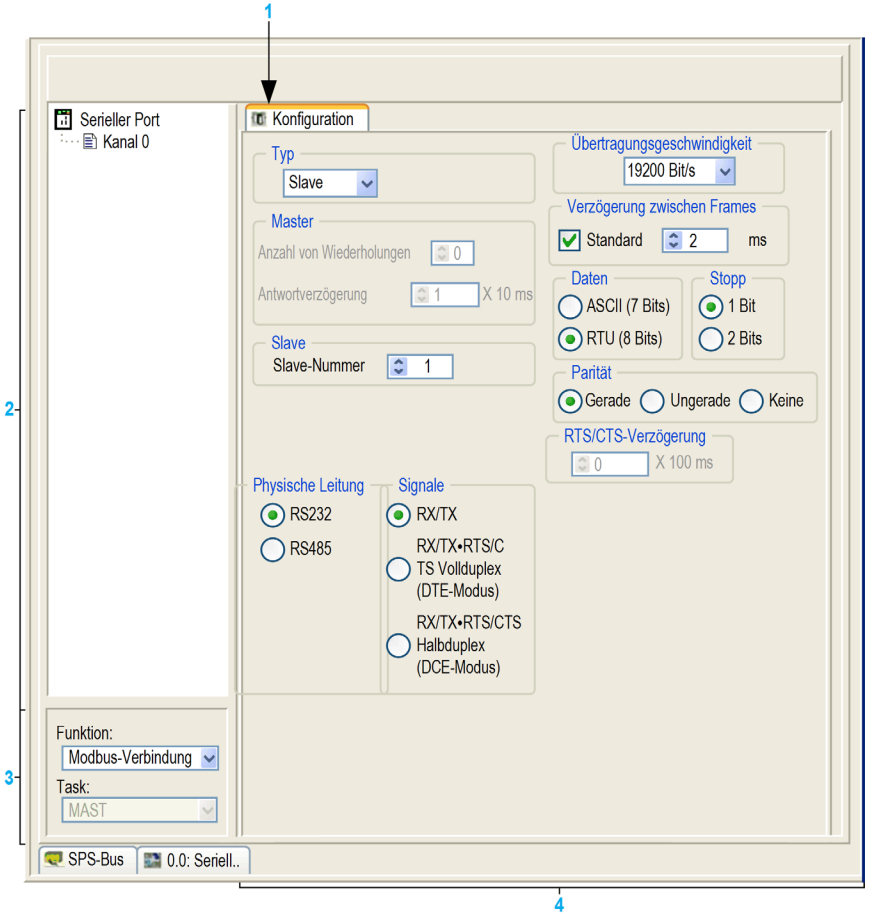
Auf den folgenden Seiten wird der Zugriff auf das Fenster zur Konfiguration des seriellen Ports für die nachstehenden Prozessoren beschrieben. Ferner werden die allgemeinen Elemente der Fenster zur Konfiguration und Fehlerbehebung für die Modbus- und Zeichenmodus-Verbindung vorgestellt:

- BMX P34 1000
- BMX P34 2000
- BMX P34 2010/20102
- BMX P34 2020

Zugriff auf die serielle Verbindung

In der folgenden Tabelle wird die Vorgehensweise für den Zugriff auf die serielle Verbindung beschrieben:

Schritt	Aktion
1	Öffnen Sie im Projektbrowser das folgende Verzeichnis: *Projekt\Konfiguration\0: PLC bus\0: BMX XBP ••••\0: BMX P34 ••••\SerialPort*. **Ergebnis**: Das folgende Fenster wird angezeigt: 

Schritt	Aktion
2	Doppelklicken Sie auf das Unterverzeichnis des seriellen Ports (SerialPort). **Ergebnis**: Das Konfigurationsfenster wird angezeigt: 

Beschreibung des Konfigurationsfensters

In der folgenden Tabelle werden die verschiedenen Elemente des Konfigurationsfensters aufgeführt:

Adresse	Element	Funktion
1	Registerkarten	Auf der im Vordergrund angezeigten Registerkarte wird der aktuelle Modus angegeben. Jeder Modus kann über die entsprechende Registerkarte ausgewählt werden. Folgende Modi sind verfügbar: ● Konfiguration ● Debug-Fenster (Zugriff nur im Online-Modus)
2	Kanalbereich	Ermöglicht Folgendes: ● Wählen Sie zwischen seriellem Port und Kanal 0, indem Sie sie jeweils anklicken. ● Durch Klicken auf den seriellen Port werden folgende Registerkarten angezeigt: ❍ Die Registerkarte „Beschreibung“, die die Merkmale des Geräts enthält. ❍ Die Registerkarte **E/A-Objekte** *(siehe EcoStruxure™ Control Expert, Betriebsarten)*, die der Vorsymbolisierung der Eingangs-/Ausgangsobjekte dient. ● Durch Klicken auf den Kanal werden folgende Registerkarten angezeigt: ❍ Konfiguration ❍ Debuggen ● Anzeige des vom Benutzer im Variableneditor festgelegten Namens und Symbols des Kanals.
3	Bereich der allgemeinen Parameter	Dieser Bereich ermöglicht die Auswahl der allgemeinen Parameter für den Kanal: ● Funktion: Die verfügbaren Funktionen sind Modbus-Verbindung und Zeichenmodus. Die Standardkonfiguration ist die Modbus-Funktion. ● Task: Legt die -Master-Task fest, in der die impliziten Austauschobjekte des Kanals ausgetauscht werden. Dieser Bereich erscheint grau abgeblendet und kann daher nicht konfiguriert werden.
4	Konfigurations- oder Debug-Bereich	Dieser Bereich dient im Konfigurationsmodus der Konfiguration der Kanalparameter. Im Debug-Modus wird er zum Debuggen des Kommunikationskanals verwendet.

Abschnitt 4.2
Konfiguration einer seriellen Modbus-Kommunikation

Inhalt dieses Abschnitts

In diesem Abschnitt wird die Softwarekonfiguration für die serielle Modbus-Kommunikation beschrieben.

Inhalt dieses Abschnitts

Dieser Abschnitt enthält die folgenden Themen:

Konfigurationsfenster für die serielle Modbus-Kommunikation

Allgemein

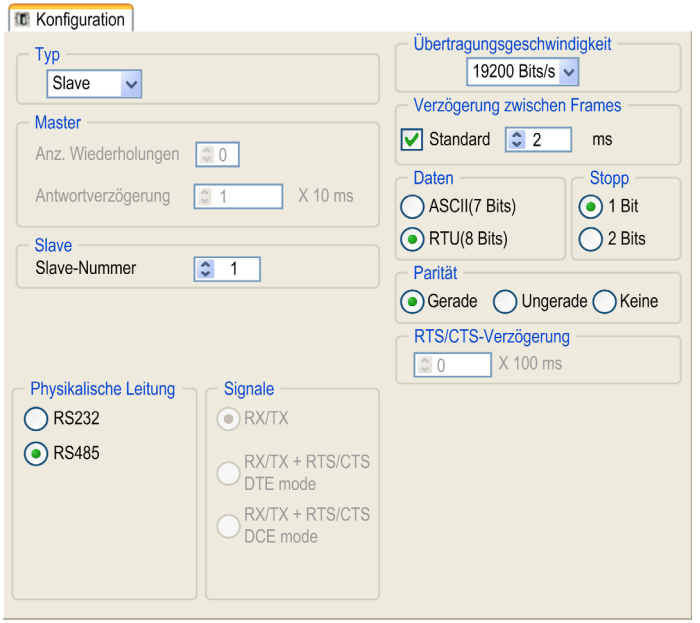
Die folgenden Seiten geben eine Einführung in das Konfigurationsfenster für die serielle Modbus-Kommunikation.

Zugriff auf das Konfigurationsfenster

Um das Konfigurationsfenster für die serielle Modbus-Kommunikation anzuzeigen, öffnen Sie das Verzeichnis „Serieller Port“ im Projekt-Browser *(siehe Seite 54)*.

Konfigurationsfenster für die serielle Modbus-Kommunikation

Die folgende Abbildung zeigt das Standardkonfigurationsfenster für die serielle Modbus-Kommunikation:



Beschreibung

Diese Bereiche dienen der Konfiguration der Kanalparameter. Der Zugriff auf diese Bereiche ist im Online-Modus möglich. Im Offline-Modus kann zwar auf diese Bereiche, jedoch möglicherweise nicht auf einige Parameter zugegriffen werden. Diese werden deshalb grau abgeblendet angezeigt.

Die folgende Tabelle zeigt die verschiedenen Bereiche des Konfigurationsfensters der Modbus-Verbindung:

Element	Kommentar
Parameter der Anwendung *(siehe Seite 61)*	Diese Parameter sind in drei verschiedenen Bereichen verfügbar: ● Typ ● Master ● Slave
Parameter der Signal- und physischen Leitung *(siehe Seite 63)*	Diese Parameter sind in drei verschiedenen Bereichen verfügbar: ● Physische Leitung ● Signale ● RTS/CTS-Verzögerung
Parameter der Übertragung *(siehe Seite 65)*	Diese Parameter sind in fünf verschiedenen Bereichen verfügbar: ● Übertragungsgeschwindigkeit ● Verzögerung zwischen Frames ● Datenbits ● Stoppbits ● Parität

HINWEIS: Bei der Konfiguration der seriellen Modbus-Kommunikation im Master-Modus wird der Bereich **Slave** grau abgeblendet dargestellt und kann nicht geändert werden - und umgekehrt.

Standardwerte

Die folgende Tabelle zeigt die Standardwerte der Parameter für die serielle Modbus-Kommunikation:

Konfigurationsparameter		**Wert**
Parameter der Anwendung	**Typ**	Slave
	Slave-Nummer	1
Parameter der Signal- und physischen Leitung	**Physische Leitung**	RS485
	Signale	RX/TX
Parameter der Übertragung	**Übertragungsgeschwindigkeit**	19200 Bit/s
	Verzögerung zwischen Frames	2 ms
	Daten	RTU (8 Bits)
	Stopp	1 Bit
	Parität	Gerade

Anwendungsrelevante Modbus-Parameter

Auf einen Blick

Nach der Konfiguration des Kommunikationskanals müssen Sie die anwendungsspezifischen Parameter eingeben.

Auf diese Parameter kann von drei Konfigurationszonen aus zugegriffen werden:

- Zone „Typ“
- Zone „Master“
- Zone „Slave“

Zone „Typ“

Diese Konfigurationszone wird wie folgt auf dem Bildschirm angezeigt:

Diese Zone ermöglicht die Auswahl des zu verwendenden seriellen Modbus:

- **Master**: Wenn die jeweilige Station Master ist.
- **Slave**: Wenn die jeweilige Station Slave ist.

Zone „Master“

Auf die unten abgebildete Konfigurationszone kann nur zugegriffen werden, wenn in der Zone „Typ“ der Wert „Master“ ausgewählt ist:

Diese Zone ermöglicht die Eingabe folgender Parameter:

- **Anzahl von Wiederholungen**: Anzahl der Verbindungsaufbauversuche, die der Master unternimmt, bevor er den Slave als nicht vorhanden betrachtet.
  Der Standardwert ist 3.
  Die möglichen Werte liegen zwischen 0 und 15.
  Der Wert 0 gibt an, dass der Master über keine Wiederholschleife verfügt.
- **Antwortverzögerung**: Die Zeit, die zwischen dem Versand der Anforderung durch den Master und dem Beginn der Wiederholschleife vergeht, wenn der Slave nicht antwortet. Sie entspricht der maximalen Zeit zwischen dem Versand des letzten Zeichens der vom Master versandten Anforderung und dem Empfang des ersten Zeichens der vom Slave zurückgesandten Anforderung.
  Der Standardwert ist 1 Sekunde (100 * 10 ms).
  Die möglichen Werte liegen zwischen 10 ms und 10 Sekunden.

HINWEIS: Die Antwortverzögerung des Masters muss mindestens der längsten Antwortverzögerung der am Bus vorhandenen Slaves entsprechen.

Zone „Slave“

Auf die unten abgebildete Konfigurationszone kann nur zugegriffen werden, wenn in der Zone „Typ“ der Wert „Slave“ ausgewählt ist:

Diese Zone ermöglicht die Eingabe der Slave-Nummer des Prozessors.

Der Standardwert ist 1.

Die möglichen Werte liegen zwischen 1 und 247.

HINWEIS: In einer Modbus-Slave-Konfiguration kann eine zusätzliche Adresse (Nummer 248) für eine serielle Punkt-zu-Punkt-Kommunikation verwendet werden.

Parameter der Signal- und der physischen Leitung im Modbus-Modus

Einführung

Die Parameter der Signal- und der physischen Leitung sind in drei verschiedenen Bereichen verfügbar:

- Bereich **Physische Leitung**
- Bereich**Signale**
- Bereich **RTS/CTS-Verzögerung**

Bereich Physische Leitung

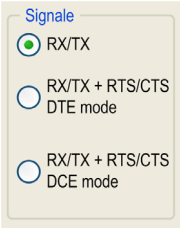
Dieser Konfigurationsbereich wird wie folgt auf dem Bildschirm angezeigt:

In diesem Bereich stehen zwei physische Leitungstypen für den seriellen Port der Prozessoren BMX P34 1000/2000/2010/20102/2020 zur Auswahl:

- RS232-Leitung
- RS485-Leitung

BereichSignale

Dieser Konfigurationsbereich wird wie folgt auf dem Bildschirm angezeigt:



In diesem Bereich können Sie die von der physischen RS232-Leitung unterstützten Signale auswählen:

- **RX/TX**
- **RX/TX + RTS/CTS DTE-Modus**
- **RX/TX + RTS/CTS DCE-Modus**

Wenn RS485 konfiguriert ist, wird der gesamte Bereich grau unterlegt (ausgeblendet) und der Standardwert ist **RX/TX**.

Bereich RTS/CTS-Verzögerung

Dieser Konfigurationsbereich wird wie folgt auf dem Bildschirm angezeigt:

Der Bereich der RTS/CTS-Verzögerung wird nur aktiviert, wenn sowohl RS232 als auch RX/TX+RTS/CTS ausgewählt sind. Ein Algorithmus zur RTS/CTS-Datenflusssteuerung wird ausgewählt, wenn der Standardwert 0 ms ist. Ein anderer Wert als 0 aktiviert einen Algorithmus zur RTS/CTS-Modemsteuerung.

Der Algorithmus zur RTS/CTS-Datenflusssteuerung (DTE <-> DTE) unterscheidet sich wie folgt vom Algorithmus zur RTS/CTS-Modemsteuerung (DTE <-> DCE):

- Der Algorithmus zur RTS/CTS-Datenflusssteuerung steht in Beziehung zum Empfangspuffer bei Überlauf (Vollduplex).
- Der Algorithmus zur RTS/CTS-Modemsteuerng befasst sich mit dem geteilten Übermittlungsprozess, beispielsweise ein Funkmodem.

Algorithmus zur RTS/CTS-Datenflusssteuerung

Ziel ist die Verhinderung eines Empfangspuffer-Überlaufs.

Das RTS-Ausgangssignal jedes Geräts ist mit dem CTS-Eingangssignal eines anderen Geräts verbunden. Das Sendegerät (M340) ist berechtigt, Daten zu übertragen, sobald es das RTS-Eingangssignal (z. B. ein anderes M340-Gerät) an seinem CTS-Eingang empfängt. Der Algorithmus ist symmetrisch und ermöglicht eine asynchrone Vollduplex-Kommunikation.

Algorithmus zur RTS/CTS-Modemsteuerung

Vor der Übertragung eines Request aktiviert der Sender (M340) das RTS-Signal und wartet auf die Auslösung des CTS-Signals durch das Modem. Wenn CTS nach der RTS/CTS-Verzögerung nicht aktiviert wird, wird der Request verworfen.

Übertragungsbezogene Modbus-Parameter

Einführung

Die Übertragungsparameter sind in fünf verschiedenen Bereichen verfügbar:

- Bereich **Übertragungsgeschwindigkeit**
- Bereich **Verzögerung zwischen Frames**
- Bereich **Daten**
- Bereich **Stopp**
- Bereich **Parität**

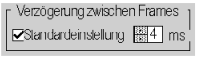
Bereich Übertragungsgeschwindigkeit

Dieser Konfigurationsbereich wird wie folgt auf dem Bildschirm angezeigt:

Sie können hier die Übertragungsgeschwindigkeit der seriellen Modbus-Kommunikation auswählen. Die ausgewählte Geschwindigkeit muss mit den anderen Geräten übereinstimmen. Konfigurierbare Werte: 600, 1200, 2400, 4800, 9600, 19200 (Standard) und 38400 Bit pro Sekunde.

Bereich Verzögerung zwischen Frames

Dieser Konfigurationsbereich wird wie folgt auf dem Bildschirm angezeigt:



Die **Verzögerung zwischen Frames** ist der Mindestzeitraum zwischen zwei Frames beim Empfang. Diese Verzögerung wird verwaltet, wenn die SPS (Master oder Slave) Nachrichten empfängt.

HINWEIS: Der Standardwert ist von der ausgewählten Übertragungsgeschwindigkeit abhängig.

HINWEIS: Die Verzögerung zwischen Frames sollte der Standardwert sein, damit die Kompatibilität mit Modbus gewährleistet werden kann. Wenn ein Slave nicht konform ist, kann der Wert geändert werden und muss dann derselbe für den Master und alle Slaves auf dem Bus sein.

Bereich Daten

Dieser Konfigurationsbereich wird wie folgt auf dem Bildschirm angezeigt:

Dieser Bereich ermöglicht Ihnen die Eingabe des für die serielle Modbus-Kommunikation verwendeten Codierungstyp. Das Feld wird in Übereinstimmung mit den anderen mit dem Bus verbundenen Geräten eingestellt. Zwei konfigurierbare Modi stehen zur Auswahl:

- RTU-Modus:
  - Die Zeichen sind über 8 Bit kodiert.
  - Das Ende des Frames wird erkannt, wenn für mindestens 3,5 Zeichen Stille herrscht.
  - Die Integrität des Frames wird mithilfe eines Worts überprüft, das als CRC-Prüfsumme bezeichnet wird und im Frame enthalten ist.
- ASCII-Modus:
  - Die Zeichen sind über 7 Bit kodiert.
  - Der Anfang des Frames wird erkannt, wenn das Zeichen „**:**“ empfangen wird.
  - Das Ende des Frames wird mittels Wagenrücklauf und Zeilenvorschub erkannt.
  - Die Integrität des Frames wird mithilfe eines Worts überprüft, das als LRC-Prüfsumme bezeichnet wird und im Frame enthalten ist.

Bereich Stopp

Dieser Konfigurationsbereich wird wie folgt auf dem Bildschirm angezeigt:

Der Stopp-Bereich ermöglicht die Eingabe der für die Kommunikation verwendeten Anzahl an Stoppbits. Das Feld wird in Übereinstimmung mit den anderen Geräten eingestellt. Konfigurierbare Werte:

- 1 Bit
- 2 Bit

Bereich Parität

Dieser Konfigurationsbereich wird wie folgt auf dem Bildschirm angezeigt:

Verwenden Sie diesen Bereich, um anzugeben, ob ein Paritätsbit hinzugefügt wird oder nicht. Hier können Sie auch den Typ des Paritätsbits festlegen. Das Feld wird in Übereinstimmung mit den anderen Geräten eingestellt. Konfigurierbare Werte:

- **Gerade**
- **Ungerade**
- **Ohne**

Abschnitt 4.3
Programmieren der seriellen Modbus-Kommunikation

Inhalt dieses Abschnitts

In diesem Abschnitt wird das Programmierverfahren für die Implementierung der seriellen Modbus-Kommunikation beschrieben.

Inhalt dieses Abschnitts

Dieser Abschnitt enthält die folgenden Themen:

Von einem Master-Prozessor über eine Modbus-Verbindung unterstützte Dienste

Kommunikationsfunktionen

Drei spezifische Kommunikationsfunktionen sind für das Senden und den Empfang von Daten über einen Modbus-Kommunikationskanal definiert:

- READ_VAR: Lesen von Variablen
- WRITE_VAR: Schreiben von Variablen
- DATA_EXCH: Senden von Modbus-Requests an ein anderes Gerät über das ausgewählte Protokoll

Detaillierte Informationen zu diesen Kommunikationsfunktionen finden Sie im Kapitel *Allgemeine Informationen zu den M340-Kommunikationsfunktionen (siehe EcoStruxure™ Control Expert, Kommunikation, Bausteinbibliothek)*.

Datenaustausch

Das Lesen oder Schreiben von Variablen erfolgt über die Adressierung der folgenden Requests an das ausgewählte Slave-Gerät:

Diese Requests greifen auf die Kommunikationsfunktionen READ_VAR, WRITE_VAR und DATA_EXCH zurück:

Modbus-Request	Funktionscode	Kommunikationsfunktion
Lesen von Bits	16#01 oder 16#02	READ_VAR
Lesen von Wörtern	16#03 oder 16#04	READ_VAR
Schreiben von Bits	16#0F	WRITE_VAR
Schreiben von Wörtern	16#10	WRITE_VAR
Andere Requests	Alle	DATA_EXCH

HINWEIS: WRITE_VAR kann im Broadcast-Modus verwendet werden (READ_VAR nicht). In diesem Fall erhält die SPS keine Antwort. Beim Senden eines Broadcast-Requests wird das Aktivitätsbit zurückgesetzt und der Code 16#01 (Stopp des Austauschs bei Timeout) wird an das zweite EF-Verwaltungswort zurückgegeben.

HINWEIS: Die von der Modicon M340-SPS gelesenen Objekte können vom Typ %I und %IW sein. In diesem Fall gibt die Funktion READ_VAR einen Modbus-Request aus: FC 0x2 oder 0x4. Bei einer Quantum-SPS erlaubt sie den Zugriff auf den Eingangsstatus oder das Eingangsstatusregister. Allgemeiner formuliert ist es möglich, über die Kommunikationsfunktion DATA_EXCH beliebige Modbus-Requests an ein Slave-Gerät zu senden.

Abbruch eines Austauschs

Für den Abbruch des von den Kommunikationsfunktionen ausgeführten Austauschs sind zwei Programmiermöglichkeiten gegeben:

- Verwendung der Funktion `CANCEL`
- Verwendung des Abbruchbits der Kommunikationsfunktion

Detaillierte Informationen zum Abbruch einer Kommunikationsfunktion finden Sie im Handbuch *EcoStruxure™ Control Expert, Kommunikation, Bausteinbibliothek*.

Von einem Slave-Prozessor in einer Modbus-Verbindung unterstützte Dienste

Auf einen Blick

Bei einer Verwendung als Slave-Prozessor in einer Modbus-Verbindung unterstützen die nachstehenden Prozessoren mehrere Dienste:

- BMX P34 1000,
- BMX P34 2000,
- BMX P34 2010/20102,
- BMX P34 2020.

Datenaustausch

Ein Slave-Prozessor verwaltet die folgenden Anforderungen:

Modbus-Anforderung	Funktionscode	Steuerungsobjekt
Lesen von "n" Ausgangsbits	16#01	%M
Lesen von „n“ Eingangsbits	16#02	%M
Lesen von "n" Ausgangswörtern	16#03	%MW
Lesen von „n“ Eingangswörtern	16#04	%MW
Schreiben von "n" Ausgangsbits	16#05	%M
Schreiben von „n“ Ausgangswort	16#06	%MW
Schreiben von "n" Ausgangsbits	16#0F	%M
Schreiben von "n" Ausgangswörtern	16#10	%MW

Diagnose und Wartung

Im Folgenden werden die über eine Modbus-Verbindung verfügbaren Diagnose- und Wartungsinformationen aufgeführt:

Bezeichnung	Funktionscode/Unterfunktionscode
Echo	16#08 / 16#00
Lesen der Diagnoseregister der Steuerung	16#08 / 16#02
Zurücksetzen der SPS-Diagnoseregister und -Zähler auf 0	16#08 / 16#0A
Lesen der Anzahl von Nachrichten auf dem Bus	16#08 / 16#0B
Lesen der Anzahl der auf dem Bus erkannten Kommunikationsfehler	16#08 / 16#0C
Lesen der Anzahl der auf dem Bus erkannten Ausnahmefehler	16#08 / 16#0D
Anzahl der vom Slave empfangenen Nachrichten lesen	16#08 / 16#0E
Anzahl der fehlenden Antworten vom Slave lesen	16#08 / 16#0F
Lesen der Anzahl von negativen Bestätigungen vom Slave	16#08 / 16#10
Lesen der Anzahl von Ausnahmeantworten vom Slave	16#08 / 16#11
Lesen der Anzahl von überlaufenden Zeichen auf dem Bus	16#08 / 16#12
Lesen des Ereigniszählers	16#0B
Lesen des Verbindungsereignisses	16#0C
Leseidentifikation	16#11
Auslesen der Geräteinformation	16#2B / 16#0E

Abschnitt 4.4
Debuggen einer seriellen Modbus-Kommunikation

Debug-Fenster der seriellen Modbus-Kommunikation

Allgemein

Der Zugriff auf das Debug-Fenster der seriellen Modbus-Kommunikation ist ausschließlich im Online-Modus möglich.

Zugriff auf das Debug-Fenster

Die folgende Tabelle beschreibt die Vorgehensweise zum Zugreifen auf das Debug-Fenster der seriellen Modbus-Kommunikation:

Schritt	Aktion
1	Rufen Sie das Konfigurationsfenster der seriellen Modbus-Kommunikation auf. *(siehe Seite 58)*
2	Wählen Sie auf dem angezeigten Fenster die Registerkarte "Debuggen" aus.

Beschreibung des Debug-Fensters

Das Debug-Fenster ist in zwei Bereiche aufgeteilt:

- Der Bereich "Typ",
- Der Bereich "Zähler".

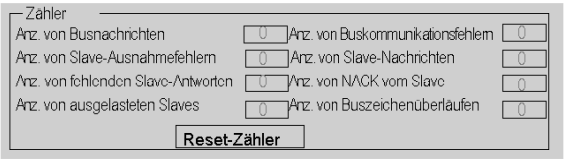
Bereich "Typ"

Dieser Bereich sieht wie folgt aus:

Er gibt den Typ der konfigurierten Modbus-Funktion an (in diesem Fall "Master").

Bereich "Zähler"

Dieser Bereich sieht wie folgt aus:



Dieser Bereich enthält die verschiedenen Debugging-Zähler.

Die Schaltfläche "Zähler zurücksetzen" setzt alle Zähler des Debug-Modus auf Null zurück.

Arbeitsweise der Zähler

Die Debugging-Zähler für die serielle Modbus-Kommunikation sind:

- Busnachrichtenzähler: Dieser Zähler gibt die Anzahl der Nachrichten an, die der Prozessor auf der seriellen Verbindung ermittelt hat. Nachrichten mit einem negativen CRC-Prüfergebnis werden nicht gezählt.
- Zähler für Buskommunikationsfehler: Dieser Zähler gibt die Anzahl der negativen CRC-Prüfergebnisse an, die vom Prozessor gezählt werde. Falls ein Zeichenfehler (Überlauf, Paritätsfehler) auftritt oder die Nachricht weniger als 3 Byte umfasst, kann das System, das die Daten empfängt, die CRC-Prüfung nicht durchführen. In solchen Fällen wird der Zähler entsprechend erhöht.
- Zähler für Slave-Ausnahmefehler: Dieser Zähler gibt die Anzahl der Modbus-Ausnahmefehler an, die vom Prozessor ermittelt werden.
- Slave-Nachrichtenzähler: Dieser Zähler gibt die Anzahl der von der Modbus-Verbindung empfangenen und verarbeiteten Nachrichten an.
- Zähler für fehlende Slave-Antworten: Dieser Zähler gibt die Anzahl der vom dezentralen System gesendeten Nachrichten an, für die keine Antwort empfangen wurde (weder normale noch Ausnahmeantwort). Es wird auch die Anzahl der im Broadcast-Modus empfangenen Nachrichten gezählt.
- Zähler für negative Slave-Bestätigungen: Dieser Zähler gibt die Anzahl der an das dezentrale System gesendeten Nachrichten an, für die es eine negative Bestätigung ausgegeben hat.
- Slave-Ausgelastet-Zähler: Dieser Zähler gibt die Anzahl der an das dezentrale System gesendeten Nachrichten an, für die es eine Ausnahmenachricht ausgegeben hat, dass der Slave ausgelastet ist.
- Buszeichenüberlauf-Zähler: Dieser Zähler gibt die Anzahl der an den Prozessor gesendeten Nachrichten an, dass er wegen eines Zeichenüberlaufs auf dem Bus keine Erfassung vornehmen kann. Ein Überlauf kann folgende Ursachen haben.
  - Daten vom Typ "Zeichen" werden auf dem seriellen Port schneller übertragen, als sie gespeichert werden können.
  - Ein Datenverlust aufgrund eines Hardwarefehlers.

HINWEIS: Alle Zähler beginnen die Zählung beim letzten Neustart, nach einer Zählerlöschung bzw. beim Prozessorstart.

Kapitel 5
Zeichenmodus-Kommunikation für Modicon M340-Prozessoren

Inhalt dieses Kapitels

In diesem Kapitel wird die Softwareimplementierung für die Kommunikation im Zeichenmodus für Modicon M340-Prozessoren beschrieben.

Inhalt dieses Kapitels

Dieses Kapitel enthält die folgenden Abschnitte:

Abschnitt 5.1
Allgemeine Informationen

Inhalt dieses Abschnitts

Dieser Abschnitt bietet einen Überblick über die allgemeinen Gesichtspunkte der Kommunikation im Zeichenmodus und ihrer Dienste.

Inhalt dieses Abschnitts

Dieser Abschnitt enthält die folgenden Themen:

Informationen zur Kommunikation im Zeichenmodus

Einführung

Bei der Kommunikation im Zeichenmodus können Dialog- und Kommunikationsfunktionen mit folgenden Geräten ausgeführt werden:

- Herkömmliche Peripheriegeräte (Drucker, Tastaturbildschirm, Endgeräte in Werkstätten usw.)
- Spezielle Peripheriegeräte (Strichcode-Lesegeräte usw.)
- Rechner (Prüfung, Produktionsmanagement usw.)
- Heterogene Geräte (numerische Befehle, Drehzahlregler usw.)
- Externe Modems

⚠ WARNUNG
KRITISCHER DATENVERLUST Verwenden Sie die Kommunikationsports ausschließlich für die nicht-kritische Datenübertragung. **Die Nichtbeachtung dieser Anweisungen kann Tod, schwere Verletzungen oder Sachschäden zur Folge haben.**

Leistung

Einführung

In der folgenden Tabelle werden die typischen Austauschzeiten im Zeichenmodus beschrieben.

Die angezeigten Ergebnisse entsprechen der durchschnittlichen Ausführungszeit der `PRINT_CHAR`-Funktion in Millisekunden.

Definition der Austauschzeit

Die Austauschzeit ist die Zeit zwischen Einrichtung und Ende eines Austauschs. Sie umfasst die Zeit der Kommunikation über die serielle Verbindung.

Der Austausch wird eingerichtet, sobald die Kommunikationsfunktion aufgerufen wird.

Er endet, wenn eines der folgenden Ereignisse auftritt:
- Empfang von Daten
- Anomalie
- Ablauf des Timeouts

Werte der Austauschzeit

Die folgende Tabelle enthält die Austauschzeiten für die Übertragung von 80 Zeichen im Zeichenmodus mit einem Prozessor BMX P34 2020 bei unterschiedlichen Baudraten und Zykluszeiten:

Austauschzeit in ms		**Zykluszeit in ms**				
		10	**20**	**50**	**100**	**255**
Baudrate der Kommunikation in Bits pro Sekunde	**1200**	805	820	850	900	980
	4800	210	220	250	300	425
	9600	110	115	145	200	305
	19200	55	60	95	100	250

Die Austauschzeiten des Prozessors BMX P34 2000/2010/20102 entsprechen denjenigen des Prozessors BMX P34 2020. Die Austauschzeiten von BMX P34 1000 fallen um 10 % kürzer aus.

HINWEIS: Alle oben aufgeführten Austauschzeiten basieren auf Messwerten mit einer Genauigkeitsmarge von +/- 10 ms.

Abschnitt 5.2
Konfiguration der Zeichenmoduskommunikation

Inhalt dieses Abschnitts

In diesem Abschnitt wird das Konfigurationsverfahren für die Implementierung der Zeichenmoduskommunikation beschrieben.

Inhalt dieses Abschnitts

Dieser Abschnitt enthält die folgenden Themen:

Konfigurationsfenster für die Zeichenmodus-Kommunikation

Allgemein

Die folgenden Seiten geben eine Einführung in das Konfigurationsfenster für die Zeichenmodus-Kommunikation.

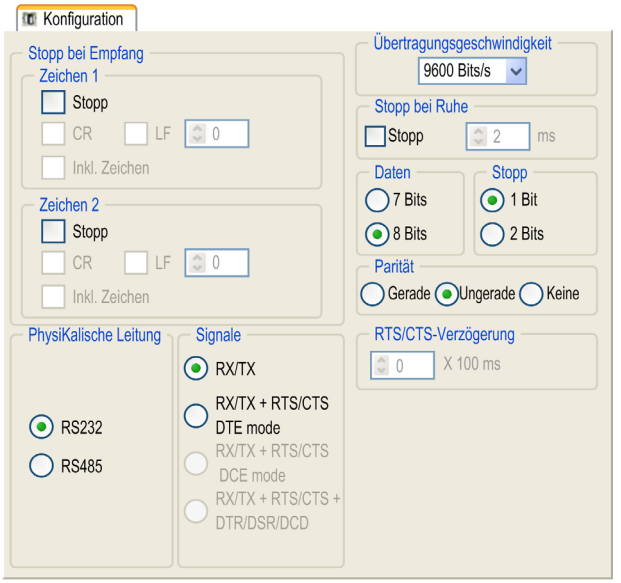
Zugriff auf das Konfigurationsfenster

Die folgende Tabelle beschreibt die Vorgehensweise zum Zugriff auf das Konfigurationsfenster für die Zeichenmodus-Kommunikation:

Schritt	Aktion
1	Öffen Sie das Unterverzeichnis des seriellen Ports im Projektbrowser *(siehe Seite 54)*.
2	Wählen Sie im daraufhin angezeigten Fenster die Option **Zeichenmodus-Verbindung** im Feld **Funktion** aus.

Konfigurationsfenster für die Zeichenmodus-Kommunikation

Die folgende Abbildung zeigt das Standardkonfigurationsfenster für die Zeichenmodus-Kommunikation:



Beschreibung

Diese Bereiche dienen der Konfiguration der Kanalparameter. Der Zugriff auf diese Bereiche ist im Online-Modus möglich. Im Offline-Modus kann zwar auf diese Bereiche, jedoch möglicherweise nicht auf einige Parameter zugegriffen werden. Diese werden deshalb grau abgeblendet.

Die folgende Tabelle zeigt der verschiedenen Bereiche des Konfigurationsfensters der Zeichenmodus-Kommunikation:

Element	Kommentar
Parameter zur Erkennung des Nachrichtenendes *(siehe Seite 82)*	Diese Parameter sind in zwei verschiedenen Bereichen verfügbar: ● Stopp bei Empfang ● Stopp bei Stille
Parameter der Signal- und physischen Leitung *(siehe Seite 84)*	Diese Parameter sind in drei verschiedenen Bereichen verfügbar: ● Physische Leitung ● Signale ● RTS/CTS-Verzögerung
Parameter der Übertragung *(siehe Seite 86)*	Diese Parameter sind in vier verschiedenen Bereichen verfügbar: ● Übertragungsgeschwindigkeit ● Daten ● Stoppbits ● Parität

Standardwerte

Die folgende Tabelle zeigt die Standardwerte der Parameter für die Zeichenmodus-Kommunikation:

Konfigurationsparameter		Wert
Parameter zur Erkennung des Nachrichtenendes	**Stopp bei Empfang**	Keine
	Stopp bei Stille	Keine
Parameter der Signal- und physischen Leitung	**Physische Leitung**	RS232
	Signale	RX/TX
Parameter der Übertragung	**Übertragungsgeschwindigkeit**	9600 Bit/s
	Daten	8 Bit
	Stopp	1 Bit
	Parität	Ungerade

Parameter zur Erkennung des Nachrichtenendes im Zeichenmodus

Einführung

Die Parameter zur Erkennung des Nachrichtenendes sind über zwei Bereiche zugänglich:

- Bereich **Stopp bei Empfang**: Stopp bei Empfang eines Sonderzeichens.
- Bereich **Stopp bei Stille**: Stopp bei Stille.

Verwendungsbedingungen

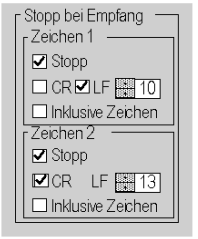
Die Auswahl von **Stopp bei Stille** bedeutet, dass die Auswahl von **Stopp bei Empfang** aufgehoben wird und umgekehrt.

HINWEIS:
Um einen Kanal im Zeichenmodus ohne Stoppparameter zu konfigurieren, müssen die **Stopp**-Kontrollkästchen für folgende Konfigurationsbereiche deaktiviert werden:

- **Stopp bei Empfang → Zeichen 1**
- **Stopp bei Empfang → Zeichen 2**
- **Stopp bei Stille**

Bereich Stopp bei Empfang

Dieser Konfigurationsbereich wird wie folgt im Fenster angezeigt:



Ein Empfangsrequest kann bei Empfang eines Sonderzeichens beendet werden.

Durch die Auswahl der Option **Stop** kann ein **Stopp bei Empfang** mit einem bestimmten Nachrichtenendzeichen aktiviert werden:

- **CR**: Das Nachrichtenende wird anhand eines Wagenrücklaufs erkannt.
- **LF**: Das Nachrichtenende wird anhand eines Zeilenvorschubs erkannt.
- Dateneingabefeld: Hier können Sie ein anderes Nachrichtenendzeichen als einen Wagenrücklauf oder Zeilenvorschub in Form eines Dezimalwerts eingeben:
  - Zwischen 0 und 255, wenn die Daten über 8 Bits codiert sind.
  - Zwischen 0 und 127, wenn die Daten über 7 Bits codiert sind.
- Zeichen enthalten: Das Nachrichtenendzeichen wird in die Empfangstabelle der SPS-Anwendung aufgenommen.

Es besteht die Möglichkeit, zwei Nachrichtenendzeichen zu konfigurieren. In obigem Fenster wird das Ende des Nachrichtenempfangs durch einen Zeilenvorschub oder Wagenrücklauf erkannt.

Bereich Stopp bei Stille

Dieser Konfigurationsbereich wird wie folgt im Fenster angezeigt:

Er ermöglicht die Erkennung des Nachrichtenendes beim Empfang anhand des Fehlens eines Nachrichtenendzeichens während eines bestimmten Zeitraums.

Stopp bei Stille wird durch Aktivierung des **Stop**-Kontrollkästchens ausgewählt. Die Dauer der Stille (ausgedrückt in Millisekunden) wird im Dateneingabefeld festgelegt.

Der minimale Wert dieser Dauer ist die Zeit, die der Übertragung von 1,5 Zeichen entspricht. In einer Anzahl von Bits und in Abhängigkeit von der Konfiguration der Start- und Stopp-Bits gilt für die minimale Dauer der Stille:

Zeichenlänge insg. (Bit)	Min. Dauer der Stille (Bit)
8	12
9	12
10	15
11	15

Konvertieren Sie die Zahl in der rechten Spalte gemäß der konfigurierten Übertragungsgeschwindigkeit in eine Dauer.

HINWEIS: Der zulässige Wertebereich reicht von 1 ms bis 10000 ms und ist von der jeweils ausgewählten Übertragungsgeschwindigkeit abhängig.

Parameter der Signal- und der physischen Leitung im Zeichenmodus

Einführung

Die Parameter der Signal- und der physischen Leitung sind in drei verschiedenen Bereichen verfügbar:

- Bereich **Physische Leitung**
- Bereich**Signale**
- Bereich **RTS/CTS-Verzögerung**

Bereich Physische Leitung

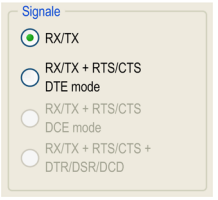
Dieser Konfigurationsbereich wird wie folgt auf dem Bildschirm angezeigt:

In diesem Bereich stehen zwei physische Leitungstypen für den seriellen Port der Prozessoren BMX P34 1000/2000/2010/20102/2020 zur Auswahl:

- RS232-Leitung
- RS485-Leitung

Bereich Signale

Dieser Konfigurationsbereich wird wie folgt auf dem Bildschirm angezeigt:



In diesem Bereich können Sie die von der physischen RS232-Leitung unterstützten Signale auswählen:

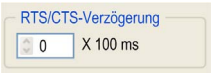
- **RX/TX**
- **RX/TX + RTS/CTS DTE-Modus**

Wenn RS485 konfiguriert ist, wird der gesamte Bereich grau unterlegt (ausgeblendet) und der Standardwert ist **RX/TX**.

HINWEIS: Bei der Konfiguration des seriellen Ports im Zeichenmodus stehen nur die Signale **RX/TX** und **RX/TX + RTS/CTS DTE-Modus** zur Auswahl.

Bereich RTS/CTS-Verzögerung

Dieser Konfigurationsbereich wird wie folgt auf dem Bildschirm angezeigt:



Der Bereich **RTS/CTS-Verzögerung** ist nur verfügbar, wenn die zwei Kontrollkästchen RS232 und **RX/TX+RTS/CTS DTE-Modus** aktiviert sind.

Ein Algorithmus zur RTS/CTS-Datenflusssteuerung ist ausgewählt: Vor der Übertragung der Zeichenfolge wartet das System auf die Aktivierung des CTS-Signals (Clear To Send). Dieser Bereich ermöglicht Ihnen die Eingabe der maximalen Wartezeit zwischen zwei Signalen. Bei Ablauf dieser Zeit wird der Request nicht auf dem Bus übertragen. Konfigurierbarer Wertebereich: 0 s bis 10 s.

HINWEIS: Der Standardwert beträgt 0 ms.

HINWEIS: Der Wert 0 s bedeutet, dass die Verzögerung zwischen zwei Signalen nicht verwaltet wird.

Algorithmus zur RTS/CTS-Datenflusssteuerung

Ziel ist die Verhinderung eines Empfangspuffer-Überlaufs.

Das RTS-Ausgangssignal jedes Geräts ist mit dem CTS-Eingangssignal eines anderen Geräts verbunden. Das Sendegerät (M340) ist berechtigt, Daten zu übertragen, sobald es das RTS-Eingangssignal (z. B. ein anderes M340-Gerät) an seinem CTS-Eingang empfängt. Der Algorithmus ist symmetrisch und ermöglicht eine asynchrone Vollduplex-Kommunikation.

Übertragungsparameter im Zeichenmodus

Einführung

Die Übertragungsparameter sind in vier verschiedenen Bereichen verfügbar:

- Bereich **Übertragungsgeschwindigkeit**
- Bereich **Daten**
- Bereich **Stopp**
- Bereich **Parität**

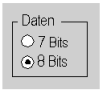
Bereich Übertragungsgeschwindigkeit

Dieser Konfigurationsbereich wird wie folgt auf dem Bildschirm angezeigt:

Sie können in diesem Bereich die Übertragungsgeschwindigkeit des Zeichenmodus-Protokolls auswählen. Die ausgewählte Geschwindigkeit muss mit den anderen Geräten übereinstimmen. Konfigurierbare Werte: 600, 1200, 2400, 4800, 9600, 19200 (Standard) und 38400 Bit pro Sekunde.

Bereich Daten

Dieser Konfigurationsbereich wird wie folgt auf dem Bildschirm angezeigt:



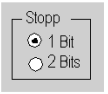
In diesem Bereich können Sie die Größe der über die Verbindung ausgetauschten Daten angeben. Konfigurierbare Werte:

- 7 Bit
- 8 Bit

Es wird empfohlen, die Anzahl der Datenbits in Übereinstimmung mit dem verwendeten dezentralen Gerät einzustellen.

Bereich Daten

Dieser Bereich sieht wie folgt aus:



Der Bereich **Stopp** ermöglicht die Eingabe der für die Kommunikation verwendeten Anzahl an Stoppbits. Es wird empfohlen, die Anzahl der Stoppbits in Übereinstimmung mit dem verwendeten dezentralen Gerät einzustellen.

Konfigurierbare Werte:

- 1 Bit
- 2 Bit

Bereich Parität

Dieser Konfigurationsbereich wird wie folgt auf dem Bildschirm angezeigt:

Verwenden Sie diesen Bereich, um anzugeben, ob ein Paritätsbit hinzugefügt wird oder nicht. Hier können Sie auch den Typ des Paritätsbits festlegen. Es wird empfohlen, die Parität in Übereinstimmung mit dem verwendeten dezentralen Gerät einzustellen. Konfigurierbare Werte:

- **Gerade**
- **Ungerade**
- **Ohne**

Abschnitt 5.3
Programmieren der Zeichenmoduskommunikation

Kommunikationsfunktionen für den Zeichenmodus

Verfügbare Funktionen

Drei bestimmte Kommunikationsfunktionen sind für das Senden und den Empfang von Daten über einen Kommunikationskanal im Zeichenmodus definiert:

- `PRINT_CHAR`: Senden einer Zeichenfolge mit maximal 1.024 Byte
- `INPUT_CHAR`: Lesen einer Zeichenfolge mit maximal 1.024 Byte
- `INPUT_BYTE`: Lesen eines Byte-Arrays mit maximal 1.024 Byte

Detaillierte Informationen zu diesen Kommunikationsfunktionen finden Sie im Kapitel *Allgemeine Informationen zu den M340-Kommunikationsfunktionen (siehe EcoStruxure™ Control Expert, Kommunikation, Bausteinbibliothek)*.

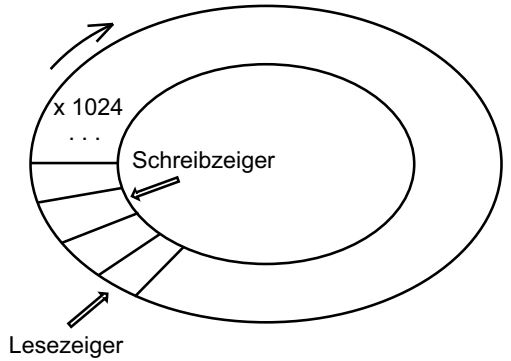
HINWEIS: Wenn der Kanal ohne „Stopp bei Stille“ konfiguriert ist, ist für die Funktion `INPUT_CHAR` ein konfigurierter Timeout erforderlich, um das Aktivitätsbit der Funktion zu quittieren. Für die Funktion `PRINT_CHAR` wird die Konfiguration eines Timeouts empfohlen, ist jedoch nicht unbedingt erforderlich.

Interner Mechanismus der CPU

Der serielle Anschluss der Modicon M340-SPS ist ein Vollduplex-Anschluss, sodass eine `PRINT_CHAR`-Funktion selbst dann gesendet werden kann, wenn eine `INPUT_CHAR`- oder `INPUT_BYTE`-Funktion gesendet und noch nicht verarbeitet wurde.

Zwei unabhängige Zeiger ermöglichen den Lese- und Schreibzugriff auf die Daten.

Die folgende Abbildung zeigt diesen Mechanismus:



Die empfangenen Daten werden in einem zyklischen 1024-Bit-Puffer gespeichert. Sobald der Puffer vollständig gefüllt ist, überschreibt das 1025. Bit das 1. Bit usw. Jedes über die Funktion `INPUT_CHAR` gelesene Pufferbit wird zurückgesetzt.

Die CPU speichert das ECHO der übertragenen Daten im gleichen Puffer wie die empfangenen Daten. Aus diesem Grund muss der Puffer der CPU nach jeder Funktion `PRINT_CHAR` bzw. vor dem Senden von Daten an den Kanal geleert werden. Andernfalls entsprechen die über `INPUT_CHAR` oder `INPUT_BYTE` empfangenen Daten nicht den erwarteten Daten.

Um den CPU-Puffer zu leeren, kann der Eingangsparameter `RAZ` der Lesefunktion auf 1 gesetzt und diese Lesefunktion vor Ablauf des Timeouts abgebrochen werden. Der Puffer wird zum ersten Mal zurückgesetzt, wenn der Prozessor auf den Empfang von Daten wartet.

HINWEIS: Es ist ratsam, diese Funktion zu verwenden, um den Empfangsprozess ordnungsgemäß zu beginnen, indem alte, eventuell noch im Puffer vorhandene Daten gelöscht werden.

Abbruch eines Austauschs

Für den Abbruch des von den Kommunikationsfunktionen ausgeführten Austauschs sind zwei Programmiermöglichkeiten gegeben:

- Verwendung der Funktion `CANCEL`
- Verwendung des Abbruchbits der Kommunikationsfunktion

Detaillierte Informationen zum Abbruch einer Kommunikationsfunktion finden Sie im Handbuch *EcoStruxure™ Control Expert, Kommunikation, Bausteinbibliothek*.

Abschnitt 5.4
Debug-Fenster für die Zeichenmoduskommunikation

Debug-Fenster der Zeichenmoduskommunikation

Allgemein

Auf das Debug-Fenster für den Zeichenmodus kann im Online-Modus zugegriffen werden.

Zugriff auf das Debug-Fenster

Die folgende Tabelle beschreibt die Vorgehensweise zum Zugreifen auf das Debug-Fenster der Zeichenmoduskommunikation:

Schritt	Aktion
1	Greifen Sie auf das Konfigurationsfenster der Zeichenmoduskommunikation zu. *(siehe Seite 80)*
2	Wählen Sie auf dem angezeigten Fenster die Registerkarte "Debuggen" aus.

Beschreibung des Debug-Fensters

Das Debug-Fenster umfasst die Zonen "Fehler" und "Signale".

Zone "Fehler"

Der Bereich Fehler sieht wie folgt aus:

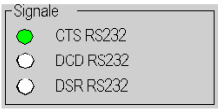
Diese Zone gibt die Anzahl der Kommunikationsunterbrechungen an, die vom Prozessor gezählt werden:

- **Beim Senden**: Entspricht der Anzahl der Unterbrechungen beim Senden (Abbild des `%MW4`-Worts).
- **Beim Empfang**: Entspricht der Anzahl der Unterbrechungen beim Empfang (Abbild des `%MW5`-Worts).

Die Schaltfläche "Zähler zurücksetzen" setzt beide Zähler auf Null zurück.

Bereich "Signale"

Der Bereich Signale sieht wie folgt aus:



Dieser Bereich gibt die Aktivität der Signale an:

- **CTS RS232**: Zeigt die Aktivität des CTS-Signals an.
- **DCD RS232**: Nicht vom Prozessor verwaltet (keine Aktivität an dieser LED).
- **DSR RS232**: Nicht vom Prozessor verwaltet (keine Aktivität an dieser LED).

Kapitel 6
Sprachobjekte der Modbus- und Zeichenmoduskommunikation

Inhalt des Kapitels

In diesem Kapitel werden die Sprachobjekte der Modbus- und der Zeichenmoduskommunikation und deren verschiedene Verwendungsmöglichkeiten beschrieben.

Inhalt dieses Kapitels

Dieses Kapitel enthält die folgenden Abschnitte:

Abschnitt 6.1
Sprachobjekte und IODDTs der Modbus- und Zeichenmoduskommunikation

Inhalt dieses Abschnitts

Dieser Abschnitt bietet einen Überblick über die allgemeinen Gesichtspunkte von IODDTs und Sprachobjekten für die Modbus- und Zeichenmoduskommunikation.

Inhalt dieses Abschnitts

Dieser Abschnitt enthält die folgenden Themen:

Einführung in die Sprachobjekte für die Modbus- und Zeichenmodus-Kommunikation

Allgemeines

IODDTs werden durch den Hersteller vordefiniert. Sie enthalten Eingangs-/Ausgangs-Sprachobjekte, die zum Kanal eines anwendungsspezifischen Moduls gehören.

Für die Modbus- und Zeichenmodus-Kommunikation sind drei zugehörige IODDTs vorhanden:

- `T_COM_STS_GEN`: Anwendbar auf alle Kommunikationsprotokolle ausgenommen Fipio und Ethernet.
- `T_COM_MB_BMX`: Spezifisch für die Modbus-Kommunikation.
- `T_COM_CHAR_BMX`: Spezifische für die Zeichenmodus-Kommunikation.

HINWEIS: IODDT-Variablen können auf zwei Arten erstellt werden:

- Verwendung der Registerkarte der E/A-Objekte *(siehe EcoStruxure™ Control Expert, Betriebsarten)*
- Verwendung des Dateneditors *(siehe EcoStruxure™ Control Expert, Betriebsarten)*

Sprachobjekttypen

Jeder IODDT beinhaltet eine Reihe von Sprachobjekten, die der Steuerung und berprüfung der ordnungsgemäßen Funktionsweise des IODDT dienen.

Es gibt zwei Arten von Sprachobjekten:

- Implizite Austauschobjekte: Diese Objekte werden automatisch bei jedem Zyklusdurchlauf der dem Modul zugeordneten Task ausgetauscht.
- Explizite Austauschobjekte: Diese Objekte werden unter Verwendung der Anweisungen zum expliziten Austausch auf Anforderung der Anwendung ausgetauscht.

Der implizite Austausch betrifft den Status der Prozessoren, Kommunikationssignale, Slaves usw.

Der explizite Austausch ermöglicht die Definition der Prozessoreinstellungen und die Durchführung von Diagnosen.

Implizite Austauschsprachobjekte der anwendungsspezifischen Funktion

Einführung

Die Verwendung einer integrierten anwendungsspezifischen Schnittstelle oder das Hinzufügen eines Moduls erweitert automatisch das Projekt von Sprachobjekten, welche das Programmieren dieser Schnittstelle oder dieses Moduls ermöglichen.

Diese Objekte entsprechen den Abbildern der Ein-/Ausgänge und Softwareinformationen des Moduls oder der integrierten anwendungsspezifischen Schnittstelle.

Zur Erinnerung

Die Moduleingänge (`%I` und `%IW`) werden zu Beginn des Tasks im SPS-Speicher aktualisiert bzw. wenn sich die SPS im Modus RUN oder STOP befindet.

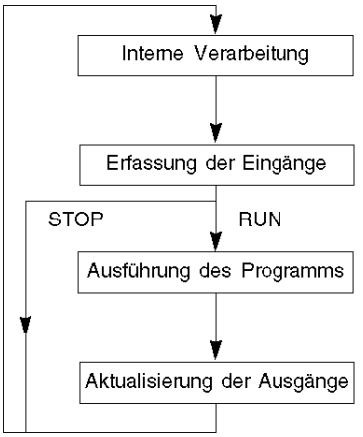
Die Ausgänge (`%Q` und `%QW`) werden am Ende des Tasks aktualisiert, jedoch nur, wenn sich die Steuerung im Modus "RUN" befindet.

HINWEIS: Befindet sich der Task im Modus STOP, so erfolgt abhängig von der gewählten Konfiguration Folgendes:

- Die Ausgänge werden auf die Position Fehlerwert gesetzt (Fehlermodus).
- Die Ausgänge werden auf ihrem letzten Wert gehalten (Modus "Wert halten").

Beschreibung

Das folgende Diagramm zeigt den Betriebszyklus der SPS-Aufgabe (zyklische Ausführung):



Explizite Austauschsprachobjekte der anwendungsspezifischen Funktion

Auf einen Blick

Ein expliziter Austausch ist ein Austausch, der auf Anforderung des Anwenderprogramms mithilfe der folgenden Anweisungen durchgeführt wird:

- READ_STS *(siehe EcoStruxure™ Control Expert, E/A-Verwaltung, Bausteinbibliothek)*: Lesen von Statuswörtern
- WRITE_CMD *(siehe EcoStruxure™ Control Expert, E/A-Verwaltung, Bausteinbibliothek)*: Schreiben von Befehlswörtern

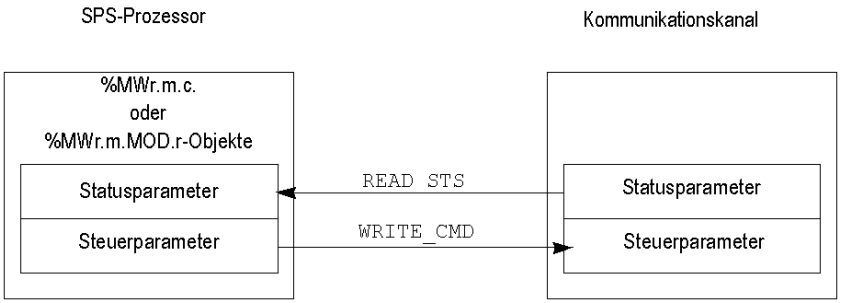
Diese Austauschvorgänge gelten für einen Satz von `%MW` Objekten desselben Typs (Status, Befehle oder Parameter), die zu einem Kanal gehören.

HINWEIS: Diese Objekte bieten Informationen über den Prozessor oder das Modul und können für Befehle an diese verwendet werden (z. B.: Steuerung der Flipflops) sowie, um deren Betriebsarten festzulegen (Speichern und Wiederherstellen der ausgeführten Regelungsparameter).

HINWEIS: Die Anweisungen `READ_STS` und `WRITE_CMD` werden gleichzeitig und immer fehlerfrei als der Task ausgeführt, von dem sie aufgerufen werden. Das Ergebnis dieser Anweisungen steht unmittelbar nach dem Ausführen zur Verfügung.

Allgemeines Nutzungsprinzip der expliziten Anweisungen

Die folgende Abbildung zeigt die unterschiedlichen Arten expliziter Austauschvorgänge, die zwischen dem Prozessor und dem Kommunikationskanal ausgeführt werden können:



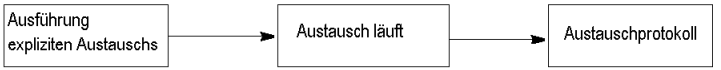
Verwaltung des Austauschs

Während eines expliziten Austauschs muss der Ablauf dieses Austauschs überwacht werden, damit die Daten nur dann berücksichtigt werden, wenn der Austausch ordnungsgemäß durchgeführt wurde.

Hierzu sind zwei Informationstypen verfügbar:

- Informationen über den laufenden Austausch *(siehe EcoStruxure™ Control Expert, E/A-Verwaltung, Bausteinbibliothek)*
- Protokoll des Austauschs *(siehe EcoStruxure™ Control Expert, E/A-Verwaltung, Bausteinbibliothek)*

Die folgende Abbildung veranschaulicht das Verwaltungsprinzip eines Austauschs:



HINWEIS: Um mehrere simultane explizite Austauschvorgänge für ein und denselben Kanal zu vermeiden, muss der Wert des Worts EXCH_STS (`%MWr.m.c.0`) des zum Kanal gehörenden IODDT getestet werden, bevor eine Elementarfunktion, die diesen Kanal nutzt, aufgerufen wird.

Verwaltung von Austauschvorgängen und Berichten mit expliziten Objekten

Auf einen Blick

Werden Daten zwischen SPS-Speicher und Modul ausgetauscht, kann die Bestätigung dieser Informationen durch das Modul mehrere Taskzyklen erfordern.

Um den Austausch zu verwalten, besitzen alle IODDT zwei Wörter:

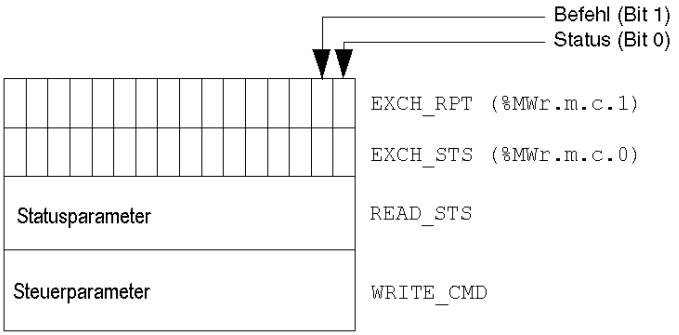
- `EXCH_STS` (`%MWr.m.c.0`): Austausch läuft.
- `EXCH_RPT` (`%MWr.m.c.1`): Rückmeldung.

HINWEIS:
Je nach Lokalisierung des Moduls wird die Verwaltung der expliziten Austauschvorgänge (z.B. `%MW0.0.MOD.0.0` von der Anwendung nicht erkannt:

- Bei Modulen im Rack erfolgt der Austausch sofort auf dem lokalen SPS-Bus und wird vor der Fertigstellung der Ausführungstask abgeschlossen, so dass beispielsweise `READ_STS` immer fertig gestellt ist, wenn das Bit `%MW0.0.mod.0.0` von der Applikation überprüft wird.
- Bei dezentralen Busmodulen (z. B. Fipio) sind explizite Austauschvorgänge nicht mit der Task-Ausführung synchronisiert, somit ist eine Erkennung für die Anwendung möglich.

Abbildung

Die folgende Abbildung zeigt die unterschiedlichen signifikanten Bits für die Verwaltung der Austauschvorgänge:



Beschreibung der signifikanten Bits

Jedes Bit der Wörter EXCH_STS (%MWr.m.c.0) und EXCH_RPT (%MWr.m.c.1) ist einem Parametertyp zugewiesen:

- Die Bits mit dem Stellenwert 0 sind den Statusparametern zugeordnet:
  - Das Bit STS_IN_PROGR (%MWr.m.c.0.0) zeigt an, ob eine aktuelle Aufforderung zum Lesen der Statuswörter vorhanden ist.
  - Das Bit STS_ERR (%MWr.m.c.1.0) zeigt an, ob eine Aufforderung zum Lesen der Statuswörter vom Kanal des Moduls akzeptiert wird.
- Die Bits mit dem Stellenwert 1 sind den Befehlsparametern zugeordnet:
  - Das Bit CMD_IN_PROGR (%MWr.m.c.0.1) gibt an, ob die Steuerparameter an den Modulkanal gesendet werden oder nicht.
  - Das Bit CMD_ERR (%MWr.m.c.1.1) zeigt an, ob die Steuerparameter vom Kanal des Moduls akzeptiert werden.

HINWEIS: „r“ entspricht der Racknummer und „m“ der Position des Moduls im Rack, während „c“ der Kanalnummer im Modul entspricht.

HINWEIS: Austausch- und Berichtswörter existieren auch auf Modulebene EXCH_STS (%MWr.m.MOD.0) und EXCH_RPT (%MWr.m.MOD.1) des IODDT-Typs T_GEN_MOD.

Ausführungsanzeiger eines expliziten Austauschs: EXCH_STS

Die folgende Tabelle zeigt die Steuerbits der expliziten Austauschvorgänge des Worts EXCH_STS (%MWr.m.c.0):

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
STS_IN_PROGR	BOOL	R	Lesen der Statuswörter des aktuellen Kanals	%MWr.m.c.0.0
CMD_IN_PROGR	BOOL	R	Befehlsparameter werden ausgetauscht	%MWr.m.c.0.1
ADJ_IN_PROGR	BOOL	R	Einstellparameter werden ausgetauscht	%MWr.m.c.0.2
RECONF_IN_PROGR	BOOL	R	Aktuelle Neueinstellung des Moduls	%MWr.m.c.0.15

HINWEIS: Wenn das Modul nicht vorhanden oder getrennt ist, werden die expliziten Austauschobjekte (z. B. READ_STS) nicht an das Modul gesendet (STS_IN_PROG (%MWr.m.c.0.0) = 0), aber die Worte werden aktualisiert.

Protokoll des expliziten Austauschs: EXCH_RPT

Die folgende Tabelle zeigt die Berichtsbits des Worts `EXCH_RPT` (`%MWr.m.c.1`):

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
STS_ERR	BOOL	R	Fehler beim Lesen der Statuswörter des Kanals erkannt (1 = Fehler)	`%MWr.m.c.1.0`
CMD_ERR	BOOL	R	Fehler beim Austausch von Befehlsparametern erkannt (1 = Fehler)	`%MWr.m.c.1.1`
ADJ_ERR	BOOL	R	Unterbrechungen beim Austausch von Einstellparametern (1 = Fehler)	`%MWr.m.c.1.2`
RECONF_ERR	BOOL	R	Unterbrechungen bei der Neukonfiguration des Kanals (1 = Fehler)	`%MWr.m.c.1.15`

Abschnitt 6.2
Allgemeine Sprachobjekte und IODDTs für Kommunikationsprotokolle

Inhalt dieses Abschnitts

Dieser Abschnitt beschreibt die generischen Sprachobjekte und IODDTs, die sich auf alle Kommunikationsprotokolle beziehen.

Inhalt dieses Abschnitts

Dieser Abschnitt enthält die folgenden Themen:

Beschreibung der impliziten IODDT-Austauschobjekte vom Typ T_COM_STS_GEN

Einführung

In der folgenden Tabelle sind die Sprachobjekte für den impliziten Austausch des IODDT des Typs `T_COM_STS_GEN` aufgeführt, der für alle Kommunikationsprotokolle außer Fipio gültig ist.

Fehlerbit

Die folgende Tabelle enthält die Bedeutung des Fehlerbits `CH_ERROR` (%Ir.m.c.ERR):

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
CH_ERROR	EBOOL	R	Fehlerbit des Kommunikationskanals	%Ir.m.c.ERR

Beschreibung der expliziten IODDT-Austauschobjekte vom Typ T_COM_STS_GEN

Auf einen Blick

Dieser Abschnitt beschreibt die Objekte mit explizitem Austausch des IODDT-Typs `T_COM_STS_GEN`, der auf alle Kommunikationsprotokolle außer Fipio und Ethernet anwendbar ist. Der Teil umfasst die Objekte des Typs Wort, deren Bits eine besondere Bedeutung haben. Diese Objekte werden im Folgenden ausführlich erläutert.

In diesem Abschnitt weist die Variable `IODDT_VAR1` den Typ `T_COM_STS_GEN` auf.

Bemerkungen

Die Bedeutung eines Bit wird im Allgemeinen für den Zustand 1 des Bit gegeben. In spezifischen Fällen wird eine Erläuterung zu jedem Zustand des Bit gegeben.

Es werden nicht alle Bits verwendet.

Ausführungsanzeiger eines expliziten Austauschs: EXCH_STS

In der folgenden Tabelle sind die Bedeutungen der Kanal-Austauschsteuerungsbits des Kanals `EXCH_STS` (`%MWr.m.c.0`) aufgeführt:

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
STS_IN_PROGR	BOOL	R	Lesen der Statuswörter des aktuellen Kanals	`%MWr.m.c.0.0`
CMD_IN_PROGR	BOOL	R	Austausch der Befehlsparameter läuft	`%MWr.m.c.0.1`

Rückmeldung von expliziten Austauschvorgängen: EXCH_RPT

In der folgenden Tabelle ist die Bedeutung der Austauschberichtsbits `EXCH_RPT` (`%MWr.m.c.1`) aufgeführt:

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
STS_ERR	BOOL	R	Fehler beim Lesen der Statuswörter des Kanals erkannt	`%MWr.m.c.1.0`
CMD_ERR	BOOL	R	Fehler beim Austausch von Befehlsparametern erkannt.	`%MWr.m.c.1.1`

Kanalspezifische Standardfehler: CH_FLT

In der folgenden Tabelle wird die Bedeutung der Bits des Statusworts `CH_FLT` (`%%MWr.m.c.2`) aufgeführt:

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
NO_DEVICE	BOOL	R	Kein Gerät auf dem Kanal funktioniert	`%MWr.m.c.2.0`
ONE_DEVICE_FLT	BOOL	R	Ein Gerät auf dem Kanal ist nicht funktionsfähig	`%MWr.m.c.2.1`
BLK	BOOL	R	Klemmenleiste ist nicht angeschlossen	`%MWr.m.c.2.2`
TO_ERR	BOOL	R	Timeout übernommen (Analyse erforderlich)	`%MWr.m.c.2.3`
INTERNAL_FLT	BOOL	R	Interner Modulfehler oder Selbsttest des Kanals	`%MWr.m.c.2.4`
CONF_FLT	BOOL	R	Unterschiedliche Hard- und Softwarekonfiguration	`%MWr.m.c.2.5`
COM_FLT	BOOL	R	Kommunikationsanalyse mit Kanal erforderlich.	`%MWr.m.c.2.6`
APPLI_FLT	BOOL	R	Anwendungsfehler erkannt (Anpassungs- oder Konfigurationsfehler)	`%MWr.m.c.2.7`

Das Lesen erfolgt durch die Anweisung `READ_STS` (`IODDT_VAR1`).

Abschnitt 6.3
Sprachobjekte und IODDTs der Modbus-Kommunikation

Inhalt dieses Abschnitts

In diesem Abschnitt werden die Sprachobjekte und IODDTs der Modbus-Kommunikation erläutert.

Inhalt dieses Abschnitts

Dieser Abschnitt enthält die folgenden Themen:

Beschreibung der expliziten Austauschsprachobjekte für eine Modbus-Funktion

Auf einen Blick

Die folgende Tabelle führt die Sprachobjekte für die Modbus-Kommunikation im Master- bzw. Slavemodus auf. Diese Objekte sind nicht in die IODDTs integriert.

Liste der expliziten Austauschobjekte im Master- bzw. Slavemodus

Die folgende Tabelle führt die expliziten Austauschobjekte auf:

Adresse	Typ	Zugriff	Bedeutung
%MWr.m.c.4	INT	R	Anzahl der fehlerfrei empfangenen Antworten.
%MWr.m.c.5	INT	R	Anzahl der Antworten, die mit CRC-Fehler empfangen wurden
%MWr.m.c.6	INT	R	Anzahl der Antworten, die im Slavemodus mit einem Ausnahmecode empfangen wurden
%MWr.m.c.7	INT	R	Anzahl der im Slavemodus gesendeten Nachrichten
%MWr.m.c.8	INT	R	Anzahl der im Slavemodus gesendeten Nachrichten ohne Antwort
%MWr.m.c.9	INT	R	Anzahl der Antworten, die mit einer negativen Bestätigung empfangen wurden.
%MWr.m.c.10	INT	R	Anzahl der im Slavemodus wiederholten Nachrichten
%MWr.m.c.11	INT	R	Anzahl der erkannten Zeichenfehler
%MWr.m.c.24.0	BOOL	RW	Zurücksetzen des Zählers für erkannte Fehler

Beschreibung der impliziten IODDT-Austauschobjekte vom Typ T_COM_MB_BMX und T_COM_MB_BMX_CONF_EXT

Einführung

Die nachstehenden Tabellen zeigen die impliziten Austauschobjekte der IODDTs vom Typ `T_COM_MB_BMX` und `T_COM_MB_BMX_CONF_EXT`, die bei der seriellen Modbus-Kommunikation zum Einsatz kommen. Sie unterscheiden sich in Bezug auf die **Verfügbarkeit der Konfigurationsobjekte** *(siehe Seite 112)*.

Bit CH_ERROR

In der folgenden Tabelle wird die Bedeutung des Fehlerbits `CH_ERROR` (`%Ir.m.c.ERR`) erläutert:

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
CH_ERROR	EBOOL	R	Fehlerbit des Kommunikationskanals	`%Ir.m.c.ERR`

Wortobjekt im Modbus-Master-Modus

In der folgenden Tabelle wird die Bedeutung des Bits des Worts `INPUT_SIGNALS` (`%IWr.m.c.0`) erläutert:

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
DCD	BOOL	R	RS232-Signal DCD (Datenträgererkennung) (nur für das Modul BMX NOM 0200)	`%IWr.m.c.0.0`
CTS	BOOL	R	Bereit zum Senden des RS232-Signals	`%IWr.m.c.0.2`
DSR	BOOL	R	RS232-Signal DSR (Datensatz bereit) (nur für das Modul BMX NOM 0200)	`%IWr.m.c.0.3`

HINWEIS: `%IWr.m.c.0.2` wird auf 1 gesetzt, wenn das CTS-Signal eine positive Spannung aufweist. Es ist ebenfalls auf DCD und DSR anwendbar.

Wortobjekt im Modbus-Slave-Modus

Die Sprachobjekte entsprechen denjenigen der Modbus-Master-Funktion. Nur die Objekte in der nachstehenden Tabelle sind unterschiedlich.

In der folgenden Tabelle wird die Bedeutung des Bits des Worts `INPUT_SIGNALS` (`%IWr.m.c.0`) erläutert:

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
LISTEN_ONLY	BOOL	R	Nur-Hören-Modus	`%IWr.m.c.0.8`

Beschreibung der expliziten IODDT-Austauschobjekte vom Typ T_COM_MB_BMX und T_COM_MB_BMX_CONF_EXT

Einführung

Dieser Abschnitt beschreibt die expliziten Austauschobjekte der IODDT-Typen `T_COM_MB_BMX` und `T_COM_MB_BMX_CONF_EXT`, die für die serielle Modbus-Kommunikation verwendet werden können und sich hinsichtlich der **Verfügbarkeit der Konfigurationsobjekte** *(siehe Seite 112)* unterscheiden. Hierzu gehören Objekte des Typs Wort, deren Bits eine besondere Bedeutung haben. Diese Objekte werden im Folgenden ausführlich erläutert.

In diesem Abschnitt weist die Variable `IODDT_VAR1` den Typ `T_COM_STS_GEN` auf.

Anmerkungen

Prinzipiell wird die Bedeutung der Bitstatus 1 angegeben. In bestimmten Fällen wird jeder Bitstatus erläutert.

Es werden nicht alle Bits verwendet.

Flags für die Ausführung des expliziten Austauschs: EXCH_STS

Die folgende Tabelle verdeutlicht die verschiedenen Bedeutungen der Austauschsteuerungsbits des Kanals `EXCH_STS` (`%MWr.m.c.0`):

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
STS_IN_PROGR	BOOL	R	Lesen der Statuswörter des aktuellen Kanals.	`%MWr.m.c.0.0`
CMD_IN_PROGR	BOOL	R	Austausch der Befehlsparameter läuft	`%MWr.m.c.0.1`
ADJ_IN_PROGR	BOOL	R	Austausch der Einstellparameter läuft (nicht für Module vom Typ BMX NOM 0200)	`%MWr.m.c.0.2`

Rückmeldung zum expliziten Austausch: EXCH_RPT

Die folgende Tabelle verdeutlicht die verschiedenen Bedeutungen der Austauschberichtsbits `EXCH_RPT` (`%MWr.m.c.1`):

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
STS_ERR	BOOL	R	Fehler beim Lesen der Statuswörter des Kanals erkannt	`%MWr.m.c.1.0`
CMD_ERR	BOOL	R	Unregelmäßigkeit während eines Austauschs von Befehlsparametern.	`%MWr.m.c.1.1`
ADJ_ERR	BOOL	R	Unregelmäßigkeit während eines Austauschs von Einstellparametern (nicht für Module des Typs BMX NOM 0200).	`%MWr.m.c.1.2`

Kanalspezifische erkannte Standardfehler: CH_FLT

Die folgende Tabelle verdeutlicht die verschiedenen Bedeutungen der Bits des Statusworts CH_FLT (%MWr.m.c.2):

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
NO_DEVICE	BOOL	R	Kein Gerät auf dem Kanal funktioniert	%MWr.m.c.2.0
ONE_DEVICE_FLT	BOOL	R	Ein Gerät auf dem Kanal ist nicht funktionsfähig	%MWr.m.c.2.1
BLK	BOOL	R	Klemmenleiste ist nicht angeschlossen	%MWr.m.c.2.2
TO_ERR	BOOL	R	Timeout übernommen (Analyse erforderlich)	%MWr.m.c.2.3
INTERNAL_FLT	BOOL	R	Interner Modulfehler oder Selbsttest des Kanals	%MWr.m.c.2.4
CONF_FLT	BOOL	R	Unterschiedliche Hard- und Softwarekonfiguration	%MWr.m.c.2.5
COM_FLT	BOOL	R	Kommunikationsanalyse mit Kanal erforderlich.	%MWr.m.c.2.6
APPLI_FLT	BOOL	R	Anwendungsfehler erkannt (Anpassungs- oder Konfigurationsfehler)	%MWr.m.c.2.7

Das Lesen erfolgt durch die Anweisung READ_STS (IODDT_VAR1).

Spezifischer Kanalstatus: %MWr.m.c.3

Die folgende Tabelle verdeutlicht die verschiedenen Bedeutungen der Bits des Kanalstatusworts PROTOCOL (%MWr.m.c.3):

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
PROTOCOL	INT	R	Byte 0 = 16#06 für Modbus-Mastermodus. Byte 0 = 16#07 für Modbus-Slavemodus. Byte 0 = 16#03 für Zeichenmodus.	%MWr.m.c.3

Das Lesen erfolgt durch die Anweisung READ_ST (SIODDT_VAR1).

Kanalbefehl: %MWr.m.c.24

In der folgenden Tabelle sind die verschiedenen Bedeutungen der Bits des Worts CONTROL (%MWr.m.c.24) aufgeführt:

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
DTR_ON	BOOL	R/W	Signal "Datenübertragungseinrichtung bereit" einstellen.	%MWr.m.c.24.8
DTR_OFF	BOOL	R/W	Signal "Datenübertragungseinrichtung bereit" zurücksetzen.	%MWr.m.c.24.9
TO_MODBUS_MASTER	BOOL	R/W	Wechseln Sie vom Zeichenmodus beziehungsweise Modbus-Slavemodus in den Modbus-Mastermodus.	%MWr.m.c.24.12
TO_MODBUS_SLAVE	BOOL	R/W	Wechseln Sie vom Zeichenmodus beziehungsweise Modbus-Mastermodus in den Modbus-Slavemodus.	%MWr.m.c.24.13
TO_CHAR_MODE	BOOL	R/W	Wechseln Sie vom Modbus- in den Zeichenmodus.	%MWr.m.c.24.14

Der Befehl wird durch die Anweisung WRITE_CMD (IODDT_VAR1) ausgeführt.

Weitere Informationen zur Änderung von Protokollen finden Sie unter **Protokolländerungen** *(siehe Seite 125)*.

Externe Konfigurationsobjekte des Typs T_COM_MB_BMX_CONF_EXT: %MWr.m.c.24.7 und %MWr.m.c.25

Die folgende Tabelle beschreibt die Bedeutung des Bits CONTROL (%MWr.m.c.24.7 und des Worts CONTROL_DATA (%MWr.m.c.25), die speziell für die Programmierung des Moduls BMX NOM 0200 vorgesehen sind:

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
SAVE_SLAVE_ADDR	BOOL	R/W	Speichert die Steuerungsdaten in den Flash-Speicher	%MWr.m.c.24.7
SLAVE_ADDR	INT	R/W	In Flash zu speichernde Modbus-Slave-Adresse, von 0 bis 248 (0 für Master). **HINWEIS:** Beachten Sie, dass diese Funktionalität optional ist und es keinen Grund gibt, sie häufig anzuwenden. Da als Technologie Flash genutzt wird, kann der Chip beschädigt werden.	%MWr.m.c.25

Beschreibung der Sprachobjekte für die Konfiguration des Modbus-Modus

Auf einen Blick

Die folgenden Tabellen führen alle Konfigurationssprachobjekte für die Kommunikation im Modbus-Modus auf. Diese Objekte sind nicht in die IODDTs integriert und können vom Anwendungsprogramm angezeigt werden.

Liste der Objekte mit explizitem Austausch für den Mastermodus

In der folgenden Tabelle sind die Objekte mit explizitem Austausch aufgeführt.

Adresse	Typ	Zugriff	Bedeutung
%KWr.m.c.0	INT	R	Byte 0 dieses Worts entspricht dem folgenden Typ: ● Wert 6 entspricht dem Master ● Wert 7 entspricht dem Slave
%KWr.m.c.1	INT	R	Byte 0 dieses Worts entspricht der Übertragungsgeschwindigkeit. Dieses Byte kann mehrere Werte haben: ● Wert -2 (0xFE) entspricht 300 Bit/s ● Wert -1 (0xFF) entspricht 600 Bit/s ● Wert 0 (0x00) entspricht 1200 Bit/s ● Wert 1 (0x01) entspricht 2400 Bit/s ● Wert 2 (0x02) entspricht 4800 Bit/s ● Wert 3 (0x03) entspricht 9600 Bit/s ● Wert 4 (0x04) entspricht 19200 Bit/s (Standardwert) ● Wert 5 (0x05) entspricht 38400 Bit/s ● Wert 6 (0x06) entspricht 57600 Bit/s (nur für BMX NOM 0200-Module) ● Wert 7 (0x07) entspricht 115200 Bit/s (nur für BMX NOM 0200-Module) Byte 1 dieses Worts entspricht dem Format: ● Bit 8: Anzahl der Bits (1 = 8 Bits (RTU), 0 = 7 Bits (ASCII)) ● Bit 9 = 1: Paritätsverwaltung (1 = mit, 0 = ohne) ● Bit 10: Paritätstyp (1 = ungerade, 0 = gerade) ● Bit 11: Anzahl der Stoppbits (1 = 1 Bit, 0 = 2 Bits) ● Bit 13: Physische Leitung (1 = RS232, 0 = RS485) ● Bit 14: DTR/DSR/DCD-Modemsignale (nur für BMX NOM 0200-Module und physische RS232-Leitung) Wenn dieses Bit 1 auf gesetzt ist, werden die Modemsignale verwaltet. ● Bit 15: Hardware-Datenflusskontrolle über RTS/CTS-Signale. Bei Auswahl von RS232 kann dieses Bit 2 Werte haben: 0 für RX/TX und 1 für RX/TX + RTS/CTS. Bei Auswahl von RS485 ist der Standardwert 0. Dieser Wert entspricht RX/TX.

Adresse	Typ	Zugriff	Bedeutung
%KWr.m.c.2	INT	R	Verzögerung zwischen Frames (nur in RTU-Modus): Wert in ms von 2 bis 10000 ms (abhängig von der ausgewählten Übertragungsgeschwindigkeit und vom ausgewählten Format). Der Standardwert ist 2 ms, wenn das Kontrollkästchen "Standard" aktiviert ist. 10 s entspricht unendlichem Warten.
%KWr.m.c.3	INT	R	Im Modbus-Mastermodus entspricht dieses Objekt der Antwortverzögerung in ms von 10 ms bis 1000 ms. 100 ms ist der Standardwert. 10 s entspricht unendlichem Warten.
%KWr.m.c.4	INT	R	Nur verfügbar im Modbus-Mastermodus. Byte 0 dieses Worts ist die Anzahl der Wiederholungen von 0 bis 15. Der Standardwert ist 3.
%KWr.m.c.5	INT	R	Bei Auswahl von RS232 entspricht dieses Wort von 0 bis 100 der RTS/CTS-Verzögerungszeit in Schritten von Hundert Millisekunden. Bei Auswahl von RS485 ist der Standardwert 0.

Liste der Objekte mit explizitem Austausch für den Slavemodus

Die Sprachobjekte für die Modbus-Slavefunktion sind identisch mit denen der Modbus-Masterfunktion. Lediglich die nachfolgend aufgeführten Objekte unterscheiden sich:

Adresse	Typ	Zugriff	Bedeutung
%KWr.m.c.3	INT	R	Im Modbus-Slavemodus entspricht Byte 0 dieses Objekts der Slave-Nummer [0/1, 247]. Für BMX NOM 0200-Module bedeutet ein Wert von 0, dass die Slave-Nummer im FLASH-Speicher abgelegt ist.
%KWr.m.c.4	INT	R	Nur im Modbus-Mastermodus.

Abschnitt 6.4
Sprachobjekte und IODDTs der Zeichenmoduskommunikation

Inhalt dieses Abschnitts

In diesem Abschnitt werden die Sprachobjekte und IODDTs der Zeichenmoduskommunikation erläutert.

Inhalt dieses Abschnitts

Dieser Abschnitt enthält die folgenden Themen:

Beschreibung der expliziten Austauschsprachobjekte für die Kommunikation im Zeichenmodus

Auf einen Blick

Die folgenden Tabellen führen alle Konfigurationssprachobjekte für die Kommunikation im Zeichenmodus auf. Diese Objekte sind nicht in die IODDTs integriert.

Liste der expliziten Austauschobjekte

Die folgende Tabelle führt die expliziten Austauschobjekte auf:

Adresse	Typ	Zugriff	Bedeutung
%MWr.m.c.4	INT	R	Unregelmäßigkeit bei gesendeten Zeichen.
%MWr.m.c.5	INT	R	Unregelmäßigkeit bei empfangenen Zeichen.
%MWr.m.c.24.0	BOOL	RW	Setzt Fehlerzähler zurück, wenn es auf 1 gesetzt ist
%QWr.m.c.0 = 16#DEAD	INT	RW	BMX NOM 0200 neu starten.

Beschreibung der impliziten IODDT-Austauschobjekte vom Typ T_COM_CHAR_BMX

Einführung

Die nachstehenden Tabellen zeigen die impliziten Austauschobjekte des IODDT vom Typ `T_COM_CHAR_BMX`, die bei der Zeichenmodus-Kommunikation zum Einsatz kommen.

Fehlerbit

In der folgenden Tabelle wird die Bedeutung des Fehlerbits `CH_ERROR` (`%Ir.m.c.ERR`) erläutert:

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
CH_ERROR	EBOOL	R	Fehlerbit des Kommunikationskanals	`%Ir.m.c.ERR`

Signalobjekt am Eingang

In der folgenden Tabelle wird die Bedeutung des Bits des Worts `INPUT_SIGNALS` (`%IWr.m.c.0`) erläutert:

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
DCD	BOOL	R	RS232-Signal DCD (Datenträgererkennung) (nur für das Modul BMX NOM 0200)	`%IWr.m.c.0.0`
CTS	BOOL	R	Bereit zum Senden des RS232-Signals	`%IWr.m.c.0.2`
DSR	BOOL	R	RS232-Signal DSR (Datensatz bereit) (nur für das Modul BMX NOM 0200)	`%IWr.m.c.0.3`

HINWEIS: `%IWr.m.c.0.2` wird auf 1 gesetzt, wenn das CTS-Signal eine positive Spannung aufweist. Es ist ebenfalls auf DCD und DSR anwendbar.

Beschreibung der expliziten IODDT-Austauschobjekte vom Typ T_COM_CHAR_BMX

Einführung

Die nachstehenden Tabellen zeigen die expliziten Austauschobjekte des IODDT vom Typ `T_COM_CHAR_BMX`, die bei der Zeichenmodus-Kommunikation zum Einsatz kommen. Hierzu gehören Objekte des Typs Wort, deren Bits eine besondere Bedeutung haben. Diese Objekte werden im Folgenden ausführlich erläutert.

In diesem Abschnitt weist die Variable `IODDT_VAR1` den Typ `T_COM_STS_GEN` auf.

Anmerkungen

Prinzipiell wird die Bedeutung für den Bitstatus 1 angegeben. In bestimmten Fällen wird jeder Bitstatus erläutert.

Es werden nicht alle Bits verwendet.

Flag für die Ausführung des expliziten Austauschs: EXCH_STS

Die folgende Tabelle verdeutlicht die verschiedenen Bedeutungen der Austauschsteuerungsbits des Kanals `EXCH_STS` (`%MWr.m.c.0`):

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
STS_IN_PROGR	BOOL	R	Lesen der Statuswörter des Kanals läuft	`%MWr.m.c.0.0`
CMD_IN_PROGR	BOOL	R	Austausch der Befehlsparameter läuft	`%MWr.m.c.0.1`
ADJ_IN_PROGR	BOOL	R	Austausch der Einstellparameter läuft (nicht für das Modul BMX NOM 0200)	`%MWr.m.c.0.2`

Rückmeldung zum expliziten Austausch: EXCH_RPT

Die Tabelle unten zeigt die Bedeutung der Austauschrückmeldungsbits `EXCH_RPT` (`%MWr.m.c.1`):

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
STS_ERR	BOOL	R	Fehler beim Lesen der Statuswörter des Kanals erkannt	`%MWr.m.c.1.0`
CMD_ERR	BOOL	R	Unregelmäßigkeit während eines Austauschs von Befehlsparametern	`%MWr.m.c.1.1`
ADJ_ERR	BOOL	R	Unregelmäßigkeit während eines Austauschs von Einstellparametern (nicht für das Modul BMX NOM 0200)	`%MWr.m.c.1.2`

Kanalspezifische Standardfehler, CH_FLT

Die folgende Tabelle verdeutlicht die verschiedenen Bedeutungen der Bits des Statusworts CH_FLT (%MWr.m.c.2):

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
NO_DEVICE	BOOL	R	Kein Gerät auf dem Kanal funktioniert.	%MWr.m.c.2.0
ONE_DEVICE_FLT	BOOL	R	Ein Gerät auf dem Kanal ist nicht funktionsfähig	%MWr.m.c.2.1
BLK	BOOL	R	Klemmenleiste ist nicht angeschlossen.	%MWr.m.c.2.2
TO_ERR	BOOL	R	Timeout übernommen (Analyse erforderlich)	%MWr.m.c.2.3
INTERNAL_FLT	BOOL	R	Interner Modulfehler oder Selbsttest des Kanals	%MWr.m.c.2.4
CONF_FLT	BOOL	R	Unterschiedliche Hard- und Softwarekonfiguration	%MWr.m.c.2.5
COM_FLT	BOOL	R	Kommunikationsanalyse erforderlich mit der SPS	%MWr.m.c.2.6
APPLI_FLT	BOOL	R	Anwendungsfehler erkannt (Anpassungs- oder Konfigurationsfehler)	%MWr.m.c.2.7

Das Lesen erfolgt durch die Anweisung READ_STS (IODDT_VAR1).

Spezifischer Kanalstatus, %MWr.m.c.3

Die folgende Tabelle verdeutlicht die verschiedenen Bedeutungen der Bits des Kanalstatusworts PROTOCOL (%MWr.m.c.3):

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
PROTOCOL	INT	R	Byte 0 = 16#03 für die Zeichenmodus-Funktion	%MWr.m.c.3

Das Lesen erfolgt durch die Anweisung READ_STS (IODDT_VAR1).

Kanalbefehl %MWr.m.c.24

Die folgende Tabelle verdeutlicht die verschiedenen Bedeutungen der Bits des Kanalstatusworts CONTROL (%MWr.m.c.24):

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
DTR_ON	BOOL	R/W	Signal „Datenübertragungseinrichtung bereit“ einstellen	%MWr.m.c.24.8
DTR_OFF	BOOL	R/W	Signal „Datenübertragungseinrichtung bereit“ zurücksetzen	%MWr.m.c.24.9

Der Befehl wird durch die Anweisung WRITE_CMD (IODDT_VAR1) ausgeführt.

Weitere Informationen zur Änderung der Protokolle finden Sie unter Ändern der Protokolle *(siehe Seite 125)*.

Wortobjekt %QWr.m.c.0

In der folgenden Tabelle wird die Bedeutung des Bits 0 des Worts `%QWr.m.c.0` beschrieben:

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
STOP_EXCH	BOOL	R/W	Anhalten des gesamten Austauschs bei steigender Flanke (nur für das Modul BMX NOM 0200)	`%QWr.m.c.0.0`

Beschreibung der Sprachobjekte für die Konfiguration des Zeichenmodus

Auf einen Blick

Die folgenden Tabellen führen alle Konfigurationssprachobjekte für die Kommunikation im Zeichenmodus auf. Diese Objekte sind nicht in die IODDTs integriert und können vom Anwendungsprogramm angezeigt werden.

Liste der Objekte mit explizitem Austausch für den Zeichenmodus

In der folgenden Tabelle sind die Objekte mit explizitem Austausch aufgeführt.

Adresse	Typ	Zugriff	Bedeutung
%KWr.m.c.0	INT	R	Byte 0 dieses Worts entspricht dem folgenden Typ. Wert 3 entspricht dem Zeichenmodus.
%KWr.m.c.1	INT	R	Byte 0 dieses Worts entspricht der Übertragungsgeschwindigkeit. Dieses Byte kann mehrere Werte haben: ● Wert -2 (0xFE) entspricht 300 Bit/s ● Wert -1 (0xFF) entspricht 600 Bit/s ● Wert 0 (0x00) entspricht 1200 Bit/s ● Wert 1 (0x01) entspricht 2400 Bit/s ● Wert 2 (0x02) entspricht 4800 Bit/s ● Wert 3 (0x03) entspricht 9600 Bit/s (Standardwert) ● Wert 4 (0x04) entspricht 19200 Bit/s ● Wert 5 (0x05) entspricht 38400 Bit/s ● Wert 6 (0x06) entspricht 57600 Bit/s (nur für BMX NOM 0200-Module wählbar) ● Wert 7 (0x07) entspricht 115200 Bit/s (nur für BMX NOM 0200-Module wählbar) Byte 1 dieses Worts entspricht dem Format: ● Bit 8: Anzahl der Bits (1 = 8 Bits (RTU), 0 = 7 Bits (ASCII)) ● Bit 9 = 1: Paritätsverwaltung (1 = mit, 0 = ohne) ● Bit 10: Paritätstyp (1 = ungerade, 0 = gerade) ● Bit 11: Anzahl der Stoppbits (1 = 1 Bit, 0 = 2 Bits) ● Bit 13: Physische Leitung (1 = RS232, 0 = RS485) ● Bit 14: DTR/DSR/DCD-Modemsignale Für BMX NOM 0200-Module und bei Auswahl von RS232 kann dieses 2 Werte haben: 1 bedeutet, dass die Modemsignale verwaltet werden, 0 bedeutet, dass sich nicht verwaltet werden (Standardwert für BMX P34 oder bei Auswahl von RS485) ● Bit 15: Hardware-Datenflusskontrolle über RTS/CTS-Signale. Bei Auswahl von RS232 kann dieses Bit 2 Werte haben: 0 für RX/TX und 1 für RX/TX + RTS/CTS. Bei Auswahl von RS485 ist der Standardwert 0. Dieser Wert entspricht RX/TX.

Adresse	Typ	Zugriff	Bedeutung
%KWr.m.c.2	INT	R	Eingegebener Wert in ms für "Stopp bei Stille" (abhängig von der ausgewählten Übertragungsgeschwindigkeit und vom ausgewählten Format). Der Wert 0 bedeutet keine Erkennung von Stille.
%KWr.m.c.3	INT	R	Dieses Wort entspricht dem Polarisierungstyp: ● Ein Wert von 0 für Bit 14 und Bit 15 entspricht keiner Polarisierung (diese ist die Standardeinstellung für BMX P34 oder bei Auswahl von RS232) ● Bit 14: Ein Wert von 1 entspricht einer Polarisierung mit niedriger Impedanz (wie Modbus) und kann nur für BMX NOM 0200-Module und bei Auswahl von RS485 verwendet werden. ● Bit 15: Ein Wert von 1 entspricht einer Polarisierung mit hoher Impedanz und kann nur für BMX NOM 0200-Module und bei Auswahl von RS485 verwendet werden.
%KWr.m.c.5	INT	R	Dieses Wort entspricht der RTS/CTS-Verzögerungszeit in Hundertstel ms von 0 bis 100, wenn RS232 ausgewählt ist. Bei Auswahl von RS485 ist der Standardwert 0.
%KWr.m.c.6	INT	R	Bit 0 von Byte 0 kann zwei Werte haben: ● Wert 1 entspricht dem aktivierten Kontrollkästchen "Stopp" im Bereich "Stopp bei Empfang" für Zeichen 1. ● Wert 0 entspricht dem deaktivierten Kontrollkästchen "Stopp" im Bereich "Stopp bei Empfang" für Zeichen 1. Bit 1 von Byte 0 kann zwei Werte haben: ● Wert 1 entspricht dem aktivierten Kontrollkästchen "Zeichen enthalten" im Bereich "Stopp bei Empfang" für Zeichen 1. ● Wert 0 entspricht dem deaktivierten Kontrollkästchen "Zeichen enthalten" im Bereich "Stopp bei Empfang" für Zeichen 1. Byte 1 dieses Worts entspricht dem für die Option "Stopp bei Empfang" von Zeichen 1 eingegebenen Wert von 0 bis 255.
%KWr.m.c.7	INT	R	Bit 0 von Byte 0 kann zwei Werte haben: ● Wert 1 entspricht dem aktivierten Kontrollkästchen "Stopp" im Bereich "Stopp bei Empfang" für Zeichen 2. ● Wert 0 entspricht dem deaktivierten Kontrollkästchen "Stopp" im Bereich "Stopp bei Empfang" für Zeichen 2. Bit 1 von Byte 0 kann zwei Werte haben: ● Wert 1 entspricht dem aktivierten Kontrollkästchen "Zeichen enthalten" im Bereich "Stopp bei Empfang" für Zeichen 2. ● Wert 0 entspricht dem deaktivierten Kontrollkästchen "Zeichen enthalten" im Bereich "Stopp bei Empfang" für Zeichen 2. Byte 1 dieses Worts entspricht dem für die Option "Stopp bei Empfang" von Zeichen 2 eingegebenen Wert von 0 bis 255.

Abschnitt 6.5
IODDT Type T_GEN_MOD, anwendbar auf alle Module

Beschreibung der Sprachobjekte des IODDT vom Typ T_GEN_MOD

Einführung

Die Modicon X80-Module verfügen über einen zugeordneten IODDT vom Typ T_GEN_MOD.

Bemerkungen

Prinzipiell wird die Bedeutung der Bits für den Bitstatus 1 angegeben. In speziellen Fällen wird jeder Status des Bits erläutert.

Einige Bits werden nicht verwendet.

Liste der Objekte

In der folgenden Tabelle werden die Objekte des IODDT aufgeführt.

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
MOD_ERROR	BOOL	R	Modulfehlerbit	%Ir.m.MOD.ERR
EXCH_STS	INT	R	Steuerwort für den Modulaustausch	%MWr.m.MOD.0
STS_IN_PROGR	BOOL	R	Lesen von Statuswörtern des Moduls	%MWr.m.MOD.0.0
EXCH_RPT	INT	R	Wort für Austauschrückmeldung	%MWr.m.MOD.1
STS_ERR	BOOL	R	Ereignis beim Lesen von Modulstatuswörtern	%MWr.m.MOD.1.0
MOD_FLT	INT	R	Internes Fehlerwort des Moduls	%MWr.m.MOD.2
MOD_FAIL	BOOL	R	Modul funktionsunfähig	%MWr.m.MOD.2.0
CH_FLT	BOOL	R	Funktionsunfähige Kanäle	%MWr.m.MOD.2.1
BLK	BOOL	R	Klemmenleiste falsch verdrahtet	%MWr.m.MOD.2.2
CONF_FLT	BOOL	R	Hardware- oder Software-Konfigurationsunregelmäßigkeit	%MWr.m.MOD.2.5
NO_MOD	BOOL	R	Modul fehlt oder nicht betriebsbereit	%MWr.m.MOD.2.6
EXT_MOD_FLT	BOOL	R	Internes Fehlerwort des Moduls (nur Fipio-Erweiterung)	%MWr.m.MOD.2.7
MOD_FAIL_EXT	BOOL	R	Interner Modulfehler, Modul nicht betriebsbereit (nur Fipio-Erweiterung)	%MWr.m.MOD.2.8
CH_FLT_EXT	BOOL	R	Funktionsunfähige Kanäle (nur Fipio-Erweiterung)	%MWr.m.MOD.2.9

Standardsymbol	Typ	Zugriff	Bedeutung	Adresse
BLK_EXT	BOOL	R	Klemmenleiste falsch verdrahtet (nur Fipio-Erweiterung)	%MWr.m.MOD.2.10
CONF_FLT_EXT	BOOL	R	Hardware- oder Software-Konfigurationsunregelmäßigkeit (nur Fipio-Erweiterung)	%MWr.m.MOD.2.13
NO_MOD_EXT	BOOL	R	Modul fehlt oder nicht betriebsbereit (nur Fipio-Erweiterung)	%MWr.m.MOD.2.14

Kapitel 7
Dynamischer Protokollwechsel

Ändern des Protokolls mit Modicon M340-Prozessoren

Allgemeines

In diesem Teil wird die Änderung des zur seriellen CPU-Kommunikation verwendeten Protokolls über den Befehl `WRITE_CMD(IODDT_VAR1)` beschrieben. Dieser Befehl kann zum Umschalten zwischen folgenden drei Protokollen eingesetzt werden:

- Modbus-Slave
- Modbus-Master
- Zeichenmodus

HINWEIS: Die Variable IODDT_VAR1 muss vom Typ T_COM_MB_BMX sein.

Ändern des Protokolls: Grundprinzip

HINWEIS: Um einen Wechsel von einem zu einem anderen Protokoll durchführen zu können, muss der Prozessor zu Anfang im Modbus-Slave-Modus konfigurier werden.

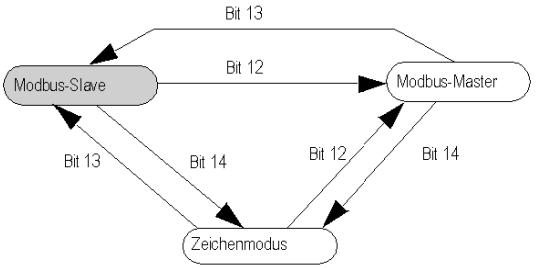
Sie müssen zunächst eine IODDT-Variable erstellen und dem seriellen Kanal des Prozessors zuordnen und dann das Bit des Worts IODDT_VAR1.CONTROL (`%MWr.m.c.24`), das der gewünschten Protokolländerung entspricht, auf 1 setzen:

- `TO_MODBUS_MASTER` (Bit 12): Das aktuelle Protokoll wird zu Modbus Master geändert.
- `TO_MODBUS_SLAVE` (Bit 13): Das aktuelle Protokoll wird zu Modbus Slave geändert.
- `TO_CHAR_MODE` (Bit 14): Das aktuelle Protokoll wird zu Zeichenmodus geändert.

HINWEIS: IODDT_VAR1.CONTROL (`%MWr.m.c.24`) ist Teil der IODDT-Variablen IODDT_VAR1.

Im Anschluss daran wenden Sie die Anweisung `WRITE_CMD` auf die dem seriellen Kanal des Prozessors zugeordnete IODDT-Variable an.

Die nachstehende Abbildung zeigt die vorzunehmenden Protokolländerungen je nach auf 1 gesetztem Bit des Worts IODDT_VAR1.CONTROL (`%MWr.m.c.24`):



Verwendung

Drei Protokolländerungen werden verwendet:

- Wechsel zu Modbus Master: Die Protokolländerung erfolgt in zwei Phasen:
  - Umschaltung von der Modbus-Slave- zur Modbus-Master-Konfiguration
  - Rückkehr zur ursprünglichen Modbus-Slave-Konfiguration

  Ziel der Modbus-Master-Konfiguration ist das Senden von Informationen über ein Ereignis an eine andere SPS. Bei einer Umschaltung von der Modbus-Slave- zur Modbus-Master-Konfiguration bleiben die Parameter für Übertragung, Signal- und physische Leitung unverändert. Nur die Werte der folgenden spezifischen Parameter der Modbus-Master-Konfiguration werden geändert:
  - Die Verzögerung zwischen Frames wird auf den Standardwert gesetzt, der von der Übertragungsgeschwindigkeit abhängig ist.
  - Die Antwortverzögerung wird auf 3.000 ms eingestellt.
  - Die Anzahl der Wiederholungen wird auf 3 gesetzt.
- Wechsel zum Zeichenmodus: Die Protokolländerung erfolgt in zwei Phasen:
  - Umschaltung von der Modbus-Slave- zur Zeichenmodus-Konfiguration
  - Rückkehr zur ursprünglichen Modbus-Slave-Konfiguration

  Ziel der Zeichenmodus-Konfiguration ist die Kommunikation mit einem privaten Protokoll (einem Modem beispielsweise). Bei einer Umschaltung von der Modbus-Slave- zur Zeichenmodus-Konfiguration bleiben die Parameter für Übertragung, Signal- und physische Leitung unverändert. Nur die spezifischen Nachrichtenendparameter für den Zeichenmodus werden auf einen Stopp bei Stille mit einem Timeout von 1000 ms eingestellt.

- Wechsel zu den Zeichenmodus- und Modbus-Master-Protokollen: Die Protokolländerung erfolgt in drei Phasen:
  - ❍ Umschaltung von der Modbus-Slave- zur Zeichenmodus-Konfiguration
  - ❍ Umschaltung von der Zeichenmodus- zur Modbus-Master-Konfiguration
  - ❍ Rückkehr zur ursprünglichen Modbus-Slave-Konfiguration

  Ziel der Zeichenmodus-Konfiguration ist die Kommunikation mit einem privaten Protokoll (einem Modem beispielsweise). Nach Abschluss des Austauschs schaltet der Benutzer zur Modbus-Master-Konfiguration um, um Informationen über ein Ereignis an eine andere SPS zu senden. Nach dem Senden der Nachricht kehrt der Benutzer zur ursprünglichen Modbus-Slave-Konfiguration zurück.

HINWEIS: In allen drei Fällen bleibt die Standardkonfiguration Modbus Slave.

Kalt- und Warmstart

Protokolländerungn werden von den Bits `%S0` und `%S1` nicht beeinflusst (die Bits werden während eines Kalt- und Warmstarts jeweils auf 1 gesetzt). Der Kalt- oder Warmstart der SPS bewirkt jedoch die Konfiguration des seriellen Ports mit dessen Standardwerten bzw. den in der Anwendung programmierten Werten.

Teil III
Kurzanleitung: Beispiel für die Implementierung einer seriellen Verbindung

Überblick

In diesem Abschnitt ist ein Beispiel für die Implementierung einer seriellen Verbindung beschrieben.

Inhalt dieses Teils

Dieser Teil enthält die folgenden Kapitel:

Kapitel 8
Beschreibung der Anwendung

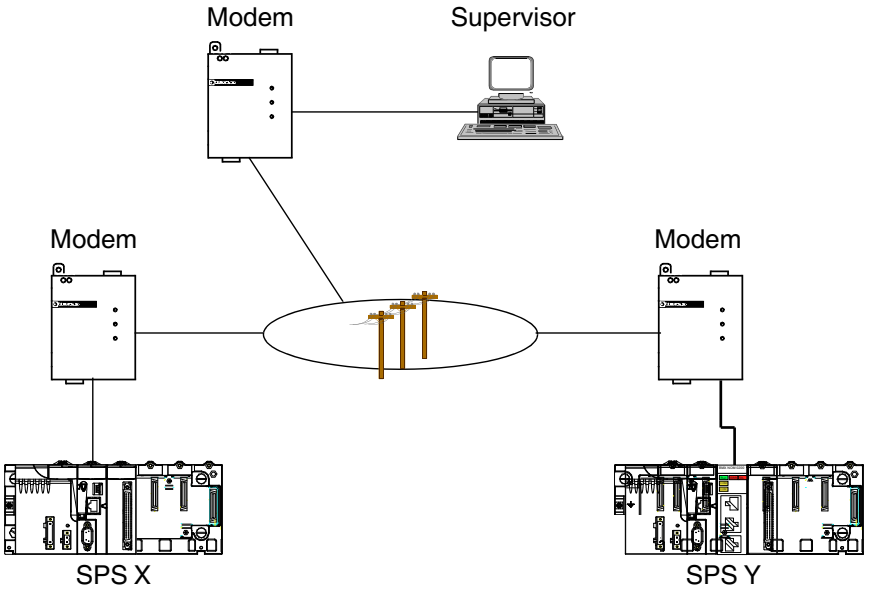
Überblick über die Anwendung

Auf einen Blick

Die in diesem Dokument beschriebene Anwendung ist eine Modbus-Kommunikationsanwendung über Modems.

Abbildung eines Beispiels

Die folgende Abbildung veranschaulicht das Beispiel.



Die Geräte kommunizieren mithilfe von Modems miteinander. Der Supervisor ist der Modbus-Master, die SPS X und Y sind Slaves.

Das Ziel dieses Beispiels ist es, die Datenwertebereiche der SPS X in die SPS Y zu schreiben.

Hierzu muss die SPS X zum Modbus-Master werden.

Der Supervisor kommuniziert jeden Tag mit den SPS, um Informationen abzufragen.

Wenn ein Alarm an der SPS X vorliegt, schaltet er in den Modbus-Master-Modus und sendet Daten an die SPS Y.

Um die Programmierung zu vereinfachen, wurden die Modems über ein Programmiergerät mit den richtigen Parametern initialisiert. Diese Parameter werden durch die AT&W-Befehle im nicht flüchtigen Speicher gespeichert.

Betriebsart

Nachfolgend ist die Funktionsweise der Anwendung beschrieben:

Schritt	Aktion
1	Der Anschluss der SPS X wird in den Zeichenmodus geschaltet.
2	Die SPS X sendet eine Wählmeldung an das Modem.
3	Der Anschluss der SPS X wird in den Modbus-Master-Modus geschaltet.
4	Die Master-SPS (X) sendet Daten an die Slave-SPS (Y).
5	Der Anschluss wird in den Zeichenmodus geschaltet.
6	Die SPS X sendet eine Trennungsmeldung an das Modem.
7	Der Anschluss der SPS X wird in den Modbus-Slave-Modus geschaltet.

Kapitel 9
Installieren der Anwendung mithilfe von Control Expert

Inhalt dieses Kapitels

In diesem Kapitel wird der zum Erstellen der Anwendung durchzuführende Prozess beschrieben. Das Kapitel enthält sowohl allgemeine als auch ausführlichere Informationen zu den Schritten, die zum Erstellen der verschiedenen Anwendungskomponenten benötigt werden.

Inhalt dieses Kapitels

Dieses Kapitel enthält die folgenden Abschnitte:

Abschnitt 9.1
Beschreibung der verwendeten Lösung

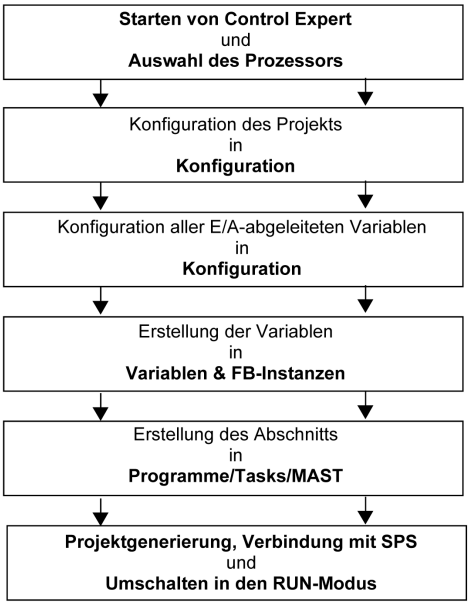
Die verschiedenen Schritte des Prozesses mithilfe von Control Expert

Einführung

Das folgende logische Diagramm zeigt die verschiedenen Schritte, die zum Erstellen der Anwendung ausgeführt werden müssen. Damit alle Anwendungselemente korrekt definiert werden können, muss eine chronologische Reihenfolge eingehalten werden.

Beschreibung

Beschreibung der verschiedenen Typen:



Abschnitt 9.2
Entwickeln der Anwendung

Inhalt dieses Kapitels

Dieser Abschnitt beschreibt die schrittweise Erstellung der Anwendung mithilfe von Control Expert.

Inhalt dieses Abschnitts

Dieser Abschnitt enthält die folgenden Themen:

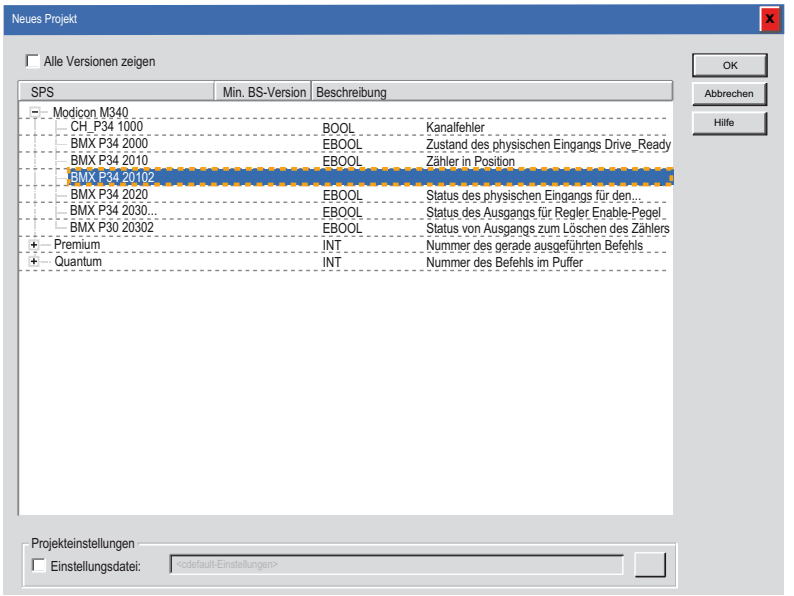
Erstellen des Projekts

Auf einen Blick

Zur Entwicklung der Beispielanwendung muss ein zur SPS X gehöriges Hauptprojekt erstellt werden, das die SPS konfiguriert sowie alle erforderlichen Variablen deklariert und das Programm enthält. Weiterhin muss ein separates Projekt zur Konfiguration von SPS Y erstellt werden.

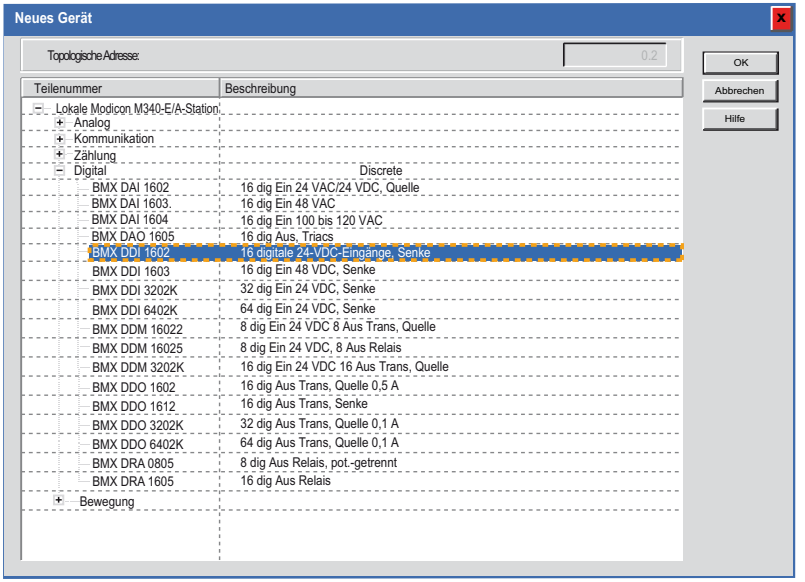
Vorgehensweise zum Erstellen eines Projekts

Die folgende Tabelle zeigt das Verfahren zum Erstellen des Projekts mit Control Expert.

Schritt	Aktion
1	Starten Sie die Software Control Expert.
2	Klicken Sie auf "Datei" und dann auf "Neu", um einen Prozessor des Typs BMX P34 20102 zu wählen. 
3	Bestätigen Sie Ihre Auswahl mit OK.

Auswahl des digitalen Eingangsmoduls

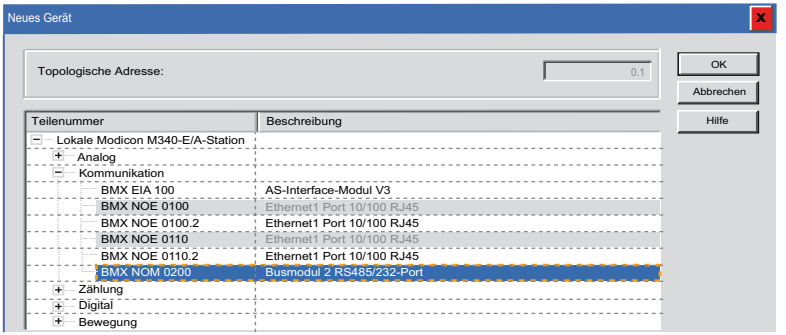
Die folgende Tabelle zeigt die Vorgehensweise zur Auswahl eines digitalen Moduls, das für SPS X erforderlich ist.

Schritt	Aktion
1	Doppelklicken Sie im `Projekt-Browser` auf `Konfiguration` und dann auf `0:SPS-Bus` und dann auf `0:BMX XBP •••` (wobei 0 der Rack-Nummer entspricht).
2	Wählen Sie im Fenster `SPS-Bus` einen Steckplatz (beispielsweise Steckplatz 1) und doppelklicken Sie darauf.
3	Wählen Sie in der Modulliste `Digital` das digitale Eingangsmodul `BMX DDI 1602`. 
4	Bestätigen Sie Ihre Auswahl mit OK.

BMX NOM 0200-Modulauswahl

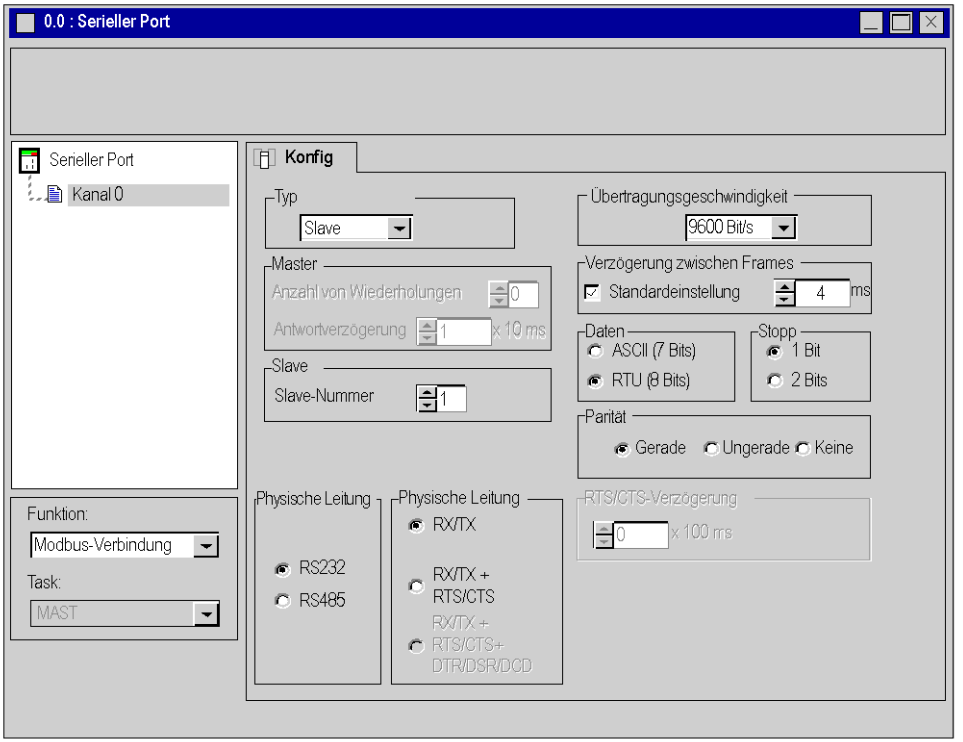
Im Beispiel wird ein BMX NOM 0200-Modul in SPS Y für die serielle Kommunikation mit dem Modem verwendet. Demzufolge muss es in dem zur SPS Y gehörigen Projekt hinzugefügt werden.

Die folgende Tabelle beschreibt die Vorgehensweise zur Auswahl des BMX NOM 0200-Moduls.

Schritt	Aktion
1	Doppelklicken Sie im `Projekt-Browser` auf `Konfiguration` und dann auf `0:SPS-Bus` und dann auf `0:BMX XBP •••` (wobei 0 der Rack-Nummer entspricht).
2	Wählen Sie im Fenster `SPS-Bus` einen Steckplatz (beispielsweise Steckplatz 1) und doppelklicken Sie darauf.
3	Wählen Sie in der Modulliste `Kommunikation` das Kommunikationsmodul `BMX NOM 0200`. 
4	Bestätigen Sie Ihre Auswahl mit OK.

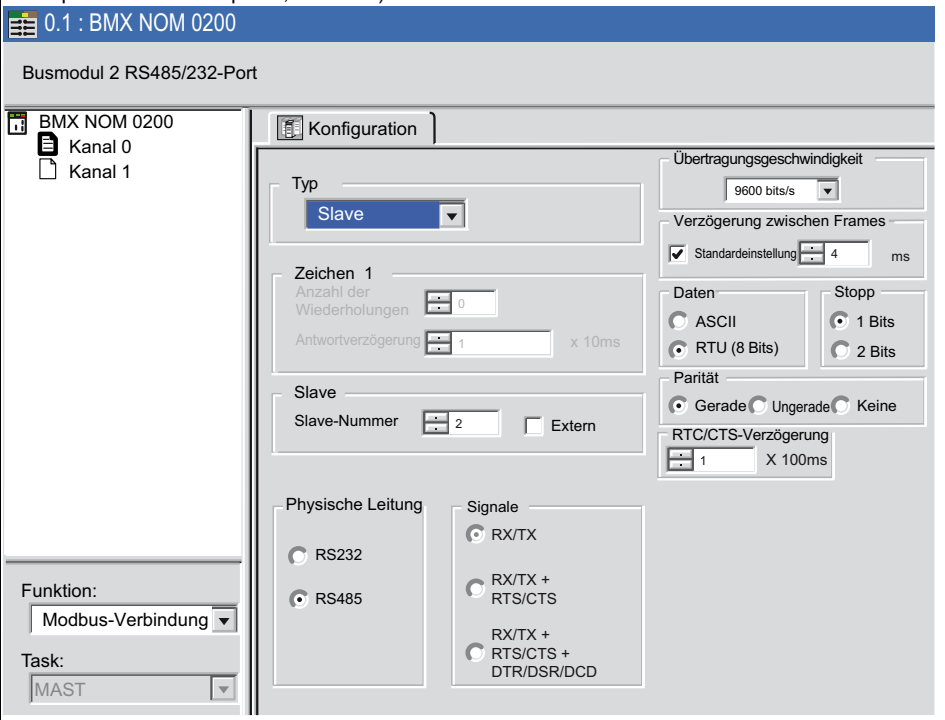
Konfiguration des seriellen Anschlusses des Prozessors

Die folgende Tabelle beschreibt das Verfahren zur Konfiguration des seriellen Anschlusses des SPS X-Prozessors als Modbus-Slave.

Schritt	Aktion
1	Doppelklicken Sie im `Projekt-Browser` auf `Konfiguration` und dann auf `0:BMX XBP 0800` und dann auf `0:BMX P34 20102`. Anschließend doppelklicken Sie auf `Serieller Port`, um das Fenster `0.0:Serieller Port` zu öffnen. 
2	Wählen Sie den Typ `Slave`.
3	Wählen Sie als Übertragungsgeschwindigkeit `9600 Bits/s`.
4	Wählen Sie als physische Leitung `RS232`.
5	Wählen Sie als Datentyp `RTU (8 Bits)`.
6	Schließen Sie das Fenster und bestätigen Sie mit "OK".

Serielle Kanalkonfiguration des BMX NOM 0200

Die folgende Tabelle beschreibt das Verfahren zur Konfiguration des seriellen Kanals des BMX NOM 0200-Moduls für SPS Y als Modbus-Slave.

Schritt	Aktion
1	Doppelklicken Sie im `Projekt-Browser` auf `Konfiguration` und dann auf `0:BMX XBP 0800` und dann auf `0:BMX NOM 0200`, um das Fenster `0.x:BMX NOM 0200` zu öffnen (wobei x der Steckplatznummer entspricht, z. B. x=1). 
2	Wählen Sie den `Kanal 0`.
3	Wählen Sie die `Modbus-Verbindung` als Funktion.
4	Wählen Sie als Typ `Slave`.
5	Wählen Sie als Übertragungsgeschwindigkeit `9600 Bits/s`.
6	Wählen Sie als physische Leitung `RS232`.
7	Wählen Sie als Signale `RX/TX + RTS/CTS + DTR/DSR/DCD`.
8	Wählen Sie als RTS/CTS-Verzögerung `100 ms`.
9	Wählen Sie als Datentyp `RTU (8 Bits)`.
10	Schließen Sie das Fenster und bestätigen Sie mit "OK".

Variablendeklaration

Einführung

Alle in den verschiedenen Abschnitten des Programms verwendeten Variablen müssen deklariert werden.

Nicht vereinbarte Variablen können im Programm nicht verwendet werden.

HINWEIS: Weitere Informationen finden Sie im Kapitel *Dateneditor (siehe EcoStruxure™ Control Expert, Betriebsarten)*.

Prozedur zum Deklarieren von Variablen

Die folgende Tabelle zeigt die Prozedur zum Deklarieren von Anwendungsvariablen:

Schritt	Aktion
1	Doppelklicken Sie im `Projekt-Browser \ Variablen und FB-Instanzen` auf `Elementare Variablen`
2	Aktivieren Sie im `Dateneditor` das Kontrollkästchen in der Spalte Name und geben Sie dann den Namen Ihrer ersten Variable ein.
3	Wählen Sie jetzt einen Typ für diese Variable.
4	Wenn alle Variablen deklariert wurden, können Sie das Fenster schließen.

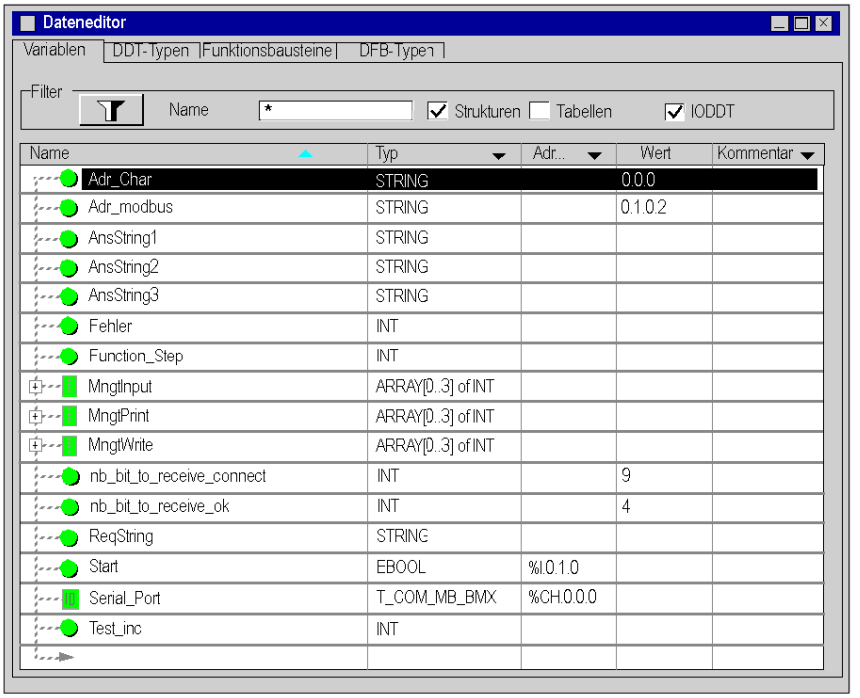
Für die Anwendung verwendete Variablen

Die folgende Tabelle zeigt die Details der in der Anwendung verwendeten Variablen, die in dem zur SPS X gehörigen Projekt deklariert sind:

Variable	Typ	Definition
Adr_Char	STRING	Adresse des seriellen Ports der Master-SPS
Adr_modbus	STRING	Serielle Kanaladresse der Modbus-Slave-SPS (Kanal 0 des BMX NOM 0200-Moduls).
AnsString1	STRING	Erste Modemantwort-Zeichenfolge
AnsString2	STRING	Zweite Modemantwort-Zeichenfolge
AnsString3	STRING	Dritte Modemantwort-Zeichenfolge
Fehler	INT	Funktionsfehlercode
Function_Step	INT	Funktionsschritt
MngtInput	ARRAY[0..3] of INT	Array der Kommunikationsparameter für den Block INPUT_CHAR
MngtPrint	ARRAY[0..3] of INT	Array der Kommunikationsparameter für den Block PRINT_CHAR

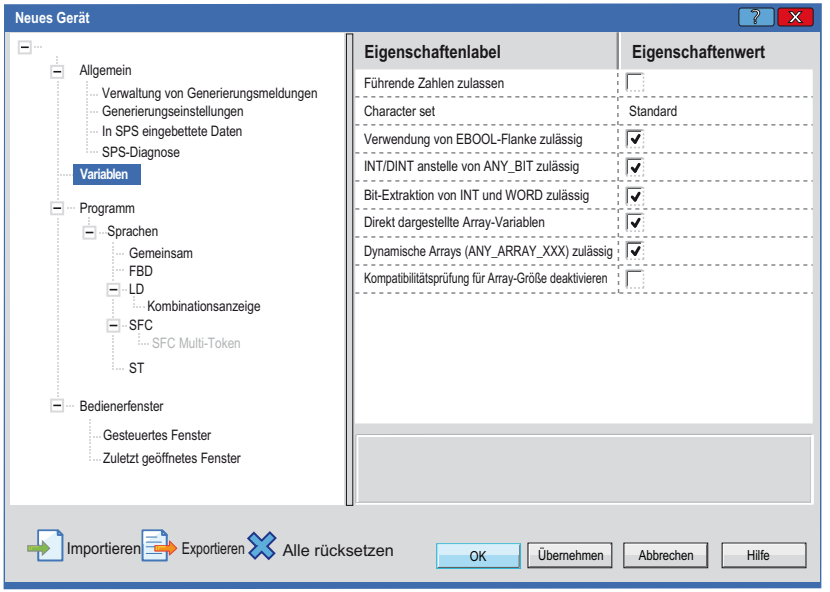
Variable	Typ	Definition
MngtWrite	ARRAY[0..3] of INT	Array der Kommunikationsparameter für den Block WRITE_VAR
nb_charac_to_receive_connect	INT	Anzahl der zu empfangenden Zeichen: Modemverbindung
nb_charac_to_receive_ok	INT	Anzahl der zu empfangenden Zeichen: Modem-Bestätigungsmeldung
ReqString	STRING	Modemantwort
Start	EBOOL	Startmodus (Signal von Kanal 0 des BMX DDI 1602-Moduls).
Serial_Port	T_COM_MB_BMX	E/A-Objekt des seriellen Ports
Test_inc	INT	Inkrementierungswert

Die folgende Abbildung zeigt die mit Hilfe des Dateneditors erstellten Anwendungsvariablen.

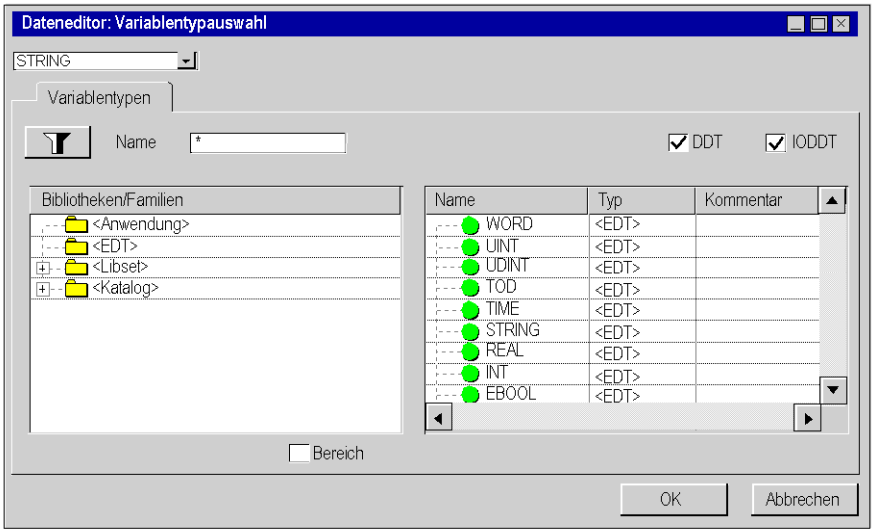


Deklarieren eines Array-Typs

Klicken Sie vor dem Deklarieren eines Array-Typs auf **Extras/Projekteinstellungen/Variablen** und aktivieren Sie dann die Optionen „Direkt dargestellte Array-Variablen“ und „Dynamische Arrays zulässig“.



Die folgende Tabelle beschreibt die Vorgehensweise zur Deklaration eines Array-Typs.

Schritt	Aktion
1	Klicken Sie im `Projekt-Browser` auf `Variablen und FB-Instanzen`.
2	Klicken Sie auf die Spalte `Name` und geben Sie einen Namen für die Variable ein.
3	Doppelklicken Sie in der Spalte `Typ` auf ... . Das Fenster `Variablentypauswahl` wird angezeigt: 
4	Wählen Sie den gewünschten Variablentyp (klicken Sie zum Beispiel auf `<EDT>` und wählen Sie `INT`) und klicken Sie dann auf das Array-Kontrollkästchen. 
5	Ändern Sie das Intervall und bestätigen Sie dann mit `OK`.

Deklaration von E/A-Objekten

Zum Deklarieren von E/A-abgeleiteten Variablen öffnen Sie das Fenster `Variablen-typauswahl`, wie in der obigen Vorgehensweise beschrieben, und klicken Sie auf `<Katalog>`, um auf die `<IODDT>` Variablentypen zuzugreifen (und im Beispiel `T COM MB BMX` auszuwählen). Bestätigen Sie mit `OK`.

Verwendung eines Modems

Beschreibung

Für die Vernetzung von Telefonmodems mit SPS sind drei Befehle erforderlich. Nachfolgend sind diese Befehle aufgeführt:

- Modem initialisieren
- Neu wählen
- Modem trennen

Vor dem Senden einer ASCII- oder Modbus-Nachricht müssen Sie zunächst eine Initialisierungsnachricht, gefolgt von einer Anwahlnachricht an das Modem senden.

Wenn die Verbindung zwischen den beiden Modems erfolgreich aufgebaut wurde, können Sie eine unbegrenzte Anzahl von ASCII- oder Modbus-Nachrichten senden.

Wenn alle Nachrichten gesendet sind, müssen Sie die Trennungs-Zeichenfolge an das Modem senden.

Initialisieren des Modems

Die beiden Modems müssen mit denselben Merkmalen wie die seriellen Anschlüsse konfiguriert werden:

- Datenrate: 9600 Baud
- Zeichenrahmen: 8 Bits/Parität: gerade/1 Stoppbit
- Leitungsmodulation: V32.

Definieren Sie dann ‘ ’ +’ ’ als Escape-Zeichen (Befehl: ATS2=43).

Beispiel für die Initialisierung des Befehls:

ATQ0&Q0E0&K0V1

mit:

- Q0: Ergebniscode aktivieren
- &Q0: DTR wird immer als (EIN) vorausgesetzt
- E0: Echo der Zeichen deaktivieren
- &K0: keine Datenflusskontrolle
- V1: Wortergebniscodes

Anwählen des Modems

Die Wählmeldung wird verwendet, um die Telefonnummer an das Modem zu senden.

Nur im Zusammenhang mit dem Wählen stehende AT-Befehle sollten in die Meldung aufgenommen werden.

Beispiel:

- **Frequenzwahl:** ATDT6800326<CR><LR>
- **Impulswahl:** ATDP6800326<CR><LF>
- **Frequenzwahl mit Warten auf den Ton:** ATDTW6800326<CR><LF>

Trennen des Modems

Das Modem wird zunächst in den Befehlsmodus zurückversetzt, indem es drei Mal das Escape-Zeichen empfängt.

Dann kann der Trennbefehl "ATH0" gesendet werden.

Escape-Sequenz: "+++" (Modem-Ergebniscode: OK).

Trennbefehl: "ATH0" (Modem-Ergebniscode: OK).

Programmierverfahren

Zu befolgendes Verfahren

In der folgenden Tabelle ist das Verfahren zum Programmieren der Anwendung beschrieben.

Schritt	Aktion	Details
1	Vorbereiten des Kommunikationsanschlusses	• Ändern Sie den Modbus-Slave-Modus in den Zeichenmodus, indem Sie an den seriellen Anschluss einen WRITE_CMD-Befehl senden *(siehe Seite 148)*. • Senden Sie bei einer Modem-Übertragung den HAYES-Befehl mithilfe des Blocks PRINT_CHAR, um das Modem zu konfigurieren *(siehe Seite 145)*. • Senden Sie bei einer Modem-Übertragung den HAYES-Befehl mithilfe des Blocks PRINT_CHAR. Die Wählmeldung wird verwendet, um die Telefonnummer an das Modem zu senden *(siehe Seite 145)*.
2	Modbus-Master-Modus	• Schalten Sie mithilfe des WRITE_CMD-Befehls *(siehe Seite 148)* in den Modbus-Master-Modus. • Senden Sie die in die Slave-SPS zu schreibenden Daten.
3	Rücksetzen des Kommunikationsanschlusses	• Schalten Sie mithilfe des WRITE_CMD-Befehls *(siehe Seite 148)* in den Zeichenmodus. • Senden Sie bei einer Modem-Übertragung das Escape-Zeichen, senden Sie dann den Trennbefehl, um mithilfe des Blocks PRINT_CHAR eine Trennnachricht an das Modem zu senden *(siehe Seite 146)*. • Kehren Sie mithilfe des WRITE_CMD-Befehls *(siehe Seite 148)* zum Startmodus des seriellen Anschlusses (Modbus-Slave) zurück.

Schreiben der Befehlswörter

Befolgen Sie die nachfolgend aufgeführten Schritte, um einen WRITE_CMD-Befehl an einen Kommunikationsanschluss zu senden.

Schritt	Aktion	Details
1	Test, um zu ermitteln, ob ein Befehl aussteht.	Testen Sie vor der Ausführung des Befehls WRITE_CMD mithilfe des Sprachobjekts `EXCH_STS` (%MWr.m.c.0), ob derzeit ein Austausch läuft. Verwenden Sie den Block READ_STS, um dieses Wort zu aktualisieren.
2	Zuweisen des Befehlsworts	Als nächstes müssen Sie den Wert des Befehls-Sprachobjekts ändern, um den gewünschten Befehl auszuführen. Für eine Modbus-Verbindung ist die Objektsprache das interne Wort `CONTROL` (%MWr.m.c.24). Um beispielsweise vom Modbus-Modus in den Zeichenmodus umzuschalten, wird Bit 14 des Worts %MWr.m.c.24 auf 1 gesetzt. **Hinweis:** Ein einzelnes Befehlsbit muss dann vor der Übertragung des Befehls WRITE_CMD von 0 auf 1 gesetzt werden.
3	Befehl senden	Um den Befehl zu quittieren muss abschließend der Befehl WRITE_CMD ausgeführt werden.

Programmierstruktur

Schrittkommentare

Schrittnummer	Schrittbeschreibung	Element
0	Initialstatus der Funktion Wenn das Startbit auf 1 wechselt, den Fehler mit 0 initialisieren und mit Schritt 5 fortfahren.	Modem
5	Status des seriellen Ports lesen und prüfen, dass kein Befehl aktiv ist. In den Zeichenmodus umschalten und Zähler Test_inc mit 0 initialisieren. Fahren Sie mit Schritt 10 fort.	
10	Status des seriellen Ports lesen und prüfen, dass kein Befehl aktiv ist. TO_CHAR_MODE-Befehlsbit zurücksetzen. ● Wenn kein Fehler am seriellen Port aufgetreten ist ❍ und der Zeichenmodus aktiv ist, mit Schritt 15 fortfahren. ❍ und der Zeichenmodus nicht aktiv ist, dann Test_inc inkrementieren und Schritt 10 bis zu 1000-mal wiederholen. Nach 1000 fehlgeschlagenen Wiederholungen Fehler auf 10 setzen und mit Schritt 130 fortfahren. ● Wenn ein Fehler am seriellen Port aufgetreten ist, dann ❍ Fehler auf 10 setzen. ❍ Fahren Sie mit Schritt 130 fort.	
15	Mit dem Block PRINT_CHAR Wählbefehl an das Modem senden. Fahren Sie mit Schritt 20 fort.	
20	Wenn das Ergebnis von PRINT_CHAR schlüssig ist, dann mit Schritt 25 fortfahren, ansonsten Fehler auf 20 setzen und mit Schritt 130 fortfahren.	
25	Mit dem Block INPUT_CHAR auf Antwort vom Modem warten. Nachdem die Antwortzeichenkette vollständig empfangen wurde, mit Schritt 30 fortfahren.	
30	Wenn das Ergebnis von INPUT_CHAR schlüssig ist, dann mit Schritt 35 fortfahren, ansonsten Fehler auf 30 setzen und mit Schritt 130 fortfahren.	
35	Wenn das Modem wie erwartet antwortet, dann mit Schritt 40 fortfahren, ansonsten Fehler auf 35 setzen und mit Schritt 130 fortfahren.	

Schrittnummer	Schrittbeschreibung	Element
40	Status des seriellen Ports lesen und prüfen, dass kein Befehl aktiv ist. In den Modbus-Master-Modus umschalten und Zähler Test_inc mit 0 initialisieren. Fahren Sie mit Schritt 45 fort.	Modbus-Master-Modus
45	Status des seriellen Ports lesen und prüfen, dass kein Befehl aktiv ist. TO_CHAR_MODE-Befehlsbit zurücksetzen. ● Wenn kein Fehler am seriellen Port aufgetreten ist ❍ und der Zeichenmodus aktiv ist, mit Schritt 50 fortfahren ❍ und der Zeichenmodus nicht aktiv ist, dann Test_inc inkrementieren und Schritt 45 bis zu 1000-mal wiederholen. Nach 1000 fehlgeschlagenen Wiederholungen Fehler auf 45 setzen und mit Schritt 130 fortfahren. ● Wenn ein Fehler am seriellen Port aufgetreten ist, dann ❍ Fehler auf 45 setzen. ❍ Fahren Sie mit Schritt 130 fort.	
50	Initialisierung der Parameter des Blocks WRITE_VAR. Senden der in die SPS zu schreibenden Daten mit Hilfe der Funktion WRITE_VAR. Fahren Sie mit Schritt 55 fort.	Funktion schreiben
55	Wenn das Ergebnis von WRITE_CHAR schlüssig ist, dann mit Schritt 60 fortfahren, ansonsten Fehler auf 55 setzen und mit Schritt 130 fortfahren.	
60	Status des seriellen Ports lesen und prüfen, dass kein Befehl aktiv ist. In den Zeichenmodus umschalten und Zähler Test_inc mit 0 initialisieren. Fahren Sie mit Schritt 65 fort	Zeichenmodus
65	Status des seriellen Ports lesen und prüfen, dass kein Befehl aktiv ist. TO_CHAR_MODE-Befehlsbit zurücksetzen. ● Wenn kein Fehler am seriellen Port aufgetreten ist ❍ und der Zeichenmodus aktiv ist, mit Schritt 70 fortfahren. ❍ und der Zeichenmodus nicht aktiv ist, dann Test_inc inkrementieren und Schritt 65 bis zu 1000-mal wiederholen. Nach 1000 fehlgeschlagenen Wiederholungen Fehler auf 65 setzen und mit Schritt 130 fortfahren. ● Wenn ein Fehler am seriellen Port aufgetreten ist, dann ❍ Fehler auf 65 setzen. ❍ Fahren Sie mit Schritt 130 fort.	

Schrittnummer	Schrittbeschreibung	Element
70	Senden einer Escape-Sequenz an das Modem mithilfe des Blocks PRINT_CHAR. Fahren Sie mit Schritt 75 fort.	Modem
75	Wenn das Ergebnis von PRINT_CHAR schlüssig ist, dann mit Schritt 80 fortfahren, ansonsten Fehler auf 75 setzen und mit Schritt 130 fortfahren.	
80	Mit dem Block INPUT_CHAR auf Antwort vom Modem warten. Nachdem die Antwortzeichenkette vollständig empfangen wurde, mit Schritt 85 fortfahren.	
85	Wenn das Ergebnis von INPUT_CHAR schlüssig ist, dann mit Schritt 90 fortfahren, ansonsten Fehler auf 85 setzen und mit Schritt 130 fortfahren.	
90	Wenn das Modem wie erwartet antwortet, dann mit Schritt 95 fortfahren, ansonsten Fehler auf 90 setzen und mit Schritt 130 fortfahren.	
95	Senden eines Trennungsbefehls an das Modem mithilfe des Blocks PRINT_CHAR. Fahren Sie mit Schritt 100 fort.	
100	Wenn das Ergebnis von PRINT_CHAR schlüssig ist, dann mit Schritt 105 fortfahren, ansonsten Fehler auf 100 setzen und mit Schritt 130 fortfahren.	
105	Mit dem Block INPUT_CHAR auf Antwort vom Modem warten. Nachdem die Antwortzeichenkette vollständig empfangen wurde, mit Schritt 110 fortfahren.	
110	Wenn das Ergebnis von INPUT_CHAR schlüssig ist, dann mit Schritt 115 fortfahren, ansonsten Fehler auf 110 setzen und mit Schritt 130 fortfahren.	
115	Wenn das Modem wie erwartet antwortet, dann mit Schritt 120 fortfahren, ansonsten Fehler auf 115 setzen und mit Schritt 130 fortfahren.	
120	Status des seriellen Ports lesen und prüfen, dass kein Befehl aktiv ist. In den Modbus-Slave-Modus umschalten und Zähler Test_inc mit 0 initialisieren. Fahren Sie mit Schritt 125 fort.	Modbus-Slave-Modus
125	Status des seriellen Ports lesen und prüfen, dass kein Befehl aktiv ist. TO_CHAR_MODE-Befehlsbit zurücksetzen. ● Wenn kein Fehler am seriellen Port aufgetreten ist ❍ und der Zeichenmodus aktiv ist, mit Schritt 130 fortfahren. ❍ und der Zeichenmodus nicht aktiv ist, dann Test_inc inkrementieren und Schritt 125 bis zu 1000-mal wiederholen. Nach 1000 fehlgeschlagenen Wiederholungen Fehler auf 125 setzen und mit Schritt 130 fortfahren. ● Wenn ein Fehler am seriellen Port aufgetreten ist, dann ❍ Fehler auf 125 setzen. ❍ Fahren Sie mit Schritt 130 fort.	
130	Rückkehr zu Schritt 0.	

Programmierung

Programmieren in der ST-Sprache

Das Beispiel ist in der ST-Sprache (Strukturierter Text) programmiert. Die entsprechende Section befindet sich unter derselben Master-Task (MAST).

```
CASE Function_Step OF
0: (* Initialisierung *)
 IF (Start) THEN (* trigger flag *)
 Error := 0;
 Function_Step := 5; (* next step *)
 END_IF;
5: (* Befehl senden, um den seriellen Port vom Modbus-Slave-Modus in den Zeichenmodus umzuschalten *)
 READ_STS(Serial_port); (* Status des seriellen Ports lesen *)
 IF (Serial_port.EXCH_STS = 0) THEN (* kein aktiver Befehl *)
 Serial_port.CONTROL := 16#00; (* Steuerwort zurücksetzen *)
 (* TO_CHAR_MODE-Befehlsbit setzen *)
 SET(Serial_port.TO_CHAR_MODE);
 WRITE_CMD (Serial_port); (* Befehl senden *)
 Test_inc := 0; (* Wiederholungszähler initialisieren *)
 Function_Step := 10; (* next step *)
 END_IF;
10: (* Ergebnis des Umschaltbefehls in den Zeichenmodus testen *)
 READ_STS(Serial_port); (* Status des seriellen Ports lesen *)
 IF (Serial_port.EXCH_STS = 0) THEN (* Befehl ausgeführt *)
 (* TO_CHAR_MODE-Befehlsbit zurücksetzen *)
 RESET(Serial_port.TO_CHAR_MODE);
 IF (Serial_port.EXCH_RPT = 0) THEN (* no Fehler *)
 IF (AND(Serial_port.PROTOCOL, 16#0F) = 03)
 THEN (* Zeichenmodus OK *)
 Function_Step := 15; (* next step *)
 ELSE
 Test_inc := Test_inc + 1;
 IF (Test_inc > 1000) THEN
```

```
 Error := 10; (* Fehler *)
 Function_Step := 130; (* nächster Schritt = Ende *)
 END_IF;
 END_IF;
 ELSE (* Fehler beim Senden des Befehls an den Port *)
 Error := 10; (* Fehler *)
 Function_Step := 130;
 END_IF;
 END_IF;

15: (* Wählbefehl an Modem senden *)
 (* Die Telefonnummer muss zwischen ‘ ATDT’ und ‘ $N’ eingefügt werden *)
 ReqString := 'ATDT4001$N'; (* Wählmeldung *)
 MngtPrint[2] := 500; (* Timeout *)
 MngtPrint[9] := 9; (* Größe des Austauschs in Byte *)
PRINT_CHAR(ADDM(Adr_Char), ReqString, MngtPrint);
 Function_Step := 20;
20: (* Ergebniss der Funktion PRINT_CHAR testen *)
 IF (NOT MngtPrint[0].0) THEN
 IF (MngtPrint[1] = 0) THEN
 Function_Step := 25; (* Erfolg: Nächster Schritt *)
 ELSE
 Error := 20; (* Fehler *)
 Function_Step := 130; (* nächster Schritt = Ende *)
 END_IF;
 END_IF;
25: (* Auf Antwort über INPUT_CHAR warten *)
 MngtInput[2] := 500; (* Timeout *)
 AnsString1:=' ';
 (* Warten auf Modem-Antwort *)
 INPUT_CHAR(ADDM(Adr_Char), 1, nb_charac_to_receive_connect, MngtInput, AnsString1);
 Function_Step := 30; (* next step *)
```

```
30: (* Ergebniss der Funktion INPUT_CHAR testen *)
 IF (NOT MngtInput[0].0) THEN
 IF (MngtInput[1] = 0) THEN
 Function_Step := 35; (* Erfolg: Nächster Schritt *)
 ELSE
 Error := 30; (* Fehler *)
 Function_Step := 130; (* nächster Schritt = Ende *)
 END_IF;
 END_IF;

35: (* Modem-Antwort testen *)
 IF (AnsString1 = '$NCONNET') THEN
 Function_Step := 40; (* Nächster Schritt *)
 ELSE
 Error := 35; (* Fehler *)
 Function_Step := 130; (* nächster Schritt = Ende *)
 END_IF;

40: (* Befehl senden, um den seriellen Port vom Zeichenmodus in den Modbus-Master-Modus
umzuschalten *)
 READ_STS(Serial_port); (* Status des seriellen Ports lesen *)
 IF (Serial_port.EXCH_STS = 0) THEN (* kein aktiver Befehl *)
 Serial_port.CONTROL := 16#00; (* Steuerwort zurücksetzen *)
 (* TO_MODBUS_MASTER-Befehlsbit setzen *)
 SET(Serial_port.TO_MODBUS_MASTER);
 WRITE_CMD (Serial_port); (* Befehl senden *)
 Test_inc := 0; (* Wiederholungszähler initialisieren *)
 Function_Step := 45; (* next step *)
 END_IF;

45: (* Ergebnis des Umschaltbefehls in den Modbus-Master-Modus testen*)
 READ_STS(Serial_port); (* Status des seriellen Ports lesen *)
 IF (Serial_port.EXCH_STS = 0) THEN (* Befehl ausgeführt *)
 (* TO_MODBUS_MASTER-Befehlsbit *)
```

```
RESET(Serial_port.TO_MODBUS_MASTER);
IF (Serial_port.EXCH_RPT = 0) THEN (* no Fehler *)
IF (AND(Serial_port.PROTOCOL, 16#0F) = 06)
THEN (* Modbus-Master-Modus OK *)
Function_Step := 50; (* next step *)
ELSE
Test_inc := Test_inc + 1;
IF (Test_inc > 1000) THEN
Error := 45; (* Fehler *)
Function_Step := 130; (* nächster Schritt = Ende *)
END_IF;
END_IF;
ELSE (* Fehler beim Senden des Befehls an den Port *)
Error := 45; (* Fehler *)
Function_Step := 130;
END_IF;
END_IF;

50: (* Informationen in die zweite CPU schreiben *)
Mngtwrite[2]:=50; (* Timeouts *)
%MW40:=5; (* zu sendender Wert *)
WRITE_VAR(ADDM(Adr_modbus),'%MW',100,2,%MW40:2,Mngtwrite);
Function_Step := 55;

55: (* Ergebniss der Funktion WRITE_VAR testen *)
IF (NOT Mngtwrite[0].0) THEN
IF (Mngtwrite[1] = 0) THEN
Function_Step := 60; (* Erfolg: Nächster Schritt *)
ELSE
Error := 55; (* Fehler *)
Function_Step := 130; (* nächster Schritt = Ende *)
END_IF;
END_IF;
```

```
60: (* Befehl senden, um den seriellen Port vom Modbus- in den Zeichenmodus umzuschalten *)
 READ_STS(Serial_port); (* Status des seriellen Ports lesen *)
 IF (Serial_port.EXCH_STS = 0) THEN (* kein aktiver Befehl *)
 Serial_port.CONTROL := 16#00; (* Steuerwort zurücksetzen *)
 (* TO_CHAR_MODE-Befehlsbit setzen *)
 SET(Serial_port.TO_CHAR_MODE);
 WRITE_CMD (Serial_port); (* Befehl senden *)
 Test_inc := 0; (* Wiederholungszähler initialisieren *)
 Function_Step := 65; (* next step *)
 END_IF;

65: (* Ergebnis des Umschaltbefehls testen*)
 READ_STS(Serial_port); (* Status des seriellen Ports lesen *)
 IF (Serial_port.EXCH_STS = 0) THEN (* Befehl ausgeführt *)
 (* TO_CHAR_MODE-Befehlsbit zurücksetzen *)
 RESET(Serial_port.TO_CHAR_MODE);
 IF (Serial_port.EXCH_RPT = 0) THEN (* no Fehler *)
 IF (AND(Serial_port.PROTOCOL, 16#0F) = 03)
 THEN (* Zeichenmodus OK *)
 Function_Step := 70; (* next step *)
 ELSE
 Test_inc := Test_inc + 1;
 IF (Test_inc > 1000) THEN
 Error := 65; (* Fehler *)
 Function_Step := 130; (* nächster Schritt = Ende *)
 END_IF;
 END_IF;
 ELSE (* Fehler beim Senden des Befehls an den Port *)
 Error := 65; (* Fehler *)
 Function_Step := 130; (* nächster Schritt = Ende *)
 END_IF;
 END_IF;
```

```
70: (* Modem auflegen: Schritt 1 *)
 ReqString := '+++'; (* Escape-Sequenz *)
 MngtPrint[3] := 3; (* Größe des Austauschs in Byte *)
 PRINT_CHAR(ADDM(Adr_Char), ReqString, MngtPrint);
 Function_Step := 75; (* next step *)

75: (* Ergebniss der Funktion PRINT_CHAR testen *)
 IF (NOT MngtPrint[0].0) THEN
 IF (MngtPrint[1] = 0) THEN
 (* Erfolg: Nächster Schritt *)
 Function_Step := 80;
 ELSE
 (* Ende bei Fehler *)
 Error := 75;
 Function_Step := 130;
 END_IF;
 END_IF;
80:
 MngtInput[2] := 50; (* Timeout *)
 INPUT_CHAR(ADDM(Adr_Char), 1, nb_charac_to_receive_ok, MngtInput, AnsString2); (*Warten
auf Modem-Antwort *)
 Function_Step := 85; (* next step*)
85: (* Ergebniss der Funktion INPUT_CHAR testen *)
 IF (NOT MngtInput[0].0) THEN
 IF (MngtInput[1] = 0) THEN
 (* Erfolg: Nächster Schritt *)
 Function_Step := 90;
 ELSE
 (* Ende bei Fehler *)
 Error := 85;
 Function_Step := 130;
 END_IF;
 END_IF;
```

```
90: (* Modem-Antwort testen *)
 IF (AnsString2 = '$NOK') THEN
 Function_Step := 95; (* Nächster Schritt *)
 ELSE
 Error := 90; (* Fehler *)
 Function_Step := 130; (* nächster Schritt = Ende *)
 END_IF;
95: (* Modem auflegen: Schritt 2 *)
 ReqString := 'ATH0$N'; (* Auflegemeldung *)
 MngtPrint[3] := 3; (* Größe des Austauschs in Byte *)
 PRINT_CHAR(ADDM(Adr_Char), ReqString, MngtPrint);
 Function_Step := 100; (* next step *)
100: (* Ergebniss der Funktion PRINT_CHAR testen *)
 IF (NOT MngtPrint[0].0) THEN
 IF (MngtPrint[1] = 0) THEN
 (* Erfolg: Nächster Schritt *)
 Function_Step := 105;
 ELSE
 (* Ende bei Fehler *)
 Error := 100;
 Function_Step := 130;
 END_IF;
 END_IF;
105:
 MngtInput[2] := 50; (* Timeout *)
 INPUT_CHAR(ADDM(Adr_Char), 1, nb_charac_to_receive_ok, MngtInput, AnsString3); (*Warten
auf Modem-Antwort *)
 Function_Step := 110; (* next step*)
110: (* Ergebniss der Funktion INPUT_CHAR testen *)
 IF (NOT MngtInput[0].0) THEN
 IF (MngtInput[1] = 0) THEN
 (* Erfolg: Nächster Schritt *)
 Function_Step := 115;
 ELSE
```

```
(* Ende bei Fehler *)
Error := 110;
Function_Step := 130;
END_IF;
END_IF;
115: (* Modem-Antwort testen *)
IF (AnsString3 = '$NOK') THEN
Function_Step := 120; (* Nächster Schritt *)
ELSE
Error := 115; (* Fehler *)
Function_Step := 130; (* nächster Schritt = Ende *)
END_IF;
120: (* Befehl senden, um den seriellen Port vom Zeichenmodus in den Modbus-Slave-Modus
umzuschalten *)
READ_STS(Serial_port); (* Status des seriellen Ports lesen *)
IF (Serial_port.EXCH_STS = 0) THEN (* kein aktiver Befehl *)
Serial_port.CONTROL := 16#00; (* Steuerwort zurücksetzen *)
(* TO_MODBUS_SLAVE-Befehlsbit setzen *)
SET(Serial_port.TO_MODBUS_SLAVE);
WRITE_CMD (Serial_port); (* Befehl senden *)
Test_inc := 0; (* Wiederholungszähler initialisieren *)
Function_Step := 125; (* next step *)
END_IF;

125: (* Ergebnis des Umschaltbefehls testen*)
READ_STS(Serial_port); (* Status des seriellen Ports lesen *)
IF (Serial_port.EXCH_STS = 0) THEN (* Befehl ausgeführt *)
(* TO_MODBUS_SLAVE-Befehlsbit zurücksetzen *)
RESET(Serial_port.TO_MODBUS_SLAVE);
IF (Serial_port.EXCH_RPT = 0) THEN (* no Fehler *)
IF (AND(Serial_port.PROTOCOL, 16#0F) = 07)
THEN (* Zeichenmodus OK *)
Function_Step := 130; (* next step *)
ELSE
```

```
 Test_inc := Test_inc + 1;
 IF (Test_inc > 1000) THEN
 Error := 125; (* Fehler *)
 Function_Step := 130; (* nächster Schritt = Ende *)
 END_IF;
 END_IF;
 ELSE (* Fehler beim Senden des Befehls an den Port *)
 Error := 125; (* Fehler *)
 Function_Step := 130; (* nächster Schritt = Ende *)
 END_IF;
 END_IF;
130: (* Ende*)
 IF (NOT Start) THEN (* trigger flag *)
 Function_Step := 0; (* in Wartezustand wechseln *)
 END_IF;
END_CASE;
```

Kapitel 10
Starten der Anwendung

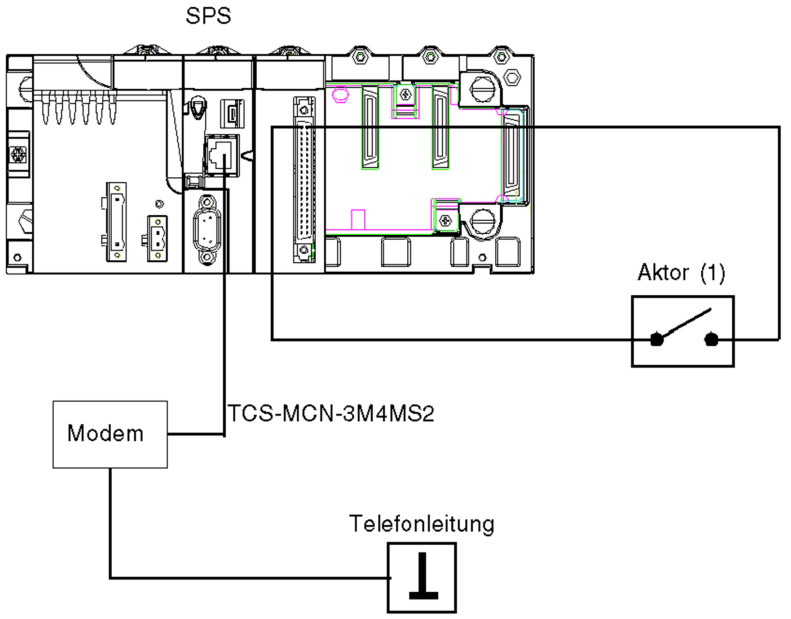
Ausführung der Anwendung im Standardmodus

Einführung

Im Beispiel werden für den Standardmodus zwei SPS, ein digitales Eingangsmodul, ein BMX NOM 0200-Modul und zwei SR2MOD01-Modems benötigt.

Verdrahtung der ersten Slave-SPS

Die erste Slave-SPS wird wie folgt angeschlossen:



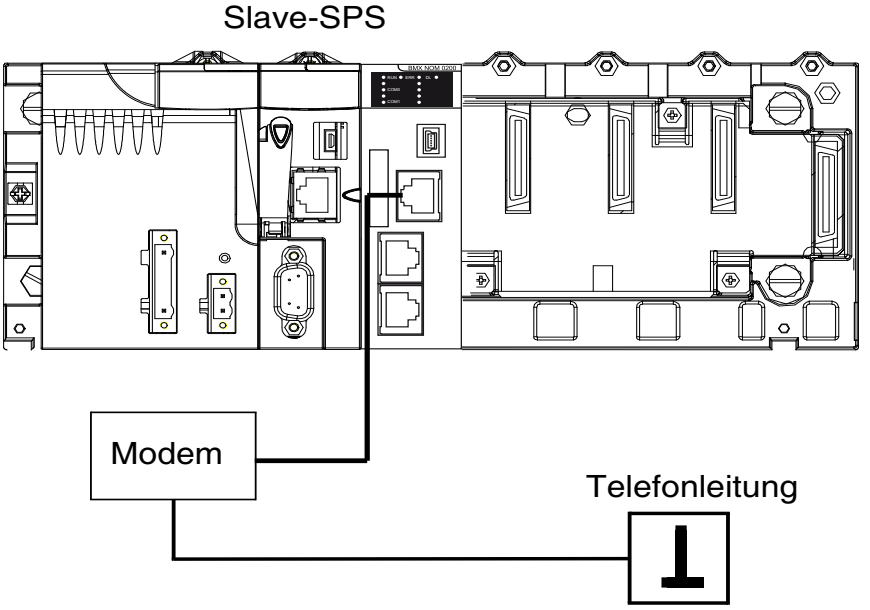
(1): Der Aktor ist mit Kanal 0 des Digitalmoduls verbunden.

Im Beispiel wird das ersten Modem an den seriellen Port des Prozessors der ersten Slave-SPS angeschlossen.

Der Aktorstatus steuert den Status der Variablen `Start` in der Anwendung.

Verdrahtung der zweiten Slave-SPS

Die zweite Slave-SPS wird wie folgt angeschlossen:



Im Beispiel wird das zweite Modem an Kanal 0 des BMX NOM 0200-Moduls der zweiten Slave-SPS angeschlossen.

Für eine höhere Zuverlässigkeit der Kommunikation wird ein Kabel TCS XCN 3M4F3S4 verwendet, das eine Verarbeitung der Modemsignale DTR/DSR/DCD in der Anwendung ermöglicht.

Konfiguration der zweiten Slave-SPS

Bevor das Projekt zum Konfigurieren der zweiten Slave-SPS übertragen wird, stellen Sie sicher, dass diese nicht mit dem Modem verbunden ist.

Die folgende Tabelle beschreibt das Verfahren zum Übertragen der Anwendung im Standardmodus:

Schritt	Aktion
1	Klicken Sie im Menü `Steuerung` auf `Standardmodus`.
2	Klicken Sie im Menü `Generierung` auf `Gesamtes Projekt generieren`. Ihr Projekt wird generiert und ist für die Übertragung an die SPS bereit.
3	Klicken Sie im Menü `Steuerung` auf `Verbinden`. Sie sind jetzt mit der Steuerung verbunden.
4	Klicken Sie im Menü `Steuerung` auf `Projekt an SPS übertragen`. Das Fenster `Projekt an SPS übertragen` wird geöffnet. Klicken Sie auf `Übertragen`. Die Anwendung wird auf die Steuerung übertragen.
5	Verbinden Sie die zweite Slave-SPS mit einem Modem des Typs SR2 MOD01.

Anwendungsübertragung an die erste Slave-SPS

Überprüfen Sie vor der Übertragung der Anwendung, dass die erste Slave-SPS nicht mit dem Modem verbunden ist.

Die folgende Tabelle beschreibt das Verfahren zum Übertragen der Anwendung im Standardmodus:

Schritt	Aktion
1	Klicken Sie im Menü `Steuerung` auf `Standardmodus`.
2	Klicken Sie im Menü `Generierung` auf `Gesamtes Projekt generieren`. Ihr Projekt wird generiert und ist für die Übertragung an die SPS bereit. Beim Generieren des Projekts wird ein Ergebnisfenster angezeigt. Falls in dem Programm ein Fehler ist, zeigt Control Expert die Fehlerstelle an (klicken Sie dazu auf die markierte Sequenz).
3	Klicken Sie im Menü `Steuerung` auf `Verbinden`. Sie sind jetzt mit der SPS verbunden.
4	Klicken Sie im Menü `SPS` auf `Projekt an SPS übertragen`. Das Fenster `Projekt an SPS übertragen` wird geöffnet. Klicken Sie auf `Übertragen`. Die Anwendung wird zur SPS übertragen.

Ausführung der Anwendung an der ersten Slave-SPS

Die folgende Tabelle beschreibt das Verfahren zur Ausführung der Anwendung im Standardmodus:

Schritt	Aktion
1	Klicken Sie im Menü `Steuerung`auf `Run`. Das Fenster `Run` wird geöffnet. Klicken Sie auf `OK`. Die Anwendung wird jetzt auf der SPS ausgeführt.
2	Trennen Sie den PC, auf dem die Software Control Expert ausgeführt wird, von der ersten Slave-SPS.
3	Verbinden Sie die erste Slave-SPS mit einem Modem des Typs SR2 MOD01.

Glossar

!

%I

In Übereinstimmung mit der IEC-Norm bezeichnet %I ein Sprachobjekt vom Typ "diskreter Eingang".

%IW

In Übereinstimmung mit der IEC-Norm bezeichnet %IW ein Sprachobjekt vom Typ "Analogeingang".

%KW

In Übereinstimmung mit der IEC-Norm bezeichnet %KW ein Sprachobjekt vom Typ "Konstantenwort".

%M

In Übereinstimmung mit der IEC-Norm bezeichnet %M ein Sprachobjekt vom Typ "Speicherbit".

%MW

In Übereinstimmung mit der IEC-Norm bezeichnet %MW ein Sprachobjekt vom Typ "Speicherwort".

%Q

In Übereinstimmung mit der IEC-Norm bezeichnet %Q ein Sprachobjekt vom Typ "diskreter Ausgang".

%QW

In Übereinstimmung mit der IEC-Norm bezeichnet %QW ein Sprachobjekt vom Typ "Analogausgang".

A

Adresse

In einen Netzwerk ist dies die Kennung einer Station. In einem Frame ist es eine Gruppe von Bits, die Quelle oder Ziel dieses Frames angeben.

Altivar

AC-Frequenzumrichter.

ARRAY

Ein ARRAY ist eine Tabelle, die Elemente eines einzelnen Typs enthält. Die Syntax lautet wie folgt: ARRAY [<Grenzen>] OF <Typ>Beispiel:ARRAY [1..2] OF BOOL ist eine eindimensionale Tabelle mit zwei Elementen des Typs BOOL. ARRAY [1..10, 1..20] OF INT ist eine zweidimensionale Tabelle mit 10x20 Elementen des Typs INT.

ASCII

ASCII ist die Abkürzung für American Standard Code for Information Interchange (amerikanischer Standardcode für den Informationsaustausch). Dies ist ein amerikanischer Code (der inzwischen zu einem internationalen Standard geworden ist), der 7 Bit zur Definition aller im Englischen verwendeten alphanumerischen Zeichen und Satzzeichen, bestimmter Graphikzeichen sowie verschiedener Befehle verwendet.

B

BOOL

BOOL bezeichnet den booleschen Datentyp. Dabei handelt es sich um ein grundlegendes Datenelement der Informatik. Eine Variable vom Typ BOOL besitzt einen der folgenden zwei Werte: 0 (FALSE) oder 1 (TRUE). Ein aus einem Wort extrahiertes Bit ist vom Typ BOOL. Beispiel: %MW10.4.

Broadcast

Bei Broadcast-Kommunikationen werden Pakete von einer Station an jedes Netzwerkziel gesendet. Broadcastnachrichten beziehen sich auf jedes Netzwerkgerät oder nur auf ein Gerät, dessen Adresse unbekannt ist.

BYTE

Eine Gruppe von 8 Bit bilden ein BYTE. Ein BYTE wird entweder im binären oder im oktalen Modus ausgedrückt. Der Datentyp BYTE wird in einem 8-Bit-Format kodiert. Im hexadezimalen Format erstreckt er sich von 16#00 bis 16#FF.

C

Control Expert

SPS-Programmiersoftware von Schneider Automation.

CPU

CPU ist die Abkürzung für Central Processing Unit (zentrale Verarbeitungseinheit): Allgemeiner Name für Prozessoren von Schneider Electric.

CRC

CRC ist die Abkürzung für den englischen Begriff Cyclic Redundancy Checksum (zyklische Redundanzprüfsumme): Sie gibt an, ob Zeichen während der Übertragung verändert, also "beschädigt" wurden.

D

DFB

DFB steht für Derived Function Block (abgeleiteter Funktionsbaustein). DFB-Typen sind Funktionsbausteine, die vom Benutzer in den Sprachen ST (Strukturierter Text), IL (Anweisungsliste), LD (Kontaktplan) oder FBD (Funktionsbausteinsprache) definiert werden können. Der Einsatz dieser DFB-Typen in Anwendungen ermöglicht Folgendes:

- Vereinfachen des Entwurfs und der Eingabe des Programms;
- Verbessern der Lesbarkeit des Programms;
- Leichtere Ausführung der Debugging-Funktion;
- Reduzierung der Menge des generierten Codes.

DINT

DINT steht für das 32-Bit-kodierte Format Double INTeger. Der gültige Wertebereich ist folgender: -(2 hoch 31) bis (2 hoch 31) - 1. Beispiel: -2147483648, 2147483647, 16#FFFFFFFF.

Diskretes Modul

Modul mit diskreten (digitalen) Eingängen/Ausgängen.

E

EBOOL

EBOOL steht für Extended BOOLean (erweiterter boolescher Datentyp). Eine Variable vom Typ EBOOL besitzt entweder den Wert 0 (FALSE) oder den Wert 1 (TRUE), sowie steigende oder fallende Flanken und Forcierungsfunktionen. Eine Variable vom Typ EBOOL belegt ein Byte im Speicher. Das Byte enthält die folgenden Informationen:

- ein Bit für den Wert;
- ein Protokollbit (jedes Mal, wenn sich der Status des Objekts ändert, wird der Wert in das Protokollbit kopiert);
- ein Forcierungsbit (0 = keine Forcierung, 1 = Forcierung).

Der Standardwert der einzelnen Bits ist 0 (FALSE).

EF

EF bedeutet Elementary Function (elementare Funktion). Es handelt sich um einen Baustein, der in einem Programm verwendet wird und dort eine vordefinierte Funktion ausführt. Eine Funktion besitzt keine Informationen über den internen Status. Jedes Mal, wenn dieselbe Funktion mit denselben Eingangsparametern aufgerufen wird, liefert sie auch dieselben Ausgangswerte. Informationen zum grafischen Aufbau des Funktionsaufrufs finden Sie im "[Funktionsbaustein (Instanz)]". Im Gegensatz zum Aufruf der Funktionsbausteine enthalten die Funktionsaufrufe lediglich einen unbenannten Ausgang, dessen Name mit dem Namen der Funktion identisch ist. In FBD wird jeder Aufruf mittels des Grafikbausteins durch eine eindeutige [Nummer] bezeichnet. Diese Nummer wird automatisch verwaltet und kann nicht geändert werden. Der Entwickler positioniert und konfiguriert diese Funktionen in seinem Programm so, wie sie in der Anwendung ausgeführt werden sollen. Mithilfe des Software Development Kits SDKC können auch andere Funktionen entwickelt werden.

F

FBD

FBD bedeutet Function Block Diagram (Funktionsbausteindiagramm). FBD ist eine grafische Programmiersprache, die wie ein Ablaufdiagramm funktioniert. Durch Hinzufügen von einfachen Logikbausteinen wie AND und OR werden die einzelnen Funktionen bzw. Funktionsbausteine des Programms in diesem grafischen Format dargestellt. Bei jedem Baustein befinden sich die Eingänge links und die Ausgänge rechts. Die Ausgänge der Bausteine können mit den Eingängen weiterer Bausteine verbunden werden und auf diese Weise komplexe Ausdrücke bilden.

Fipio

Ein Feldbus zur Verbindung von Feldgeräten wie Sensoren oder Aktoren.

FLASH-Speicher

Der FLASH-Speicher ist ein nicht flüchtiger, überschreibbarer Speicher. Er wird in einem speziellen EEPROM gespeichert, der gelöscht und neu programmiert werden kann.

Frame

Ein Frame ist eine Gruppe von Bits, die einen digitalen Informationsblock bilden. Frames enthalten Netzwerk-Steuerungsinformationen oder Daten. Die Größe und Zusammensetzung eines Frames wird durch die verwendete Netzwerktechnologie bestimmt.

H

Halbduplex

Eine Methode, bei der Daten in beiden Richtungen übertragen werden können, jedoch zu einem Zeitpunkt immer nur in eine Richtung.

Hub

Ein Gerät, das eine Reihe flexibler und zentralisierter Module verbindet, um ein Netzwerk einzurichten.

I

INT

INT steht für das Format Single INTeger (Ganzzahl vom Typ Single, 16-Bit-kodiert). Der gültige Wertebereich ist folgender: -(2 hoch 15) bis (2 hoch 15) - 1. Beispiel: 32768, 32767, 2#1111110001001001, 16#9FA4.

IODDT

IODDT bedeutet Input/Output Derived Data Type (abgeleiteter E/A-Datentyp). Mit dem Begriff IODDT wird ein strukturierter Datentyp bezeichnet, der ein Modul oder einen Kanal eines SPS-Moduls darstellt. Jedes Expertenmodul besitzt seine eigenen IODDTs.

K

Konfiguration

In der Konfiguration werden Daten zusammengefasst, die die Maschine (invariant) charakterisiert und die für den Betrieb des Moduls erforderlich sind. Alle diese Informationen werden im Speicherbereich %KW für die Steuerungskonstanten abgelegt. Sie können von der SPS-Anwendung nicht geändert werden.

L

LED

LED ist die Abkürzung für den Begriff "Licht emittierende Diode". Ein zur Anzeige verwendetes Bauelement, das im stromdurchflossenen Zustand leuchtet. Sie zeigt den Betriebsstatus des Kommunikationsmoduls an.

LRC

LRC ist die Abkürzung für den englischen Begriff Longitudinal Redundancy Check (Längssummenprüfung): Sie wurde entwickelt, weil die Fehlererkennung durch die Paritätsprüfung nicht hoch genug ist.

M

Master-Task

Hauptprogramm-Task. Sie ist obligatorisch und wird zur Ausführung der sequenziellen Verarbeitung der SPS verwendet.

Momentum

E/A-Module, die mehrere offene Standardkommunikationsnetzwerke verwenden.

N

Netzwerk

Das Wort "Netzwerk" hat zweierlei Bedeutung.

- Im Kontaktplan (LD): Ein Netzwerk ist eine Gruppe von untereinander verknüpften Grafikelementen. Die Reichweite eines Netzes beziehungsweise Netzwerks gilt in Bezug auf die organisatorische Einheit (Section) des Programms, in dem sich das Netz befindet, als lokal.
- Für Expertenkommunikationsmodule: Hier bedeuten die Begriffe Netz bzw. Netzwerk eine Gruppe von Stationen, die miteinander kommunizieren. Außerdem werden die Begriffe "Netz" und "Netzwerk" auch hier verwendet, um eine Gruppe von miteinander verbundenen grafischen Elementen zu bezeichnen. Eine solche Gruppe bildet dann einen Teil eines Programms, das aus einer Netzgruppe bestehen kann.

P

Protokoll

Beschreibt Meldungsformate und eine Reihe von Regeln, die von zwei oder mehr Geräten verwendet werden, um mittels dieser Formate zu kommunizieren.

R

RS232

Serieller Kommunikationsstandard, der die Spannung für den folgenden Dienst definiert:

- ein Signal von +12 V steht für logisch 0,
- ein Signal von -12 V steht für logisch 1.

Es gibt jedoch für den Fall eines gedämpften Signals eine mögliche Erkennung bis zu den Grenzwerten –3 V und +3 V. Zwischen diesen beiden Grenzwerten wird das Signal als ungültig betrachtet. RS-232-Verbindungen reagieren recht empfindlich auf Störungen. Die Norm gibt an, dass ein Abstand von 15 m oder maximal 9600 Baud (Bit/s) nicht überschritten werden darf.

RS485

Serieller Kommunikationsstandard, der bei 10 V/+5 V arbeitet. Er verwendet zwei Drähte für Senden/Empfangen. Deren Ausgänge mit drei möglichen Zuständen ermöglichen ihnen, am Ende der Übertragung in den Lausch-Modus umzuschalten.

RTU

RTU ist die Abkürzung für den englischen Begriff "Remote Terminal Unit" (entfernte Datenerfassungsstation). Im RTU-Modus werden Daten als zwei Hexadezimalzeichen mit vier Bit gesendet, wodurch bei gleicher Baudrate ein höherer Durchsatz als im ASCII-Modus erreicht wird. Modbus RTU ist ein binäres Protokoll und empfindlicher gegenüber Zeitverzögerungen als das ASCII-Protokoll.

S

Section

Programmmodul, das zu einer Task gehört und in einer vom Programmierer gewählten Sprache geschrieben sein kann (FBD, LD, ST, IL oder SFC). Eine Task kann aus mehreren Sections bestehen; die Reihenfolge der Ausführung der Sections entspricht der Reihenfolge, in der sie erzeugt werden. Die Reihenfolge kann bearbeitet werden.

SEPAM

Digitales Schutzrelais für Schutz, Steuerung und Überwachung in elektrischen Energieverteilungsnetzen.

Socket

Eine Kombination aus Port und IP-Adresse, die zur Identifikation von Sender oder Empfänger dient.

SPS

SPS ist die Abkürzung für speicherprogrammierbare Steuerung. Die SPS ist das Gehirn eines industriellen Fertigungsverfahrens. Im Gegensatz zu Relaisregelungssystemen automatisiert sie einen Prozess. Eine SPS ist ein Computer, der für die rauen Bedingungen industrieller Umgebungen geeignet ist.

ST

ST bedeutet Structured Text (Strukturierter Text). Unter "Strukturierter Text" versteht man eine Programmiersprache, die an die Hochsprachen der Programmierung angelehnt ist. Sie ermöglicht die Strukturierung einer Folge von Anweisungen.

STRING

Eine Variable vom Typ STRING ist eine aus ASCII-Zeichen zusammengesetzte Zeichenfolge. Eine solche Zeichenfolge kann maximal 65.534 Zeichen enthalten.

T

TAP

TAP die Abkürzung für den englischen Begriff "Transmission Access Point" (Übertragungszugangspunkt): die Busanschlusseinheit.

Task

Eine Gruppe von Sections und Unterprogrammen, die zyklisch oder periodisch für die Task MAST oder periodisch für die Task FAST ausgeführt werden. Eine Task besitzt eine bestimmte Prioritätsstufe und ist mit Ein- und Ausgängen der SPS verknüpft. Diese E/A werden nacheinander aktualisiert.

V

Variable

Speichereinheit vom Typ BOOL, WORD, DWORD usw., deren Inhalt durch das aktuell ausgeführte Programm geändert werden kann.

Vollduplex

Eine Methode zur Datenübertragung, bei der über den gleichen Kanal gleichzeitig gesendet und empfangen werden kann.

W

WORT

Der 16-Bit-kodierte Datentyp WORD wird verwendet, um Bitketten zu verarbeiten.

Die folgende Tabelle zeigt die Wertebereiche der einzelnen verwendbaren Basissysteme an:

Basis	Unterer Grenzwert	Oberer Grenzwert
Hexadezimal	16#0	16#FFFF
Oktal	8#0	8#177777
Binär	2#0	2#1111111111111111

Beispiele für die Darstellung:

Daten	Darstellung im entsprechenden Zahlensystem
0000000011010011	16#D3
1010101010101010	8#125252
0000000011010011	2#11010011

X

XBT

Grafikbasiertes Bediengerät.

XPS

Sicherheitsmodul zur Verarbeitung von Sicherheitssignalen zur Überwachung von Komponenten und Verdrahtung eines Sicherheitssystem, bestehend aus Geräten zur allgemeinen Überwachung sowie anwendungsspezifischen Modellen.

Index