10/10 MN05006004Z-DE
ersetzt 08/07 AWB2786-1593de

Datenübertragung zwischen easy- und IEC-Steuerungen easyNet





Powering Business Worldwide

Alle Marken- und Produktnamen sind Warenzeichen oder eingetragene Warenzeichen der jeweiligen Titelhalter.

Störfallservice
Bitte rufen Sie Ihre lokale Vertretung an:
http://www.eaton.com/moeller/aftersales
oder
Hotline After Sales Service:
+49 (0) 180 5 223822 (de, en)
AfterSalesEGBonn@eaton.com

Originalbetriebsanleitung
Die deutsche Ausführung dieses Dokuments ist die Originalbetriebsanleitung.

Übersetzung der Originalbetriebsanleitung
Alle nicht deutschen Sprachausgaben dieses Dokuments sind Übersetzungen der Originalbetriebsanleitung.

1. Auflage 2007, Redaktionsdatum 08/07
2. Auflage 2010, Redaktionsdatum 10/10

© 2007 Eaton Industries GmbH, 53105 Bonn

Autor: Peter Roersch, Andreas Lüngen
Redaktion: Heidrun Riege

Alle Rechte, auch die der Übersetzung, vorbehalten.

Kein Teil dieses Handbuches darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder einem anderen Verfahren) ohne schriftliche Zustimmung der Firma Eaton Industries GmbH, Bonn, reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.

Änderungen vorbehalten.

Inhalt

Zu diesem Handbuch **3**
Abkürzungen und Symbole 3

1 Das Netzwerk easyNet **5**
Netzwerkübersicht 5
Teilnehmer und Netzwerk konfigurieren 5
Übersicht Datenübertragung 5

2 Funktionen **7**
Funktionen zur Datenübertragung 7
Beschreibung der Funktionen 7
– Funktion NET_UPDATE für den Datenaustausch 7
– Funktion NET_GET zum Abholen der im Netz abgelegten Daten 8

3 Strukturen **9**
Struktur EASY_NET_MAIN 9
– Aufbau der Struktur 9
– Komponenten der Struktur 9
Struktur EASY_NET_INFO 9
– Aufbau der Struktur 9
– Komponenten der Struktur 9
Struktur EASY_NET_DEVICE 10
– Aufbau der Struktur 10
– Komponenten der Struktur 10
Struktur EASY_NET_SND 11
– Aufbau der Struktur 11
– Komponenten der Struktur 11
Struktur EASY_NET_SNDx 12
– Aufbau der Struktur 12
– Komponenten der Struktur 12
Struktur EASY_NET_PUT 12
– Aufbau der Struktur 12
– Komponenten der Struktur 12
Struktur EASY_NET_RCV 13
– Aufbau der Struktur 13
– Komponenten der Struktur 13
Struktur EASY_NET_GET 14
– Aufbau der Struktur 14
– Komponenten der Struktur 14

4 Anwendungen **15**

IEC-Teilnehmer <- IEC-/easy-Teilnehmer (I/O-Information) 15
IEC-Teilnehmer (Net-ID1) -> Remote I/O-Teilnehmer, z. B. easy800 (I/O-Information) 16
IEC-Teilnehmer (Net-IDx) <-> EASY223-SWIRE-Gateway (I/O-Information) 17
– IEC-Teilnehmer (Net-IDx) <- EASY223-SWIRE-Gateway (I/O-Information) 17
– IEC-Teilnehmer (Net-IDn) -> EASY223-SWIRE-Gateway (I/O-Information) 19
IEC-Teilnehmer <-> IEC-/easy-Teilnehmer (Bit-Information) 20
IEC-Teilnehmer <-> mehrere IEC-/easy-Teilnehmer (Doppelwort-Datenaustausch) 22
– PUT-Funktion (Strukturelement PUT) 22
– Die NET_GET-Funktion 25
Synchronisation der Echtzeituhr 26

5 Netzwerk (easyNet) konfigurieren **27**

Konfiguration durchführen 27
– Konfigurieren mit easySoft-CoDeSys (Net-ID1 = IEC-Teilnehmer) 27
– Konfigurieren mit easySoft (Net-ID1 = easy- oder IEC-Teilnehmer)28
– Konfigurieren über das Gerätedisplay 28
– Konfigurieren mit der Funktion Net_Config 29

6 Kombination von easyNet mit CAN/CANopen **31**

Kombinationsmöglichkeiten 31
Hinweise zum Betrieb 31
– Buslast steuern 31
– Auslastung des CAN-Busses anzeigen 31
– easyNet-Telegramm überwachen 32

7 Programmieren über easyNet (Routing) **33**

8 Bus-Topologie **35**

Übersicht 35
CANopen/easyNet-Schnittstelle 36
– XC200 36
– EC4-200, easy800, MFD-(AC-)CP8 36

Stichwortverzeichnis **37**

Zu diesem Handbuch

Änderungsprotokoll

Gegenüber der letzten Ausgabe hat es folgende wesentliche Änderung gegeben:

Redaktions-datum	Seite	Stichwort	Änderung
10/10	allg.	Umstellung auf Eaton-Bezeichnungen	✓

Dieses Handbuch beinhaltet im Wesentlichen den Zugriff eines IEC-Teilnehmers, z. B. einer XC200, auf andere Teilnehmer im Netzwerk easyNet. Der Zugriff der IEC-Teilnehmer erfolgt mit Hilfe des Programms, in das die Funktionen zum Senden und Empfangen der Daten eingebunden werden.

Die Funktionen enthalten Variablen vom Typ Struktur. Die Funktionen und die Strukturen werden in Kapitel 2 und Kapitel 3 stichwortartig beschrieben.

Der Zusammenhang zwischen den Funktionen und Strukturen sowie deren Einbindung in das Programm gehen aus Kapitel 4 hervor.

Die Möglichkeiten der Datenübertragung werden in Kapitel 2 beschrieben.

Das Handbuch ist in folgende Kapitel gegliedert:

- Das Netzwerk easyNet
  Gibt eine Einführung in das Netzwerk easyNet,
  → Kapitel 1, Seite 5.
- Funktionen
  Beschreibt die Funktionen und ihre Eigenschaften,
  → Kapitel 2, Seite 7.
- Strukturen
  Beschreibt die Struktuen, → Kapitel 3, Seite 9.
- Anwendung
  Zeigt die Möglichkeiten der Datenübertragung zwischen IEC-Teilnehmern und easy-Teilnehmern, z. B. easy800, in Form von Beispielen. In diesen Beispielen werden die Funktionen und Strukturen benutzt, → Kapitel 4, Seite 15.
  Zusätzliche Hinweise, wie der Zugriff der easy-Teilnehmer auf das Netzwerk, finden Sie in den Handbüchern der easy-Teilnehmer.
- Netzwerk konfigurieren
  Beschreibt, wie über den ersten Teilnehmer (Net-ID 1) am easyNet die anderen Teilnehmer konfiguriert werden,
  → Kapitel 5, Seite 27.
- Kombination von easyNet mit CAN/CANopen
  Gibt Hinweise zu den Kombinationen easyNet – CANopen, CAN-Direkt, CAN-Netzwerkvariablen,
  → Kapitel 6, Seite 31.
- Routing
  Zeigt die Möglichkeit der Kommunikation zwischen PC und Teilnehmer indirekt über das easyNet, → Kapitel 7, Seite 33.
- Bus-Topologie
  Zeigt verschiedene Topologien auf, → Kapitel 8, Seite 35.

Abkürzungen und Symbole

In diesem Handbuch werden Abkürzungen und Symbole eingesetzt, die folgende Bedeutung haben:

In einigen Abbildungen ist der Begriff „Station" gleichbedeutend mit dem Begriff„Teilnehmer" im Fließtext verwendet.

▶ zeigt Handlungsanweisungen an

→ macht Sie aufmerksam auf interessante Tipps und Zusatzinformationen

Für eine gute Übersichtlichkeit finden Sie auf den linken Seiten im Kopf die Kapitelüberschrift und auf den rechten Seiten den aktuellen Abschnitt. Ausnahmen sind Kapitelanfangsseiten und leere Seiten am Kapitelende.

1 Das Netzwerk easyNet

Netzwerkübersicht

easyNet basiert auf dem CAN-Netzwerk, das einen Austausch von Prozess- und Systemdaten ermöglicht. Es ist für 8 Teilnehmer, z. B. Steuerungen, ausgelegt. Es gibt zwei Teilnehmertypen, die unterschiedlich programmiert werden:

- easy-Teilnehmer (Programmierung in Form eines Schaltplans)
  - easy800
  - MFD-CP8-…
  - EASY223-SWIRE (SmartWire-Gateway ohne Programm, mit Konfiguration)

- IEC-Teilnehmer (Programmierung nach der Norm IEC 1131)
  - XC200
  - EC4-200
  - MFD4

Teilnehmer und Netzwerk konfigurieren

Im easyNet ist jeder Teilnehmer zu konfigurieren. Zur Identifikation im Netzwerk erhält der Teilnehmer eine Adresse (Net-ID), sowie Parameter, die das Verhalten des Teilnehmers im Netzwerk festlegen (z. B. Baudrate, SEND IO, Remote Run). easy-Teilnehmer konfigurieren Sie im Konfigurator der Programmiersoftware oder über Tastatur/Display am Gerät.

Die IEC-Teilnehmer können Sie einzeln konfigurieren. Im Konfigurator der easySoft-CoDeSys parametrieren Sie die Kommunikationsbedingungen und laden sie mit dem Anwenderprogramm in den Teilnehmer.

easy-Teilnehmer werden im Netzverbund mit der easySoft konfiguriert. Sind easy- und IEC-Teilnehmer in einem Projekt vorhanden, so sind beide Teilnehmertypen im Netzverbund zu konfigurieren.

In der easySoft wählen Sie in der „Projektansicht" ein Symbol für den Teilnehmer aus und ziehen das Symbol auf die Werkbank. Unter der Bezeichnung „Net-Teilnehmer" steht für die IEC-Teilnehmer ein allgemeines Symbol zur Verfügung. Positionieren Sie die Geräte und geben Sie die Kommunikationparameter ein.

→ Das Gerät an der Position 1 in der Projektansicht erhält die Net-ID = 1 fest zugeordnet und muss auch das erste Gerät am physikalischen Netzstrang sein.

Wenn Sie alle Teilnehmer konfiguriert haben, können Sie die Konfiguration in die easy-Teilnehmer laden. Die Konfiguration und das Programm für die IEC-Teilnehmer sind separat mit der Programmiersoftware easySoft-CoDeSys zu laden!

Übersicht Datenübertragung

Jeder Teilnehmer erhält eine Net-ID-Nummer (1, … ,8) und kann mit allen anderen Teilnehmern Daten austauschen. Als Datenformat stehen Bit, Byte, Wort und Doppelwort zur Verfügung. Ein Teilnehmer mit der Net-ID = 1 muss immer vorhanden sein. Er übernimmt die Verwaltung der Kommunikation im Netzwerk. Als einziger Teilnehmer kann er die Ausgänge der Remote I/O-Teilnehmern setzen.

Remote I/O-Teilnehmer sind Geräte, wie z. B. eine easy800, ohne Programm. Die Möglichkeiten der Datenübertragung sind → Kapitel „Funktionen", Seite 7 aufgeführt.

→ Ausführliche Hinweise zur Datenübertragung zwischen den easy-Teilnehmern finden Sie in den folgenden Bedienungshandbüchern:

- easy800 Steuerrelais (MN04902001Z-DE; frühere Bezeichnung AWB2528-1423D),
- EASY223-SWIRE (MN05006003Z-DE; frühere Bezeichnung AWB2528+1251-1589D),
- MFD-Titan Multi-Funktions-Display (MN05002001Z-DE; frühere Bezeichnung AWB2528-1480D).

Für die easy-Teilnehmer stellt die easySoft zum Senden und Empfangen der Daten über das Netz spezielle Operanden bereit.

Für die IEC-Teilnehmer wird das Senden und Empfangen der Daten mit Hilfe von Funktionen der easySoft-CoDeSys durchgeführt. Die Funktionen sind Bestandteil der Bibliothek „SysLibEasyNet.lib". Um sie in das Projekt zu integrieren, fügen Sie die Bibliothek „SysLibEasyNet.lib" in den Bibliotheksverwalter ein. Durch den Aufruf der Funktionen im Programm werden die Daten zwischen Anwenderprogramm und der für den Anwender nicht sichtbaren Protokolltask ausgetauscht, die unabhängig vom Anwenderprogramm arbeitet. Die Protokolltask ist ein Programm, das die eigentliche Datenübertragung zwischen dem IEC-Teilnehmer und den einzelnen Teilnehmern übernimmt. Sie übernimmt auch administrative Aufgaben wie Teilnehmerüberwachung und die Unterstützung des Routings.

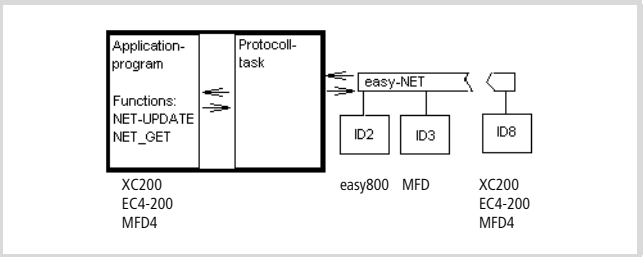


Abbildung 1: Datenübertragung zwischen IEC- und easy-/IEC-Teilnehmern

2 Funktionen

Funktionen zur Datenübertragung

Die Bibliothek „SysLibEasyNet.lib" stellt die beiden Funktionen NET_UPDATE und NET_GET zum Senden und Empfangen der Daten zur Verfügung, sowie NET_CONFIG zur Konfiguration des Netzwerks. Die Funktionen NET_UPDATE und NET_GET bieten folgende Übertragungsmöglichkeiten:

- NET_UPDATE
  - Ausgangszustände (Bit) des Teilnehmers mit der Net-ID = 1 an einen Teilnehmer ohne Programm (Remote I/O) senden (Operand Q/S).
    → Abschnitt „IEC-Teilnehmer (Net-ID1) -> Remote I/O-Teilnehmer, z. B. easy800 (I/O-Information)", Seite 16
    → Abschnitt „IEC-Teilnehmer (Net-IDn) -> EASY223-SWIRE-Gateway (I/O-Information)", Seite 19
  - Ein-/Ausgangzustände (Bit) von jedem Teilnehmer (auch Remote I/O) abfragen (Operand Q/S und I/R).
    → Abschnitt „IEC-Teilnehmer <- IEC-/easy-Teilnehmer (I/O-Information)", Seite 15,
    → Abschnitt „IEC-Teilnehmer (Net-IDx) <- EASY223-SWIRE-Gateway (I/O-Information)", Seite 17.
  - Bit-Informationen zwischen zwei Teilnehmern mit Programm austauschen. Der Sendeteilnehmer überträgt das Bit mit dem Operand SN. Der Empfangsteilnehmer erfasst das Bit mit dem Operand RN.
    → Abschnitt „IEC-Teilnehmer <-> IEC-/easy-Teilnehmer (Bit-Information)", Seite 20,
  - Werte (DWORD-Format) ins Netzwerk stellen, die von anderen Teilnehmern mit Hilfe der NET-GET-Funktion abgeholt werden.
    → Abschnitt „IEC-Teilnehmer <-> mehrere IEC-/easy-Teilnehmer (Doppelwort-Datenaustausch)", Seite 22.

- NET_GET
  - Werte (DWORD-Format) aus dem Netzwerk abholen, die ein anderer Teilnehmer mit der Funktion NET_UPDATE ins Netz gestellt hat.
    → Abschnitt „IEC-Teilnehmer <-> mehrere IEC-/easy-Teilnehmer (Doppelwort-Datenaustausch)", Seite 22.

Beschreibung der Funktionen

In diesem Abschnitt werden die Parameter der Funktionen beschrieben. Detaillierte Informationen sowie die Handhabung der Funktionen finden Sie → Kapitel „Anwendungen", Seite 15.

Funktion NET_UPDATE für den Datenaustausch

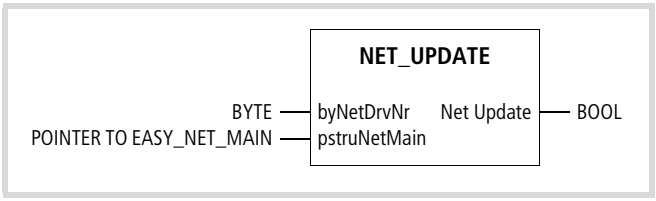


Abbildung 2: Funktion NET_UPDATE

Parameter der Funktion

byNetDrvNr	easyNet-Treibernummer (Defaulteinstellung = 0)
pstruNetMain	Pointer auf eine Datenstruktur vom Typ EASY_NET_MAIN
Net_Update	(Rückgabewert) TRUE: Funktion erfolgreich ausgeführt. FALSE: Funktionsabarbeitung mit Fehler abgebrochen.

Der Rückgabewert ist im Anwenderprogramm zu berücksichtigen (Fehlermeldung ausgeben).

Funktion aufrufen (Beispiel)

```
PROGRAM PLC_PRG
VAR
myEASY_NET_MAIN:EASY_NET_MAIN;
(*Deklaration lokal/global möglich*)
xResult: BOOL;
END_VAR
xResult:=Net_Update(0 ,ADR(myEASY_NET_MAIN));
```

→ Die Funktion NET_UPDATE muss in jedem Programmzyklus einmal aufgerufen werden. Der Aufruf darf nur aus einer Task erfolgen!

Beschreibung
Mit Hilfe der NET_UPDATE-Funktion können Sie Daten senden und empfangen. Die Funktion enthält den Parameter „pstruNetMain", an den Sie einen Zeiger anlegen, der auf die Struktur EASY_NET_MAIN bzw. auf eine Variable vom Typ EASY_NET_MAIN, z. B. „myEASY_NET_MAIN", zeigt.

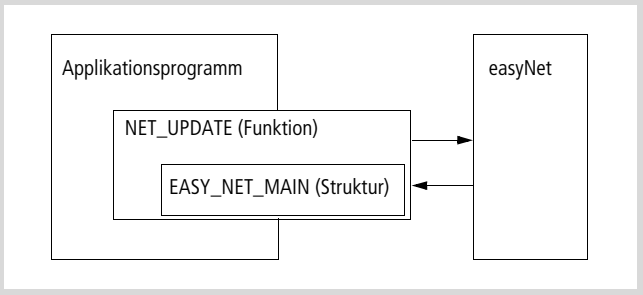


Abbildung 3: Zugriff des Anwenderprogramms auf easyNet

Die Struktur EASY_NET_MAIN und die Folgestrukturen werden → Kapitel „Strukturen", Seite 9 beschrieben.

Funktion NET_GET zum Abholen der im Netz abgelegten Daten

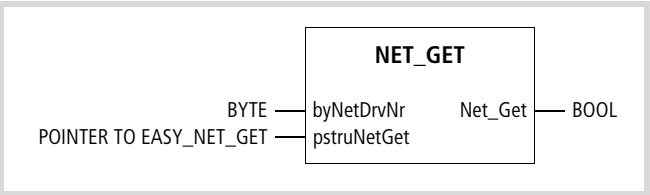


Abbildung 4: Funktion NET_GET

Parameter der Funktion

byNetDrvNr:	
	EasyNet Treibernummer. Defaulteinstellung = „0".
pstruNetGet:	
	Pointer auf eine Datenstruktur vom Typ EASY_NET_GET. Diese Struktur wird zum Datenaustausch zwischen easyNet und dem Anwenderprogramm des Teilnehmers verwendet.
Net_Get (Rückgabewert):	
	TRUE: Funktion erfolgreich ausgeführt.
	FALSE: Funktionsabarbeitung mit Fehler abgebrochen.

Funktion aufrufen (Beispiel)

```
PROGRAM PLC_PRG
VAR
myEASY_NET_GET : EASY_NET_GET;
xResult:BOOL;
END_VAR

xResult:=NET_GET(0,ADR(myEASY_NET_GET));
```

Beschreibung
Mit dem Aufruf der NET_GET-Funktion können Sie Daten aus dem Netzwerk lesen (abholen), die von anderen Teilnehmern mit der Struktur EASY_NET_PUT ins Netz gestellt wurden. Mit einem Funktionsaufruf können Sie nur ein Datum „abholen". Haben mehrere Teilnehmer ein Datum ins Netz gestellt, so kann ein Teilnehmer nacheinander alle Daten abfragen. Zur Abfrage hat er eine der folgenden Möglichkeiten:

- Die Funktion einmal in das Programm einzubinden und für jedes Datum aufzurufen.
  Vor jedem Funktionsaufruf ist die Net-ID des Teilnehmers und die Modulnummer des Datums anzugeben.
- Für jedes Datum die Funktion in das Programm einzubinden. Jeder Funktion wird eine Net-ID und eine Modulnummer fest zugeordnet. Alle Funktionen werden in einem Programmzyklus aufgerufen.

Pro Programmzyklus können beliebig viele NET_GET-Funktionsaufrufe erfolgen.

Die Funktion enthält den Parameter „pstruNetGet", an den Sie einen Zeiger auf die Struktur EASY_NET_GET anlegen bzw. auf eine Variable vom Typ EASY_NET_GET, z. B. „myEASY_NET_GET".

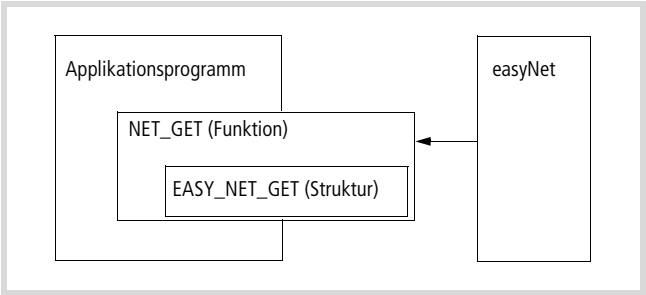


Abbildung 5: Abgelegte Daten aus dem Netz abholen

→ Abschnitt „Struktur EASY_NET_GET", Seite 14.

3 Strukturen

Struktur EASY_NET_MAIN

Aufbau der Struktur

```
TYPE EASY_NET_MAIN:
STRUCT
INFO: EASY_NET_INFO;
SND: EASY_NET_SND;
RCV: ARRAY [1..8] OF EASY_NET_RCV;
END_STRUCT
END_TYPE
```

Komponenten der Struktur

Die Komponenten der umfangreichen Struktur enthalten folgende Informationen:

INFO (Statusinformationen der Teilnehmer):
Die Informationen werden in die Struktur EASY_NET_INFO eingetragen.
Die Daten werden mit dem Aufruf der NET_UPDATE-Funktion aktualisiert.
SND (Sendedaten):
Die Informationen sind in die Struktur EASY_NET_SND einzutragen.
Sie werden mit dem nächsten Aufruf der NET_UPDATE Funktion übergeben.
RCV (Empfangsdaten):
Array aus 8 Elementen RCV[1, … ,8]. Jedes Element ist einem Teilnehmer zugeordnet. Die Zuordnung ergibt sich aus der Elementnummer und der Net-ID des Teilnehmers (1,…,8), die identisch sind. Ein Element ist vom Strukturtyp EASY_NET_RCV.
Die Daten werden mit dem Aufruf der NET_UPDATE-Funktion aktualisiert.

Struktur EASY_NET_INFO

Aus den Komponenten der Struktur können Sie die Zustände und den Status eines Teilnehmers abfragen. Die Komponente „aNET_STATE" enthält ein Array mit einem Element für jeden Teilnehmer. Ein Element enthält den Zustand und Status eines Teilnehmers.

Aufbau der Struktur

```
TYPE EASY_NET_INFO:
STRUCT
byNrOfActiveDevices: BYTE;
byNrOfConfDevices: BYTE;
uiBaudrate: UINT;
byMY_NET_ID: BYTE;
aNET_STATE: ARRAY [1..8] OF EASY_NET_DEVICE;
byError: BYTE;
END_STRUCT
END_TYPE
```

Komponenten der Struktur

byNrOfActiveDevices (Anzahl der aktiven Teilnehmer)
byNrOfConfDevices IEC-Teilnehmer mit der Net-ID1 (ausgenommen XC200):
Anzahl der Teilnehmer in der Konfigurationsliste. IEC-Teilnehmer mit der Net-ID2, … ,8:
Max. Anzahl der gleichzeitig als aktiv erkannten Teilnehmer.
uiBaudrate (Baudrate im easyNet)
byMY_NET_ID (Variable zeigt die Net-ID des Teilnehmers)
aNET_STATE (Zustand der Teilnehmer):
Array aus 8 Elementen „aNET_STATE[1, … ,8]". Jedes Element ist einem Teilnehmer zugeordnet. Die Zuordnung ergibt sich aus der Elementnummer und der Net-ID des Teilnehmers (1,…,8), die identisch sind. Ein Element ist vom Strukturtyp EASY_NET_DEVICE. In dieser Struktur wird der aktuelle Zustand des Teilnehmers hinterlegt.
byError (zeigt den letzten erkannten Fehler im easyNet an, → Tabelle 1).

Tabelle 1: Mögliche Fehlercodes

0x00	No Error.
0xC0	Undefined NET-Device missing.
0xC1	NET-Device 1 missing.
0xC2	NET-Device 2 missing.
0xC3	NET-Device 3 missing.
0xC4	NET-Device 4 missing.
0xC5	NET-Device 5 missing.
0xC6	NET-Device 6 missing.
0xC7	NET-Device 7 missing.
0xC8	NET-Device 8 missing.
0xD0	Daisy-Chain Error.
0xD2	Daisy-Chain Error, to few devices on the NET-Bus available.
0xD3	Daisy-Chain Error, device with unexpected NET-ID found - BUS STOPPED.
0xD4	Daisy-Chain Error, device with unexpected NET-ID found, Net-ID changed - BUS STARTED.
0xE0	CAN-Error.
0xE1	CAN-Error, failed to open CAN-Port. Device stopped.
0xE2	CAN-Error, failed to set Baud-Rate. Device stopped.
0xE3	CAN-Error, not all COB-IDs cold be registered for reading
0xF0	Undefined Error…

Die Variable „byError" kann durch Setzen der Sende-Variable „xAcknowledgeActiveError" zurückgesetzt werden.

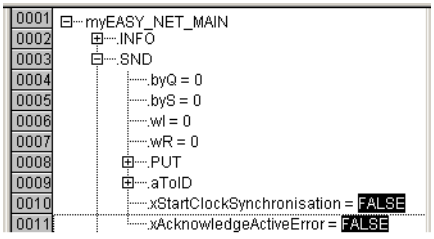


Abbildung 6: Zustandsanzeige der Struktur myEASY_NET_MAIN

„xAcknowledgeActiveError":

Ein Wechsel von FALSE auf TRUE setzt die Variable „byError" auf Null. Nach dem Rücksetzen wird die Variable vom System wieder auf FALSE gesetzt.

Struktur EASY_NET_DEVICE

Aufbau der Struktur

```
TYPE EASY_NET_DEVICE:
STRUCT
xDeviceAvailable: BOOL;
xProgram: BOOL;
xRUN: BOOL;
byDelay: BYTE;
END_STRUCT
END_TYP
```

Komponenten der Struktur

xDeviceAvailable:
TRUE = Zum Teilnehmer ist eine Kommunikation möglich.
FALSE = Zum Teilnehmer ist keine Kommunikation möglich.
xProgram: BOOL;
TRUE = Der Teilnehmer enthält ein Programm.
FALSE = Der Teilnehmer enthält kein Programm.
xRUN: BOOL;
TRUE = Der Teilnehmer ist im „RUN".
FALSE = Der Teilnehmer ist im „STOP".
byDelay: BYTE;
easy-Teilnehmer: Einstellung der Bus-Delay.
IEC-Teilnehmer: Grundsätzlich „0".

Struktur EASY_NET_SND

Mit Hilfe der EASY_NET_SND-Struktur werden die Daten vom Anwenderprogramm zur Protokolltask gesendet. In die Komponenten byQ, byS, wI und wR werden die Werte der lokalen Aus- und Eingänge inclusive der Aus- und Eingänge eines Erweiterungsmoduls eingetragen.

→ Die EC4-200-Steuerung wertet diese Komponenten nicht aus.

Aufbau der Struktur

```
TYPE EASY_NET_SEND:
STRUCT
byQ:BYTE;
byS:BYTE;
wI:WORD;
wR:WORD;
PUT:EASY_NET_PUT;
aToID:ARRAY[1..8] OF EASY_NET_SNDx;
xStartClockSynchronisation:BOOL;
xAcknowledgeActiveError:BOOL;
END_STRUCT
END_TYPE
```

Komponenten der Struktur

byQ:

Lokale Bit-Ausgänge Q1 bis Q8
(IEC-Schreibweise = byQ.0 bis byQ.7).
Die Zustände der Ausgänge werden von allen anderen Teilnehmern als aktuelle Ausgangswerte interpretiert.

byS:

Lokale Bit-Ausgänge S1 bis S8
(IEC-Schreibweise = byS.0 bis byS.7).
Der Wert wird aus dem Anwenderprogramm gesetzt.

wI:

Lokale Bit-Eingänge I1 bis I16
(IEC-Schreibweise = wI.0 bis wI.15).
Die Zustände der Eingänge werden von allen anderen Teilnehmern als aktuelle Eingangswerte interpretiert.

wR:

Lokale Bit-Eingänge R1 bis R16
(IEC-Schreibweise = wR.0 bis wR.15).
Der Wert wird aus dem Anwenderprogramm gesetzt.

PUT:

Mit Hilfe der PUT-Komponente vom Strukturtyp EASY_NET_PUT lassen sich 32 Datenblöcke von jeweils 4 Byte nacheinander versenden. Diese Datenblöcke können von einem anderen Teilnehmer mit Hilfe eines GET-Bausteins bzw. Funktion gelesen werden, → Abschnitt „IEC-Teilnehmer <-> mehrere IEC-/easy-Teilnehmer (Doppelwort-Datenaustausch)", Seite 22.

aToID:

Array aus 8 Elementen „aToID[1,…,8]". Jedes Element ist einem Teilnehmer (1,…,8) zugeordnet. Die Anwahl des Empfangsteilnehmers geschieht durch die Auswahl des Elements.
Tragen Sie in das Element die Daten ein, die Sie versenden möchten.
Ein Element ist vom Strukturtyp EASY_NET_SNDx.

xStartClockSynchronisation:

Ein Wechsel von FALSE auf TRUE startet die Synchronisierung der Teilnehmeruhren auf den Wert der lokal eingestellten Uhr.
Die Synchronisation erfolgt zur nächsten vollen Minute.
Nach erfolgreicher Ausführung wird die Variable „xStartClockSynchronisation" auf FALSE gesetzt.

xAcknowledgeActiveError:

Ein Wechsel von FALSE auf TRUE setzt die Variable „byError" auf Null. Nach erfolgtem Rücksetzen wird die Variable „xAcknowledgeActiveError" auf FALSE gesetzt, → Abschnitt „Struktur EASY_NET_INFO", Seite 9.

Struktur EASY_NET_SNDx

Aufbau der Struktur

```
TYPE EASY_NET_SNDx :
STRUCT
byQ:BYTE;
byS:BYTE;
dwSN:DWORD;
END_STRUCT
END_TYPE
```

Komponenten der Struktur

byQ/byS:

Der sendende Teilnehmer muss die Net-ID = 1 haben.
Der Empfangsteilnehmer muss vom Typ Remote I/O sein!

byQ:

Ausgangsbits Q1 bis Q8 des angewählten Teilnehmers (IEC-Schreibweise = byQ.0 bis byQ.7).

byS:

Ausgangsbits S1 bis S8 des Erweiterungsgerätes des angewählten Teilnehmers (IEC-Schreibweise = byS.0 bis byS.7).

dwSN:

32 Bit, die an den ausgewählten Teilnehmer gesendet werden, (IEC-Schreibweise = dwSN.0 bis dwSN.31), → Abschnitt „IEC-Teilnehmer <-> IEC-/easy-Teilnehmer (Bit-Information)", Seite 20.

Struktur EASY_NET_PUT

Mit Hilfe der Struktur lässt sich ein Datum/Wert (DWORD-Format) in das Netz stellen, der von einen anderen Teilnehmer mit der NET_GET-Funktion aus dem Netz geholt wird. Es kann immer nur ein Wert versendet werden.

→ Versenden Sie Werte mit Hilfe dieser Struktur, so sollten Sie nur eine Task benutzen.

Aufbau der Struktur

```
TYPE EASY_NET_PUT :
STRUCT
xStartTransfer: BOOL;
byModuleNumber: BYTE;dwData: DWORD;
xTransferPending: BOOL;
xTransferFailed: BOOL;
END_STRUCT
END_TYPE
```

Komponenten der Struktur

xStartTransfer:

Ein Wechsel von FALSE nach TRUE startet das Versenden der in die Komponente „dwData" eingetragenen Daten.

byModuleNumber:

Modulnummer der zu sendenden Daten (1 bis 32).

dwData:

Datum/Wert (DWORD-Format).

xTransferPending:

TRUE = Das Senden der Daten wird durchgeführt.
Es kann kein weiterer Sendeauftrag erfolgen.

FALSE = Der Sendebaustein ist frei.
Es kann ein neuer Sendeauftrag erfolgen.

xTransferFailed:

TRUE = Die Datenübertragung war fehlerhaft.

FALSE = Die Datenübertragung war ok.

→ Nimmt die Daten kein Teilnehmer entgegen (NET_GET), so wird an „xTransferFailed" kein Fehler angezeigt.

→ Abschnitt „IEC-Teilnehmer <-> mehrere IEC-/easy-Teilnehmer (Doppelwort-Datenaustausch)", Seite 22.

Tabelle 2: Gegenüberstellung der Parameter der Struktur EASY_NET_PUT (easySoft-CoDeSys) und den Ein-/ Ausgängen des PUT-Bausteins (easySoft)

Parameter	easySoft	easySoft-CoDeSys
Daten	I1	dwData
PUT-Nummer	PTxx	byModuleNumber
Strobe	T_	xStartTransfer
Aktiv	AC	xTransferPending
Gestartet	Q1	–
Fehler	E1	xTransferFailed

Datum/Wert ins Netz stellen

Nach dem Eintragen der Parameter „byModuleNumber" und „dwData" und einer steigenden Flanke an „xStartTransfer" wird das Datum oder der Wert ins Netz gestellt. Als Rückmeldung geht der Ausgang „xTransferPending" kurzzeitig auf TRUE. Danach kann ein neuer Wert abgesetzt werden.

Struktur EASY_NET_RCV

Mit Hilfe der EASY_NET_RCV-Struktur holt das Anwenderprogramm die Daten von der Protokolltask. Die Strukturkomponente „RCV" ist ein Array aus 8 Elementen RCV[1,…,8]. Die Elemente sind vom Strukturtyp „EASY_NET_RCV". Jedes Element ist einem Teilnehmer (1,…,8) zugeordnet, in das der Teilnehmer die Daten schreibt. Die Abfrage der empfangenen Daten geschieht durch das Auslesen eines Elements, z. B. „myEASY_NET_MAIN.RCV[2].byQ".

Ein Auffrischen der Daten findet mit jedem Aufruf der Net_Update Funktion statt.

Aufbau der Struktur

```
TYPE EASY_NET_RCV:
STRUCT
byQ:BYTE;
byS:BYTE;
wI:WORD;
wR:WORD;
dwRN:DWORD;
END_STRUCT
END_TYPE
```

Komponenten der Struktur

byQ:	
	Ausgangsbits Q1 bis Q8 des sendenden Teilnehmers (IEC-Schreibweise = byQ.0 bis byQ.7).
byS:	
	Ausgangsbits S1 bis S8 des sendenden Teilnehmers (Erweiterungsgerät) oder einer XC200 -Steuerung (IEC-Schreibweise = byS.0 bis byS.7).
wI:	
	Eingangsbits I1 bis I16 des sendenden Teilnehmers (IEC-Schreibweise = wI.0 bis wI.15).
wR:	
	Eingangsbits R1 bis R16 des sendenden Teilnehmers (Erweiterungsgerät) (IEC-Schreibweise = wR.0 bis wR.15).
dwRN:	
	32-Bit Daten des sendenden Teilnehmers (IEC-Schreibweise = dwRN.0 bis dwRN.31).

→ Abschnitt „IEC-Teilnehmer <-> IEC-/easy-Teilnehmer (Bit-Information)", Seite 20.

Struktur EASY_NET_GET

Aufbau der Struktur

```
TYPE EASY_NET_GET :
STRUCT
byNET_ID: BYTE;
byModuleNumber: BYTE;
xNewDataReceived: BOOL;
byReceiveCount: BYTE;
dwGet_Data: DWORD;
byError: BYTE;
END_STRUCT
END_TYPE
```

Komponenten der Struktur

Voraussetzung ist, dass Sie eine Variable, z. B. „myEasy_Net_Get" vom Strukturtyp EASY_NET_GET, parametriert haben. Die Komponenten der Struktur enthalten folgende Informationen.

byNET_ID:
Hier wird die Net-ID des Teilnehmers eingetragen, der das Datum ins Netz gestellt hat.
byModuleNumber:
Hier wird die Modulnummer des Datums eingetragen, das abzuholen ist.
xNewDataReceived:
TRUE: Es wurden neue Daten empfangen.
FALSE: Unter der angegebenen Adresse liegen keine neuen Daten. Es wurde der letzte eingetragene Stand übermittelt.
byReceiveCount:
Diese Variable enthält die Anzahl der Telegramme, die empfangen wurden (Zyklischer Zähler, der von 16#FF auf 16#00 zurückspringt).
dwGET_Data:
Hier liegen die empfangenen Daten in einem Doppelwort.
byError:
0: Daten erfolgreich ausgelesen.
1: Fehler beim Auslesen der Daten.

Tabelle 3: Gegenüberstellung der Parameter der NET_GET-Funktion (easySoft-CoDeSys) und den Ein-/Ausgängen des GET-Bausteins (easySoft)

Parameter	easySoft	easySoft-CoDeSys
Bausteinnummer	GTxx	–
Empfangsdaten	Qv	dwGet_Data
Modul Nummer	PTxx	byModuleNumber
Net-ID des Sendeteilnehmers	NET_ID	byNET_ID
Neuer Wert liegt an	Q1	xNewDataReceived
Empfangszähler	–	byReceiveCount
Fehler	–	byError

4 Anwendungen

Sie haben mehrere Möglichkeiten Daten zu senden und zu empfangen. Die Datenübertragung ist abhängig von der Art der Teilnehmer (IEC- oder easy-Teilnehmer) und wird im Folgenden beschrieben. In diesem Zusammenhang werden in einer Strukturübersicht auch die Komponenten der Strukturen aufgeführt, die in Abhängigkeit der zu übertragenden Datenart zu parametrieren sind. Es wird der gesamte Strukturbaum aufgezeigt, so wie er im online-Betrieb in der Zustandsanzeige dargestellt wird.

IEC-Teilnehmer <- IEC-/easy-Teilnehmer (I/O-Information)

Ein IEC-Teilnehmer fragt die Ein-/Ausgänge (Bit-Daten) von IEC- oder easy-Teilnehmern ab, z. B. EC4-200/easy800. Die Datenübertragung erfolgt zyklisch.

Die Teilnehmer EC4-200 und easy800 (auch als Remote I/O) mit ihren Erweiterungen EASY202-RE oder EASY6...-RE/-TE stellen die Bit-Daten ihrer lokalen Ein-/Ausgänge zur Verfügung. Sie können von jedem Teilnehmer mit Anwenderprogramm, auch XC200, mit der Funktion „Net_Update" über die Struktur „myEASY_NET_MAIN.RCV" abgefragt werden.

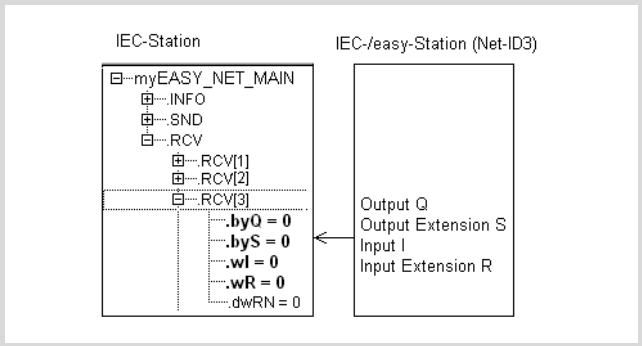
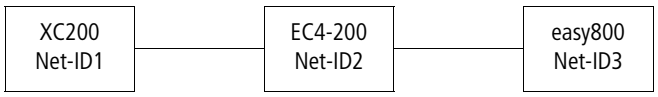


Abbildung 7: Online-Darstellung: IEC-Teilnehmer (Net-ID1) fragt die Ein-/Ausgänge vom Gerät (Net-ID3) ab.

Beispiel



Die XC200 fragt:

1. den lokalen Eingang 1 der EC4-200 (Net-ID2) ab,
2. den lokalen Eingang 3 der easy800 (Net-ID3).

Dazu ist folgendes Programm erforderlich:

```
VAR
myEASY_NET_MAIN: EASY_NET_MAIN;
x1NetID2 :BOOL;
x3NetID3 :BOOL;
END_VAR

Net_Update(byNetDrvNr:=0,
pstruNetMain:=ADR(myEASY_NET_MAIN));
(* read input *)
X1NetID2:=myEASY_NET_MAIN.RCV[2].wI.0;
(*input1 von EC4-200*)
X3NetID3:=myEASY_NET_MAIN.RCV[3].wI.2;
(*input3 von easy800*)
```

Sind in der easy 800 (Net-ID3) die Eingänge 3, 5, 8, 9 = TRUE und die Ausgänge 1, 3 = TRUE, so zeigt die Zustandsanzeige der XC200 (Net-ID1) folgende Struktur an:

Struktur „myEASY_NET_MAIN"

INFO	
SND	
RCV	
RCV[3]	E/A der easy800 (Net-ID3)
.byQ=16#05	Die Ausgänge 1 und 3 sind TRUE.
.byS=16#00	Die Ausgänge der Erweiterung (nicht vorhanden).
.wI=16#2194	Die Eingänge 3, 5, 8 und 9 sind TRUE.
.wR=16#0000	Die Eingänge der Erweiterung (nicht vorhanden).

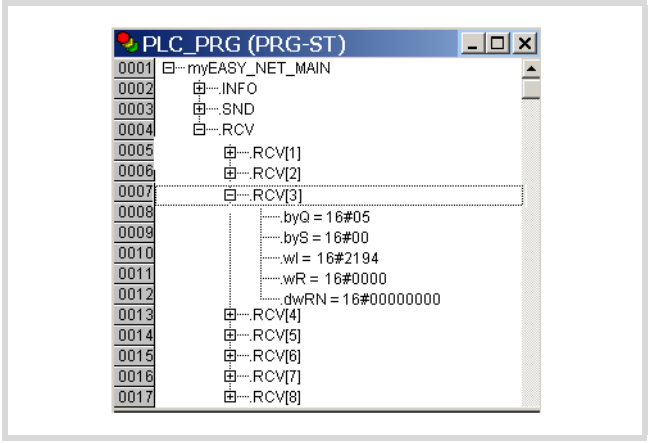


Abbildung 8: Online-Zustandsanzeige: IEC-Teilnehmer (Net-ID1) fragt die Ein-/Ausgänge vom Gerät (Net-ID3) ab

IEC-Teilnehmer (Net-ID1) -> Remote I/O-Teilnehmer, z. B. easy800 (I/O-Information)

Ein IEC-Teilnehmer setzt die Ausgänge eines Remote I/O-Teilnehmers. Die Datenübertragung erfolgt zyklisch.

→ Es kann nur der Teilnehmer mit der Net-ID1 die Ausgänge eines Remote I/O-Teilnehmer, z. B. easy800, setzen.

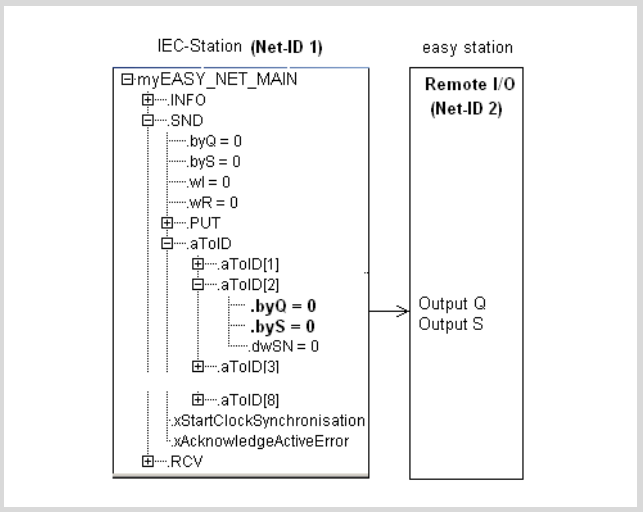
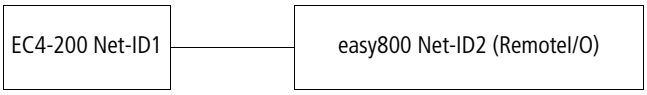


Abbildung 9: Online-Darstellung: Ausgänge von Remote I/O-Gerät setzen

Beispiel
Der IEC-Teilnehmer (Net-ID1) setzt den Ausgang 3 von der easy800 (Net-ID2).



Dazu ist folgendes Programm erforderlich:

```
VAR
myEASY_NET_MAIN: EASY_NET_MAIN;
END_VAR

Net_Update(byNetDrvNr:=0,
pstruNetMain:= ADR(myEASY_NET_MAIN));
(* write Output *)
myEASY_NET_MAIN.SND.aToID[2].byQ.2:=TRUE;
```

In der Zustandsanzeige des Gerätes mit der Net-ID1 wird für das Gerät (Net-ID2) folgende Struktur angezeigt:

Struktur „myEASY_NET_MAIN“

INFO	
SND	
–	
aToID[1]	
aToID[2]	Ausgänge der easy800.
.byQ=16#04	Der Ausgang 3 (TRUE) wird an Net-ID2 gesendet.
.byS=16#00	Die Ausgänge der Erweiterung (nicht vorhanden).
.dwSN= 16#00000000	Keine Funktion.

IEC-Teilnehmer (Net-IDx) <-> EASY223-SWIRE-Gateway (I/O-Information)

Die Datenübertragung zwischen einem IEC-Teilnehmer und dem Gateway EASY223-SWIRE über das easyNet erfolgt mit den gleichen Funktionen wie zwischen IEC-Teilnehmer und easy800.

An das Gateway können 16 Module angeschlossen werden. Ein Modul kann bis zu 4 Eingangssignale (SI) und ein Diagnosesignal (SD) an das Gateway senden. Die Signale können von mehreren Teilnehmern abgefragt werden. Umgekehrt kann ein ausgewählter Teilnehmer zwei Signale (SQ) über das Gateway an ein Modul senden.

Aus Abbildung 10 gehen die Signale zwischen IEC-Teilnehmer/ EASY223-SWIRE und zwei SmartWire-Modulen hervor. Die Bedeutung der Signale geht aus den Beschreibungen der Module hervor.

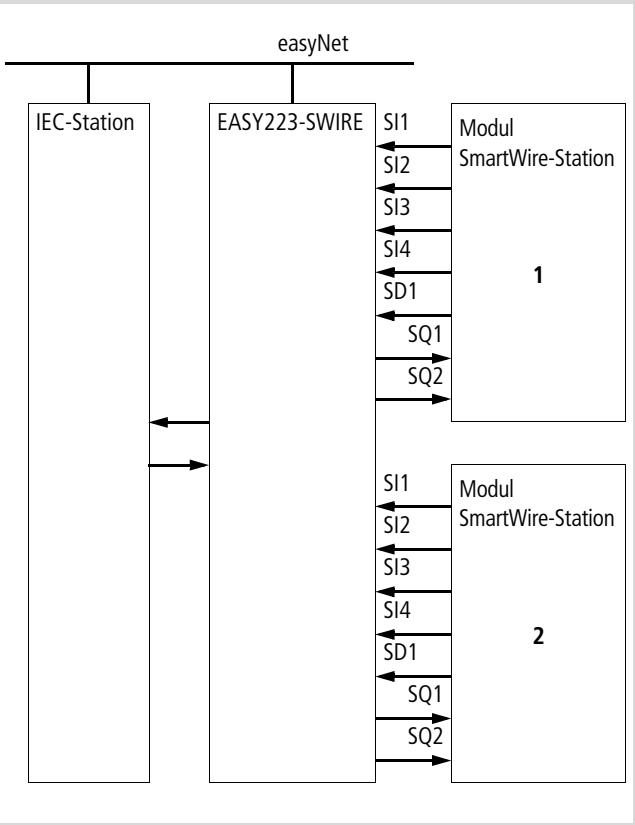


Abbildung 10: Datenübertragung
IEC <-> EASY223-SWIRE <-> Module

SI = Input signal
SQ = Output signal
SD = Diagnose signal

Beispiel

easy-Net-Verbindung zwischen IEC-Teilnehmer und EASY223-SWIRE.

IEC station Net-ID1	easyNet	EASY223-SWIRE (Gateway) Net-ID2	1	2	3	…	16
			SmartWire-Module				

IEC-Teilnehmer (Net-IDx) <- EASY223-SWIRE-Gateway (I/O-Information)

Ein IEC-Teilnehmer fragt die Eingangs-/Diagnose-Daten der Module ab. Die Datenübertragung erfolgt zyklisch.
Die Eingänge/Diagnoseeingänge der Module können von jedem Teilnehmer abgefragt werden, „→ Abschnitt „IEC-Teilnehmer <- IEC-/easy-Teilnehmer (I/O-Information)".

Aus der Struktur „myEASY_NET_MAIN.RCV[2]" in der Abbildung 11 geht die Zuordnung zwischen den Operanden des IEC-Teilnehmers und den Ausgängen eines SmartWire-Moduls hervor. Das Gateway hat in diesem Beispiel die Net-ID 2.

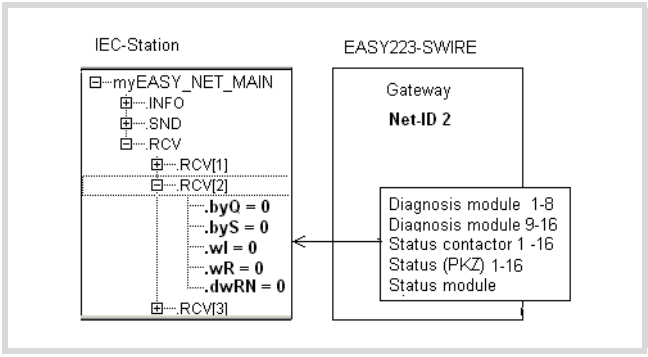


Abbildung 11: Abfrage der Modulsignale von EASY223-SWIRE (Net-ID2)

Signalabfrage des EASY223-SWIRE

Modulnr. 1

Signal	EASY223-SWIRE Signalbedeutung Beispiel SWIRE-DIL	IEC-Teilnehmer Operand RCV[x]
SI 1	Spulenstatus	.wI.0
SI 2	PKZ	.wR.0
SI 3	Keine Funktion	.dwRN.0
SI 4	Keine Funktion	.dwRN.16
SD 1	Diagnose	.byQ.0

Modulnr. 2

Signal	EASY223-SWIRE Signalbedeutung Beispiel SWIRE-DIL	IEC-Teilnehmer Operand RCV[x]
SI 1	Spulenstatus	.wI.1
SI 2	PKZ	.wR.1
SI 3	Keine Funktion	.dwRN.1
SI 4	Keine Funktion	.dwRN.17
SD 1	Diagnose	.byQ.1

usw. bis

Modulnr. 8

Signal	EASY223-SWIRE Signalbedeutung Beispiel SWIRE-DIL	IEC-Teilnehmer Operand RCV[x]
SI 1	Spulenstatus	.wI.7
SI 2	PKZ	.wR.7
SI 3	Keine Funktion	.dwRN.7
SI 4	Keine Funktion	.dwRN.23
SD 1	Diagnose	.byQ.7

Modulnr. 9

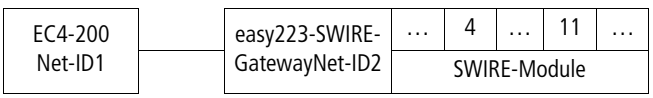
Signal	EASY223-SWIRE Signalbedeutung Beispiel SWIRE-DIL	IEC-Teilnehmer Operand RCV[x]
SI 1	Spulenstatus	.wI.8
SI 2	PKZ	.wR.8
SI 3	Keine Funktion	.dwRN8
SI 4	Keine Funktion	.dwRN.24
SD 1	Diagnose	.byS.0

usw. bis

Modulnr. 16

Signal	EASY223-SWIRE Signalbedeutung Beispiel SWIRE-DIL	IEC-Teilnehmer Operand RCV[x]
SI 1	Spulenstatus	.wI.15
SI 2	PKZ	.wR.15
SI 3	keine Funktion	.dwRN.15
SI 4	keine Funktion	.dwRN.31
SD 1	Diagnose	.byS.7

Beispiel
Die EC4-200 (Net-ID1) liest den Eingang I1 vom Modul 4 und I1 vom Modul 11.



Dazu ist folgendes Programm erforderlich:

```
VAR
myEASY_NET_MAIN: EASY_NET_MAIN;
X3NetID2
X1ONetID2
END_VAR

Net_Update(
byNetDrvNr=0,
pstruNetMain = ADR(myEASY_NET_MAIN));
(* read input *)
X3NetID2:= myEASY_NET_MAIN.RCV[2].wI.3;
X1ONetID2:=myEASY_NET_MAIN.RCV[2].wI.10;
```

IEC-Teilnehmer (Net-IDn) -> EASY223-SWIRE-Gateway (I/O-Information)

Ein IEC-Teilnehmer sendet Ausgangssignale an die Module (→ Beispiel, Seite 19). Die Datenübertragung erfolgt zyklisch.

Ist ein SmartWire-Gateway im easyNet-Netzwerk eingebunden, so legen Sie einen Teilnehmer (n) in der easySoft fest, der die Ausgänge der Module setzt, die am Gateway angeschlossen sind.

→ Es kann nur ein ausgewählter Teilnehmer die Ausgänge an einem oder mehreren SmartWire-Gateways setzen.

Aus der Struktur in der Zustandsanzeige eines IEC-Teilnehmers geht die Zuordnung zwischen dem Operand eines IEC-Teilnehmers und dem Ausgang eines SmartWire-Moduls hervor.

Struktur„myEASY_NET_MAIN"

INFO	
SND	
–	
aToID[1]	
aToID[2]	Ausgänge der Module des EASY223-SWIRE
.byQ=16#00	
.byS=16#00	
.dwSN= 16#00000000	dwSN.0 bis dwSN.15 = Ausgang Modul 1 bis 16 (z. B. Ansteuern der Schützspule)

Signale senden an EASY223-SWIRE
Modulnr. 1

Signal	EASY223-SWIRE Signalbedeutung Beispiel SWIRE-DIL	IEC-Teilnehmer Operand aToID[x]
SQ 1	Spule	.dwSN.0
SQ 2	Keine Funktion	.dwSN.16

Modulnr. 2

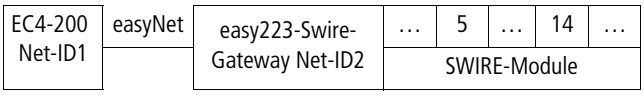
Signal	EASY223-SWIRE Signalbedeutung Beispiel SWIRE-DIL	IEC-Teilnehmer Operand aToID[x]
SQ 1	Spule	.dwSN.1
SQ 2	Keine Funktion	.dwSN.17

usw. bis

Modulnr.15

Signal	EASY223-SWIRE Signalbedeutung Beispiel SWIRE-DIL	IEC-Teilnehmer Operand aToID[x]
SQ 1	Spule	.dwSN.15
SQ 2	Keine Funktion	.dwSN.31

Beispiel
Die EC4-200 (Net-ID1) setzt den Ausgang „SQ1" vom Modul 5 und 14.



Dazu ist folgendes Programm erforderlich:

```
VAR
myEASY_NET_MAIN: EASY_NET_MAIN;
END_VAR

Net_Update(
byNetDrvNr:=0 ,
pstruNetMain:= ADR(myEASY_NET_MAIN));
(* write Output *)
myEASY_NET_MAIN.SND.aToID[2].dwSN.4:=TRUE;
myEASY_NET_MAIN.SND.aToID[2].dwSN.13:=TRUE;
```

IEC-Teilnehmer <-> IEC-/easy-Teilnehmer (Bit-Information)

Jeder Teilnehmer kann einem anderen Teilnehmer eine einzelne Bit-Information oder einen Datenblock mit bis zu 32 Bit-Informationen senden (Operand SN). Im Empfänger sind diese Bits abzufragen (Operand RN). Die Bits, auch als Netzwerkmerker bezeichnet, sind in einem 32 Bit-Datenblock zusammengefasst.

Jeder Teilnehmer hat zur Kommunikation mit den sieben weiteren Netzteilnehmern jeweils einen Sende- und einen Empfangsdatenblock (→ Abbildung 12). Die Sende- und Empfangsdatenblöcke sind in eindimensionalen Arrays, bestehend aus 8 Elementen, zusammengefasst. Zwischen dem Sendedatenblock eines Teilnehmers und dem Empfangsdatenblock eines anderen Teilnehmers besteht eine direkte Zuordnung. Dabei entspricht die Position des Datenblocks im Sendearray, die z. B. bei einem IEC-Teilnehmer im Stukturelement aToID angegeben wird, der Net-ID des Empfangsteilnehmers.

Die Position des Datenblocks im Empfangsarray, die z. B. bei einem IEC-Teilnehmer in der Stukturkomponente RCV angegeben wird, entspricht der Net-ID des Sendeteilnehmers. Der Datenaustausch auf dem Bus erfolgt zyklisch.

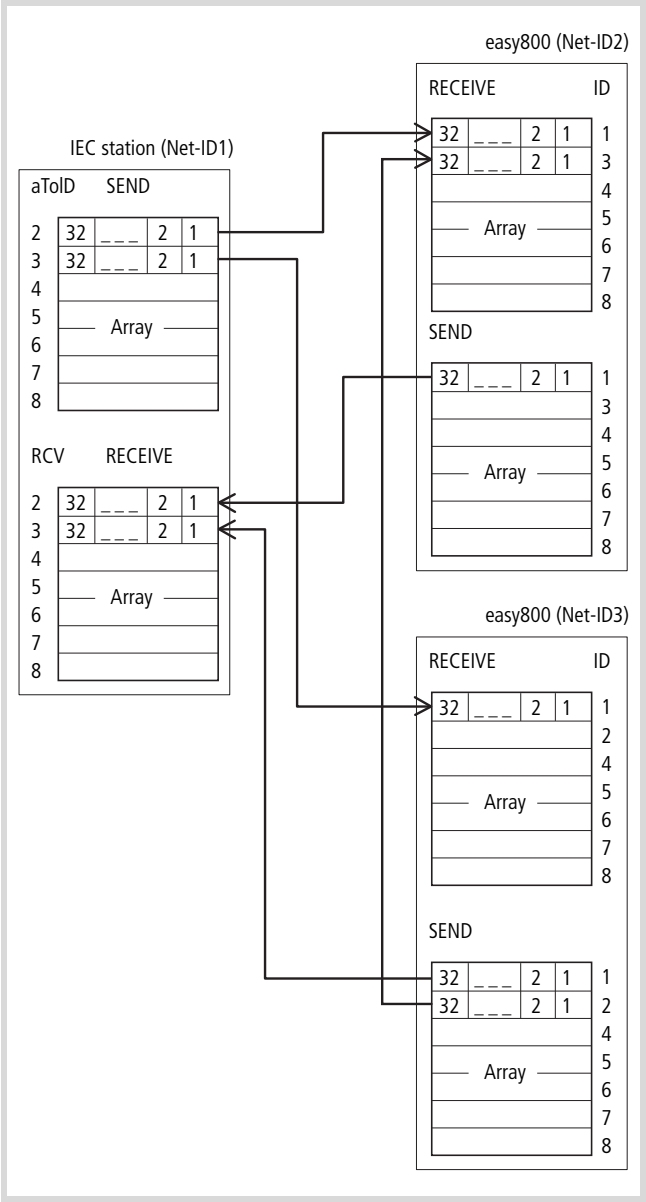


Abbildung 12: Übersicht der Datenblöcke

Beispiel 1
Übertragung IEC-Teilnehmer (Net-ID1) -> easy-Teilnehmer (Net-ID2).

Es soll vom IEC-Teilnehmer (Net-ID1) der Netzwerkmerker 1 (dwSN.0) oder ein 32Bit-Wert zum easy- Teilnehmer Net-ID2 übertragen werden.

Der Zugriff auf die Netzwerk-Merker erfolgt mit Hilfe der Datenstruktur „myEASY_NET_MAIN“ mit der Komponente „.SND.aToID[x].dwSN“ (Doppelwort)
(IEC-Schreibweise = dwSN.0 bis dwSN.31).

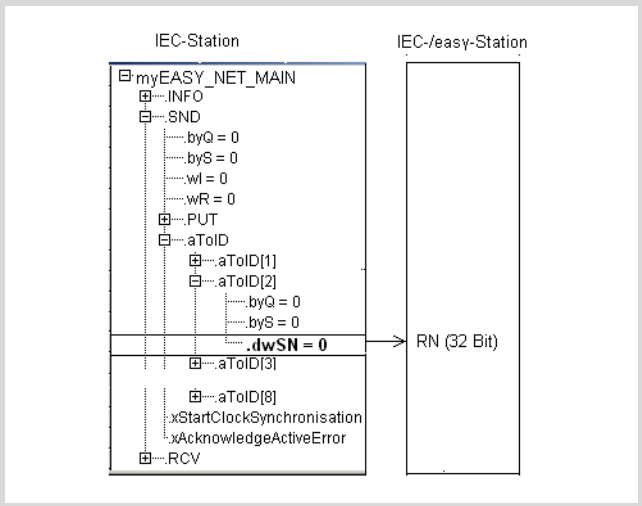


Abbildung 13: Strukturübersicht 32 Bit senden

Für den IEC-Teilnehmer Net-ID1 ist folgendes Programm zu erstellen:

```
VAR
myEASY_NET_MAIN: EASY_NET_MAIN;
END_VAR

Net_Update(
byNetDrvNr:=0 ,
pstruNetMain :=ADR(myEASY_NET_MAIN));
(* set dwSN.0 = TRUE*)
myEASY_NET_MAIN.SND.aTo[2].dwSN.0:=TRUE;
oder
(* set value = 5 *)
myEASY_NET_MAIN.SND.aToID[2].dwSN:=5;
```

Im easy-Teilnehmer (Net-ID2) kann der einzelne Netzwerkmerker 1 verarbeitet werden, indem der Netzwerkmerker 1RN1 abgefragt wird. Oder der Wert kann abgefragt werden, indem die Netzwerkmerker 1RN1 bis 1RN32 abgefragt werden.

Beispiel 2
Übertragung IEC-Teilnehmer (Net-ID1) <- easy-Teilnehmer (Net-ID3).

Es soll vom easy-Teilnehmer (Net-ID3) der Wert = 7 (3hex) an den IEC-Teilnehmer (Net-ID1) übertragen werden. Im easy-Teilnehmer sind die Netzwerkmerker 1SN1,1SN2,1SN3 auf TRUE zu setzen. Im IEC-Teilnehmer (Net-ID1) ist die Strukturkomponente „RCV[x].dwRN" (Doppelwort) zu programmieren (IEC-Schreibweise = dwRN.0 bis dwRN.31).

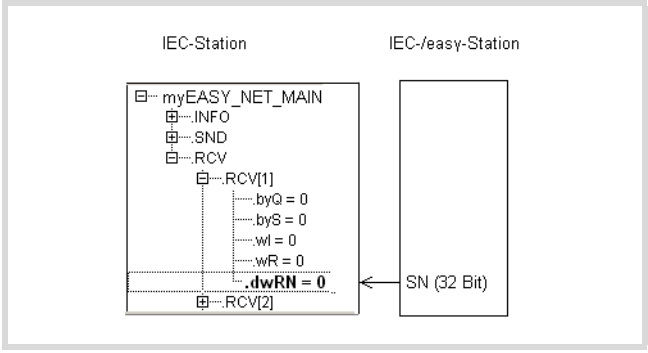


Abbildung 14: Strukturübersicht 32 Bit empfangen

Zum Abfragen des Wertes im IEC-Teilnehmer Net-ID1 ist folgendes Programm zu erstellen:

```
VAR
myEASY_NET_MAIN: EASY_NET_MAIN;
Inpval: DWORD;
END_VAR

Net_Update(byNetDrvNr:=0,
pstruNetMain := ADR(myEASY_NET_MAIN));
(* Demand value *)
Inpval := myEASY_NET_MAIN.RCV[3].dwRN;
```

IEC-Teilnehmer <-> mehrere IEC-/easy-Teilnehmer (Doppelwort-Datenaustausch)

Er erfolgt nach dem PUT-GET-Prinzip mit der Bedeutung:

- PUT = Datum ins Netz stellen,
- GET = Datum aus dem Netz holen.

Nach diesem Prinzip stellt ein IEC-Teilnehmer ein Datum (D-Wort) in das Netz, das durch eine Modul-Nummer (MN-Nummer) gekennzeichnet ist. Dazu benutzt er das Strukturelement PUT (PUT-Funktion), das Bestandteil der Funktion NET_UPDATE ist.

Easy-Teilnehmer benutzen dazu den PUT-Funktionsbaustein.

Das Abholen der Daten ist vom Teilnehmertyp abhängig:

- IEC-Teilnehmer holen das Datum mit der Funktion NET_GET ab,
- easy-Teilnehmer benutzen einen GET-Funktionsbaustein.

Zur Auswahl des Datums muss die Funktion NET_GET oder der Baustein die Angabe der Net-ID des Sendeteilnehmers und die Modulnummer des Datums enthalten.

Die Abbildung 15 zeigt den Zugriff auf die Daten im Netz mit Hilfe der PUT- und NET_GET-Funktionen.

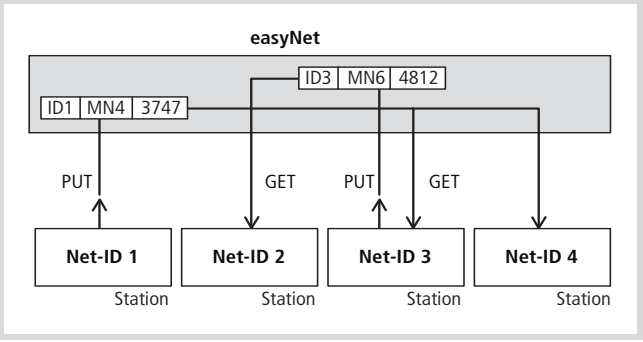


Abbildung 15: Beispiel mit PUT- und NET_GET-Funktionen

Zwei der Teilnehmer stellen mit Hilfe der PUT-Funktion jeweils ein Datum, gekennzeichnet durch die Net-ID-Nummer des Senders und die Modulnummer (MN) ins Netz. Net-ID1 den Wert 3747, Modulnummer 4, und Net-ID3 den Wert 4812, Modulnummer 6.

Die Teilnehmer (Net-ID3) und (Net-ID4) holen das Datum 3747 mit der NET_GET-Funktion ab, der Teilnehmer (Net-ID2) das Datum 4812.

Die folgenden Tabellen zeigen die Funktionen und Funktionsbausteine zum Senden und Abholen der Daten.

IEC-Teilnehmer (Sender)	Netzwerk	IEC-Teilnehmer (Abholer)
Funktion: NET_UPDATE (PUT)	Daten	Funktion: NET_GET

IEC-Teilnehmer (Sender)	Netzwerk	easy-Teilnehmer (Abholer)
Funktion: NET_UPDATE (PUT)	Daten	Funktionsbaustein: GET

easy-Teilnehmer (Sender)	Netzwerk	IEC-Teilnehmer (Abholer)
Funktionsbaustein: PUT	Daten	Funktion: NET_GET

PUT-Funktion (Strukturelement PUT)

Jeder IEC-Teilnehmer kann mit Hilfe der PUT-Funktion sequentiell Daten, gekennzeichnet durch die MN/PT-Nummern 1, 2, … ,32 nacheinander in das Netz stellen. Die Daten werden über eine „Datenschleuse" ins Netz gestellt, die pro Zyklus ein Datum absetzt.

Zur Ausführung der PUT-Funktion rufen Sie die Funktion NET_UPDATE auf.

Beispiel
Zum Aufrufen der PUT-Funktion ist folgendes Programm zu erstellen:

```
PROGRAM PLC_PRG
VAR
myEASY_NET_MAIN: EASY_NET_MAIN;
END_VAR

Net_Update(
byNetDrvNr:= 0 ,pstruNetMain:= ADR(myEASY_NET_MAIN));
```

Strukturübersicht EASY_NET_MAIN mit PUT
PUT, vom Typ Easy_Net_PUT, hat folgende Struktur:

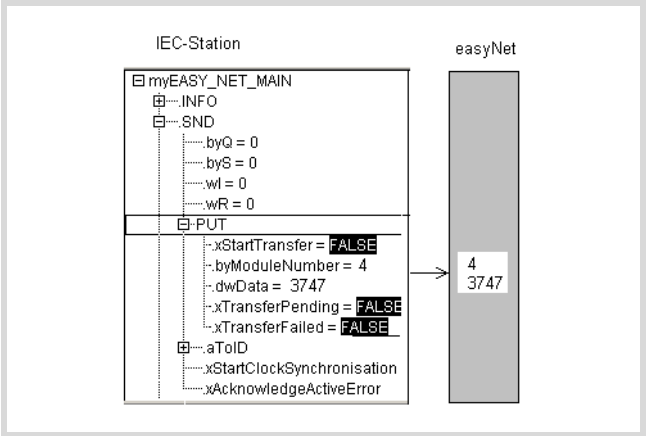


Abbildung 16: Struktur von EASY_NET_PUT

Die PUT-Funktion durchführen

Mit der folgenden Programmsequenz kann ein IEC-Teilnehmer eine PUT-Funktion durchführen.

PLC_PRG (PRG-ST)

```
0001 PROGRAM PLC_PRG   (* DWORD -> easyNet *)
0002 VAR
0003     myEASY_NET_MAIN  :EASY_NET_MAIN;
0004     xNewPUT          :BOOL:= FALSE;
0005     Input AT%IX0.0   :BOOL;
0006 END_VAR
```

```
0001 Net_Update(byNetDrvNr:=0,
0002 pstruNetMain:=ADR(myEASY_NET_MAIN));
0003
0004 IF NOT myEASY_NET_MAIN.SND.PUT.xTransferPending AND NOT
0005         myEASY_NET_MAIN.SND.PUT.xStartTransfer THEN
0006     IF Input THEN
0007     myEASY_NET_MAIN.SND.PUT.byModuleNumber:=4;
0008     myEASY_NET_MAIN.SND.PUT.dwData:=3747;
0009     xNewPUT:=TRUE;
0010     END_IF
0011 END_IF
0012 IF xNewPUT THEN
0013     xNEWPUT:=FALSE;
0014     myEASY_NET_MAIN.SND.PUT.xStartTransfer:=TRUE;
0015     END_IF
0016 IF myEASY_NET_MAIN.SND.PUT.xTransferPending THEN
0017     myEASY_NET_MAIN.SND.PUT.xStartTransfer:=FALSE;
0018 END_IF
```

Abbildung 17: Programmbeispiel für die PUT-Funktion

Nach dem Eintragen der Parameter „byModuleNumber“ = 4 und „dwData“ = 3747 und einer steigenden Flanke an „xStart-Transfer“ wird das Datum auf das Netz gelegt. Als Rückmeldung geht der Ausgang „.xTransferPending“ kurzzeitig auf TRUE, danach kann eine neue PUT-Funktion abgesetzt werden.

→ Bei „.xTransferFailed“ wird kein Fehler angezeigt, wenn kein Teilnehmer die Daten mit der Funktion NET_GET entgegennimmt.

Beispiel
Um Daten nacheinander ins Netz zu stellen, ist folgendes Programm zu erstellen:

```
PROGRAM PLC_PRG (*Programm für XC200*)
(*Die XC200 sendet 3 Datentelegramme ins Netz*)
VAR
     sEasyNetMain           :EASY_NET_MAIN;
     bNetUpdate             :BOOL;
     byState                :BYTE;
     xNewPUTMsgToTransmitt  :BOOL := FALSE;
END_VAR
VAR_INPUT
     dwzahl: DWORD;
END_VAR

(*Programm*)
dwzahl:=dwzahl+1;

bNetUpdate:=Net_Update(0, ADR(sEasyNetMain));
IF NOT sEasyNetMain.SND.PUT.xTransferPending AND NOT
                          sEasyNetMain.SND.PUT.xStartTransfer THEN
     (*Ein neues Daten-Telegramm kann übertragen werden.*)
     IF sEasyNetMain.SND.PUT.xTransferFailed THEN
                          ; (* Fehler bei der Übertragung des letzten Daten-Telegramms. *)
     END_IF
     (* In der Case-Anweisung werden die Werte (Adressierung und Daten) des nächsten zu versendenden Daten-Telegramms
     eingetragen.*)
     CASE (byState) OF
     0:                   (* Application Data Telegram 1*)
                            sEasyNetMain.SND.PUT.byModuleNumber := 1;
                            sEasyNetMain.SND.PUT.dwData := dwzahl;
                            xNewPUTMsgToTransmitt := TRUE; (* ready for send*)
                            byState:=1;
     1:                   (* Application Data Telegram 2 *)
                            sEasyNetMain.SND.PUT.byModuleNumber := 2;
                            sEasyNetMain.SND.PUT.dwData := 22;              xNewPUTMsgToTransmitt := TRUE; (* ready for send*)
                            byState:=2;
     2:                   (* Application Data Telegram 3*)
                            sEasyNetMain.SND.PUT.byModuleNumber := 3;
                            sEasyNetMain.SND.PUT.dwData := 33;              xNewPUTMsgToTransmitt := TRUE; (* ready for send*)
                            byState:=0;
     END_CASE
END_IF
IF xNewPUTMsgToTransmitt THEN
     (* Starten des Sendevorgangs durch Setzen des Start-Flags.*)
     xNewPUTMsgToTransmitt := FALSE;
     sEasyNetMain.SND.PUT.xStartTransfer := TRUE;
END_IF
IF sEasyNetMain.SND.PUT.xTransferPending THEN
     (* Die Datenübertragung ist aktiv, Start-Flag kann zurückgesetzt werden.*)
     sEasyNetMain.SND.PUT.xStartTransfer := FALSE;
END_IF
```

Die NET_GET-Funktion

Mit dem Aufruf der NET_GET-Funktion können Daten aus dem Netzwerk gelesen (abgeholt) werden, die von anderen Teilnehmern mit der PUT-Funktion ins Netz gestellt wurden. Mit einem Funktionsaufruf kann nur ein Datum „abgeholt" werden.

Zur Ausführung der NET_GET-Funktion rufen Sie die Funktion NET_GET auf.

Beispiel
Zum Aufrufen der NET_GET-Funktion ist folgendes Programm zu erstellen:

```
PROGRAM PLC_PRG
VAR

myEasyNetGet : EASY_NET_GET;
END_VAR
NET_GET(0,ADR(myEasyNetGet));
```

Die Struktur von EASY_NET_GET geht aus der folgenden Online-Darstellung hervor:

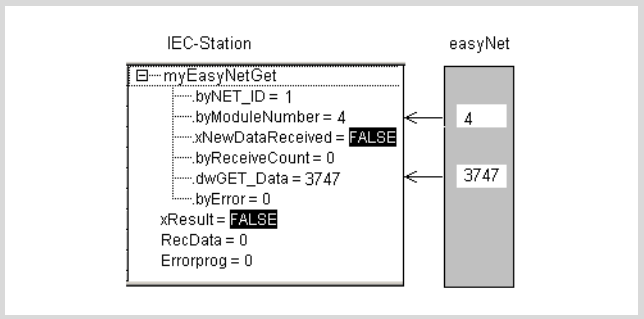


Abbildung 18: Struktur von EASY_NET_GET

Die NET_GET-Funktion durchführen

Mit der folgenden Programmsequenz kann ein IEC-Teilnehmer eine NET_GET-Funktion durchführen.

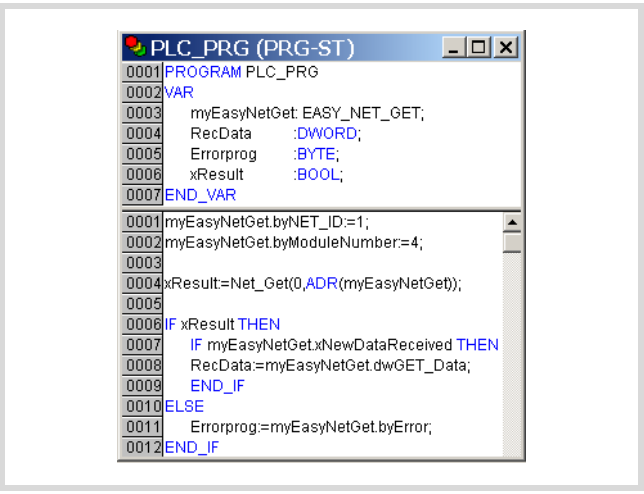


Abbildung 19: Programmbeispiel für die NET-GET-Funktion

Nach dem Parametrieren der Komponenten „byNET_ID" = 1 und „byModuleNumber" = 4 erfolgt ein zyklischer Aufruf der Funktion NET_GET. Die Variable „xNewDataReceived" ist ständig abzufragen. Hat die Variable den Zustand TRUE, so wurde ein neues Datum empfangen. Es steht an der Komponente „dwData" an.

Beispiel
Abholen mehrerer Module vom Netz.

Mit dem folgenden Programm einer EC4-200, Net-ID2, werden die Module 1, 2 und 3 vom Netz geholt, die vom Teilnehmer mit der Net-ID1 ins Netz gestellt wurden.

```
PROGRAM PLC_PRG
VAR
myEasyNetGet: EASY_NET_GET;
bNetUpdate: BOOL;
byState: BYTE;
xSuccess: BOOL;
RecData: DWORD;
xReceive: BOOL;
Errorflag: BYTE;
RecData1: DWORD;
RecData2: DWORD;
RecData3: DWORD;
END_VAR

IF xReceive = FALSE THEN
CASE (byState) OF
0:
    myEasyNetGet.byModuleNumber := 1;
    myEasyNetGet.byNet_ID := 1;
    xReceive := TRUE;
    byState:=1;
1:
    myEasyNetGet.byModuleNumber := 2;
    myEasyNetGet.byNet_ID := 1;
    xReceive := TRUE;
    byState:=2;
2:
    myEasyNetGet.byModuleNumber := 3;
    myEasyNetGet.byNet_ID := 1;
    xReceive := TRUE;
    byState:=0;
END_CASE
END_IF
(* GET-Daten...*)
xSuccess:=Net_Get(0, ADR(myEasyNetGet));
IF xSuccess AND xReceive THEN
  IF myEasyNetGet.xNewDataReceived THEN
    IF myEasyNetGet.xNewDataReceived THEN
    IF byState = 1 THEN
    RecData1:=myEasyNetGet.dwGet_Data;
    END_IF
    IF byState = 2 THEN
    RecData2:=myEasyNetGet.dwGet_Data;
    END_IF
    IF byState = 0 THEN
    RecData3:=myEasyNetGet.dwGet_Data;
    END_IF
  xReceive:=FALSE;
  END_IF
  ELSE
  Errorflag:=myEASYNetGet.byError;
END_IF
```

Synchronisation der Echtzeituhr

Die Echtzeituhren aller easyNet Teilnehmer können auf die aktuelle Zeit der lokalen Echtzeituhr gestellt werden. Diese Synchronisation kann auf einfachste Weise durchgeführt werden.

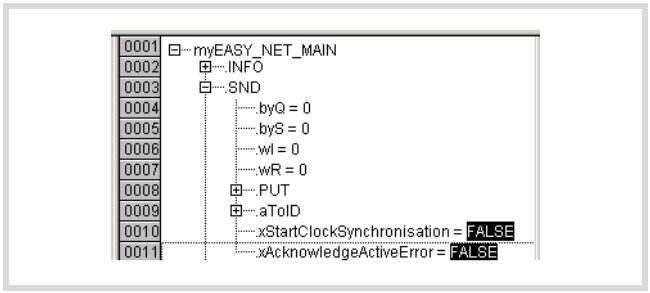


Abbildung 20: Datenstruktur zur Synchronisation der Echtzeituhr

In SND-Bereich der Datenstruktur „myEASY_NET_MAIN" gibt es die Variable „xStartClockSynchronisation". Ein Wechsel von FALSE auf TRUE startet die Synchronisierung der Uhren auf den Wert der lokal eingestellten Uhr. Der Versand erfolgt zur nächsten vollen Minute. Nach erfolgreicher Ausführung der Synchronisierung wird die Variable „xStartClockSynchronisation" automatisch auf FALSE zurückgesetzt.

5 Netzwerk (easyNet) konfigurieren

Mit Hilfe des easy-Teilnehmers (Net-ID = 1) können Sie alle weitere Teilnehmer am Netz, wie easy800, MFD-CP8-… und EC4P, konfigurieren. Die mit der Net-ID 2, … ,8 adressierten Teilnehmer erhalten die Kommunikationsparameter wie Net-ID, Baudrate, Bus-Delay und Send IO. Der Teilnehmer mit der Net-ID1 kann eine EC4-200 oder ein easy-Teilnehmer wie easy800 oder MFD-CP8-… sein.

Voraussetzungen zur Konfiguration sind:

- Es darf keine XC200 im Netzwerk sein.
  Enthält das Netzwerk eine XC200, so ist jeder Teilnehmer separat zu konfigurieren!
- Der Netzwerk-Strang darf keine Stichleitungen haben.
- Die Teilnehmer sind für die Netzwerkkonfiguration befähigt.
- Soll ein IEC-Teilnehmer über das Netzwerk konfiguriert werden, so muss er ein Projekt beinhalten, in dem die easyNet-Funktion aktiviert ist.
- Der durchführende Teilnehmer muss die Net-ID = 1 haben und sich am Anfang des Stranges befinden (Position 1).

Konfiguration durchführen

Die Konfiguration erfolgt in zwei Schritten:

1. Konfiguration erstellen (Kommunikationsparameter eingeben),
2. Parameter in die einzelnen Teilnehmer übertragen.

Konfigurieren mit easySoft-CoDeSys (Net-ID1 = IEC-Teilnehmer)

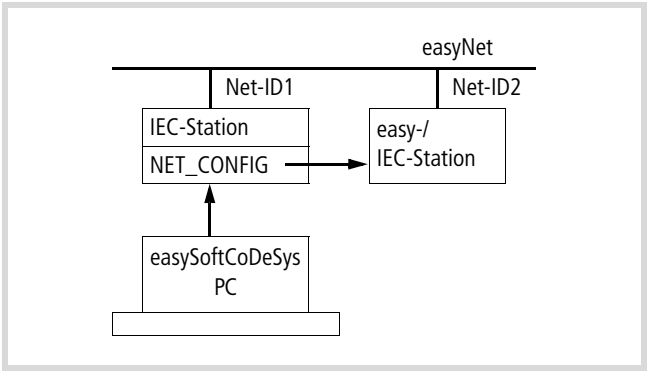


Abbildung 21: Netzwerkkonfiguration über PC (easySoft-CoDeSys)

1. Kommunikationsparameter eingeben
In der Programmiersoftware easySoft-CoDeSys geben Sie die Parameter für alle Teilnehmer ein.

- ▶ Rufen Sie in der Programmiersoftware den Steuerungskonfigurator auf.
- ▶ Wählen Sie das Register „CAN/easyNET" an.
- ▶ Machen Sie im Feld „easyNET-Einstellungen" im Feld „Aktivieren" einen Haken.
- ▶ Setzen Sie im Feld „easyNET-ID" eine „1" ein.
- ▶ Aktivieren/deaktivieren Sie die Funktionen „Remote Run" (Net-ID = 2…8) und „Send I/O".

Ordnen Sie den weiteren Teilnehmern eine Net-ID zu:

- ▶ Klicken Sie auf die Schaltfläche „easy-NET konfigurieren" und wählen Sie, wie in Abbildung 22 dargestellt, die Net-IDs für die Teilnehmer aus.

Für alle ausgewählten Teilnehmer gelten die aktivierten/deaktivierten Funktionen „Remote Run" und „Send I/O".

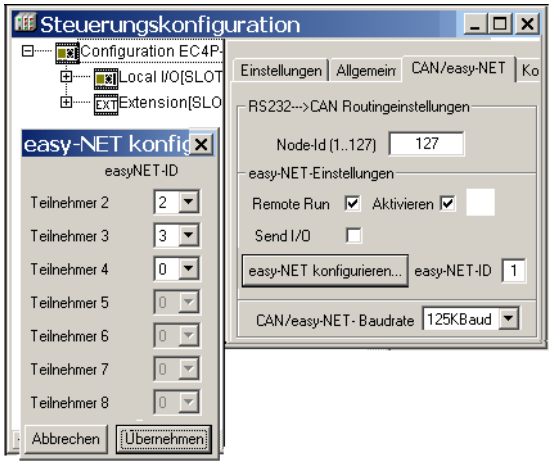


Abbildung 22: Parameter eingeben und Teilnehmer Net-ID festlegen

- ▶ Tragen Sie im Feld „CAN/easy-NET-Baudrate" die Baudrate ein.

Erstellen Sie das Anwenderprogramm und binden Sie die Funktion NET_CONFIG mit ein, → Abschnitt „Konfigurieren mit der Funktion Net_Config", Seite 29.

2. Übertragen der Parameter in die einzelnen Teilnehmer mit Hilfe der Software-Funktion NET_CONFIG
Übertragen Sie das Projekt zum Teilnehmer mit der Net-ID =1. Nach dem Start des Teilnehmers übernimmt die NET_CONFIG-Funktion die Übertragung der Parameter zu den einzelnen Teilnehmern.

Konfigurieren mit easySoft (Net-ID1 = easy- oder IEC-Teilnehmer)

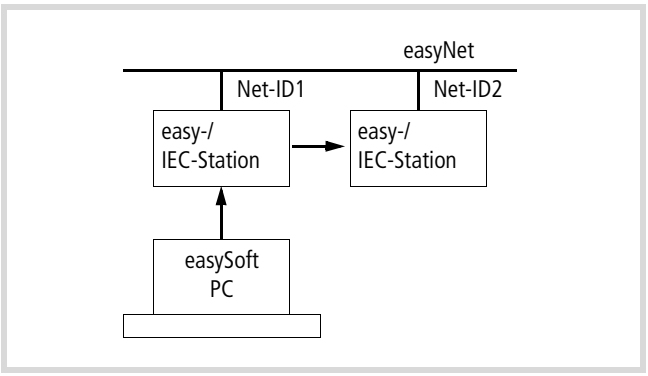


Abbildung 23: Netzwerkkonfiguration über PC (easySoft)

In der Programmiersoftware easySoft erstellen Sie das gesamte Netzwerk mit den Kommunikationsparametern für die einzelnen Teilnehmer. Verbinden Sie den PC mit dem Teilnehmer mit der Net-ID1. Mit dem Menübefehl <Kommunikation-> Konfiguration -> NET> stoßen Sie die Übertragung der Parameter vom PC zu den einzelnen Teilnehmern an.

Voraussetzung:

- Laden Sie zunächst in den Teilnehmer mit der Net-ID1 sein Programm und seine Konfiguration, die ihm die Net-ID = 1 zuordnet.
- Soll ein IEC-Teilnehmer die Net-ID1 erhalten, so muss zusätzlich die easyNet-Einstellung aktiviert werden, → Abschnitt „Konfigurieren mit easySoft-CoDeSys (Net-ID1 = IEC-Teilnehmer)“, Seite 27.
- Danach kann mit Hilfe der easySoft auf den Teilnehmer Net-ID1 zugegriffen werden und über ihn können die weiteren Teilnehmer konfiguriert werden.

1. Kommunikationsparameter eingeben

Haben Sie in der easySoft im Register „Projekt“ die Teilnehmer plaziert, so wurde jedem Teilnehmer aufgrund seiner Plazierung am Strang eine Net-ID zugeordnet oder von Ihnen festgelegt.

Sie können nun nach der Anwahl eines Teilnehmers die Parameter, wie z. B. „Remote-RUN“ und „Sende I/O“, eintragen.

2. Übertragen der Parameter und Zuordnen der Net-IDs

▶ Verbinden Sie den PC mit dem Teilnehmer Net-ID1.
▶ Wählen Sie den online-Betrieb.
▶ Klicken Sie im Menü <Kommunikation -> Konfiguration> die Schaltfläche „NET“ an.
▶ Beantworten Sie die Frage im Fenster „Konfiguration NET“ mit ja, so laden Sie die Parameter inclusive der Net-ID in alle Teilnehmer.

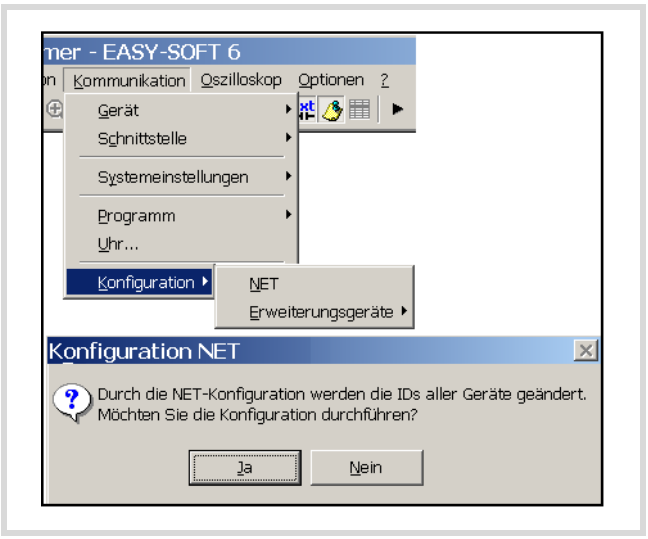


Abbildung 24: Konfiguration durchführen, Menüübersicht

Konfigurieren über das Gerätedisplay

Setzen Sie eine easy800, MFD-CP8-…oder EC4-200 mit Display ein, können Sie die Parameter auch über das Display eingeben.

Voraussetzung:

- Sie haben in das Gerät die NET-ID = 1 und die Baudrate übertragen.
- Die Konfiguration führen Sie durch, wenn Sie unter <KONFIGURATOR -> NET -> KONFIGURIEREN> die Frage „konfigurieren?“ mit ok beantworten.

Konfigurieren mit der Funktion Net_Config

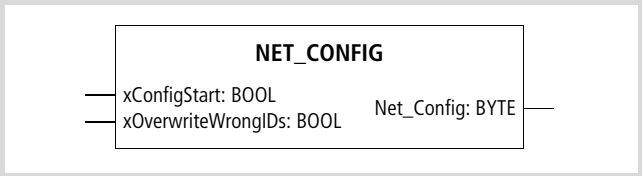


Abbildung 25: Funktion NET_CONFIG

Die Funktion ist Bestandteil der Bibliothek „SysLibEasyNet.lib".

Beschreibung
Mit Hilfe dieser Funktion wird eine Konfiguration der Teilnehmer durchgeführt.

→ Beachten Sie die Voraussetzungen zur Konfiguration → Abschnitt „Konfigurieren mit easySoft-CoDeSys (Net-ID1 = IEC-Teilnehmer)", Seite 27!

Mit einer positiven Flanke am Eingang „xConfigStart" wird die Funktion gestartet. Den Ablauf können Sie überwachen, indem Sie die Rückgabewerte kontrollieren.

Rufen Sie die Funktion Net_Config in einem separaten Programmabschnitt auf, der nur bei Bedarf (Konfigurationswunsch) durchlaufen wird.

Eingänge der Funktion
- xConfigStart:
  Eine Flanke (FALSE->TRUE) startet die Funktion.
- xOverwriteWrongIDs:
  Sind in einem Netzwerk Teilnehmer angeschlossen, die bereits eine Adresse haben, so kann der Anwender durch das Anlegen von TRUE/FALSE am Eingang vorgeben, ob die bestehenden Adressen bleiben oder verändert werden sollen.
  – TRUE:
    Teilnehmer mit bestehenden Adressen erhalten die Adressen aus der Konfiguration.
  – FALSE:
    Die bestehenden Adressen und die Reihenfolge der Teilnehmer werden mit den konfigurierten Adressen und Reihenfolge verglichen. Bei Ungleichheit wird die Konfiguration nicht durchgeführt und an der Ausgangsvariable eine Fehlermeldung (0xFE) ausgegeben.
- Ausgangsvariable der Funktion Net_Config:
  – 0x00(0): keine Aktion
  – 0x01(1): Konfiguration ordnungsgemäß abgeschlossen
  – 0x02(2): Konfiguration wird durchgeführt
  – 0xFF(-1): Teilnehmer, der die Konfiguration durchführen soll, hat nicht die Net-ID = 1, die Adressfolge ist falsch
  – 0xFE(-2): Net-ID-Konflikt
  – 0xFD(-3): Timeout-Fehler, keine Antwort vom angesprochenen Teilnehmer
  – 0xFC(-4): Anzahl der bestehenden Teilnehmer ist größer als die der konfigurierten Teilnehmer
  – 0xFB(-5): easyNet nicht aktiv

Beispiel
Programmierung der Funktion Net-Config.

```
Program NETConfig
VAR
xStartAutoconfig          AT%IX0.0    :BOOL;
xOverwriteWrongIDs        AT%IX0.1    :BOOL;
xConfigStart                          :BOOL:=FALSE;
xConfigurationDone                    :BOOL:=FALSE;
siStateOf Autoconfig                  :SINT;
siErrorCode                           :SINT;
fbTriggerConfig                       :R_TRIG;
END_VAR

fbTriggerConfig(CLK:=xStartAutoconfig );
IF fbTriggerConfig.Q AND siStateOfAutoconfig = 0 THEN
  xConfigStart:= TRUE;
  xConfigurationDone:= FALSE;
  siErrorCode:= 0;
END_IF
siStateOfAutoconfig := Net_Config
(xConfigStart, xOverwriteWrongIDs);
CASE siStateOfAutoconfig OF
0:(* Waiting for start of configuration *)
; (* Do nothing *)
1:(* Configuration successfully terminated *)
  xConfigurationDone := TRUE;
  xConfigStart := FALSE;
2:(* Configuration pending*)
; (* Do nothing *)
ELSE
  (* Configuration done with error *)
  siErrorCode := siStateOfAutoconfig;(* Error handling *);
  xConfigStart := FALSE;
END_CASE
```

6 Kombination von easyNet mit CAN/CANopen

Kombinationsmöglichkeiten

Grundsätzlich ist die Kombination von easyNet mit CAN (verschiedenen Arten) über einen Bus möglich. In Abhängigkeit von der CAN-Kommunikation sind einige Regeln zu beachten.

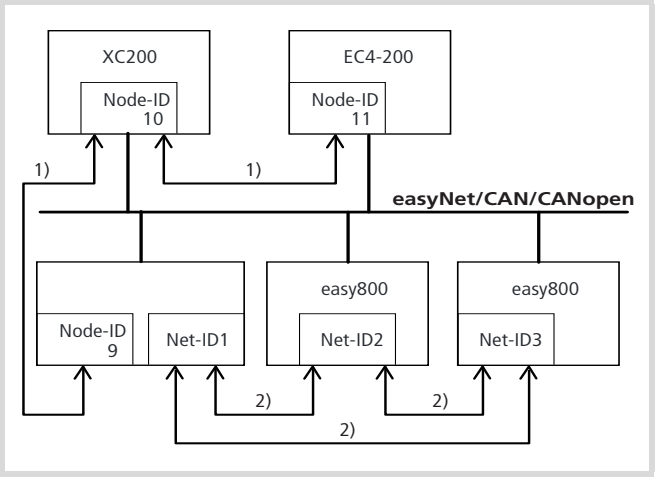


Abbildung 26: Möglichkeiten der Datenübertragung im easyNet, CAN, CANopen Netzwerk

1) CANopen, CAN-Direkt, CAN-Netzwerkvariablen.

2) easyNet.

- easyNet und CANopen
  Eine gleichzeitige Nutzung von easyNet und CANopen führt dazu, dass für die CANopen-Teilnehmer nur die folgenden Node-IDs zur Verfügung stehen:
  – Node-IDs: 9…12; 16; 25…32.
  – Eine Verwendung anderer Node-IDs kann zur Kollisionen mit easy-Teilnehmern führen.
- easyNet und CAN-Direkt[1)]
  Wird easyNet mit CAN-Direkt genutzt, so stehen folgende COB-ID Bereiche zur Verfügung (COB = Communication-Object) COB-IDs 0x1:
  – 0x7f für hochpriore CAN-Telegramme 0x781;
  – 0x7FF für niederpriore CAN-Telegramme.
- easyNet und CAN-Netzwerkvariablen[1)]
  Für die gleichzeitige Verwendung von easyNet und CAN-Netzwerkvariablen stehen folgende COB-ID Bereiche zur Verfügung (COB =Communication-Object) COB-IDs 0x1:
  – 0x7f für hochpriore CAN-Telegramme 0x781;
  – 0x7FF für niederpriore CAN-Telegramme.

1) Bei gleichzeitiger Verwendung von CAN-Direkt und CAN-Netzwerkvariablen dürfen sich die COB-IDs nicht überschneiden.

Hinweise zum Betrieb

Buslast steuern

In einem easyNet-Netzwerk steuern die Teilnehmer die Buslast. Damit ist sichergestellt, dass alle Nachrichten übertragen werden.

In einem CAN-/CANopen-Netzwerk sowie der Kombination von easyNet und CAN/CANopen ist die Buslast höher als in einem easyNet-Netzwerk. In diesem Fall ist der Parameter „SendIO/xSendIO“ bei jedem easy-Teilnehmer zu deaktivieren, damit eine grundsätzliche Reduzierung der Buslast erreicht wird.

Um die Buslast weiter zu senken, können Sie die Pausenzeit zwischen den Übertragungszyklen der Nutzdaten für einen easy-Teilnehmer erhöhen. Erhöhen Sie in der Projektansicht der easySoft im Register Kommunikationsparameter den Parameter „Bus-Delay“. Der Wert „1“ bedeutet, dass sich die Pausenzeit verdoppelt, der Wert „15“ bedeutet, die Pausenzeit erhöht sich um den Faktor 16, siehe easySoft-Onlinehilfe -> Konfiguration des Netzwerkes.

Um einen ordnungsgemäßen Datenaustausch sicherzustellen, darf die Buslast 75 % nicht überschreiten.

→ Beachten Sie, dass beim Routing, dem Zugriff vom PC auf einen Zielteilnehmer (nicht auf denTeilnehmer, der mit dem PC verbunden ist), eine zusätzliche Busbelastung von 15 % entstehen kann.

Versuchen Sie eine möglichst gleichmäßige Auslastung des Busses zu erreichen.

Programmieren Sie die Teilnehmer so, dass diese die Daten nicht gleichzeitig senden!

Auslastung des CAN-Busses anzeigen

Die Buslast kann der Anwender mit Hilfe des Browserbefehls „canload“ in der easySoft-CoDeSys kontrollieren.

Der PLC-Browser-Befehl „canload“ ist Bestandteil der Library „XC200_Util.lib“. Er zeigt die Auslastung des CAN-Busses an.

Beispiele zur Anzeige → Abb. 27 und Abb. 28, Seite 32.

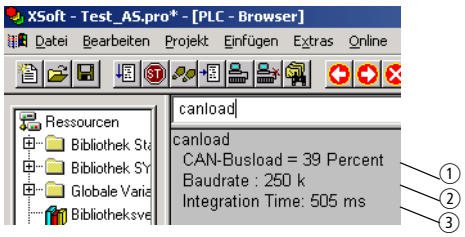


Abbildung 27: Auslastung des CAN-Busses (Beispiel 1)

① Auslastung des CAN-Bus im letzten Integrationsintervall.

② Aktuelle Baudrate des CAN-Bus

③ Zeit des Integrationsintervalls.
Die Integrationszeit ist defaultmäßig auf 500 ms eingestellt und lässt sich über den Browser nicht ändern.

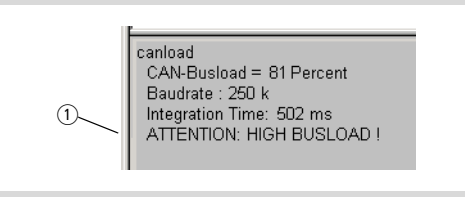


Abbildung 28: Auslastung des CAN-Busses mit Warnmeldung (Beispiel 2)

① Warnmeldung, → Tabelle 4

Tabelle 4: Mögliche Warnmeldungen

Warnmeldung	Bedeutung
ATTENTION: HIGH BUSLOAD	Auslastung des CAN-Busses ≧ 75 %
CAN-Bus not activated	Der CAN-Bus ist nicht aktiv
CAN-Busload = Invalid Calculation	Die Überwachung der Busbelastung ist fehlgeschlagen

easyNet-Telegramm überwachen

Bei einer XC200- oder EC4-200- Steuerung kann durch folgendes Beispielprogramm der Empfang von easyNet-Telegrammen überwacht werden. Ein Verlust eines Telegramms durch den Controller oder durch den Überlauf eines Empfangspeichers kann vom Anwenderprogramm erkannt werden. In diesem Fall ist einer der beiden Parameter „CAN_GET_HW_OVERFLOW_COUNT“ oder „CAN_GET_QUEUE_OVERFLOW_COUNT“ größer als 0. Um die Telegrammverluste auszuschließen ist die Buslast zu senken.

Zur Programmerstellung ist die Einbindung der Bibliothek „SysLibCan.lib“ erforderlich. Der Parameter „dwHandle“ ist abhängig vom Teilnehmertyp einzutragen:

- für die XC200 ist eine „5“,
- für die EC4-200 eine „6“.

Programm zur Überwachung von Telegrammen:

```
PROGRAM PLC_PRG
VAR
MyNetMainStruct : EASY_NET_MAIN;
iHwOverflowCount: INT := 0;
iQueueOverflowCount: INT :=0;
END_VAR
VAR CONSTANT
dwHandle_XC200:DWORD:=05;
dwHandle_EC4-200:DWORD:=06;
END_VAR

iHwOverflowCount:= SysCanControl(
dwHandle:=dwHandle_XC200,
wFunction:=CAN_GET_HW_OVERFLOW_COUNT,dwParam := 0);
iQueueOverflowCount:= SysCanControl(
dwHandle := dwHandle_XC200,
wFunction := CAN_GET_QUEUE_OVERFLOW_COUNT,dwParam := 0);
IF iHwOverflowCount > 0 OR iQueueOverflowCount > 0 THEN
; (*Telegramm verloren.*)
END_IF
(* Allgemeiner Buszugriff *)
Net_Update(ByNetDrvNr:=0 ,
pstruNetMain:= ADR(MyNetMainStruct));
```

→ Der Parameter „CAN_GET_HW_OVERFLOW_COUNT“ für die XC200 ist ab der Hardwareversion 5 (V05) verfügbar.

7 Programmieren über easyNet (Routing)

Als Routing bezeichnet man die Möglichkeit, eine online-Verbindung von einem Programmiergerät (PC) zu einem beliebigen (routingfähigen) Teilnehmer in einem easyNet-Netzwerk aufzubauen, ohne dass das Programmiergerät direkt mit dem Zielteilnehmer verbunden sein muss. Es kann an einen anderen Teilnehmer im Netzwerk angeschlossen werden.

Über die Routing-Verbindung können Sie Aktionen durchführen, die auch bei einer direkten online-Verbindung zwischen Programmiergerät und Teilnehmer möglich sind:

- Programmdownload,
- Online-Änderungen,
- Programmtest (Debugging).

Das Routing bietet den Vorteil, dass Sie von einem Teilnehmer, der mit dem Programmiergerät verbunden ist, Zugriff auf alle routingfähigen Teilnehmer am easyNet erhalten. So lassen sich dezentral angeordnete Geräte leicht bedienen.

Allerdings ist die Datenübertragung von Routing-Verbindungen deutlich langsamer als bei Direktverbindungen (Seriell oder TCP/IP). Dies macht sich beispielsweise durch langsamere Aktualisierungszeiten von Visualisierungselementen (Variablen) oder langsamere Download-Geschwindigkeiten bemerkbar.

Um das Routing einsetzen zu können, müssen folgende Voraussetzungen erfüllt sein:

- Sowohl der routende Teilnehmer als auch der Zielteilnehmer müssen das Routing unterstützen.
- Beide Teilnehmer müssen über den Bus verbunden sein.
- Die Teilnehmer müssen über die gleiche aktive Bus-Baudrate verfügen.

In der folgenden Abb. 29 sind die Routing-Möglichkeiten aufgeführt. Wenn Sie den PC an einen Teilnehmer ankoppeln, können Sie über das easyNet zu einem anderen Teilnehmer eine Verbindung aufbauen.

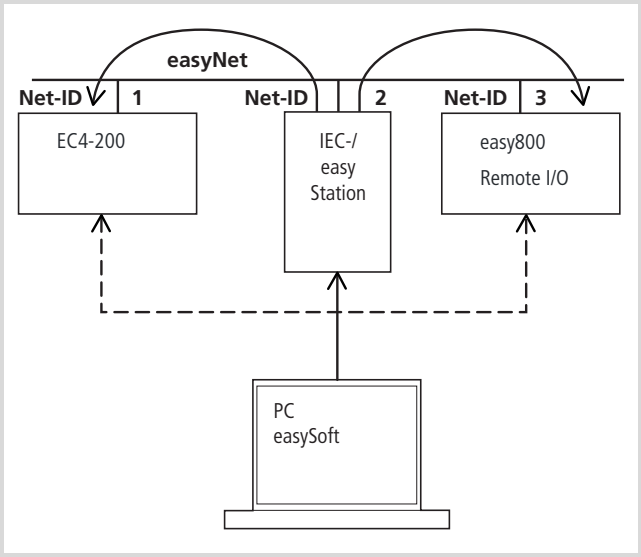


Abbildung 29: Routingmöglichkeiten

Die Verbindung stellen Sie über eine der Programmierschnittstellen des Teilnehmers her, z. B. RS232, USB oder Ethernet.

▶ Klicken Sie in der easySoft zuerst die Schaltfläche „Kommunikation" und dann „Verbindung" an.
▶ Wählen Sie die Schnittstelle aus:
  – Ethernet:
    Unter der Überschrift „Ethernet-Profile" klicken Sie die Schaltfläche „Editieren" an und geben Sie die IP-Adresse an, z. B. „192.168.119.200. Port = 10001"
  – COMx:
    Wählen Sie eine Schnittstelle und die Baudrate aus (Standardbaudrate der IEC-Teilnehmer: 38400 Bit/s, der easy Teilnehmer 19200 Bit/s).
  – COM3 (USB) nur EC4-200

▶ Klicken Sie auf die Schaltfläche „online".
  Der PC wird mit dem Teilnehmer verbunden.
  Um die Verbindung zum Zielteilnehmer aufbauen zu können, muss er über eine Konfiguration mit einer Net-ID verfügen.
▶ Wählen Sie unter der Überschrift „Gerät" einen Zielteilnehmer aus, mit dem Sie kommunizieren möchten.



Abbildung 30: Programmdownload im Online-Betrieb

Im online-Betrieb können Sie die Funktionen in den Bedienfeldern ausführen:

Feld	Funktion
Uhr	Die lokale Uhr setzen, die Geräteuhren im Netzwerk synchronisieren.
Programm	Zwischen „RUN" und „STOP" wechseln.
Anzeige	Zustände der easy-NET Variablen und der Geräteinformation ansehen.

Wenn Sie einen easy-Teilnehmer als Zielteilnehmer ausgewählt haben, können Sie einen Programmdownload durchführen (Schaltfläche „PC -> Gerät" anklicken) oder Parameter ändern.

Sie können folgende Parameter ändern:

Baudrate	im Menü „Projekt" in der Netzwerkübersicht
Busdelay	
REMOTE-RUN	In Abhängigkeit der angewählten Steuerung im Register „Kommunikationsparameter" → NET-Konfigurator
Sende IO	
NET-ID	

So übertragen Sie die neuen Einstellungen in die Teilnehmer:

- ▶ Klicken Sie zuerst die Schaltfläche „Kommunikation → Programm" dann die Schaltfläche „PC => Gerät".
- ▶ Die Einstellungen können Sie durch Anklicken der Schaltfläche „Gerät => PC" aus dem Teilnehmer auslesen.

Die Routingmöglichkeiten sind auch gegeben, wenn Sie, wie in Abbildung 31, Seite 34, einen easy-Teilnehmer mit der Net-ID1 und einen IEC-Teilnehmer mit der Net-ID2 konfigurieren. Wenn Sie dann den IEC-Teilnehmer als Zielteilnehmer auswählen, können Sie nach dem Aufruf des Kommunikationsmenüs der easySoft folgende Aktionen mit dem IEC-Teilnehmer durchführen:

- starten/stoppen,
- Uhrzeit einstellen,
- Geräteinformationen anzeigen lassen.

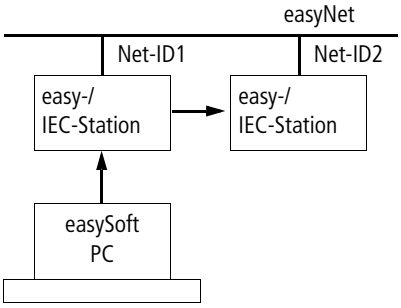


Abbildung 31: Routing über easy-Teilnehmer Net-ID1 auf IEC-Teilnehmer Net-ID2

8 Bus-Topologie

Übersicht

Als Bus-Topologie ist eine Linientopologie mit optionalen Stichleitungen möglich. Die Enden des Busses werden durch Busabschlusswiderstände (120 Ω) abgeschlossen.

Zur Übertragung der Daten im easyNet können Sie an der XC200 eine der Baudraten von 50 kBit/s oder 125 kBit/s einstellen.
An den weiteren Teilnehmern ist die Baudrate anzupassen, so dass alle Teilnehmer mit der gleichen Baudrate kommunizieren.

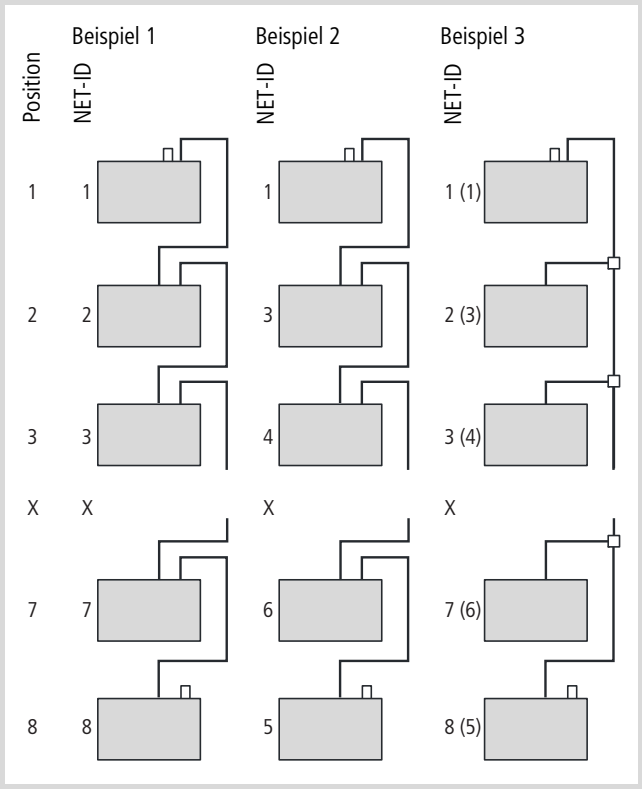


Abbildung 32: Verbindungsmöglichkeiten der Steuerungen

easy-/IEC-Teilnehmer

Beispiel 1
Geografische Position gleich Net-ID
Teilnehmer: EC4-200, easy800,MFD-CP8, MFD-AC-CP8
Netzkonfiguration möglich

Beispiel 2
Geografische Position ungleich Net-ID
Teilnehmer: EC4-200, easy800,MFD-CP8, MFD-AC-CP8
Netzkonfiguration möglich

Beispiel 3
Geografische Position gleich oder ungleich Net-ID
Teilnehmer: EC4-200, easy800,MFD-CP8, MFD-AC-CP8, XC200
Netzkonfiguration nicht möglich, Geräte sind einzeln zu konfigurieren.

Der Teilnehmer auf der Position „1" erhält immer die Net-ID =1. An diesen Teilnehmer werden über das easyNet die 7 weiteren Teilnehmer angeschlossen.

Die IEC-Teilnehmer, die über zwei easyNet-Anschlüsse mit den Signalleitungen (SEL_IN/SEL_OUT) verfügen, können wie in das easyNet eingebunden werden, → Abb. 32.

→ Die XC200 hat zwei CAN/easy-NET-Schnittstellen in einem Steckverbinder. Im Gegensatz zu anderen Geräten hat sie jedoch keine Signalleitung (SEL_IN/OUT)!
Die Leitungen beider Schnittstellen sind intern miteinander verbunden. Sie können, wie in Abb. 32, Beispiel 3, ins Netzwerk eingabunden werden.

Verbinden Sie die Teilnehmer wie in Abbildung 32, Beispiel 3, so müssen Sie das Verbindungskabel zwischen den Teilnehmern und den T-Stücken nach folgendem Verfahren anschließen:

Die easyNet-Teilnehmer haben für den Anschluss an das easyNet einen Eingangs- und einen Ausgangskanal. Die Busleitung wird am Teilnehmer mit der Net-ID 1 am Ausgangskanal angeschlossen und über die T-Stücke (T-connector) auf die Eingangskanäle der anderen Teilnehmer weitergeführt.

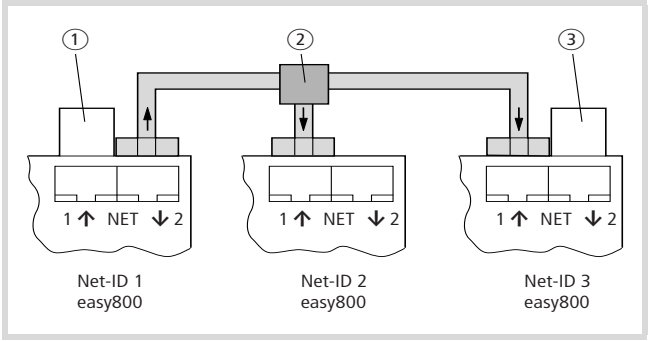


Abbildung 33: easyNet-Netzwerk mit T-Stück

① Busabschlusswiderstand
② T-Stück
③ Busabschlusswiderstand

Tabelle 5: Signalverbindung zwischen den Teilnehmern

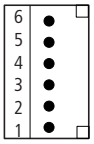
IEC-Teilnehmer	↔	easy-Teilnehmer
ECAN_H		ECAN_H
ECAN_L		ECAN_L
GND		GND
SEL_IN (Net-IDn)		SEL_OUT (Net-IDn-1)
SEL_OUT (Net-IDn)		SEL_IN (Net-IDn+1)

n = 1, … ,8

CANopen/easyNet-Schnittstelle

XC200

Tabelle 6: Belegung der CANopen/easyNet-Schnittstelle der XC200

	Klemme	Signal	
		CANopen	easyNet
	6	GND	GND
	5	CAN_L	ECAN_L
	4	CAN_H	ECAN_H
	3	GND	GND
	2	CAN_L	ECAN_L
	1	CAN_H	ECAN_H

Die Klemmen 1 und 4, 2 und 5, 3 und 6 sind intern verbunden.

EC4-200, easy800, MFD-(AC-)CP8

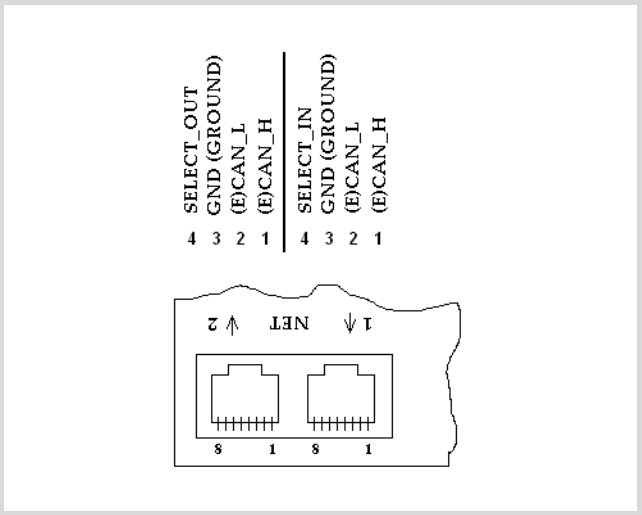


Abbildung 34: CAN/easyNet-Schnittstellen

Stichwortverzeichnis

A
Adresse Net-ID 5
Ausgangsbits 12

B
Baudrate (easyNet) 35
Bibliothek 5
Bit-Ausgänge 11
Bit-Eingänge 11
Busabschlusswiderstände (easyNet) 35
Bus-Topologie (easyNet) 35

C
CAN
- CANopen 31
- -Direkt 31
- -Netzwerk 5
- -Netzwerkvariable 31
- -Schnittstelle 36

canload 31
COB-ID 31
Communication-Object 31

D
Datenaustausch 20
Datenblock 20
Datenübertragung über easyNet 5
Debugging 33
disk_sys 8

E
easyNet
- easy-Teilnehmer 5
- IEC-Teilnehmer 5
- Remote I/O-Teilnehmer 5
- -Schnittstelle 36

Eingangsbits 13
Empfangsdaten 9
Erweiterung 15

F
Flash 8

G
Gateway EASY223-SWIRE 17
Geräteinformationen anzeigen (easyNet) 34
GET-Baustein 14

K
Kommunikationsparameter 5, 27

L
Linientopologie (easyNet) 35

M
Modulnummer 8

N
NET_GET-Funktion 22
Net-ID 5
Netzwerk
- easyNet 5
- easyNet konfigurieren 27
- -merker 20

O
Online-Änderungen 33

P
Programm
- -download 33, 34
- -test 33

Protokolltask 5
PUT-Baustein 13
PUT-Funktion 22

S
Schnittstelle
- CAN 36
- easyNet 36

Sendedaten 9
Starten/Stoppen über easy-Net 34
Statusinformationen der Teilnehmer 9
Stichleitung 27
SWIRE-DIL 17
Systemspeicher 8

T
Task 7
Teilnehmer
- easy- 5
- IEC- 5
- konfigurieren 5
- Routing 33
- Statusinformationen 9
- -überwachung 5

Teilnehmeruhren sychronisieren 11
Topologie (easyNet) 35

U
Uhren synchronisieren 11
Uhrzeit einstellen (über easyNet) 34

V
Verbindung herstellen (PC-Steuerung) 33

W
Wert (DWORD-Format) 12

X
XC200 27

Z
Zielteilnehmer 33