

Industrie-Automatisierung **System *HIMatrix***

TCP S/R **Handbuch**



HIMA Paul Hildebrandt GmbH
Industrie-Automatisierung

HI 800 116 CDA

Wichtige Hinweise

Alle in diesem Handbuch genannten HIMA-Produkte sind mit dem HIMA-Warenzeichen geschützt. Dies gilt ebenfalls, soweit nicht anders vermerkt, auch für andere genannte Hersteller und deren Produkte.

Alle technischen Angaben und Hinweise in diesem Handbuch wurden mit größter Sorgfalt erarbeitet und unter Einschaltung wirksamer Kontrollmaßnahmen zusammengestellt. Trotzdem sind Fehler nicht ganz auszuschließen.

HIMA sieht sich deshalb veranlasst, darauf hinzuweisen, dass weder eine Garantie noch die juristische Verantwortung oder irgend eine Haftung übernommen werden kann für die Folgen, die auf fehlerhafte Angaben zurückgehen. Für die Mitteilung eventueller Fehler ist HIMA dankbar.

Technische Änderungen vorbehalten.

Weitere Informationen sind in der Dokumentation auf der CD-ROM „ELOP II Factory“ und auf unserer Website unter www.hima.de zu finden.

Informationsanfragen sind zu richten an:

HIMA Paul Hildebrandt GmbH
Postfach 1261
68777 Brühl

Tel: +49(6202)709 0
Fax: +49(6202)709 107

e-mail: info@hima.com

Zu diesem Handbuch

Ziel dieses Handbuchs ist es, den Anwender mit dem Protokoll „Send/Receive over TCP“ (TCP S/R Protokoll) vertraut zu machen und ihn bei der Einrichtung, Konfiguration und dem Betrieb des TCP S/R Protokolls zu unterstützen.

Um das TCP S/R Protokoll einzurichten, benötigt der Anwender das Programmiertool **ELOP II Factory**, das auf einem PC mit dem Betriebssystem Microsoft Windows NT[®] oder Windows 2000[®] installiert sein muss.

Der Anwender sollte den Umgang mit dem Programmiertool **ELOP II Factory** und den HIMA *HIMatrix* Steuerungen beherrschen. Für ein Selbststudium wird das Handbuch „Erste Schritte **ELOP II Factory**“ und die Online-Hilfe von **ELOP II Factory** empfohlen. Darüber hinaus bietet HIMA Kundens Schulungen an.

Dieses Handbuch ist in vier Teile gegliedert:

- Der erste Teil „Einführung“ gibt einen Überblick über die Eigenschaften und der Verwendung des TCP S/R Protokolls.
- Der zweite Teil „Programmbeschreibung“ erklärt die Menüfunktionen und Dialogfenster in **ELOP II Factory** zur Konfiguration des TCP S/R Protokolls.
- Der dritte Teil „TCP S/R Funktionsbausteine“ beschreibt die Funktion und Konfiguration der TCP S/R Funktionsbausteine.
- Im vierten Teil „Anwendungen“ wird ein Beispiel des TCP S/R Protokolls beschrieben, das der Anwender in einer Schrittanleitung nachvollziehen kann.

Wir wünschen Ihnen viel Erfolg bei der Umsetzung Ihrer TCP S/R Projekte. Sollten Sie weitere Fragen haben, wenden Sie sich bitte direkt an HIMA.

Alle Rechte und technische Änderungen vorbehalten.

© HIMA Paul Hildebrandt GmbH
Postfach 1261
D - 68777 Brühl bei Mannheim

	Inhaltsverzeichnis	Seite
1	Einführung	5
1.1	Benötigte Ausstattung und Systemanforderungen	5
1.2	Eigenschaften des TCP S/R Protokolls	5
1.3	Kommunikation über TCP S/R	7
1.3.1	TCP-Verbindungen	7
1.3.2	Zyklischer Datenaustausch	8
1.3.3	Azyklischer Datenaustausch mit Funktionsbausteine	8
1.3.4	Gleichzeitiger zyklischer und azyklischer Datenaustausch	9
1.3.5	Flusskontrolle	9
1.4	Fremdsysteme mit so genannten „Pad Bytes“	10
2	Programmbeschreibung	11
2.1	Kontextmenü „Send/Receive over TCP“	12
2.1.1	Menüfunktion „Signale verbinden“	12
2.1.2	Menüfunktion „Validieren“	12
2.1.3	Menüfunktion „Neu“	12
2.1.4	Menüfunktion „Kopieren, Einfügen, Löschen, Drucken“	13
2.2	Kontextmenü „TCP-Verbindung“	13
2.2.1	Menüfunktion „Signale verbinden“	13
2.2.2	Menüfunktion „Validieren“	14
2.2.3	Menüfunktion „Kopieren, Einfügen, Löschen, Drucken“	14
2.2.4	Menüfunktion „Eigenschaften“	15
2.3	Kontextmenü „Funktionsbausteine“	18
2.3.1	Menüfunktion „Neu“	18
2.3.2	Menüfunktion „Drucken“	18
2.3.3	Kontextmenü der COM-Funktionsbausteine	19
2.3.4	Menüfunktion „Signale verbinden“	19
2.3.5	Menüfunktion „Kopieren, Einfügen, Löschen, Drucken“	19
2.3.6	Menüfunktion „Eigenschaften“	19
2.4	Status und Fehlercodes	20
3	TCP S/R Funktionsbausteine	23
3.1	Funktionsweise der Funktionsbausteine	24
3.2	Funktionsbaustein „TCP_Receive“	26
3.2.1	CPU-Funktionsbaustein „TCP_Receive“	26
3.2.2	Funktionsablauf	28
3.2.3	COM-Funktionsbaustein „Empfangen“	29
3.3	Funktionsbaustein „TCP_Receive_Line“	31
3.3.1	CPU-Funktionsbaustein „TCP_Receive_Line“	31
3.3.2	Funktionsablauf	33
3.3.3	COM-Funktionsbaustein „Zeilenweises Empfangen“	34

3.4	Funktionsbaustein „TCP_Receive_Var“	36
3.4.1	CPU-Funktionsbaustein „TCP_Receive_Var“	37
3.4.2	Funktionsablauf	39
3.4.3	COM-Funktionsbaustein „TCP_Receive_Var“	40
3.5	Funktionsbaustein „TCP_Reset“	42
3.5.1	CPU-Funktionsbaustein „TCP_Reset“	42
3.5.2	Funktionsablauf	43
3.5.3	COM-Funktionsbaustein „TCP_Reset“	44
3.6	Funktionsbaustein „TCP_Send“	46
3.6.1	CPU-Funktionsbaustein „TCP_Send“	46
3.6.2	Funktionsablauf	48
3.6.3	COM-Funktionsbaustein „Senden“	49
3.7	Hilfsfunktionsbausteine	51
4	Anwendung	53
4.1	Zyklischer Datenverkehr zwischen Siemens und HIMA	53
4.1.1	Konfiguration des Datenaustauschs	54
4.1.2	Konfiguration der Siemens SIMATIC 300	56
4.1.3	Konfiguration des TCP S/R Protokolls in einer <i>HIMatrix</i> F60	60

1 Einführung

TCP S/R ist ein herstellerunabhängiges, nicht sicherheitsgerichtetes Protokoll für zyklischen und azyklischen Datenaustausch und verwendet außer TCP/IP kein spezielles Protokoll.

Mit dem TCP S/R Protokoll unterstützen die HIMA *HIMatrix* Steuerungen nahezu jedes Fremdsystem und auch PC's mit vorhandener Socket-Schnittstelle (z.B. Winsock.dll) zu TCP/IP.

Hinweis Das nicht sicherheitsgerichtete TCP S/R Protokoll dient in erster Linie als zusätzliche Schnittstelle zur Kommunikation mit Fremdsystemen.
Für die Kommunikation zwischen HIMA *HIMatrix* Steuerungen über Ethernet sollte das sicherheitsgerichtete HIMA Peer-to-Peer-Protokoll eingesetzt werden.

1.1 Benötigte Ausstattung und Systemanforderungen

HIMA ELOP II Factory	ab Version 5.xx
<i>HIMatrix</i> Steuerungen	F20, F30, F35 und F60 ab Hardware Revision: 00
Betriebssystemversionen der <i>HIMatrix</i> Steuerungen	-COM BS ab Version 8.4 -CPU BS ab Version 4.xx
Lizenznummer	Für die Freischaltung des TCP S/R Protokolls

1.2 Eigenschaften des TCP S/R Protokolls

Sicherheitsgerichtet	Nein
Schnittstelle	Ethernet 10/100BaseT
Datenaustausch	Zyklischer und azyklischer Datenaustausch über TCP/IP.
Funktionsbausteine	Die TCP S/R Funktionsbausteine müssen beim azyklischen Datenaustausch verwendet werden (Siehe Kapitel 3).
TCP-Verbindungen	Es können bis zu 32 TCP-Verbindungen in einer Steuerung konfiguriert werden, sofern nicht die maximale Größe der Sendedaten oder Empfangsdaten überschritten wird.
Sendedaten	Maximal 8192 Bytes Sendedaten können insgesamt gesendet werden. Um die maximale Anzahl Nutzdaten zu ermitteln, müssen alle Statussignale der verwendeten TCP-Verbindungen und der TCP/SR Funktionsbausteine, von der maximalen Anzahl Sendedaten (8192 Bytes) abgezogen werden. Die Aufteilung auf die einzelnen TCP-Verbindungen ist beliebig.

Empfangsdaten	Maximal 8192 Bytes Empfangsdaten können insgesamt empfangen werden. Um die maximale Anzahl Nutzdaten zu ermitteln, müssen alle Statussignale der verwendeten TCP-Verbindungen und der TCP/SR Funktionsbausteine, von der maximalen Anzahl Senddaten (8192 Bytes) abgezogen werden. Die Aufteilung auf die einzelnen TCP-Verbindungen ist beliebig.
---------------	--

Hinweis	<p>Neben dem TCP S/R Protokoll können gleichzeitig noch weitere Protokolle (z.B. Profibus-DP, Modbus...) auf der <i>HIMatrix</i>-Steuerung betrieben werden.</p> <p>Insgesamt können pro <i>HIMatrix</i>-Steuerung 16284 Byte Daten gesendet und 16284 Byte Daten empfangen werden.</p> <p>Die 16284 Byte Daten, können beliebig zwischen den Protokollen aufgeteilt werden, jedoch nicht mehr als 8192 Byte für ein Protokoll und Richtung.</p>
----------------	--

1.3 Kommunikation über TCP S/R

TCP S/R arbeitet gemäß dem Client/Server Prinzip. Der Verbindungsaufbau muss durch den Kommunikationspartner initiiert werden, der als Client konfiguriert ist. Nach dem ersten Verbindungsaufbau sind aber beide Kommunikationspartner gleichberechtigt und können zu jedem Zeitpunkt Daten senden.

TCP S/R besitzt kein eigenes Protokoll zur Datensicherung, sondern benutzt dafür direkt das TCP/IP Protokoll. Da TCP die Daten in einem „Daten-Stream“ sendet, muss sichergestellt sein, dass die Offsets und die Typen der auszutauschenden Signale auf der Empfangsseite und auf der Sendeseite identisch sind.

TCP S/R ist kompatibel zu der Siemens SEND/RECEIVE-Schnittstelle und erlaubt den zyklischen Datenaustausch mit den Siemens S7-Funktionsbausteinen AG_SEND (FC5) und AG_RECV (FC6) (siehe auch 4.1).

Zudem stellt HIMA fünf TCP S/R Funktionsbausteine bereit, mit denen die Kommunikation über das Anwenderprogramm gesteuert und individuell angepasst werden kann. Mit den TCP S/R Funktionsbausteinen können beliebige Protokolle (z.B. Modbus), die über TCP übertragen werden gesendet und empfangen werden.

1.3.1 TCP-Verbindungen

Für jede Verbindung über TCP S/R mit einem Kommunikationspartner muss mindestens eine TCP-Verbindung in der HIMA *HIMatrix* Steuerung erstellt werden.

In den „Eigenschaften“ der TCP-Verbindung muss die Identifikationsnummer der TCP-Verbindung und die Adressen/Ports der eigenen Steuerung und die des Kommunikationspartners eingetragen werden.

- Eine neue TCP-Verbindung wird über das Kontextmenü *Neu->TCP-Verbindung* von „Send/Receive over TCP“ angelegt.
- Die Konfiguration der TCP-Verbindung muss im Dialogfenster „Eigenschaften“ der TCP-Verbindung durchgeführt werden (siehe 2.2.4).
- Maximal 32 TCP-Verbindungen können in einer HIMA *HIMatrix* Steuerung erstellt werden.
- Die erstellten TCP-Verbindungen müssen unterschiedliche Identifikationsnummern und unterschiedliche Adressen/Ports besitzen.

Hinweis Die HIMA *HIMatrix* und das Fremdsystem müssen sich im gleichen Subnet befinden oder bei Verwendung eines Routers die entsprechenden Routingeinträge besitzen.
(siehe **ELOP II Factory** Online Hilfe “IP Einstellungen”)

1.3.2 Zyklischer Datenaustausch

Wird zyklischer Datenaustausch verwendet, dann muss ein Sendeintervall in der HIMA *HIMatrix* Steuerung und im Kommunikationspartner festgelegt werden.

Das Sendeintervall legt das zyklische Intervall fest, innerhalb dem der sendende Kommunikationspartner seine Signale an den empfangenden Kommunikationspartner sendet.

- Um einen kontinuierlichen Datenaustausch zu gewährleisten, sollte bei beiden Kommunikationspartnern ungefähr das gleiche Sendeintervalle festgelegt werden (siehe 1.3.5).
- Die Option „Zyklischer Datenversand“ muss in der verwendeten TCP-Verbindung für den zyklischen Datenaustausch aktiviert sein.
- In einer TCP-Verbindung in der „zyklischer Datenversand“ aktiviert ist, dürfen keine Funktionsbausteine verwendet werden.
- Die zu sendenden und zu empfangenden Signale werden im Dialogfenster „Signal Zuordnungen“ der TCP-Verbindung zugewiesen. Empfangssignale müssen vorhanden sein, Sendesignale sind optional.

Hinweis Die gleichen Signale (gleiche Offsets und Typen), die in der einen Station als Sendesignale definiert sind, müssen in der anderen Station als Empfangssignale definiert werden.

1.3.3 Azyklischer Datenaustausch mit Funktionsbausteine

Der azyklischen Datenaustausch wird in der HIMA *HIMatrix* Steuerung vom Anwenderprogramm über die TCP S/R Funktionsbausteine gesteuert.

Somit ist es möglich, mit einem Timer oder einen mechanischen Schalter an einem physikalischen Eingang der HIMA *HIMatrix* Steuerung, den Datenaustausch zu steuern.

- Die Option „Zyklischer Datenversand“ muss in der verwendeten TCP-Verbindung deaktiviert sein.
- Zu einem Zeitpunkt darf immer nur ein TCP S/R Funktionsbaustein senden. Die Funktion und die Konfiguration der TCP S/R Funktionsbausteine wird im Kapitel 3 beschrieben.
- Die zu sendenden oder zu empfangenden Signale werden im Register „Daten“ im Dialog „Signale Zuordnen der TCP S/R Funktionsbausteins (alle außer „Reset“) zugewiesen.

Hinweis Die gleichen Signale (gleiche Offsets und Typen), die in der einen Station als Sendesignale definiert sind, müssen in der anderen Station als Empfangssignale definiert werden.

1.3.4 Gleichzeitiger zyklischer und azyklischer Datenaustausch

Eine HIMA *HIMatrix* Steuerung kann gleichzeitig zyklische und azyklische Daten mit einem Kommunikationspartner austauschen. Hierzu muss eine TCP-Verbindung für zyklische Daten und eine zweite TCP-Verbindung für die azyklische Daten konfiguriert werden.

Eine einzelne TCP-Verbindung kann nicht für zyklischen und azyklischen Datenaustausch gemeinsam verwendet werden.

1.3.5 Flusskontrolle

Die Flusskontrolle ist ein Bestandteil von TCP und überwacht den kontinuierlichen Datenverkehr zwischen zwei Kommunikationspartnern.

Wenn nach fünf gesendeten Datenpaketen kein Datenpaket empfangen wird, dann wird das Senden blockiert. Wird jetzt nicht innerhalb der Sende-Timeout (45 Sekunden) ein Datenpaket empfangen, dann schließt die TCP-Verbindungsüberwachung diese TCP-Verbindung.

- Bei der Projektierung ist darauf zu achten, dass keine der beiden Stationen mehr Daten sendet, als die andere synchron verarbeiten kann.
- Um einen kontinuierlichen Datenaustausch zu gewährleisten, muss die Flusskontrolle für den zyklischen und azyklischen Datenaustausch beachtet werden.
 - Für den zyklischen Datenaustausch muss bei beiden Kommunikationspartnern ungefähr das gleiche Sendeintervall eingestellt werden.
 - Für den azyklischen Datenaustausch muss der Anwender die TCP S/R Funktionsbausteine so konfigurieren, dass zu einem Zeitpunkt immer nur ein Kommunikationspartner senden kann.

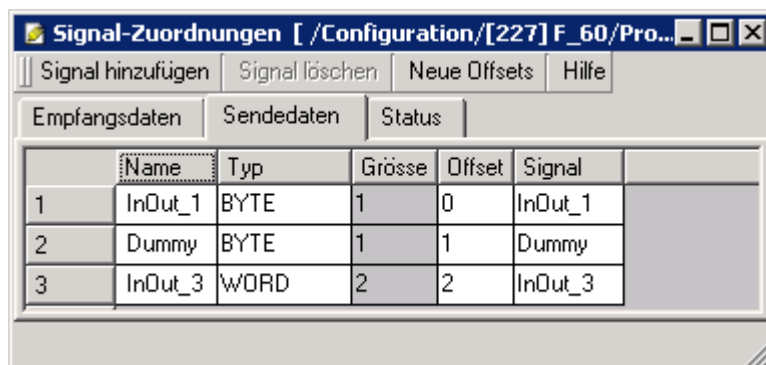
1.4 Fremdsysteme mit so genannten „Pad Bytes“

Beim zyklischem und azyklischen Datenaustausch ist zu beachten, dass manche Steuerungen (z.B. SIMATIC 300) so genannte „Pad Bytes“ einfügen. Damit wird sichergestellt, dass alle Datentypen die größer als ein Byte sind, immer an einem geraden Offset beginnen und dass die Gesamtlänge der Pakete (in Byte) ebenfalls immer gerade ist.

In der HIMA Steuerung müssen für die „Pad Bytes“, „Dummy-Bytes“ an den entsprechenden Stellen eingefügt werden.

Adresse	Name	Typ	Anfangswert
0.0		STRUCT	
+0.0	InOut_1	BYTE	B#16#0
+2.0	InOut_3	WORD	W#16#0
=4.0		END_STRUCT	

Bild 1: In der Siemens Steuerung wird ein „Pad-Byte“ (nicht sichtbar) eingefügt, damit die Variable „InOut_3“ an einem geraden Offset beginnt.



	Name	Typ	Grösse	Offset	Signal
1	InOut_1	BYTE	1	0	InOut_1
2	Dummy	BYTE	1	1	Dummy
3	InOut_3	WORD	2	2	InOut_3

Bild 2: In der HIMA Steuerung muss eine „Dummy Bytes“ eingefügt werden, damit das Signal „InOut_3“ den gleichen Offset wie in der Siemens Steuerung hat.

2 Programmbeschreibung

Die Programmbeschreibung erklärt die Menüfunktionen und Dialoge in **ELOP II Factory**, die zur Konfiguration des TCP S/R Protokolls benötigt werden.

Hinweis Das TCP S/R Protokoll kann in den *HIMatrix*-Steuerungen F20, F30, F35 und F60 ab Hardware-Version „00“ konfiguriert werden.

Starten Sie **ELOP II Factory** und erstellen Sie ein neues Projekt, oder laden Sie ein vorhandenes Projekt. Wechseln Sie danach ins Hardware-Management und wählen Sie aus dem Kontextmenü für Protokolle *Neu->Send/Receive over TCP*, um einen neuen TCP S/R Protokoll in der Ressource anzulegen.

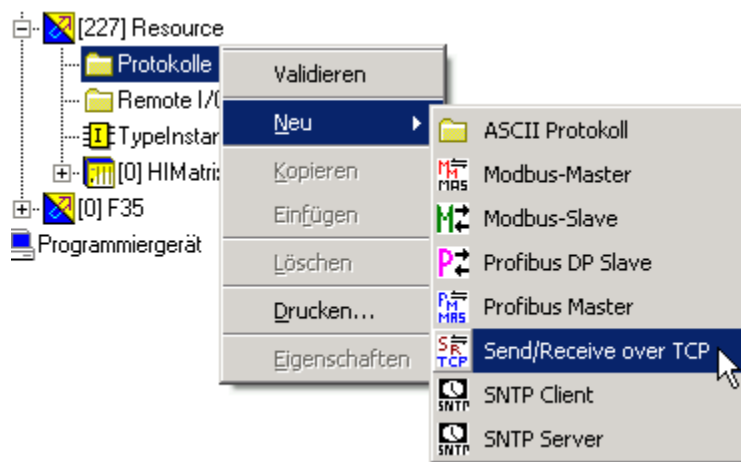


Bild 3: Neue TCP S/R-Verbindung

2.1 Kontextmenü „Send/Receive over TCP“

Das Kontextmenü des TCP S/R Protokolls enthält die folgenden Funktionen.

TCP S/R Protokoll
Signale verbinden
Validieren
Neu
Kopieren
Einfügen
Löschen
Drucken
Eigenschaften

2.1.1 Menüfunktion „Signale verbinden“

Die Menüfunktion *Signale verbinden* aus dem Kontextmenü des TCP S/R Protokolls öffnet das Dialogfenster „Signal Zuordnungen“.

Signal	Beschreibung	Typ
Status	Keine Funktion! (Um Status-Informationen des Anwenderprogramms auszuwerten siehe Kapitel 2.4)	DWORD

Tabelle 1: Register „Status“ im Dialogfenster „Signal-Zuordnungen“

2.1.2 Menüfunktion „Validieren“

Vor der Codegenerierung kann die Parametrierung des TCP S/R Protokoll getestet werden. In der Strukturansicht wird „Send/Receive over TCP“ selektiert und im Kontextmenü wird *Validieren* gewählt. In der Fehler-Status-Anzeige werden dann eventuelle Fehler und Warnungen angezeigt.

Die Validation wird zudem automatisch vor jeder Codegeneration durchgeführt. Wird bei der Validation ein Fehler festgestellt, dann wird die Codegeneration abgebrochen.

2.1.3 Menüfunktion „Neu“

Mit *Neu->TCP-Verbindung* aus dem Kontextmenü des TCP S/R Protokolls wird dem TCP S/R Protokoll eine neue TCP-Verbindung hinzugefügt.

2.1.4 Menüfunktion „Kopieren, Einfügen, Löschen, Drucken“

- Kopieren:* Kopiert ein TCP S/R Protokoll inklusive Konfiguration in die Zwischenablage
- Einfügen:* Fügt einer Ressource das TCP S/R Protokoll aus der Zwischenablage hinzu.
- Löschen:* Löscht das gewählte TCP S/R Protokoll aus der Ressource.
- Drucken:* Druckt alle Signale und Konfigurationen des TCP S/R Protokolls dieser Ressource.

2.2 Kontextmenü „TCP-Verbindung“

Das Kontextmenü der TCP-Verbindung enthält die folgenden Funktionen.

TCP-Verbindung
Signale verbinden
Validieren
Neu
Kopieren
Einfügen
Löschen
Drucken
Eigenschaften

2.2.1 Menüfunktion „Signale verbinden“

Die Menüfunktion *Signale verbinden* aus dem Kontextmenü der TCP-Verbindung öffnet den Dialog „Signal-Zuordnungen“.

Der Dialog „Signal-Zuordnungen“ enthält die drei Register

- Empfangsdaten,
- Sendedaten und
- Status

Die Signale für den zyklischen Datenaustausch, die von dieser Steuerung empfangen werden sollen, werden im Register „Empfangsdaten“ eingetragen.

Empfangsdaten	Beschreibung
Signale für den zyklischen Datenaustausch	Im Register „Empfangsdaten“ können beliebige Signale angelegt werden. Die Offsets und Typen der Signale müssen allerdings identisch mit den Offsets und den Typen der Signale (Sendedaten) des Kommunikationspartners sein.

Tabelle 2: Register „Empfangsdaten“ im Dialogfenster „Signal-Zuordnungen“

Die Signale für den zyklischen Datenaustausch, die von dieser Steuerung gesendet werden sollen, werden im Register „Sendedaten“ eingetragen.

Sendedaten	Beschreibung
Signale für den zyklischen Datenaustausch	Im Register „Sendedaten“ können beliebige Signale angelegt werden. Die Offsets und Typen der Signale müssen allerdings identisch mit den Offsets und den Typen der Signale (Empfangsdaten) des Kommunikationspartners sein.

Tabelle 3: Register „Sendedaten“ im Dialogfenster „Signal-Zuordnungen“

Mit den Signalen im Register „Status“ kann der Zustand der TCP-Verbindung im Anwenderprogramm ausgewertet werden.

Status	Beschreibung	Typ
Bytes empfangen	Anzahl Bytes, die bisher empfangen wurden	UDINT
Bytes versenden	Anzahl Bytes, die bisher verschickt wurden	UDINT
Status	Verbindungszustand und Fehlercode der TCP-Verbindung. (Siehe 2.4 Status und Fehlercodes)	DWORD

Tabelle 4: Register „Status“ im Dialogfenster „Signal-Zuordnungen“

2.2.2 Menüfunktion „Validieren“

Vor der Codegenerierung kann die Parametrierung der TCP-Verbindung getestet werden. In der Strukturansicht wird die TCP-Verbindung selektiert und im Kontextmenü wird *Validieren* gewählt. In der Fehler-Status-Anzeige werden dann eventuelle Fehler und Warnungen angezeigt.

Die Validation wird zudem automatisch vor jeder Codegeneration durchgeführt. Wird bei der Validation ein Fehler festgestellt, dann wird die Codegeneration abgebrochen.

2.2.3 Menüfunktion „Kopieren, Einfügen, Löschen, Drucken“

Kopieren: Kopiert eine TCP-Verbindung inklusive Konfiguration in die Zwischenablage.

Einfügen: Fügt einem TCP S/R Protokoll eine TCP-Verbindung aus der Zwischenablage hinzu.

Löschen: Löscht die gewählte TCP-Verbindung aus dem TCP S/R Protokoll.

Drucken: Druckt alle Signale und Konfigurationen dieser TCP-Verbindung.

2.2.4 Menüfunktion „Eigenschaften“

Mit *Eigenschaften* im Kontextmenü des TCP S/R Protokolls wird der Dialog „Eigenschaften“ geöffnet.

Hinweis Der Datenaustausch über eine TCP-Verbindung erfolgt entweder zyklisch oder azyklisch. Für den azyklischen Datenaustausch werden die TCP S/R Funktionsbausteine benötigt. Beim zyklischen Datenverkehr ist der Betrieb von TCP S/R Funktionsbausteinen nicht möglich.

Name	Erklärung	Wert
Typ	TCP-Verbindung	Nur Anzeige
Name	Beliebiger, eindeutiger Name für eine TCP-Verbindung.	Max. 32 Zeichen
Id	Beliebige, aber eindeutige Identifikationsnummer „Id“ für jede TCP-Verbindung. Die „Id“ wird auch als Referenz in den TCP S/R Funktionsbausteinen benötigt.	0..255 Default: 0
Modus	Server: Diese Station arbeitet als Server, d.h. im passiven Modus. Der Verbindungsaufbau muss durch den Kommunikationspartner (Client) initiiert werden. Nach dem ersten Verbindungsaufbau sind aber beide Stationen gleichberechtigt und können zu jedem Zeitpunkt Daten senden. Benötigt wird die Angabe des eigenen Ports.	Default: Server
	Server mit definiertem Partner: Diese Station arbeitet als Server, d.h. im passiven Modus. Der Verbindungsaufbau muss durch den Kommunikationspartner (Client) initiiert werden. Nach dem ersten Verbindungsaufbau sind aber beide Stationen gleichberechtigt und können zu jedem Zeitpunkt Daten senden. Wird hier die IP-Adresse und/oder Port des Kommunikationspartners eingetragen, dann kann nur der definierte Kommunikationspartner eine Verbindung aufnehmen. Alle anderen Stationen werden ignoriert. Wird einer der Parameter (IP-Adresse oder Port) auf Null gesetzt, findet für diesen Parameter keine Überprüfung statt.	

Name	Erklärung	Wert
	<p>Client:</p> <p>Diese Station arbeitet als Client, d.h. die Station initiiert den Verbindungsaufbau mit dem Kommunikationspartner.</p> <p>Benötigt die Angabe von IP-Adresse und Port des Kommunikationspartners.</p> <p>Optional kann auch ein eigener Port angegeben werden.</p>	
Partner IP-Adresse	<p>IP-Adresse des Kommunikationspartners.</p> <p>0.0.0.0 bedeutet beliebige IP-Adresse ist erlaubt.</p> <p>Gültiger Bereich: 1.0.0.0 bis 223.255.255.255, außer: 127.x.x.x</p>	<p>Gültige IP-Adresse</p> <p>Default: 0</p>
Partner Port	<p>Port des Kommunikationspartners.</p> <p>Null bedeutet einen beliebigen Port.</p> <p>Reservierte oder bereits belegte Ports (1 bis 1024) werden vom COM-BS abgelehnt.</p>	<p>0..65535</p> <p>Default: 0</p>
Eigener Port	<p>Eigener Port.</p> <p>Null bedeutet einen beliebigen Port.</p> <p>Reservierte oder bereits belegte Ports (1 bis 1024) werden vom COM-BS abgelehnt.</p>	<p>0..65535</p> <p>Default: 0</p>
Zyklischer Datenversand	<p>Deaktiviert:</p> <p>Zyklischer Datenversand ist deaktiviert.</p> <p>Der Datenaustausch über diese TCP-Verbindung muss mit Funktionsbausteinen programmiert werden.</p> <p>Es dürfen keine zyklischen E/A-Daten definiert sein.</p>	Default: Nein
	<p>Aktiviert:</p> <p>Zyklischer Datenversand ist aktiv.</p> <p>Die Daten werden im Dialog „Signal Zuordnungen“ der TCP-Verbindung definiert.</p> <p>Es müssen Empfangsdaten definiert sein.</p> <p>Es können keine Funktionsbausteine betrieben werden.</p>	
Sendeintervall	<p>Nur editierbar bei zyklischem Datenversand.</p> <p>Hier wird das Sendeintervall eingestellt.</p> <p>Es sind nur Schritte im Raster von 10 ms möglich.</p>	<p>0..2147483647 ms</p> <p>Default: 0</p>

Name	Erklärung	Wert
KeepAlive	<p>Ist die Zeit, bis die von TCP bereitgestellte Verbindungsüberwachung aktiv wird.</p> <p>Null deaktiviert die Verbindungsüberwachung.</p> <p>Werden innerhalb dem eingestellten KeepAlive-Interval keine Daten ausgetauscht, werden KeepAlive-Proben an den Kommunikationspartner geschickt. Besteht die Verbindung noch, werden die KeepAlive-Proben vom Kommunikationspartner bestätigt.</p> <p>Nach zehn verschickten KeepAlive-Proben, ohne eine Bestätigung vom Kommunikationspartner, wird die Verbindung geschlossen.</p> <p>Da das kleinste mögliche Zeitintervall für die KeepAlive-Proben 64 Sekunden beträgt, dauert es also mindestens 640 Sekunden bis die Verbindung geschlossen wird.</p> <p>Schneller greift das Sende-Timeout (kann nicht konfiguriert werden).</p> <p>Das Sende-Timeout schließt eine Verbindung, wenn der Kommunikationspartner nicht innerhalb von 45 Sekunden ein gesendetes Paket bestätigt.</p>	<p>0, 64.. 65535s</p> <p>Default: 0</p>

Tabelle 5: Einstellungen der TCP-Verbindung

2.3 Kontextmenü „Funktionsbausteine“

Das Kontextmenü des Verzeichnis „Funktionsbausteine“ enthält die folgenden Funktionen:

Funktionsbausteine
Neu
Kopieren
Einfügen
Löschen
Drucken
Eigenschaften

2.3.1 Menüfunktion „Neu“

Mit der Menüfunktion *Neu* wird dem TCP S/R Protokoll ein neuer Funktionsbaustein hinzugefügt.

Dem Anwender stehen die folgenden Funktionsbausteine zur Verfügung:

- Empfangen
- Reset
- Senden
- Variabel Empfangen
- Zeilenweises Empfangen

2.3.2 Menüfunktion „Drucken“

Durch Auswahl von *Drucken* öffnet sich ein Standard-Dialogfenster zum Drucken.

Betätigt der Anwender die Schaltfläche *OK*, werden alle konfigurierten Funktionsbausteine mit den aktuell belegten Signalen ausgedruckt.

2.3.3 Kontextmenü der COM-Funktionsbausteine

Die Com-Funktionsbausteine werden auf dem Kommunikations-Prozessor (COM) ausgeführt. Weitere Informationen zu den COM-Funktionsbausteinen siehe Kapitel 3.

Das Kontextmenü der COM-Funktionsbausteine enthält die folgenden Funktionen:

COM-Funktionsbausteine
Signale verbinden
Kopieren
Einfügen
Löschen
Drucken
Eigenschaften

2.3.4 Menüfunktion „Signale verbinden“

Mit *Signale verbinden* wird das Dialogfenster „Signal-Zuordnungen“ geöffnet.

Im Dialogfenster „Signal-Zuordnungen“ befinden sich die drei Register „Ausgänge“ , „Eingänge“ und „Daten“.

Die Register „Ausgänge“ und „Eingänge“ enthalten vordefinierte Systemvariablen, die der Anwender durch Zuweisen von Signalen verbinden muss.

Das Register „Daten“ wird in allen Funktionsbausteinen außer „TCP_Reset“ verwendet. Im Register „Daten“ werden die Signale eingetragen die mit dem jeweiligen Funktionsbaustein gesendet/empfangen werden sollen.

2.3.5 Menüfunktion „Kopieren, Einfügen, Löschen, Drucken“

Kopieren: Kopiert einen Funktionsbaustein inklusive Konfiguration in die Zwischenablage

Einfügen: Fügt einem TCP S/R Protokoll den Funktionsbaustein aus der Zwischenablage hinzu.

Löschen: Löscht den gewählten Funktionsbaustein aus dem TCP S/R Protokoll.

Drucken: Druckt den markierte Funktionsbaustein mit allen Signalen und Konfigurationen aus.

2.3.6 Menüfunktion „Eigenschaften“

Mit *Eigenschaften* wird das gleichnamige Dialogfenster geöffnet. Hier kann der Name des Funktionsbausteins geändert werden.

2.4 Status und Fehlercodes

Der Status und die Fehlercodes können aus dem Signal „Status“ (siehe Tabelle 4) oder dem Ausgang der Funktionsbausteine „A_Status“ (z.B. Tabelle 12) gelesen werden. Die Fehlercodes der Funktionsbausteine (z.B. „16xx8x“) werden nur an „A_Status“ der TCP S/R Funktionsbausteinen ausgegeben.

In den zwei niederwertigen Bytes wird der Fehlercode der letzten Operation angezeigt.

In den zwei höherwertigen Bytes wird der aktuellen Protokollzustand angezeigt.

Fehlercode	Erklärung
16#xx00	OK
16#xx23	Operation ist blockiert
16#xx30	Adresse ist bereits in Verwendung
16#xx32	Netzwerk läuft nicht
16#xx35	Software hat die Verbindung abgebrochen
16#xx36	Verbindung wurde vom Kommunikationspartner zurückgesetzt
16#xx37	Kein Pufferspeicher mehr verfügbar
16#xx3C	Zeit für Operation abgelaufen (Timeout)
16#xx3D	Verbindung abgewiesen
16#xx41	Keine Route zum Kommunikationspartner
16#xxFF	Verbindung durch Partner geschlossen

Tabelle 6: Fehlercode der TCP-Verbindung

Fehlercode	Erklärung
16#xx81	Unbekannte Verbindungs-Id
16#xx82	Unzulässige Länge
16#xx83	Nur zyklische Daten sind auf dieser Verbindung erlaubt.
16#xx84	Verbindung ist momentan nicht verfügbar
16#xx85	Der Timeout-Wert ist zu groß
16#xx86	Fataler Programmfehler
16#xx87	Interner Konfigurationsfehler
16#xx88	Daten passen nicht zur konfigurierten Datenstruktur.
16#xx89	Programm wurde gestoppt
16#xx8A	Timeout-Zeit abgelaufen
16#xx8B	Ein anderer Funktionsbaustein ist bereits aktiv

Tabelle 7: Fehlercodetabelle der Funktionsbausteine

Protokollzustand	Erklärung
16#00xx	Verbindung OK
16#01xx	Verbindung geschlossen
16#02xx	Server wartet auf Verbindungsaufbau
16#04xx	Client versucht eine Verbindung aufzubauen
16#08xx	Verbindung ist blockiert

Tabelle 8: Protokollzustand der TCP-Verbindung

3 TCP S/R Funktionsbausteine

Wenn die zyklische Datenübertragung zu unflexibel ist, können Daten auch mittels der TCP S/R Funktionsbausteinen gesendet und empfangen werden. Die Option „Zyklischer Datenversand“ muss in der verwendeten TCP-Verbindung deaktiviert werden.

Mit den TCP S/R Funktionsbausteinen kann der Anwender die Datenübertragung über TCP/IP optimal den Erfordernissen seines Projekts anpassen.

Die Funktionsbausteine werden im Anwenderprogramm parametrisiert. So können die Funktionen (Senden, Empfangen, Reset) der *HIMatrix*-Steuerung im Anwenderprogramm gesetzt und ausgewertet werden.

Hinweis TCP S/R Funktionsbausteine werden nur für den azyklischen Datenaustausch benötigt. Für den zyklischen Datenaustausch zwischen Server und Client sind diese Funktionsbausteine nicht erforderlich!

Es stehen die folgenden Funktionsbausteine zur Verfügung:

Funktionsbaustein	Beschreibung der Funktion
TCP_Receive	Empfangen von Datenpaketen fester Länge
TCP_ReceiveLine	Empfangen einer ASCII-Zeilen
TCP_ReceiveVar	Empfangen von Datenpaketen variabler Länge (mit Längensfeld)
TCP_Reset	Zurücksetzen einer TCP-Verbindung
TCP_Send	Senden von Daten

Tabelle 9: Beschreibung der Funktionsbausteine

3.1 Funktionsweise der Funktionsbausteine

TCP S/R Funktionsbausteine werden sowohl vom Kommunikations-Prozessor (COM) als auch von der CPU einer *HIMatrix*-Steuerung ausgeführt und müssen daher zweifach angelegt werden.

COM-Funktionsbausteine werden im Hardware-Management im Strukturbaum von **ELOP II Factory** angelegt.

CPU-Funktionsbausteine werden im Projektmanagement von **ELOP II Factory** im Anwenderprogramm angelegt. CPU-Funktionsbausteine werden verwendet, um COM-Funktionsbausteine anzusteuern.

Der Datenaustausch zwischen COM- und CPU-Funktionsbausteinen erfolgt über Signale, die vom Anwender im Signaleditor definiert und über *Signale verbinden* per Drag&Drop mit den Ein- und Ausgängen der Funktionsbausteine verbunden werden müssen.

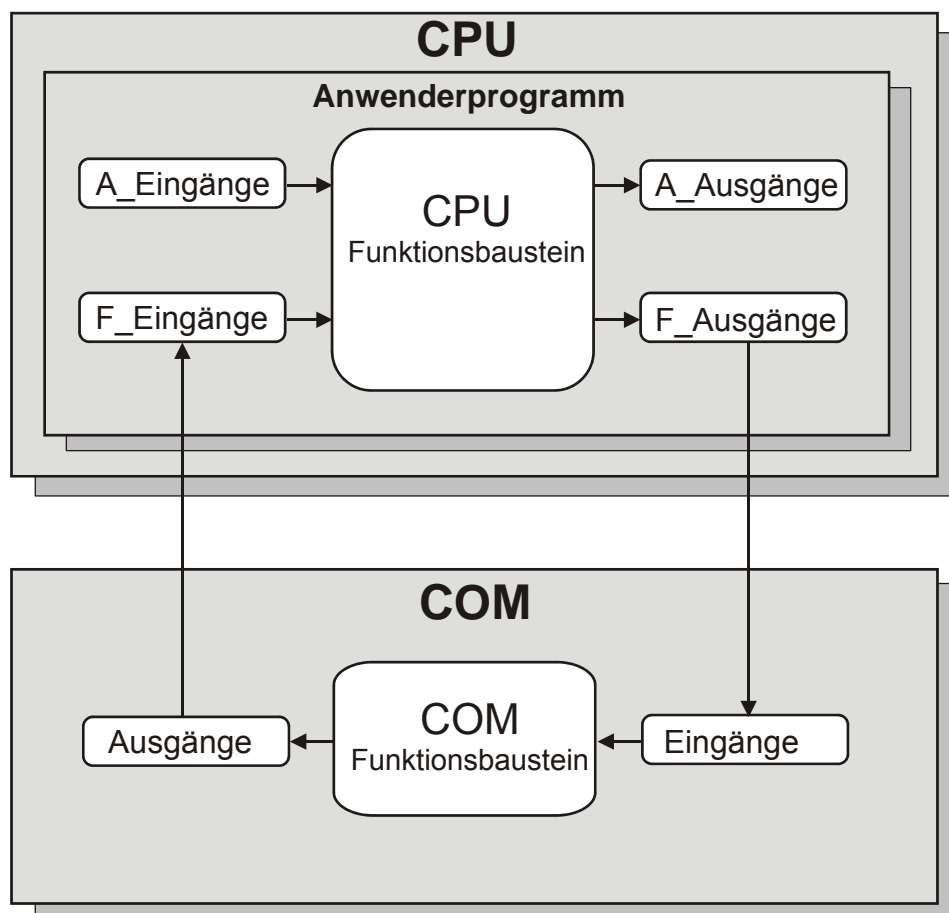


Bild 4: Kommunikation zwischen dem CPU-FB und dem COM-FB

CPU-Funktionsbausteine

Die CPU-Funktionsbausteine befinden sich im Projektmanagement im Verzeichnis „TCPLib“ und werden wie Standard-Funktionsbausteine per Drag&Drop in das Anwenderprogramm kopiert.

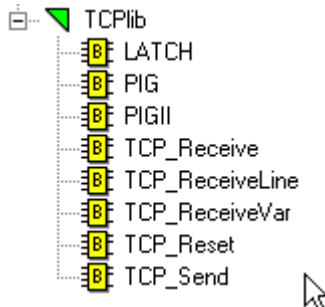


Bild 5: CPU-Funktionsbausteine

COM-Funktionsbausteine

Durch Auswahl von *Funktionsbausteine->Neu* im Strukturbaum des Hardware-Managements werden alle Funktionsbausteine der TCP S/R-Kommunikation angezeigt.



Bild 6: COM-Funktionsbausteine

Die CPU- und COM-Funktionsbausteine sind einander folgendermaßen zugeordnet:

CPU-Funktionsbausteine -> COM-Funktionsbausteine

TCP_Receive	-> Empfangen
TCP_ReceiveLine	-> Zeilenweises Empfangen
TCP_ReceiveVar	-> Variabel Empfangen
TCP_Reset	-> Reset
TCP_Send	-> Senden

3.2 Funktionsbaustein „TCP_Receive“

Mit dem Funktionsbaustein „TCP_Receive“ können definierte Signale vom Kommunikationspartner empfangen werden.

Hinweis Alle Signale für den CPU- und COM-Funktionsbaustein „TCP_Receive“ müssen im Signaleditor des Hardware-Managements erstellt werden. Die Signale werden dann per Drag&Drop in das Anwenderprogramm eingefügt.

Es wird empfohlen, die Signale passend zu den Ein- und Ausgängen des Funktionsbausteins „Empfangen“ zu benennen .

3.2.1 CPU-Funktionsbaustein „TCP_Receive“

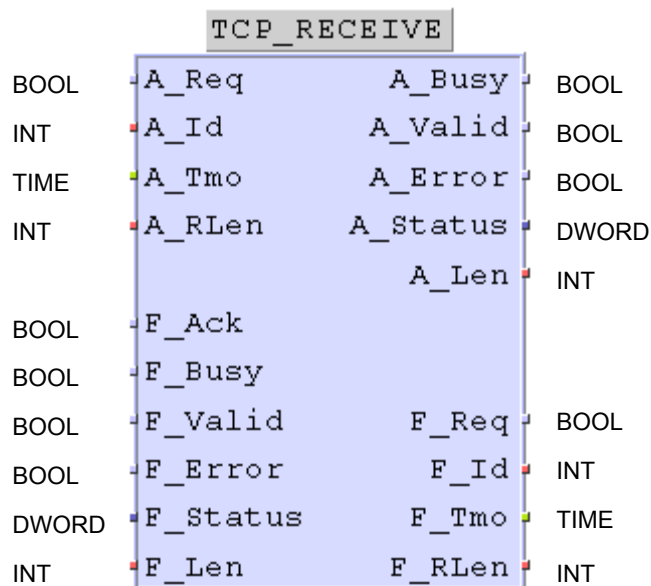


Bild 7: CPU-Funktionsbaustein „TCP_Receive“

„A_xxx“-Eing.	Beschreibung	Typ
A_Req	Positive Flanke startet den Funktionsbaustein.	BOOL
A_Id	Identifikationsnummer der konfigurierten TCP-Verbindung zu dem Kommunikationspartner, von welchem die Daten empfangen werden sollen.	INT
A_Tmo	Empfangs-Timeout. Wenn innerhalb dieser Zeit keine Daten empfangen wurden, terminiert der Baustein mit einer Fehlermeldung. Wird der Eingang „A_Tmo „nicht belegt, oder Null angelegt, ist der Timeout deaktiviert.	TIME
A_RLen	A_RLen ist die erwartete Länge der zu empfangenden Signale in Bytes. A_RLen muss größer als Null sein und darf nicht innerhalb eines Signals enden.	INT

Tabelle 10: „A_xxx“-Eingänge des CPU-Funktionsbausteins „TCP_Receive“

„F_xxx“-Eing.	Beschreibung	Typ
F_Ack	Diese Eingänge müssen mit den entsprechenden Ausgangssignalen des COM-Funktionsbausteins verbunden werden (siehe Tabelle 15).	BOOL
F_Busy		BOOL
F_Valid		BOOL
F_Error		BOOL
F_Status		DWORD
F_Len		INT

Tabelle 11: „F_xxx“- Eingänge des CPU-Funktionsbausteins „TCP_Receive“

„A_xxx“-Ausg.	Beschreibung	Typ
A_Busy	TRUE: Der Empfang der Daten ist noch nicht beendet.	BOOL
A_Valid	TRUE: Der Empfang der Daten wurde fehlerfrei beendet.	BOOL
A_Error	TRUE: Ein Fehler ist aufgetreten FALSE: Kein Fehler	BOOL
A_Status	Am Ausgang „A_Status“ werden Status und Fehlercode des Funktionsbausteins und der TCP-Verbindung ausgegeben. (Siehe 2.4 Status und Fehlercodes)	DWORD
A_Len	Anzahl der empfangenen Bytes.	INT

Tabelle 12: „A_xxx“-Ausgänge des CPU-Funktionsbausteins „TCP_Receive“

„F_xxx“-Ausg.	Beschreibung	Typ
F_Req	Diese Ausgänge müssen mit den Eingangssignalen des COM-Funktionsbausteins verbunden werden (siehe Tabelle 14).	BOOL
F_Id		DWORD
F_Tmo		INT
F_RLen		INT

Tabelle 13: „F_xxx“-Ausgänge des CPU-Funktionsbausteins „TCP_Receive“

3.2.2 Funktionsablauf

Für die Bedienung des CPU-Funktionsbaustein „TCP_Receive“ sind die folgenden Schritte erforderlich:

Hinweis Die Empfangssignale müssen im Register „Daten“ des COM-Funktionsbausteins „Empfangen“ angelegt werden. Die Offsets und Typen der Empfangssignale müssen identisch mit den Offsets und den Typen der Sendesignale des Kommunikationspartners sein.

1. Im Anwenderprogramm die Identifikationsnummer der TCP-Verbindung am Eingang „A_Id“ setzen.
2. Im Anwenderprogramm die Empfangs-Timeout am Eingang „A_Tmo“ setzen.
3. Im Anwenderprogramm die erwartete Länge der zu empfangenden Signale am Eingang „A_RLen“ setzen.
4. Im Anwenderprogramm den Eingang „A_Req“ auf TRUE setzen.

Hinweis Der Funktionsbaustein startet mit einem positiven Flankenwechsel an „A_Req“. Der Funktionsbaustein „TCP_Receive“ ist jetzt empfangsbereit.

5. Der Ausgang „A_Busy“ ist TRUE, bis die Signale empfangen wurden, oder der Empfangs-Timeout abgelaufen ist. Danach wechseln die Ausgänge „A_Busy“ auf FALSE und „A_Valid“ oder „A_Error“ auf TRUE.
6. Ist der Empfang der Signale fehlerfrei, wechselt der Ausgang „A_Valid“ auf TRUE. Die Signale, die im Register „Daten“ definiert wurden, können ausgewertet werden.
7. Ist der Empfang der Signale nicht fehlerfrei, wechselt der Ausgang „A_Error“ auf TRUE, und am Ausgang „A_Status“ wird ein Fehlercode ausgegeben.

3.2.3 COM-Funktionsbaustein „Empfangen“

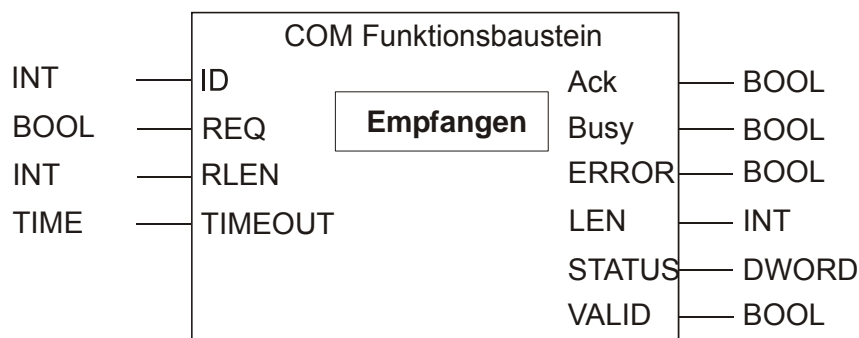


Bild 8: Signalbelegung des COM-Funktionsbausteins „Empfangen“

Der COM-Funktionsbaustein „Empfangen“ ist das Gegenstück zum CPU-Funktionsbaustein „TCP_Receive“ und kommuniziert mit diesem über Signale.

Die Signale für den COM-Funktionsbaustein „Empfangen“ werden beim Konfigurieren des CPU-Funktionsbausteins „TCP_Receive“ für die „F_Eingänge“ und „F_Ausgänge“ erstellt.

Hinweis Die Darstellung des COM-Funktionsbausteins „Empfangen“ im Bild 8 ist im Hardware-Management nicht sichtbar und dient nur der Veranschaulichung der Funktion.

Die Eingänge des COM-Funktionsbausteins müssen über Signale mit den „F_Ausgängen“ des CPU-Funktionsbausteins verbunden werden.

Eingänge	Beschreibung	Typ
ID	Identifikationsnummer der konfigurierten TCP-Verbindung zu dem Kommunikationspartner, von welchem die Daten empfangen werden sollen.	INT
REQ	Positive Flanke startet den Baustein und setzt die Verbindung empfangsbereit	BOOL
RLEN	Anzahl der zu empfangenden Signale in Bytes. A_RLen muss größer als Null sein und darf nicht mitten in einem Signal enden.	INT
TIMEOUT	Empfangs-Timeout Wenn innerhalb dieser Zeit keine Daten empfangen wurden, terminiert der Baustein mit einer Fehlermeldung. Wird der Eingang offen gelassen, oder Null angelegt, ist der Timeout ausgeschaltet.	TIME

Tabelle 14: Eingänge des COM Funktionsbausteins „Empfangen“

Die Ausgänge des COM-Funktionsbausteins müssen über Signale mit den „F_Eingängen“ des CPU-Funktionsbausteins verbunden werden.

Ausgänge	Beschreibung	Typ
Ack	TRUE: Ausgangssignale gültig FALSE: Ausgangssignale ungültig	BOOL
Busy	TRUE: Der Empfang der Daten ist noch nicht beendet.	BOOL
ERROR	TRUE: Ein Fehler ist aufgetreten FALSE: Kein Fehler	BOOL
LEN	Anzahl der empfangenen Bytes.	INT
STATUS	Am Ausgang „A_Status“ wird Status und Fehlercode des Funktionsbausteins und der TCP-Verbindung ausgegeben. (Siehe 2.4 Status und Fehlercodes)	DWORD
VALID	TRUE: Der Empfang der Daten wurde Fehlerfrei beendet.	BOOL

Tabelle 15: Ausgänge des COM-Funktionsbausteins „Empfangen“

Daten	Beschreibung
Empfangs-Signale	Im Register „Daten“ können beliebige Signale angelegt werden. Die Offsets und Typen der Signale müssen allerdings identisch mit den Offsets und den Typen der Signale des Kommunikationspartners sein.

Tabelle 16: Im Register „Daten“ werden die zu empfangenden Signale eingetragen.

3.3 Funktionsbaustein „TCP_Receive_Line“

Der Funktionsbaustein „TCP_Receive_Line“ dient zum Empfang einer ASCII Zeichenketten inklusive LineFeed (16#0A), eines Kommunikationspartners.

Hinweis: Alle Signale für den CPU- und COM-Funktionsbaustein „TCP_Receive_Line“ müssen im Signaleditor des Hardware-Managements erstellt werden. Die Signale werden dann per Drag&Drop in das Anwenderprogramm eingefügt.

Es wird empfohlen, die Signale passend zu den Ein- und Ausgängen des Funktionsbausteins „Zeilenweises Empfangen“ zu benennen.

3.3.1 CPU-Funktionsbaustein „TCP_Receive_Line“

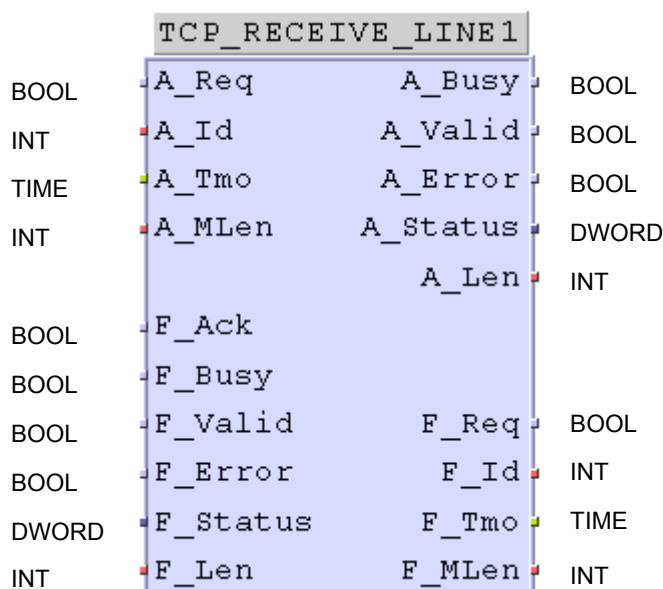


Bild 9: CPU-Funktionsbaustein „TCP_Receive_Line“

„A_xxx“-Eing.	Beschreibung	Typ
A_Req	Positive Flanke startet den Baustein und setzt die Verbindung empfangsbereit	BOOL
A_Id	Identifikationsnummer der konfigurierten TCP-Verbindung zu dem Kommunikationspartner, von welchem die Daten empfangen werden sollen.	INT
A_Tmo	Empfangs-Timeout Wenn innerhalb dieser Zeit keine Daten empfangen wurden, terminiert der Baustein mit einer Fehlermeldung. Wird der Eingang offen gelassen, oder Null angelegt, ist der Timeout ausgeschaltet.	TIME

„A_xxx“-Eing.	Beschreibung	Typ
A_MLen	„A_MLen“ ist die maximale Länge einer zu empfangenden Zeile in Bytes. Die Empfangssignale müssen im Register „Daten“ im Com Funktionsbaustein angelegt werden. Übertragene Bytes = Min (A_MLen, Zeilenlänge, Länge des Datenbereichs).	INT

Tabelle 17: „A_xxx“-Eingänge des CPU-Funktionsbausteins „TCP_Receive_Line“

„F_xxx“-Eing.	Beschreibung	Typ
F_Ack	Diese Eingänge müssen mit den entsprechenden Ausgangssignalen des COM-Funktionsbausteins verbunden werden (siehe Tabelle 22).	BOOL
F_Busy		BOOL
F_Valid		BOOL
F_Error		BOOL
F_Status		DWORD
F_Len		INT

Tabelle 18: „F_xxx“-Eingänge des CPU-Funktionsbausteins „TCP_Receive_Line“

„A_xxx“-Ausg.	Beschreibung	Typ
A_Busy	TRUE: Der Empfang der Daten ist noch nicht beendet.	BOOL
A_Valid	TRUE: Der Empfang der Daten wurde Fehlerfrei beendet.	BOOL
A_Error	TRUE: Ein Fehler ist aufgetreten FALSE: Kein Fehler	BOOL
A_Status	Am Ausgang „A_Status“ wird Status und Fehlercode des Funktionsbausteins und der TCP-Verbindung ausgegeben. (Siehe 2.4 Status und Fehlercodes)	DWORD
A_Len	Anzahl der empfangenen Bytes.	INT

Tabelle 19: „A_xxx“-Ausgänge des CPU-Funktionsbausteins „TCP_Receive_Line“

„F_xxx“-Ausg.	Beschreibung	Typ
A_Req	Diese Ausgänge müssen mit den Eingangssignalen des COM-Funktionsbausteins verbunden werden (siehe Tabelle 21).	BOOL
A_Id		INT
A_Tmo		TIME
A_MLen		INT

Tabelle 20: „F_xxx“-Ausgänge des CPU-Funktionsbausteins „TCP_Receive_Line“

3.3.2 Funktionsablauf

Für die Bedienung des CPU-Funktionsbausteins „TCP_Receive_Line“ sind die folgenden Schritte erforderlich:

Hinweis Die zu empfangenden Signale müssen im COM-Funktionsbaustein „Zeilenweises Empfangen“ angelegt werden. Die Offsets und Typen der Signale müssen identisch mit den Offsets und den Typen der Signale des Kommunikationspartners sein.

1. Im Anwenderprogramm die TCP-Verbindungs Id am Eingang „A_Id“ setzen.
2. Im Anwenderprogramm die Empfangs-Timeout am Eingang „A_Tmo“ setzen.
3. Im Anwenderprogramm die maximale Länge der zu empfangenden Zeile am Eingang „A_MLen“ setzen.

Hinweis A_MLen muss größer als Null sein und bestimmt die Größe des Empfangspuffers in Byte.

Wenn der Empfangspuffer gefüllt ist, und noch kein Zeilenende aufgetreten ist, wird der Lesevorgang ohne Fehlermeldung beendet.

Am Ausgang „A_Len“ wird die Anzahl der empfangenen Bytes zur Verfügung gestellt:

Empfangene Bytes = Min (A_MLen, Zeilenlänge, Länge des Datenbereichs)

4. Im Anwenderprogramm den Eingang „A_Req“ auf TRUE setzen.

Hinweis Der Funktionsbaustein startet mit einem positiven Flankenwechsel an „A_Req“. Der Funktionsbaustein „TCP_Receive_Line“ ist jetzt empfangsbereit.

5. Der Ausgang „A_Busy“ geht solange auf TRUE bis:

- der Empfangspuffer voll ist oder
- das Zeilenende „LineFeed“ empfangen wurde oder
- der Empfangs-Timeout abgelaufen ist.

Danach gehen die Ausgänge „A_Busy“ auf FALSE und „A_Valid“ oder „A_Error“ auf TRUE.

6. Ist der Empfang der Zeile fehlerfrei, geht der Ausgang „A_Valid“ auf TRUE. Die Signale, die im Register „Daten“ definiert wurden, können ausgewertet werden.
7. Ist der Empfang der Zeile nicht Fehlerfrei, geht der Ausgang „A_Error“ auf TRUE, und am Ausgang „A_Status“ wird ein Fehlercode ausgegeben.

3.3.3 COM-Funktionsbaustein „Zeilenweises Empfangen“

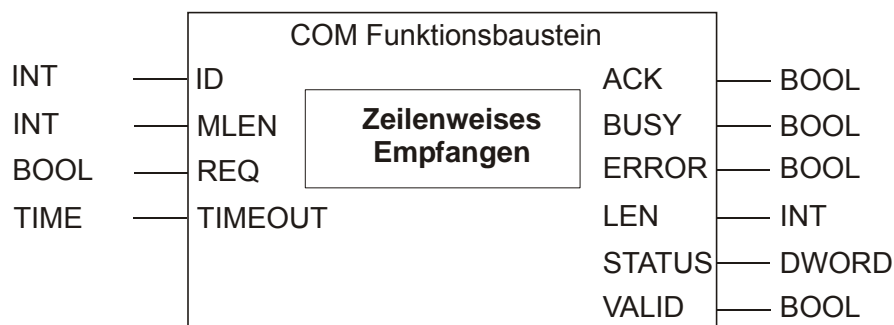


Bild 10: COM-Funktionsbaustein „Zeilenweises Empfangen“

Der COM-Funktionsbaustein „Zeilenweises Empfangen“ ist das Gegenstück zum CPU-Funktionsbaustein „TCP_Receive_Line“ und kommuniziert mit diesem über Signale.

Die Signale für den COM-Funktionsbaustein „Zeilenweises Empfangen“ werden beim Konfigurieren des CPU-Funktionsbaustein „TCP_Receive_Line“ für die „F_Eingänge“ und „F_Ausgänge“ erstellt.

Hinweis: Die Darstellung des COM-Funktionsbausteins „Zeilenweises Empfangen“ im Bild oben ist im Hardware-Management nicht sichtbar und dient nur der Veranschaulichung der Funktion.

Die Eingänge des COM-Funktionsbausteins müssen über Signale mit den „F_Ausgängen“ des CPU-Funktionsbausteins verbunden werden.

Eingänge	Beschreibung	Typ
ID	Identifikationsnummer der konfigurierten TCP-Verbindung zu dem Kommunikationspartner, von welchem die Daten empfangen werden sollen.	INT
MLEN	Maximale Länge einer zu empfangenden Zeile in Bytes. „MLEN“ muss größer als Null sein und darf nicht mitten in einem Signal enden.	INT
REQ	Positive Flanke startet den Baustein und setzt die Verbindung empfangsbereit	BOOL
TIMEOUT	Empfangs-Timeout Wenn innerhalb dieser Zeit keine Daten empfangen wurden, terminiert der Baustein mit einer Fehlermeldung. Wird der Eingang offen gelassen, oder Null angelegt, ist der Timeout ausgeschaltet.	TIME

Tabelle 21: Eingänge des COM-Funktionsbausteins „Zeilenweises Empfangen“

Die Ausgänge des COM-Funktionsbausteins müssen über Signale mit den „F_Eingängen“ des CPU-Funktionsbausteins verbunden werden.

Ausgänge	Beschreibung	Typ
ACK	TRUE: Ausgangssignale gültig FALSE: Ausgangssignale ungültig	BOOL
BUSY	TRUE: Der Empfang der Daten ist noch nicht beendet.	BOOL
ERROR	TRUE: Ein Fehler ist aufgetreten FALSE: Kein Fehler	BOOL
LEN	Anzahl der empfangenen Bytes.	INT
STATUS	Am Ausgang „A_Status“ wird Status und Fehlercode des Funktionsbausteins und der TCP-Verbindung ausgegeben. (Siehe 2.4 Status und Fehlercodes)	DWORD
VALID	TRUE: Der Empfang der Daten wurde Fehlerfrei beendet.	BOOL

Tabelle 22: Ausgänge des COM Funktionsbaustein „Zeilenweises Empfangen“

Daten	Beschreibung
Empfangs-Signale	Im Register „Daten“ können beliebige Signale angelegt werden. Die Offsets und Typen der Signale müssen allerdings identisch mit den Offsets und den Typen der Signale des Kommunikationspartners sein.

Tabelle 23: Im Register „Daten“ werden die zu empfangenden Signale eingetragen.

3.4 Funktionsbaustein „TCP_Receive_Var“

Mit diesem Funktionsbaustein können Datenpakete variabler Länge, die mit einem Längenfeld ausgestattet sind, ausgewertet werden.

Die empfangenen Datenpakete müssen den in Bild 11 dargestellten Aufbau besitzen (z.B. Modbus Protokoll). Eine Anpassung an ein beliebiges Protokoll-Format erfolgt über die Einstellung der Eingabeparameter „A_LfPos, A_LfLen, A_LfFac, A_LfLen“.

Das empfangene Datenpaket besteht aus einem Kopf- und einem Nutzdatenbereich. Der Kopfbereich enthält Daten wie Teilnehmer-Adresse, Telegrammfunktion, Längenfeld usw., die für die Kommunikationsverbindung erforderlich sind. Um den Nutzdatenbereich auszuwerten muss der Kopfbereich abgetrennt und das Längenfeld ausgelesen werden.

Die Größe des Kopfbereichs wird im Parameter „A_LfAdd“ eingetragen.

Die Länge des Nutzdatenbereichs muss aus dem Längenfeld des aktuell gelesenen Datenpakets ausgelesen werden. Die Position des Längenfeldes wird im Parameter „A_LfPos“ eingetragen. Die Größe des Längenfeldes wird in „LfLen“ in Byte eingetragen. Falls die Länge nicht als Byte angegeben ist, muss der Umrechnungsfaktor hierfür in „A_LfFac“ eingetragen werden (z.B. 2 für Word oder 4 für Double Word).

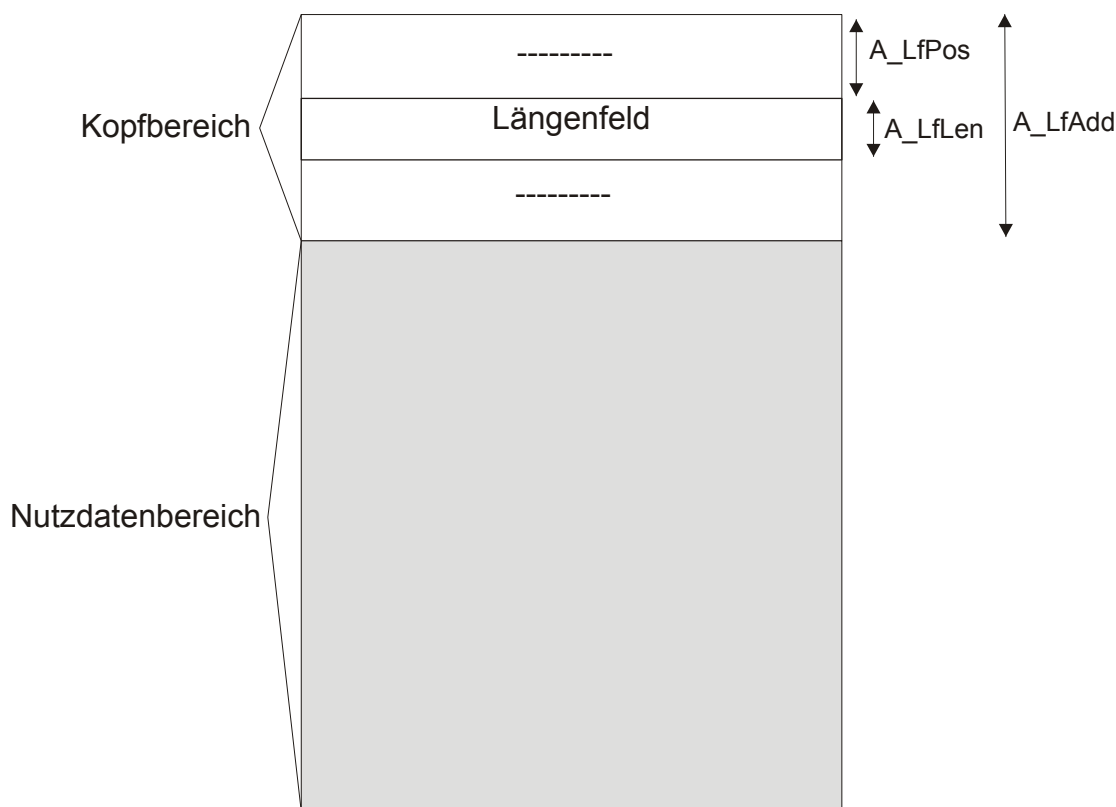


Bild 11: Aufbau des Datenpakets

Hinweis Alle Signale für den CPU- und COM-Funktionsbaustein „TCP_Receive_Var“ müssen im Signaleditor des Hardware-Managements erstellt werden. Die Signale werden dann per Drag&Drop in das Anwenderprogramm eingefügt.

Es wird empfohlen, die Signale passend zu den Ein- und Ausgängen des Funktionsbausteins „TCP_Receive_Var“ zu benennen.

3.4.1 CPU-Funktionsbaustein „TCP_Receive_Var“

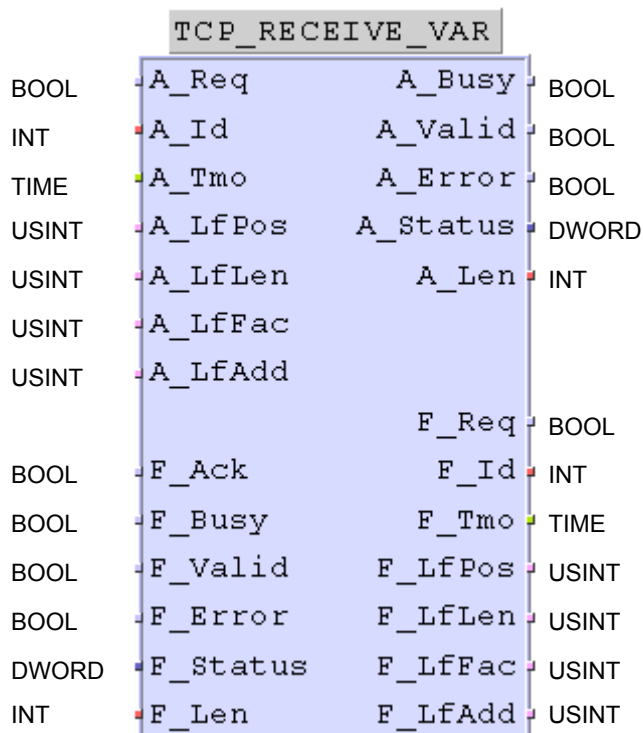


Bild 12: CPU-Funktionsbaustein „TCP_Receive_Var“

„A_xxx“-Eing.	Beschreibung	Typ
A_Req	Mit der positiven Flanke wird der CPU-Funktionsbaustein gestartet	BOOL
A_Id	Identifikationsnummer „Id“ der konfigurierten TCP-Verbindung zu einem Kommunikationspartner, von dem das Datenpaket empfangen werden soll.	DWORD
A_Tmo	Empfangs-Timeout Wenn innerhalb dieser Zeit keine Daten empfangen wurden, terminiert der Baustein mit einer Fehlermeldung. Wird der Eingang offen gelassen, oder Null angelegt, ist der Timeout ausgeschaltet.	INT
A_LfPos	Startposition des Längenfeldes im Datenpaket; die Nummerierung beginnt mit Null (gemessen in Bytes).	USINT
A_LfLen	Größe des Längenfeldes „A_LfLen“ in Bytes. Erlaubt sind 1, 2 oder 4 Bytes.	USINT
A_LfFac	Umrechnungsfaktor in Bytes, falls der Eintrag im Längenfeld nicht in Bytes ist. Wird der Eingang offen gelassen, oder mit Null belegt, wird „1“ als Defaultwert genommen.	USINT
A_LfAdd	Größe des Kopffeldes in Bytes	USINT

Tabelle 24: „A_xxx“-Eingänge des CPU-Funktionsbausteins „TCP_Receive_Var“

„F_xxx“-Eing.	Beschreibung	Typ
F_Ack	Diese Eingänge müssen mit den entsprechenden Ausgangssignalen des COM-Funktionsbausteins verbunden werden (siehe Tabelle 29).	BOOL
F_Busy		BOOL
F_Valid		BOOL
F_Error		BOOL
F_Status		DWORD
F_Len		INT

Tabelle 25: „F_xxx“-Eingänge des CPU-Funktionsbausteins „TCP_Receive_Var“

„F_xxx“-Ausg.	Beschreibung	Typ
F_Req	Diese Ausgänge müssen mit den Eingangssignalen des COM-Funktionsbausteins verbunden werden (siehe Tabelle 28).	BOOL
F_Id		INT
F_Tmo		TIME
F_LfPos		USINT
A_LfLen		USINT
A_LfFac		USINT
A_LfAdd		USINT

Tabelle 26: „F_xxx“-Ausgänge des CPU-Funktionsbausteins „TCP_Receive_Var“

„A_xxx“-Ausg.	Beschreibung	Typ
A_Busy	TRUE: Der Empfang der Daten ist noch nicht beendet.	BOOL
A_Valid	TRUE: Der Empfang der Daten wurde Fehlerfrei beendet.	BOOL
A_Error	TRUE: Beim Lesen trat ein Fehler auf FALSE: Kein Fehler	BOOL
A_Status	Am Ausgang „A_Status“ wird Status und Fehlercode des Funktionsbausteins und der TCP-Verbindung ausgegeben. (Siehe 2.4 Status und Fehlercodes)	DWORD
A_Len	Anzahl der empfangenen Bytes.	INT

Tabelle 27: „A_xxx“-Ausgänge des CPU-Funktionsbausteins „TCP_Receive_Var“

3.4.2 Funktionsablauf

Für die Bedienung des CPU-Funktionsbausteins „TCP_Receive_Var“ sind die folgenden Schritte erforderlich:

Hinweis Die zu empfangenden Signale müssen im COM-Funktionsbaustein „TCP_Receive_Var“ angelegt werden. Die Offsets und Typen der Signale müssen identisch mit den Offsets und den Typen der Signale des Kommunikationspartners sein.

1. Im Anwenderprogramm die TCP-Verbindungs Id der TCP-Verbindung am Eingang „A_Id“ setzen.
2. Im Anwenderprogramm die Empfangs-Timeout am Eingang „A_Tmo“ setzen.
3. Im Anwenderprogramm die Parameter A_LfPos, A_LfLen, A_LfFac und A_LfAdd setzen.
4. Im Anwenderprogramm den Eingang „A_Req“ auf TRUE setzen.

Hinweis Der Funktionsbaustein startet mit einem positiven Flankenwechsel an „A_Req“. Der Funktionsbaustein „TCP_Receive_Var“ ist jetzt empfangsbereit.

5. Der Ausgang „A_Busy“ geht solange auf TRUE, bis die Signale empfangen wurden. Danach gehen die Ausgänge „A_Busy“ auf FALSE und „A_Valid“ oder „A_Error“ auf TRUE.
6. Ist der Empfang der Signale fehlerfrei, geht der Ausgang „A_Valid“ auf TRUE. Die Signale, die im Register „Daten“ definiert wurden, können ausgewertet werden.. Der Ausgang „A_Len“ enthält die Anzahl der Bytes, die tatsächlich ausgelesen wurden.
7. Ist der Empfang der Signale nicht Fehlerfrei, geht der Ausgang „A_Error“ auf TRUE, und am Ausgang „A_Status“ wird ein Fehlercode ausgegeben.

3.4.3 COM-Funktionsbaustein „TCP_Receive_Var“

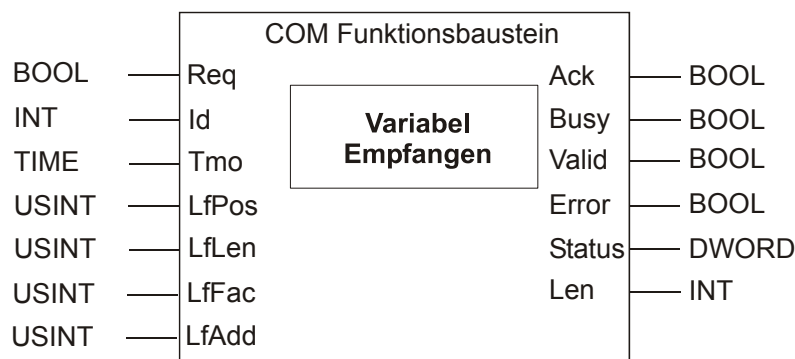


Bild 13: COM-Funktionsbaustein „TCP_Receive_Var“

Der COM-Funktionsbaustein „TCP_Receive_Var“ ist das Gegenstück zum CPU-Funktionsbaustein „TCP_Receive_Var“ und kommuniziert mit diesem über Signale.

Die Signale für den COM-Funktionsbaustein „TCP_Receive_Var“ werden beim Konfigurieren des CPU-Funktionsbausteins „TCP_Receive_Var“ für die „F_Eingänge“ und „F_Ausgänge“ erstellt.

Hinweis Die Darstellung des COM-Funktionsbausteins „TCP_Receive_Var“ im Bild oben ist im Hardware-Management nicht sichtbar und dient nur der Veranschaulichung der Funktion.

Die Eingänge des COM-Funktionsbausteins im Dialog „Signal-Zuordnungen“ müssen über Signale mit den „F_Ausgängen“ des CPU-Funktionsbausteins verbunden werden.

Eingänge	Beschreibung	Typ
ID	Identifikationsnummer „Id“ der konfigurierten TCP-Verbindung zu einem Kommunikationspartner, von dem das Datenpaket empfangen werden soll.	INT
LfAdd	Größe des Kopffeldes in Byte.	USINT
LfFac	Umrechnungsfaktor in Byte, falls der Eintrag im Längenfeld nicht in Byte ist. Wird der Eingang offen gelassen, oder mit Null belegt, wird „1“ als Defaultwert genommen.	USINT
LfLen	Größe des Längenfelds „A_LfLen“ in Bytes. Erlaubt sind 1, 2 oder 4 Bytes.	USINT
LfPos	Startposition des Längenfeldes im Datenpaket; die Nummerierung beginnt mit Null (gemessen in Bytes).	USINT
REQ	Wenn TRUE, wird der COM-Funktionsbaustein gestartet.	BOOL

Eingänge	Beschreibung	Typ
TIMEOUT	Empfangs-Timeout Wenn innerhalb dieser Zeit keine Daten empfangen wurden, terminiert der Baustein mit einer Fehlermeldung. Wird der Eingang offen gelassen, oder Null angelegt, ist der Timeout ausgeschaltet.	TIME

Tabelle 28: Eingänge des COM-Funktionsbausteins „TCP_Receive_Var“

Die Ausgänge des COM-Funktionsbausteins im Dialog „Signal-Zuordnungen“ müssen über Signale mit den „F_Eingängen“ des CPU-Funktionsbausteins verbunden werden.

Ausgänge	Beschreibung	Typ
ACK	TRUE: Ausgangssignale gültig FALSE: Ausgangssignale ungültig	BOOL
BUSY	TRUE: Der Empfang der Daten ist noch nicht beendet.	BOOL
ERROR	TRUE: Ein Fehler ist aufgetreten FALSE: Kein Fehler	BOOL
LEN	Anzahl der empfangenen Bytes.	INT
STATUS	Am Ausgang „A_Status“ wird Status und Fehlercode des Funktionsbausteins und der TCP-Verbindung ausgegeben. (Siehe 2.4 Status und Fehlercodes)	DWORD
VALID	TRUE: Der Empfang der Daten wurde fehlerfrei beendet.	BOOL

Tabelle 29: Ausgänge des COM-Funktionsbausteins „TCP_Receive_Var“

Daten	Beschreibung
Empfangs-Signale	Im Register „Daten“ können beliebige Signale angelegt werden. Die Offsets und Typen der Signale müssen allerdings identisch mit den Offsets und den Typen der Signale des Kommunikationspartners sein.

Tabelle 30: Im Register „Daten“ werden die zu empfangenden Signale eingetragen.

3.5 Funktionsbaustein „TCP_Reset“

Mit diesem Baustein kann eine gestörte Verbindung wiederhergestellt werden, wenn sich ein Send- oder Receive-Funktionsbaustein mit einem TIMEOUT-Fehler meldet (16#8A).

Hinweis Alle Signale für den CPU- und COM-Funktionsbaustein „TCP_Reset“ müssen im Signaleditor des Hardware-Managements erstellt werden. Die Signale werden dann per Drag&Drop in das Anwenderprogramm eingefügt.

Es wird empfohlen, die Signale passend zu den Ein- und Ausgängen des Funktionsbausteins „TCP_Reset“ zu benennen.

3.5.1 CPU-Funktionsbaustein „TCP_Reset“

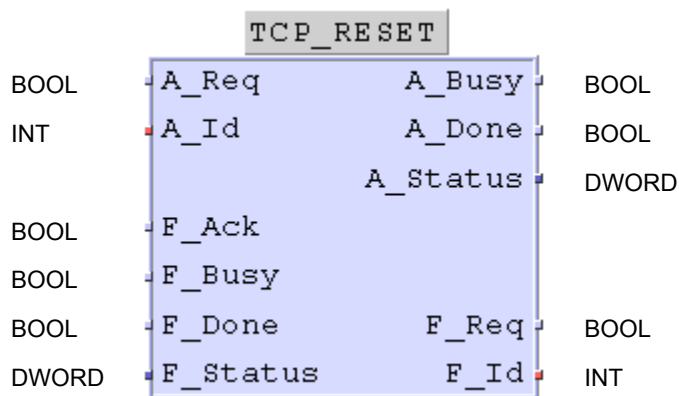


Bild 14: CPU-Funktionsbausteins „TCP_Reset“

„A_xxx“-Eing.	Beschreibung	Typ
A_Req	Positive Flanke startet den Baustein	BOOL
A_Id	Identifikationsnummer „Id“ der gestörten TCP-Verbindung, die zurückgesetzt werden soll.	INT

Tabelle 31: „A_xxx“-Eingänge des CPU-Funktionsbausteins „TCP_Reset“

„F_xxx“-Eing.	Beschreibung	Typ
F_Ack	Diese Eingänge müssen mit den entsprechenden Ausgangssignalen des COM-Funktionsbausteins verbunden werden (siehe Tabelle 36).	BOOL
F_Busy		BOOL
F_Done		BOOL
F_Status		DWORD

Tabelle 32: „F_xxx“-Eingänge des CPU-Funktionsbausteins „TCP_Reset“

„A_xxx“-Ausg.	Beschreibung	Typ
A_Busy	TRUE: Der Reset des Funktionsbausteins ist noch nicht beendet.	BOOL
A_Done	TRUE: Der Sendevorgang wurde fehlerfrei beendet	BOOL
A_Status	Am Ausgang „A_Status“ wird Status und Fehlercode des Funktionsbausteins und der TCP-Verbindung ausgegeben. (Siehe 2.4 Status und Fehlercodes)	DWORD

Tabelle 33: „A_xxx“-Ausgänge des CPU-Funktionsbausteins „TCP_Reset“

„F_xxx“-Ausg.	Beschreibung	Typ
F_Req	Diese Ausgänge müssen mit den Eingangssignalen des COM-Funktionsbausteins verbunden werden (siehe Tabelle 35).	BOOL
F_Id		DWORD

Tabelle 34: „F_xxx“-Ausgänge des CPU-Funktionsbaustein „TCP_Reset“

3.5.2 Funktionsablauf

Für die Bedienung des CPU-Funktionsbaustein „TCP_Reset“ sind die folgenden Schritte erforderlich:

1. Im Anwenderprogramm die TCP-Verbindungs Id am Eingang „A_Id“ setzen.
2. Im Anwenderprogramm den Eingang „A_Req“ auf TRUE setzen.

Hinweis Der Funktionsbaustein startet mit einem positiven Flankenwechsel an „A_Req“.

3. Der Ausgang „A_Busy“ wechselt auf TRUE, bis ein Reset an die definierte TCP-Verbindung gesendet wurde. Danach wechseln die Ausgänge „A_Busy“ auf FALSE und „A_Done“ auf TRUE.

3.5.3 COM-Funktionsbaustein „TCP_Reset“

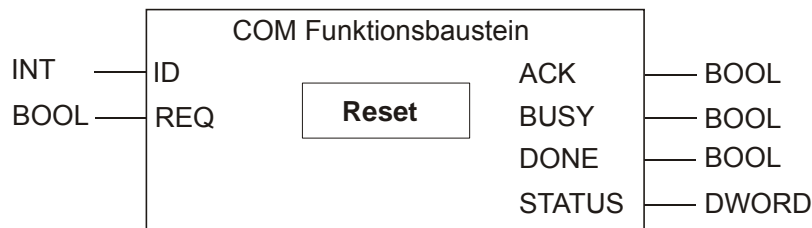


Bild 15: COM-Funktionsbaustein „TCP_Reset“

Der COM-Funktionsbaustein „TCP_Reset“ ist das Gegenstück zum CPU-Funktionsbaustein „TCP_Reset“ und kommuniziert mit diesem über Signale.

Die Signale für den COM-Funktionsbaustein „TCP_Reset“ werden vom Anwender beim Konfigurieren des CPU-Funktionsbaustein „TCP_Reset“ für die Eingänge und Ausgänge im Signaleditor erstellt.

Hinweis Die Darstellung des COM-Funktionsbausteins „TCP_Reset“ im Bild oben ist im Hardware-Management nicht sichtbar und dient nur der Veranschaulichung der Funktion.

Die Eingänge des COM-Funktionsbausteins müssen über Signale mit den Ausgängen des CPU-Funktionsbausteins im Anwenderprogramm verbunden werden.

Eingänge	Beschreibung	Typ
ID	Identifikationsnummer „Id“ der gestörten TCP-Verbindung, die zurückgesetzt werden soll.	DWORD
REQ	Mit der positiven Flanke wird der COM-Funktionsbaustein gestartet.	BOOL

Tabelle 35: Eingänge des COM-Funktionsbausteins „TCP_Reset“

Die Ausgänge des COM-Funktionsbausteins müssen über Signale mit den „F_Eingängen“ des CPU-Funktionsbausteins verbunden werden.

Ausgänge	Beschreibung	Typ
ACK	TRUE: Ausgangssignale gültig FALSE: Ausgangssignale ungültig	BOOL
BUSY	TRUE: CPU-Funktionsbaustein wartet auf die Rückmeldung vom COM-Funktionsbaustein, dass der Reset gesendet wurde	BOOL
DONE	TRUE: Das Reset-Signal wurde gesendet.	BOOL
STATUS	Am Ausgang „A_Status“ wird Status und Fehlercode des Funktionsbausteins und der TCP-Verbindung ausgegeben. (Siehe 2.4 Status und Fehlercodes)	DWORD

Tabelle 36: Ausgänge des COM-Funktionsbausteins „TCP_Reset“

3.6 Funktionsbaustein „TCP_Send“

Der Funktionsbaustein „TCP_SEND“ dient zum azyklischen senden von Signalen zu einem Kommunikationspartner. Im Kommunikationspartner muss ein Funktionsbaustein z.B. „Empfangen“ mit den gleichen Signalen und Offsets Konfiguriert werden.

Hinweis Alle Signale für den CPU- und COM-Funktionsbaustein „TCP_SEND“ müssen im Signaleditor des Hardware-Managements erstellt werden. Die Signale werden dann per Drag&Drop in das Anwenderprogramm eingefügt.

Es wird empfohlen, die Signale passend zu den Ein- und Ausgängen des Funktionsbausteins „TCP_SEND“ zu benennen .

3.6.1 CPU-Funktionsbaustein „TCP_Send“

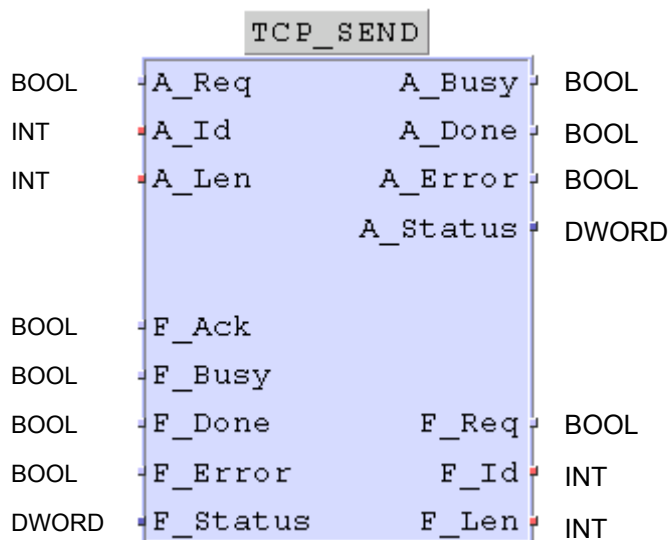


Bild 16: Der CPU Funktionsbaustein „TCP_Send“ im Anwenderprogramm

„A_xxx“-Eing.	Beschreibung	Typ
A_Req	Positive Flanke startet den Baustein.	BOOL
A_Id	Identifikationsnummer der konfigurierten TCP-Verbindung zu dem Kommunikationspartner, zu welchem die Daten gesendet werden sollen.	INT
A_Len	Anzahl der zu sendenden Signale in Bytes. A_Len muss größer als Null sein und darf nicht innerhalb eines Signals enden.	INT

Tabelle 37: „A_xxx“-Eingänge des CPU-Funktionsbausteins „TCP_Send“

„F_xxx“-Eing.	Beschreibung	Typ
F_Ack	Diese Eingänge müssen mit den entsprechenden Ausgangssignalen des COM-Funktionsbausteins verbunden werden (siehe Tabelle 42).	BOOL
F_Busy		BOOL
F_Done		BOOL
F_Error		BOOL
F_Status		DWORD

Tabelle 38: „F_xxx“-Eingänge des CPU-Funktionsbausteins „TCP_Send“

„A_xxx“-Ausg.	Beschreibung	Typ
A_Busy	TRUE: Der Sendevorgang ist noch nicht beendet.	BOOL
A_Done	TRUE: Der Sendevorgang wurde fehlerfrei beendet	BOOL
A_Error	TRUE: Ein Fehler ist aufgetreten FALSE: Kein Fehler	BOOL
A_Status	Am Ausgang „A_Status“ wird Status und Fehlercode des Funktionsbausteins und der TCP-Verbindung ausgegeben. (Siehe 2.4 Status und Fehlercodes)	DWORD

Tabelle 39: „A_xxx“-Ausgänge des CPU-Funktionsbausteins „TCP_Send“

„F_xxx“-Ausg.	Beschreibung	Typ
F_Req	Diese Ausgänge müssen mit den Eingangssignalen des COM-Funktionsbausteins verbunden werden (siehe Tabelle 41).	BOOL
F_Id		DWORD
F_Len		INT

Tabelle 40: „F_xxx“-Ausgänge des CPU-Funktionsbausteins „TCP_Send“

3.6.2 Funktionsablauf

Für die Bedienung des CPU-Funktionsbausteins „TCP_Send“ sind die folgenden Schritte erforderlich:

Hinweis Die zu sendenden Signale müssen im COM-Funktionsbaustein „Senden“ angelegt werden. Die Offsets und Typen der Signale müssen identisch mit den Offsets und den Typen der Signale des Kommunikationspartners sein.

1. Im Anwenderprogramm die TCP-Verbindungs Id am Eingang „A_Id“ setzen.
2. Im Anwenderprogramm die Länge der zu sendenden Signale am Eingang „A_Len“ in Byte setzen.
3. Im Anwenderprogramm den Eingang „A_Req“ auf TRUE setzen.

Hinweis Der Funktionsbaustein reagiert auf einen positiven Flankenwechsel an „A_Req“.

4. Der Ausgang „A_Busy“ geht solange auf TRUE, bis die Signale gesendet wurden. Danach gehen die Ausgänge „A_Busy“ auf FALSE und „A_Done“ auf TRUE.
5. Konnte der Sendevorgang nicht erfolgreich ausgeführt werden, geht der Ausgang „A_Error“ auf TRUE und am Ausgang „A_Status“ wird ein Fehlercode ausgegeben.

3.6.3 COM-Funktionsbaustein „Senden“

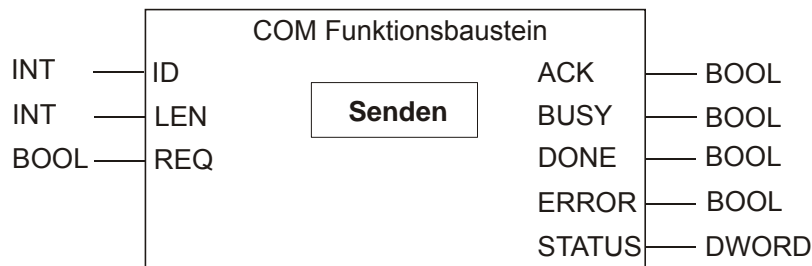


Bild 17: COM-Funktionsbaustein „Senden“

Der COM-Funktionsbaustein „Senden“ ist das Gegenstück zum CPU-Funktionsbaustein „TCP_Send“ und kommuniziert mit diesem über Signale.

Die Signale für den COM-Funktionsbaustein „Senden“ werden vom Anwender beim Konfigurieren des CPU-Funktionsbausteins „TCP_Send“ für die „F_Eingänge“ und „F_Ausgänge“ im Signaleditor erstellt.

Hinweis Die Darstellung des COM-Funktionsbausteins „Senden“ im Bild 17 ist im Hardware-Management nicht sichtbar und dient nur der Veranschaulichung der Funktion.

Die Eingänge des COM-Funktionsbausteins müssen über Signale mit den „F_Ausgängen“ des CPU-Funktionsbausteins verbunden werden.

Eingänge	Beschreibung	Typ
ID	Identifikationsnummer der konfigurierten TCP-Verbindung zu dem Kommunikationspartner, zu welchem die Daten gesendet werden sollen.	DWORD
LEN	Anzahl der zu sendenden Signale in Bytes. „LEN“ muss größer als Null sein und darf nicht innerhalb eines Signals enden.	INT
REQ	Positive Flanke startet den Baustein.	BOOL

Tabelle 41: Eingänge des COM-Funktionsbausteins „Senden“

Die Ausgänge des COM-Funktionsbausteins müssen über Signale mit den „F_Eingängen“ des CPU-Funktionsbausteins verbunden werden.

Ausgänge	Beschreibung	Typ
ACK	TRUE: Ausgangssignale gültig FALSE: Ausgangssignale ungültig	BOOL
BUSY	TRUE: Der Sendevorgang ist noch nicht beendet.	BOOL
Done	TRUE: Der Sendevorgang wurde ausgeführt.	BOOL
ERROR	TRUE: Ein Fehler ist aufgetreten FALSE: Kein Fehler	BOOL
STATUS	Am Ausgang „A_Status“ wird Status und Fehlercode des Funktionsbausteins und der TCP-Verbindung ausgegeben. (Siehe 2.4 Status und Fehlercodes)	DWORD

Tabelle 42: Ausgänge des COM-Funktionsbausteins „Senden“

Daten	Beschreibung
Sende-Signale	Im Register „Daten“ können beliebige Signale angelegt werden. Die Offsets und Typen der Signale müssen allerdings identisch mit den Offsets und den Typen der Signale des Kommunikationspartners sein.

Tabelle 43: Im Register „Daten“ werden die zu sendenden Signale eingetragen.

3.7 Hilfsfunktionsbausteine

Die folgenden Hilfsfunktionsbausteine laufen komplett im Anwenderprogramm auf der CPU ab.

Die Hilfsfunktionsbausteine befinden sich im Projekt-Verzeichnis in der Bibliothek „TCPLib“. Die Hilfsfunktionsbausteine werden nur innerhalb der anderen Funktionsbausteine gebraucht.

Die folgenden Hilfsfunktionsbausteine befinden sich in der Bibliothek „TCPLib“.

Hilfsfunktionsbausteine	Beschreibung
LATCH	Die Funktion ID generiert aus vier Bytes einen Identifier.
PIG	Erstellt aus einer Slot-Nummer eine SLOT Identifikationsnummer.
PIGII	Erstellt fortlaufende Identifikationsnummer für die Slot.

Tabelle 44: Die Hilfsfunktionsbausteine und Ihre Funktion

Hinweis	Die Signale für die Hilfsfunktionsbausteine müssen im Signaleditor des Hardware-Managements erstellt werden. Die Signale werden dann per Drag&Drop in das Anwenderprogramm eingefügt.
----------------	---

4 Anwendung

4.1 Zyklischer Datenverkehr zwischen Siemens und HIMA

In diesem Beispiel wird das Protokoll „Send/Receive over TCP“ in einer HIMA *HIMatrix* F60 Steuerung eingerichtet. Die HIMA *HIMatrix* F60 soll zyklisch über TCP S/R mit einer Siemens Steuerung (z.B. SIMATIC 300) kommunizieren.

In diesem Beispiel ist die HIMA *HIMatrix* F60 (Client) die aktive Station, welche die TCP-Verbindung zur passiven Siemens SIMATIC 300 (Server) aufbaut. Nach dem Verbindungsaufbau sind aber beide Steuerungen gleichberechtigt und können jederzeit senden und empfangen.

Bei der Zusammenschaltung der *HIMatrix* F60 und der Siemens SIMATIC 300 ist folgendes zu beachten:

1. Für die HIMA *HIMatrix* F60 gelten die im Kapitel 1.1 beschriebenen Anforderungen.
2. Die HIMA *HIMatrix* F60 und die Siemens SIMATIC 300 werden über ihre Ethernet Schnittstellen miteinander verbunden.
3. Die HIMA *HIMatrix* F60 und die Siemens SIMATIC 300 müssen sich im gleichen Subnet befinden oder bei Verwendung eines Routers die entsprechenden Routing Einträge besitzen.

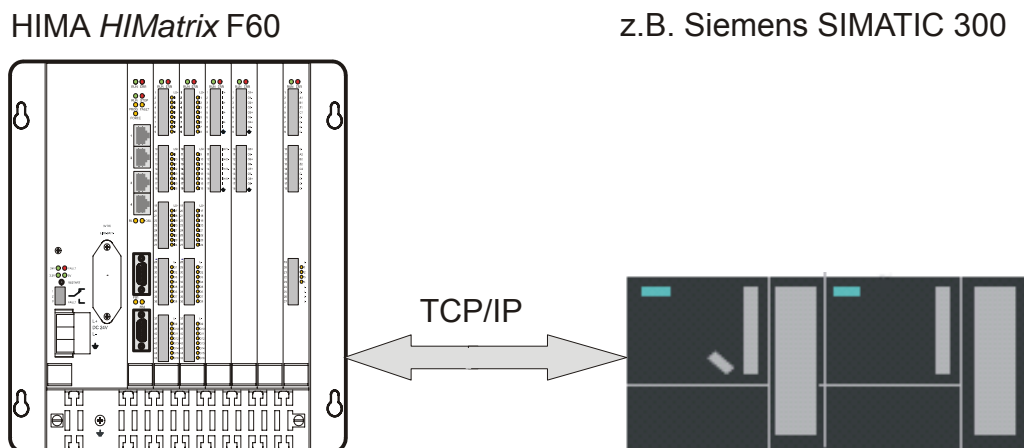


Bild 18: Die HIMA *HIMatrix* F60 und die Siemens SIMATIC 300 werden über TCP/IP verbunden.

4.1.1 Konfiguration des Datenaustauschs

In dieser Anwendung sollen zwei BYTES und ein WORD von der HIMA *HIMatrix* F60 zur Siemens SIMATIC 300 gesendet werden. Die Signale werden in der SIMATIC 300 vom Baustein „AG_RECV“ (FC 6) empfangen und intern an den Baustein „AG_SEND“ (FC 5) übergeben. Über den Baustein „AG_SEND“ (FC 5) sendet die SIMATIC 300 die Signale unverändert an die HIMA *HIMatrix* F60 zurück. Die Übertragung der Signale kann der Anwender nach der Konfiguration mit dem HIMA Forceeditor prüfen.

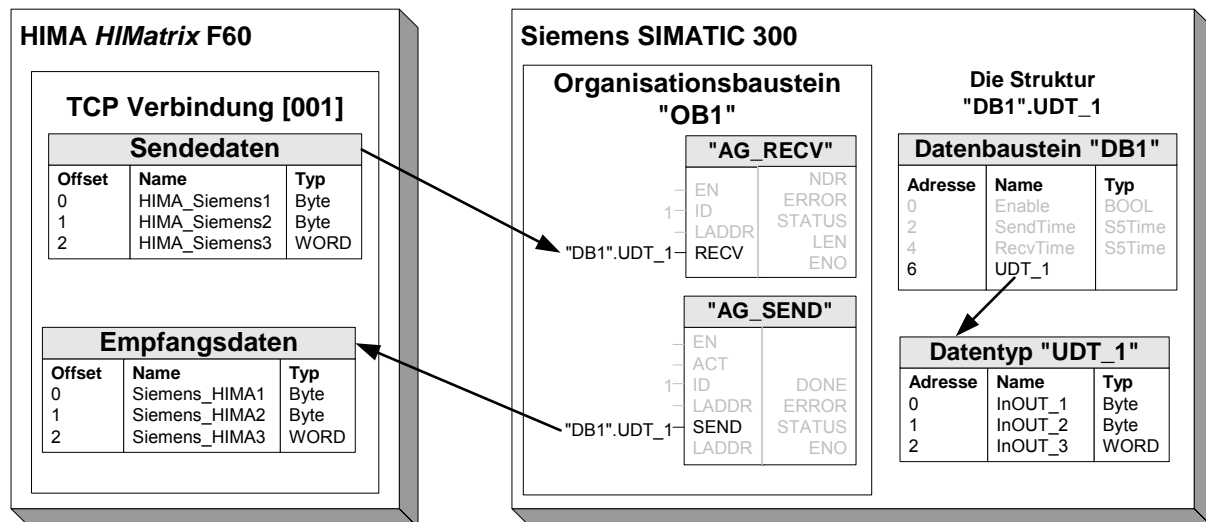


Bild 19: Datenaustausch zwischen HIMA und Siemens über TCP S/R

Beschreibung zum Bild 19	
HIMA <i>HIMatrix</i> F60	
TCP-Verbindung [001]	In diesem Dialog stehen alle Parameter, die für die Kommunikation mit dem Kommunikationspartner (Siemens Simatic 300) notwendig sind.
Sendedaten	Die Offsets und Typen der Signale in der <i>HIMatrix</i> F60 müssen mit der Adresse und den Typen der Variablen im Datentyp „UDT_1“ der SIMATIC 300 übereinstimmen.
Empfangsdaten	Die Offsets und Typen der Signale in der <i>HIMatrix</i> F60 müssen mit der Adresse und den Typen der Variablen im Datentyp „UDT_1“ der SIMATIC 300 übereinstimmen.
Siemens SIMATIC 300	
Organisationsbaustein „OB1“	Die Funktionsbausteine „AG_RECV“ (FC6) und „AG_SEND“ (FC 5) müssen im Organisationsbaustein „OB1“ angelegt und konfiguriert werden.
„AG_RECV“ (FC 6)	Der Funktionsbaustein „AG_RECV“ (FC 6) übernimmt die empfangenen Daten vom Kommunikationspartner in den Datentyp „DB1“.UDT_1. Die Eingänge „ID“ und „LADDR“ müssen für die Kommuni-

	kation mit dem Kommunikationspartner entsprechend konfiguriert werden.
„AG_SEND“ (FC 5)	Der Funktionsbaustein „AG_SEND“ (FC 5) überträgt die Daten aus dem Datentyp „DB1“.UDT_1 zum Kommunikationspartner. Die Eingänge „ID“ und „LADDR“ müssen für die Kommunikation mit dem Kommunikationspartner entsprechend konfiguriert werden.
Datenbaustein „DB1“	Der Datentyp „UDT_1“ wird im Datenbaustein „DB1“ definiert.
Datentyp „UDT_1“	Die Adressen und Typen der Variablen in der SIMATIC 300 müssen mit den Offsets und den Typen der <i>HIMatrix</i> F60 übereinstimmen. Der Datentyp „UDT_1“ übernimmt die empfangenen Nutzdaten und speichert diese bis zur Übertragung an den Kommunikationspartner.

Tabelle 45: Beschreibung zum Bild 19.

4.1.2 Konfiguration der Siemens SIMATIC 300

Achtung Die folgende Schrittanleitung zur Konfiguration der Siemens-Steuerung erhebt keinen Anspruch auf Vollständigkeit.
Alle Angaben sind ohne Gewähr, maßgebend zur Projektierung der Siemens-Steuerung ist die Dokumentation von Siemens.

Schritt 1: Erstellen Sie ein neues SIMATIC 300 Projekt:

- ❑ Starten Sie den SIMATIC Manager.
- ❑ Wählen Sie über das Hauptmenü *Datei->Assistent „Neues Projekt“* um ein neues Projekt anzulegen.
- ❑ Folgen Sie den Anweisungen des Assistenten.
- ❑ Konfigurieren Sie die „Industrial Ethernet“ und „MPI“ Verbindungen.

Schritt 2: Erstellen Sie den Datentyp „UDT1“ mit den folgenden Variablen:

- ❑ Wechseln Sie in den Ordner „Bausteine“ im Siemens SIMATIC Manager.
- ❑ Wählen Sie im Hauptmenü *Einfügen->S7 Baustein->Datentyp* und erstellen Sie einen Datentyp.
- ❑ Geben Sie dem Datentyp den Namen „UDT1“
- ❑ Geben Sie dem Datentyp den Symbolischen Namen „UDT_1“
- ❑ Erstellen Sie im Datentyp „UDT_1“ die Variablen wie in Bild 20.

Adresse	Name	Typ	Anfangswert	Kommentar
0.0		STRUCT		
+0.0	InOut_1	BYTE	B#16#0	
+1.0	InOut_2	BYTE	B#16#0	
+2.0	InOut_3	WORD	W#16#0	
=4.0		END_STRUCT		

Bild 20: Im Datentyp „UDT_1“ werden die „InOut_x“ Variablen erstellt.

Hinweis Beim zyklischem und azyklischem Datenaustausch ist zu beachten, dass manche Steuerungen (z.B. SIMATIC 300) so genannte „Pad Bytes“ einfügen. Damit wird sichergestellt, dass alle Datentypen die größer als ein Byte sind, immer an einem geraden Offset beginnen und dass die Gesamtlänge aller definierten Variablen ebenfalls immer gerade ist.
In diesen Fällen müssen an den entsprechenden Stellen Dummy-Bytes in der HIMA-Steuerung eingefügt werden (siehe Kapitel 1.4).

Schritt 3: Erstellen Sie den Datenbaustein „DB1“ für die Funktionsbausteine „FC 5“ and „FC 6“:

- Wählen Sie im Hauptmenü *Einfügen->S7 Baustein->Datenbaustein* und erstellen Sie einen Datenbaustein.
 - Geben Sie dem Datenbaustein den Namen „DB1“.
 - Geben Sie dem Datenbaustein den Symbolischen Namen „DB1“.
- Weisen Sie dem Datenbaustein „DB1“ den Datentyp „UDT_1“ zu.
- Erstellen Sie im Datenbaustein „DB1“ die Datentypen wie in Bild 21.

Adresse	Name	Typ	Anfangswert
0.0		STRUCT	
+0.0	Enable	BOOL	FALSE
+2.0	SendTime	S5TIME	S5T#100MS
+4.0	RecvTime	S5TIME	S5T#10MS
+6.0	UDT_1	"UDT_1"	
=10.0		END_STRUCT	

Bild 21: Die Variablen für die Funktionsbausteine „AG_RECV“ (FC 6)“ und „AG_SEND“ (FC 5) werden im Datenbaustein „DB1“ angelegt.

Schritt 4: Erstellen Sie im Symboleditor die folgenden Symbole:

- Öffnen Sie das Dialogfenster „KOP/AWL/FUP“ mit einem Doppelklick auf den Organisationsbaustein „OB1“.
- Öffnen Sie über das Hauptmenü mit *Extras->Symboltabelle* den Symboleditor.
- Ergänzen Sie den Symboleditor mit den Variablen wie in Bild 22.

S7-Programm(1) (Symbole) -- S7_Pro5\SIMATIC 300-Stat				
	Status	Symbol ▲	Adresse	Datentyp
1		Cycle Execution	OB 1	OB 1
2		DB1	DB 1	DB 1
3		UDT_1	UDT 1	UDT 1
4		RecDone	M 1.0	BOOL
5		RecError	M 1.1	BOOL
6		SendDone	M 1.2	BOOL
7		SendError	M 1.3	BOOL
8		RecStatus	MW 1	WORD
9		RecLen	MW 3	INT
10		SendStatus	MW 5	WORD
11				

Bild 22: Erstellen Sie die Variablen M1.0 bis MW 5.

Schritt 5: Erstellen Sie den FC-Baustein „AG_RECV“ (FC 6):

- Wechseln Sie in das Dialogfenster „KOP/AWL/FUP“.
- Wählen Sie nacheinander aus der Struktur im linken Teil des Fenster
 - ein „Oder-Gatter“
 - ein „S_VIMP“
 - ein „AG_RECV“ (FC 6)und ziehen Sie diese Funktionsbausteine per Drag&Drop in den Organisationsbaustein „OB1“.
- Verbinden und Konfigurieren Sie die Funktionsbausteine wie in Bild 23.
- Öffnen Sie das Kontextmenü mit einem rechten Mausklick auf den FC-Baustein „AG_RECV“ (FC 6).
 - Wählen Sie *Eigenschaften* um das Dialogfenster „Eigenschaften“ der TCP-Verbindung zu öffnen.
 - Deaktivieren Sie *Aktiver Verbindungsaufbau*
 - Konfigurieren Sie die *Ports*.
 - Notieren Sie den Bausteinparameter „LADDR“ und tragen Sie diese im Funktionsplan am Baustein „AG_RECV“ (FC 6) ein.

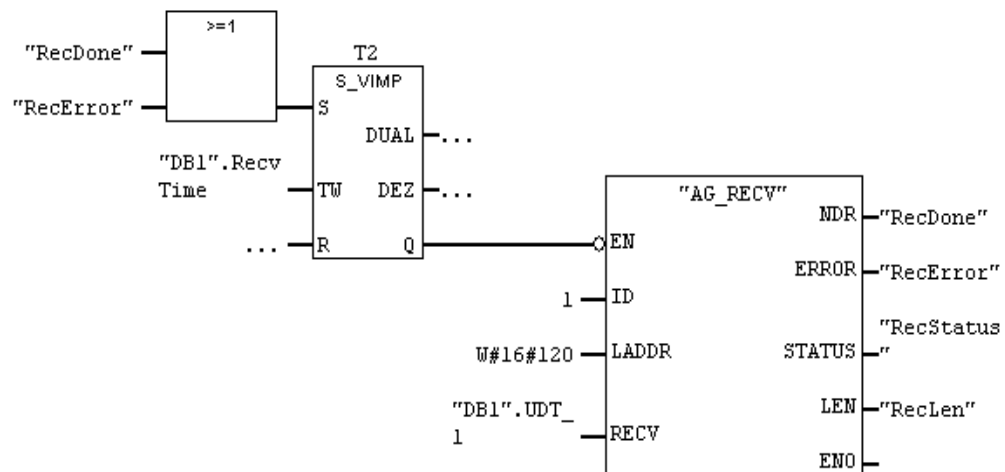


Bild 23: Die Konfiguration für „AG_RECV“ (FC 6)

Schritt 6: Erstellen Sie den FC-Baustein „AG_SEND“ (FC 5):

- Wechseln Sie in das Dialogfenster „KOP/AWL/FUP“.
- Wählen Sie nacheinander aus der Struktur im linken Teil des Fenster
 - ein „Oder-Gatter“
 - ein „S_VIMP“
 - ein „AG_SEND“ (FC 5)und ziehen Sie diese Funktionsbausteine per Drag&Drop in den Organisationsbaustein „OB1“.
- Verbinden und Konfigurieren Sie die Funktionsbausteine wie in Bild 24.
- Öffnen Sie das Kontextmenü mit einem rechten Mausklick auf den FC-Baustein „AG_SEND“ (FC 5).
 - Wählen Sie *Eigenschaften* um den Dialog Eigenschaften der TCP-Verbindung zu öffnen.
 - Deaktivieren Sie *Aktiver Verbindungsaufbau*.
 - Konfigurieren Sie die *Ports*.
 - Notieren Sie den Bausteinparameter „LADDR“ und tragen Sie diesen im Funktionsplan am Baustein „AG_SEND“ (FC 5) ein.

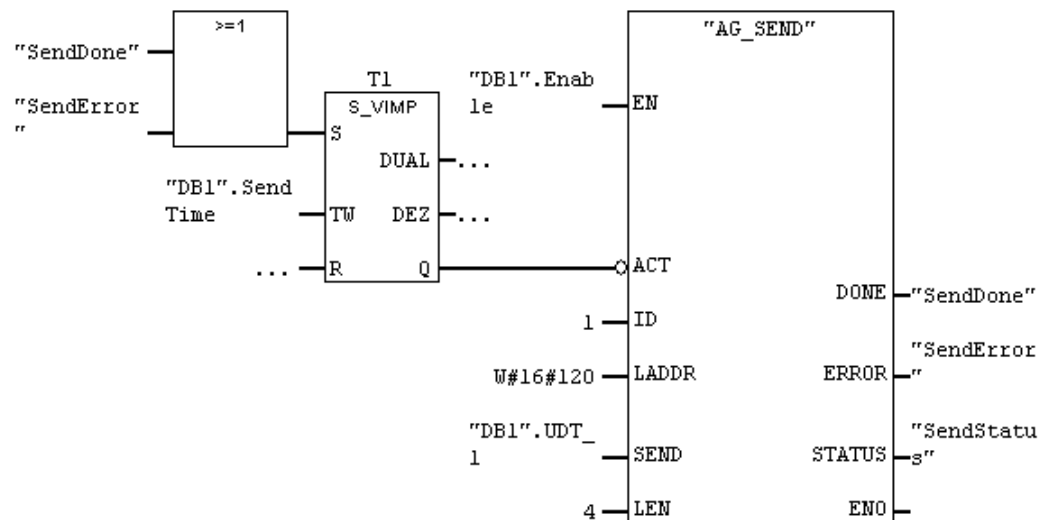


Bild 24: Die Konfiguration für „AG_SEND“ (FC 5)

Schritt 7: Laden Sie den Code in die SIMATIC 300 Steuerung:

- Starten Sie den Codegenerator für das Programm.
- Stellen Sie sicher, dass der code ohne Fehler generiert wurde.
- Laden Sie das Programm in die SIMATIC 300 Steuerung.

4.1.3 Konfiguration des TCP S/R Protokolls in einer *HIMatrix* F60

Schritt 1: Erstellen Sie ein neues HIMA *HIMatrix* F60 Projekt:

- ❑ Starten Sie **ELOP II Factory**.
- ❑ Starten Sie im Projektmanagement über das Hauptmenü *Projekt->Projekt-Assistent...* den Projekt-Assistenten.
- ❑ Folgen Sie den Anweisungen des Projekt-Assistenten um ein neues Projekt anzulegen.
 - ❑ Geben Sie dem Projekt einen Namen (z.B. „SR_Test“).
 - ❑ Geben Sie der Ressource einen Namen (z.B. „F60“), die automatisch mit einem neuen Project angelegt wird.

Hinweis Für die Konfiguration der HIMA *HIMatrix* Steuerungen und dem Umgang mit dem Programmierwerkzeug **ELOP II Factory** wird das Handbuch „Erste Schritte **ELOP II Factory**“ und die Online-Hilfe von **ELOP II Factory** empfohlen.

Schritt 2: Konfigurieren Sie die Ressource F60:

- ❑ Wechseln Sie in das Hardware-Management.
- ❑ Öffnen Sie das Kontextmenü der Ressource F60 und ändern Sie die folgenden Parameter:
 - ❑ Typ „F60“ zuweisen
 - ❑ System Id [SRS] zuweisen
 - ❑ „Forcen erlauben“ aktivieren.

Schritt 3: Legen Sie das TCP S/R Protokoll in der Ressource F60 an:

- ❑ Öffnen Sie im Strukturbaum das Verzeichnis der F60.
- ❑ Klicken Sie mit der rechten Maustaste auf „Protokolle“.
- ❑ Wählen Sie *Neu->Send/Receive over TCP*.

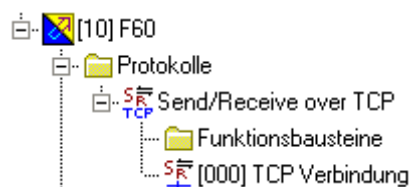


Bild 25: Ansicht der Struktur eines neu angelegten TCP S/R Protokolls

Schritt 4: Konfigurieren Sie die TCP/IP Verbindung der Ressource F60:

- ❑ Wählen Sie in der Struktur die TCP-Verbindung.
- ❑ Öffnen Sie das Kontextmenü der TCP-Verbindung.
- ❑ Wählen Sie *Eigenschaften* um das Dialogfenster „Eigenschaften“ zu öffnen.
- ❑ Konfigurieren Sie die Eigenschaften wie in Bild 26: .
- ❑ Um weitere TCP-Verbindungen zu erstellen, wählen Sie im Kontextmenü von „Send/Receive over TCP“ *Neu->TCP-Verbindung*.

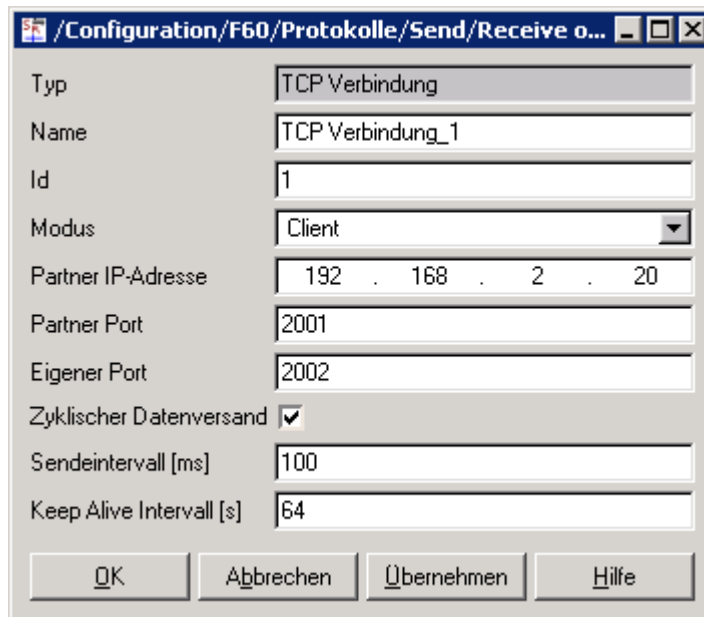


Bild 26: Die *HIMatrix F60* ist der aktive Kommunikationspartner (Client)

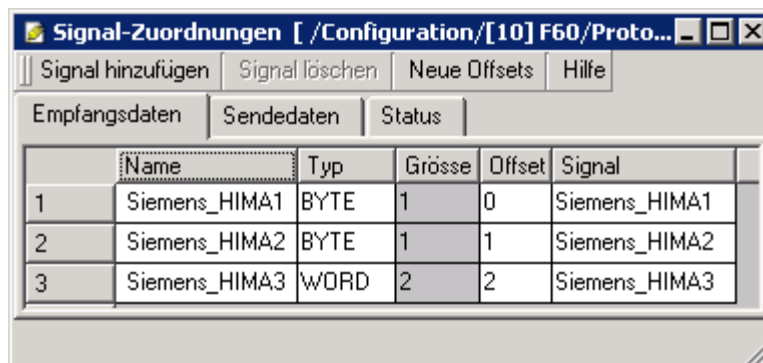
Hinweis Wenn zyklischer Datenaustausch zwischen zwei Steuerungen parametrisiert werden soll, muss im Dialog „Eigenschaften“ der TCP-Verbindung die Option „Zyklischer Datenversand“ aktiviert sein.

Schritt 5: Erstellen Sie die folgenden Signale im Signaleditor der F60:

- ❑ Öffnen Sie mit *Signale->Editor* den Signaleditor im Hardware-Management.
- ❑ Erstellen Sie die folgenden Signale, die als Empfangsdaten für die F60 verwendet werden:
 - ❑ „Siemens_HIMA1“ vom Typ „Byte“
 - ❑ „Siemens_HIMA2“ vom Typ „Byte“
 - ❑ „Siemens_HIMA3“ vom Typ „WORD“
- ❑ Erstellen Sie die folgenden Signale, die als Sendedaten für die F60 verwendet werden:
 - ❑ „HIMA_Siemens1“ vom Typ „Byte“
 - ❑ „HIMA_Siemens2“ vom Typ „Byte“
 - ❑ „HIMA_Siemens3“ vom Typ „WORD“

Schritt 6: Verbinden der „Empfangsdaten“ der F60 mit den SIMATIC 300 Variablen:

- ❑ Wählen Sie die in Schritt 4 konfigurierte TCP-Verbindung mit der Id „001“.
- ❑ Öffnen Sie das Kontextmenü und wählen Sie *Signale verbinden*.
- ❑ Öffnen Sie das Register „Empfangsdaten“ im Dialogfenster „Signal-Zuordnungen“.
- ❑ Öffnen Sie den Signaleditor mit *Signale->Editor* im Hardware-Management.
- ❑ Klicken Sie im Signaleditor auf den „Namen“ des Signals „Siemens_HIMA1“ und ziehen Sie das Signal per Drag & Drop in das Register „Empfangsdaten“ des Dialogfensters „Signal Zuordnungen“.
- ❑ Wiederholen Sie den vorherigen Schritt, wenn Sie mehr als ein Eingangssignal für die TCP-Verbindung der F60 definiert haben.
- ❑ Klicken Sie im Dialogfenster „Signal-Zuordnungen“ auf die Schaltfläche *Neue Offsets*.
- ❑ Klicken Sie im Popup-Fenster „Offsets nummerieren“ auf die Schaltfläche *Nummerieren*.
- ❑ Schließen Sie das Dialogfenster.



	Name	Typ	Grösse	Offset	Signal
1	Siemens_HIMA1	BYTE	1	0	Siemens_HIMA1
2	Siemens_HIMA2	BYTE	1	1	Siemens_HIMA2
3	Siemens_HIMA3	WORD	2	2	Siemens_HIMA3

Bild 27: Empfangsdaten der *HIMatrix* F60

Hinweis Beachten Sie, dass die Offsets der Signale in der *HIMatrix* F60 mit den Adressen der Variablen im Datentyp „UDT_1“ der SIMATIC 300 übereinstimmen müssen.

Schritt 7: Verbinden der „Sendedaten“ der F60 mit den SIMATIC 300 Variablen:

- ❑ Wählen Sie die in Schritt 4 erstellte TCP-Verbindung mit der Id „001“.
- ❑ Öffnen Sie das Kontextmenü und wählen Sie *Signale verbinden*.
- ❑ Öffnen Sie das Register „Sendedaten“ im Dialogfenster „Signal-Zuordnungen“.
- ❑ Öffnen Sie im Hardware-Management den Signaleditor mit *Signale->Editor*.
- ❑ Klicken Sie im Signaleditor auf den „Namen“ des Signals „HIMA_Siemens1“ und ziehen Sie das Signal per Drag & Drop in das Register „Sendedaten“ des Dialogfensters „Signal Zuordnungen“.
- ❑ Wiederholen Sie den vorherigen Schritt, wenn Sie mehr als ein Ausgangssignal für die TCP S/R-Verbindung der F60 definiert haben.
- ❑ Klicken Sie im Dialogfenster „Signal-Zuordnungen“ auf die Schaltfläche *Neue Offsets*.
- ❑ Klicken Sie im Popup-Fenster „Offsets nummerieren“ auf die Schaltfläche *Nummerieren*.
- ❑ Schließen Sie das Dialogfenster.

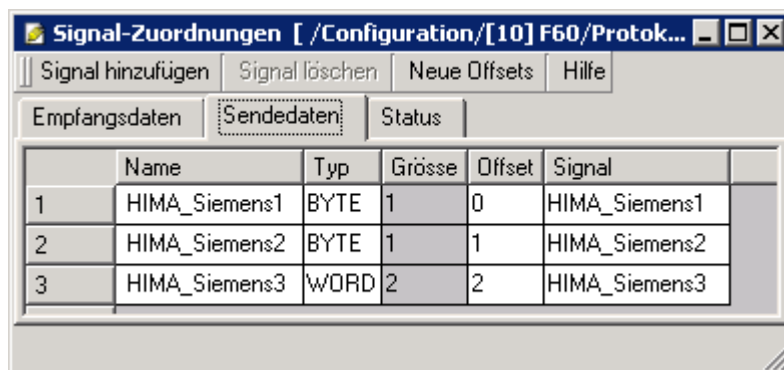


Bild 28: Sendedaten der HIMatrix F60

Hinweis Beachten Sie, dass die Offsets der Signale in der *HIMatrix* F60 mit den Adressen der Variablen im Datentyp „UDT_1“ der SIMATIC 300 übereinstimmen müssen.

Schritt 8: Laden Sie den Code in die Ressource F60:

- ❑ Starten Sie den Code Generator für die Ressource F60.
- ❑ Stellen Sie sicher, dass der Code fehlerfrei generiert wurde (siehe Fehler-Status-Anzeige).
- ❑ Laden Sie den Code in die Ressource F60.

Schritt 9: Kontrollieren Sie die Kommunikation mit dem HIMA Force Editor:

- ❑ Öffnen Sie das Kontextmenü der Ressource F60 und öffnen Sie den Forceeditor über *Online->Force Editor*.
- ❑ Wählen Sie alle Signale aus, die Sie im Schritt 5 erstellt haben.
- ❑ Forcen Sie die Signale aus dem Register „Sendedaten“.

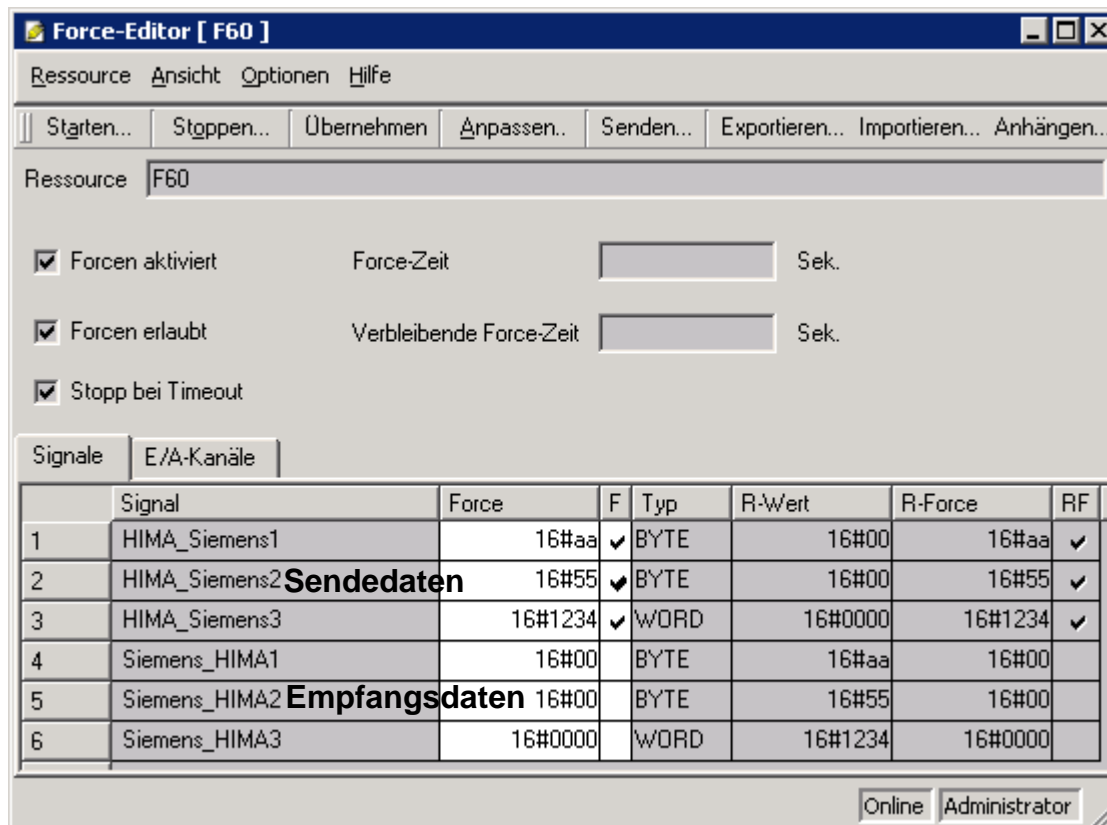


Bild 29: Die Sendedaten werden von der Siemens Steuerung an die HIMA Steuerung zurück gesendet und in die Empfangsdaten geschrieben.

HIMA
...die sichere Entscheidung.



HIMA Paul Hildebrandt GmbH
Industrie-Automatisierung
Postfach 1261 • 68777 Brühl

Telefon: (06202) 709-0 • Telefax: (06202) 709-107
E-mail: info@hima.com • Internet: www.hima.de

(0921)