EtherCAN-S7

Industrial Ethernet / UDP Gateway

Software Handbuch

zu Artikel C.2050.07

esd electronic system design gmbh
Vahrenwalder Str. 207 • 30165 Hannover • Germany
www.esd-electronics.com • Fax: 0511/37 29 8-68
Phone: 0511/37 29 80 • International: +49-5 11-37 29 80

Der Inhalt dieses Handbuches wurde mit größter Sorgfalt erarbeitet und geprüft. **esd** übernimmt jedoch keine Verantwortung für Schäden, die aus Fehlern in der Dokumentation resultieren könnten. Insbesondere Beschreibungen und technische Daten sind keine zugesicherten Eigenschaften im rechtlichen Sinne.

esd hat das Recht, Änderungen am beschriebenen Produkt oder an der Dokumentation ohne vorherige Ankündigung vorzunehmen, wenn sie aus Gründen der Zuverlässigkeit oder Qualitätssicherung vorgenommen werden oder dem technischen Fortschritt dienen.

esd electronic system design gmbh
Vahrenwalder Str. 207
30165 Hannover

Tel.: 0511/372 98-0
FAX : 0511/372 98-68
E-Mail: info@esd.electronics.com
Internet: www.esd-electronics.com

Dokument-Datei:	I:\Texte\Doku\MANUALS\CAN\EtherCAN\Deutsch\EtherCAN_S7\EtherCAN_an S7_11S.ma9
Datum des Ausdrucks:	2008-05-19

Software -Version:	EtherCAN 0.1

Änderungen in den Kapiteln

Die hier aufgeführten Änderungen im Dokument betreffen sowohl Änderungen in der *Hardware* als auch reine Änderungen in der *Beschreibung* der Sachverhalte.

Kapitel	Änderungen gegenüber Vorversion
-	Erste freigegebene Ausgabe
-	-

Weitere technische Änderungen vorbehalten.

Diese Seite ist bewusst unbedruckt.

Inhalt

Diese Seite ist bewusst unbedruckt.

1. Einleitung

Dieses Handbuch beschreibt die Funktionalität des EtherCAN-S7. Das EtherCAN-S7 verbindet die Siemens-SPS S7 über einen FB77-Funktionsbaustein mit dem CAN-Bus.

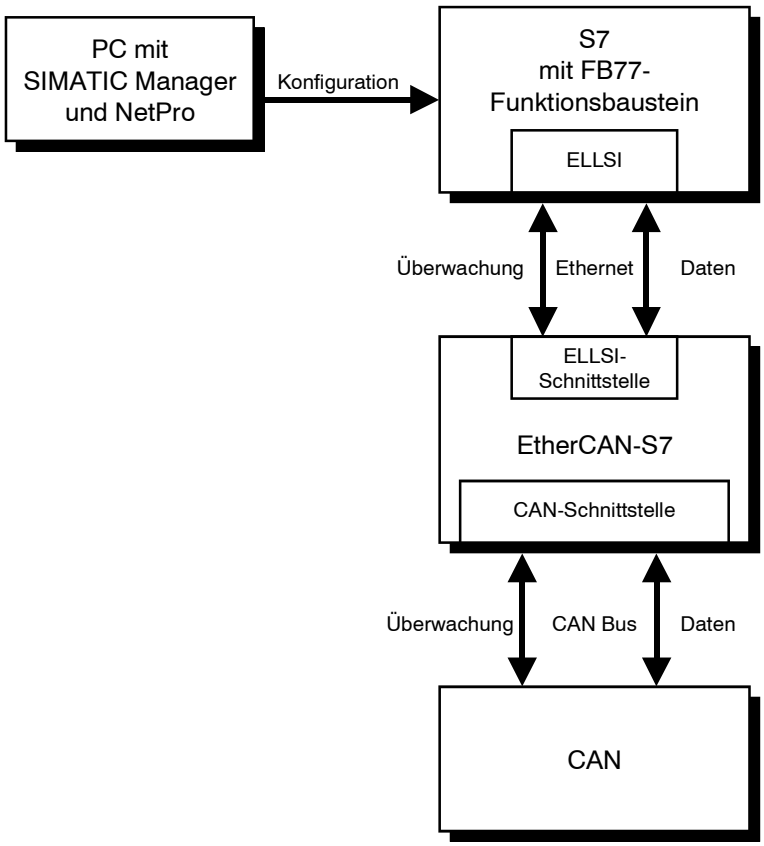


Abb. 1: Anschluss der Siemens-SPS S7 über EtherCAN-S7 an einen CAN-Bus

Mit Hilfe des Funktionsbausteins FB77 und des EtherCAN-S7 ist es einer Siemens-SPS-S7 mit Industrial Ethernet-Schnittstelle möglich, CAN-Telegramme zu versenden und zu empfangen und den Zustand des angeschlossenen CAN-Busses zu überwachen.
Die Konfiguration des Netzwerkes erfolgt über das Siemens Tool NetPro.

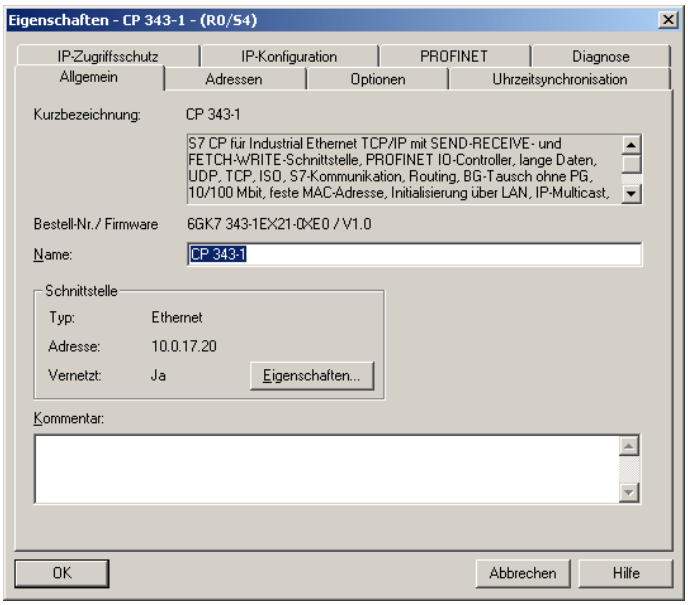
Das EtherCAN-S7 kommuniziert mit der Siemens-SPS S7 über das esd-eigene Protokoll ELLSI. Weiterführende Informationen über ELLSI können dem ELLSI Software-Handbuch entnommen werden. Kenntnisse über ELLSI werden für diese Anwendung jedoch nicht benötigt.

2. Vorgehensweise zur Konfiguration

Zunächst starten Sie den SIMATIC-Manager und erzeugen ein neues Projekt. Fügen Sie eine Station ein und starten Sie den Hardware-Manager. Fügen Sie ein CP ein.
Sollte Ihre SPS bereits Industrial Ethernet unterstützen, konfigurieren Sie sie entsprechend (bitte Handbuch Ihrer SPS beachten).
Mit einem Doppelklick auf *CP* öffnen Sie das Eigenschaftsfenster für die CP-Hardware-Konfiguration.

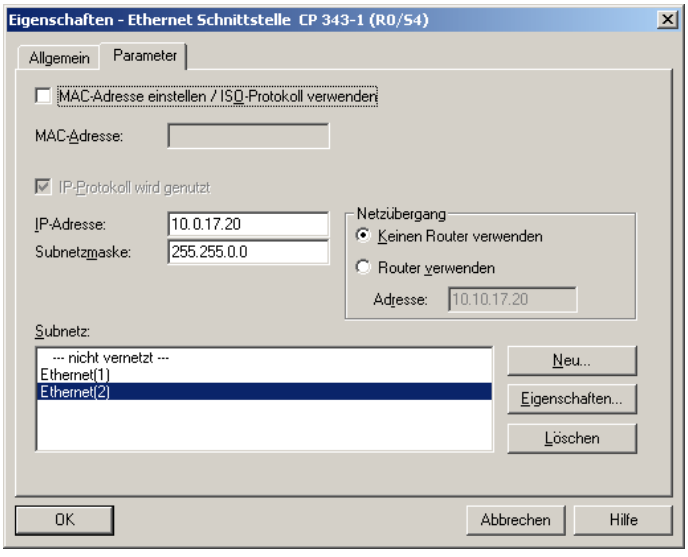
2.1 CP-Hardware Konfigurieren

- Eigenschaften des CP:



Zum Konfigurieren der Eigenschaften des Ethernet-S7 klicken Sie auf die Schaltfläche *Eigenschaften*.
Es öffnet sich das in Abb. 3 dargestellte Fenster.

Abb. 2: CP-Eigenschaften



Im Fenster *Eigenschaften-Ethernet Schnittstelle* können Sie jetzt die eigene IP-Adresse des CP und die Subnetzmaske konfigurieren.
Bestätigen Sie Ihre Eingaben mit *OK*.

Abb. 3: Parameter

Im SIMATIC-Manager ist nun auf Steckplatz 4 das CP eingetragen.

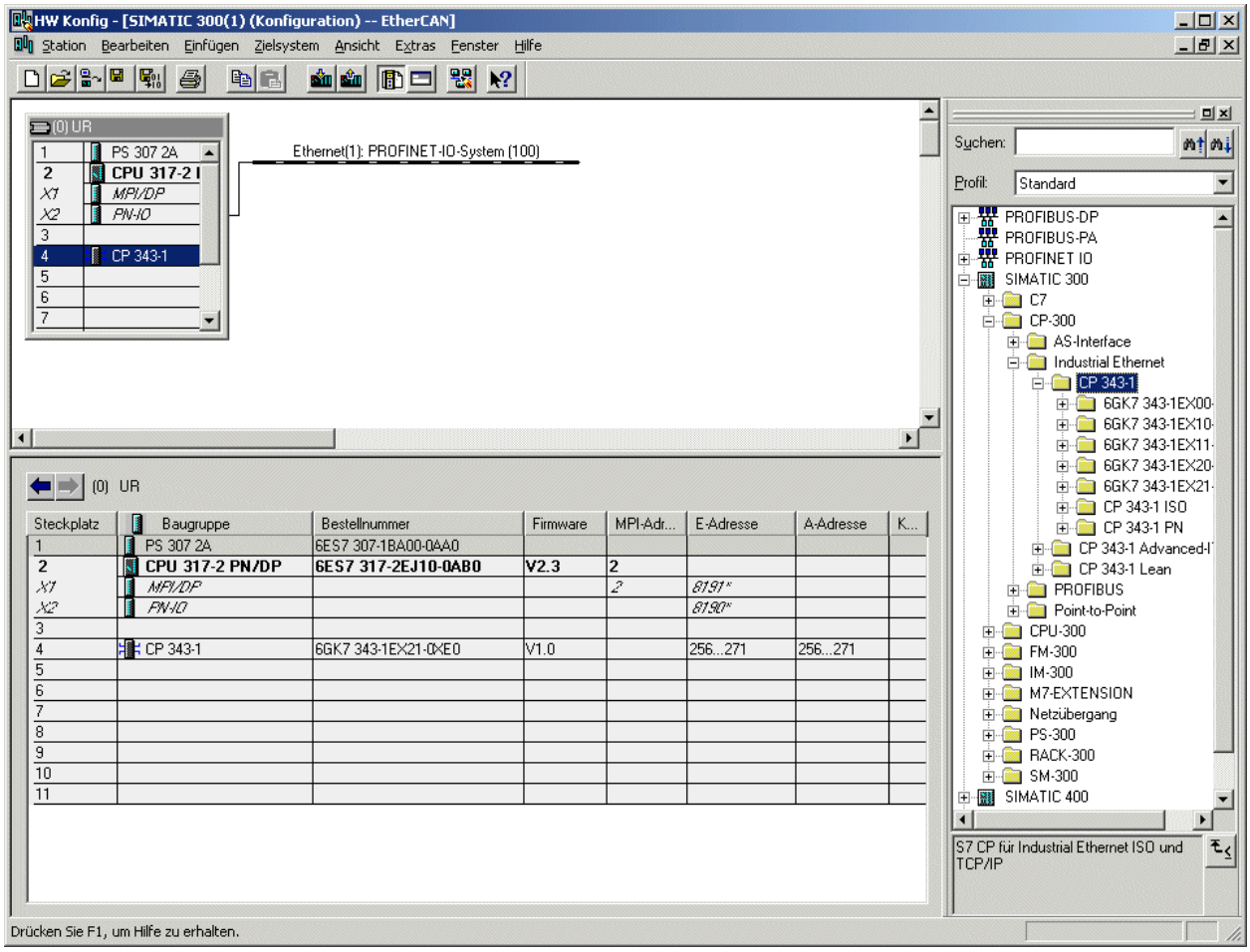


Abb. 4: Eintrag des CP im SIMATIC-Manager

- Hardware-Konfiguration speichern

Speichern Sie die neue Hardware-Konfiguration.

2.2 Konfiguration des Netzwerkes mit NetPro

2.2.1 NetPro Programmaufruf

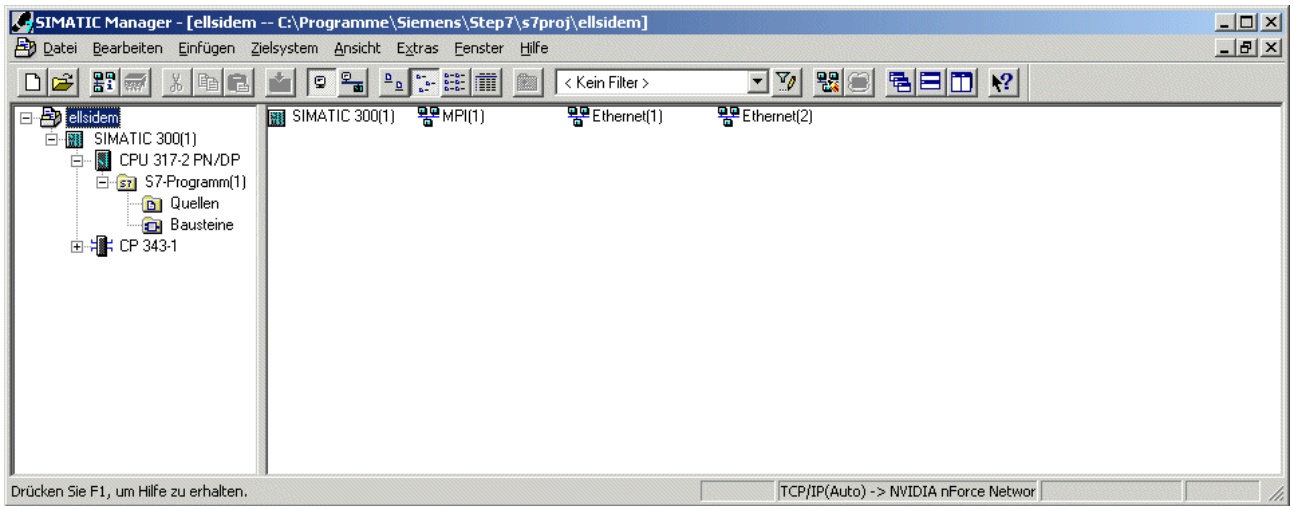


Abb. 5: Aufruf NetPro

Mit einem Doppelklick auf das Symbol für *Ethernet(2)*, das sich rechts im Fenster befindet, starten sie das Programm NetPro.

2.2.2 NetPro Programmfenster

Es öffnet sich das Programmfenster des Programms NetPro.

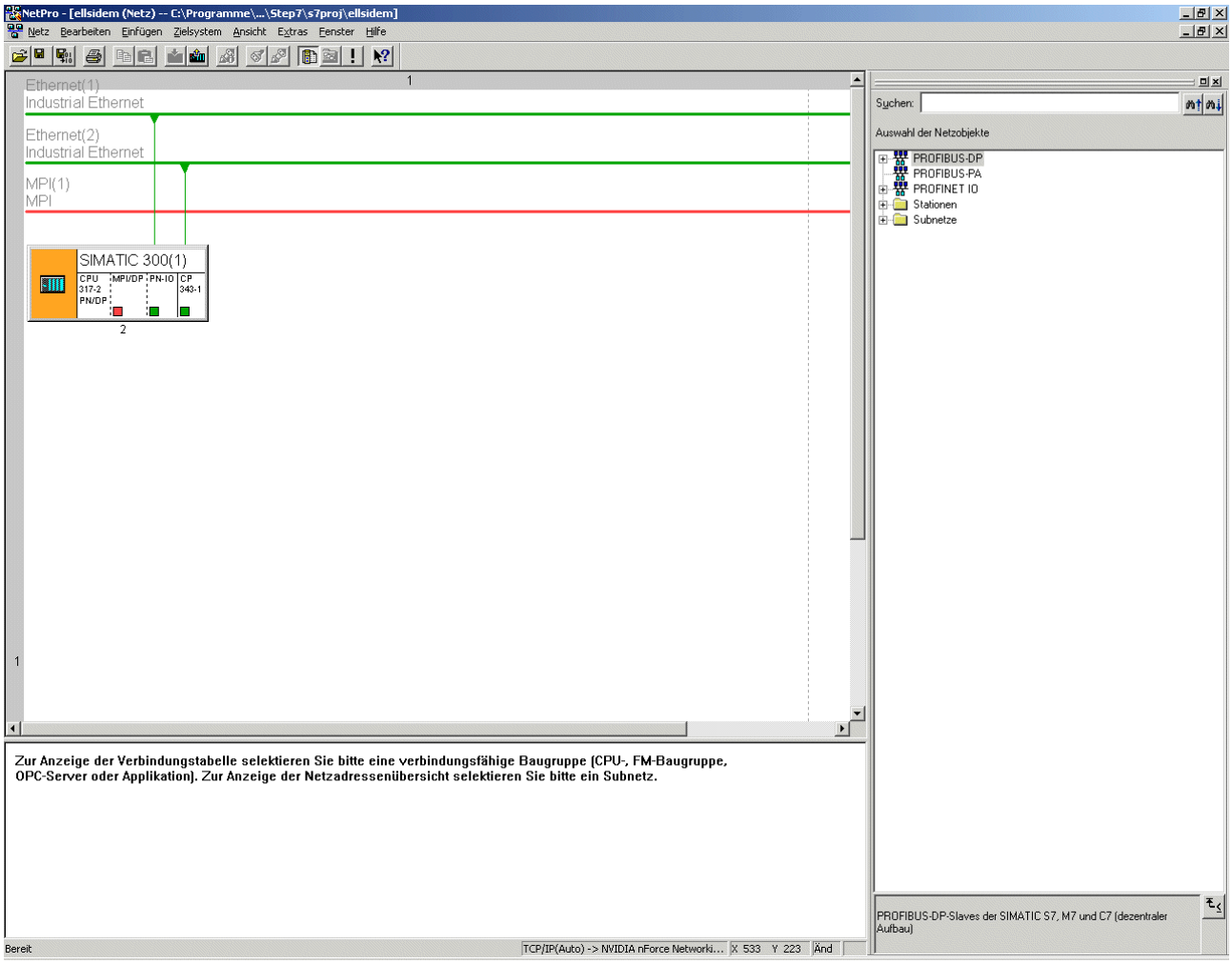


Abb. 6: NetPro Programmfenster

2.2.3 Einfügen einer Station

Hier können Sie nun „Andere Station“ einfügen. Klicken Sie dazu in der Baumstruktur *Auswahl der Netzobjekte* (rechts) auf das Verzeichnis *Stationen*. Wählen Sie nun *Andere Station* aus und ziehen Sie diese mit der Maus (Drag-and-Drop) in das Hauptfenster des Programms NetPro. Wie im folgenden Fenster abgebildet, wird dann das Symbol für die *Andere Station* im Hauptfenster dargestellt.

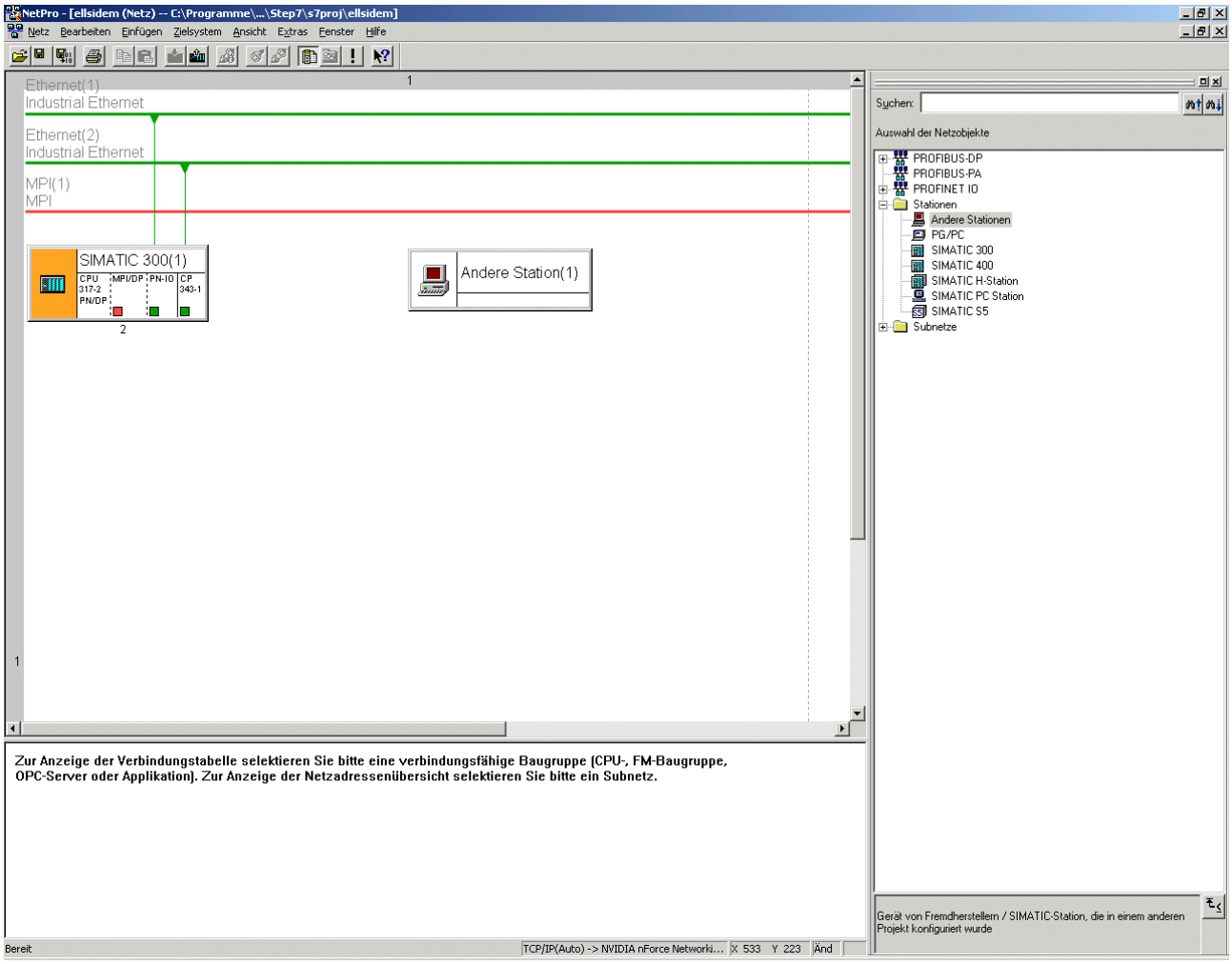
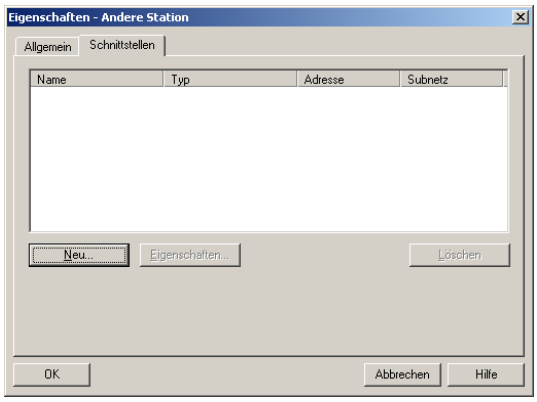


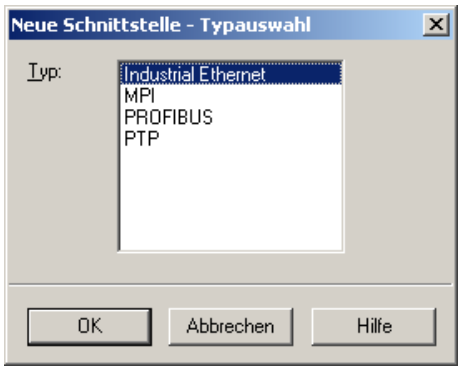
Abb. 7: Einfügen einer Station

2.2.4 Konfiguration *Andere Station*



Durch Doppelklick auf das Symbol *Andere Station* öffnet sich das nebenstehende Eingabefenster *Eigenschaften-Andere Station*, in dem Sie neue Schnittstellen konfigurieren können.
Wechseln Sie im Eigenschaftsfenster auf den Reiter *Schnittstellen* und wählen Sie die Schaltfläche *Neu.*. um eine neue Schnittstelle einzufügen.

Abb. 8: Eigenschaften *Andere Station*



Wählen Sie in dem Auswahlfenster *Neue Schnittstelle-Typauswahl* als Typ *Industrial Ethernet* und bestätigen Sie die Auswahl mit *OK*.

Abb. 9: Typauswahl

Es öffnet sich daraufhin das folgende Eigenschaftsfenster für die Konfiguration der Ethernet-Schnittstelle.

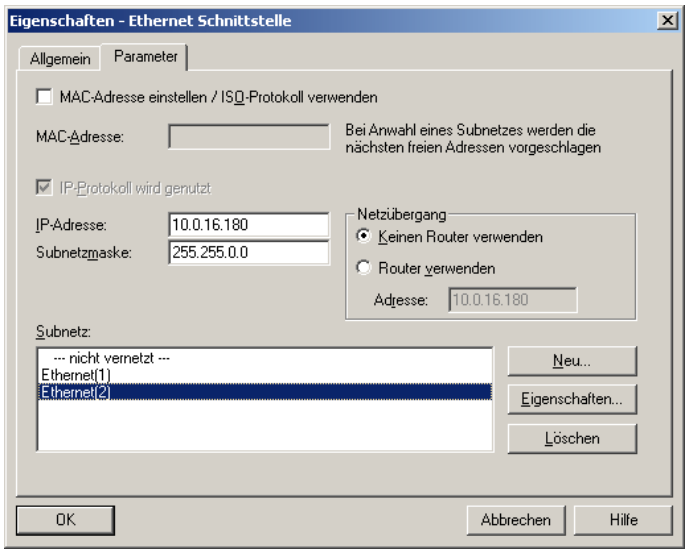


Abb. 10:
Eigenschaften *Ethernet Schnittstelle*

In diesem Eigenschaftsfenster zur Ethernet Schnittstelle wählen Sie den Reiter *Parameter* und können nun die IP-Adresse sowie die Subnetzmaske des EtherCAN-S7-Moduls eingeben (siehe dazu auch das Hardware-Handbuch des EtherCAN-Moduls).
Wählen Sie das Ethernet, mit dem auch das CP verbunden ist (hier Ethernet (2)).

Bestätigen Sie Ihre Eingaben mit OK.

2.3 Konfiguration der Verbindung

- Industrial Ethernet:

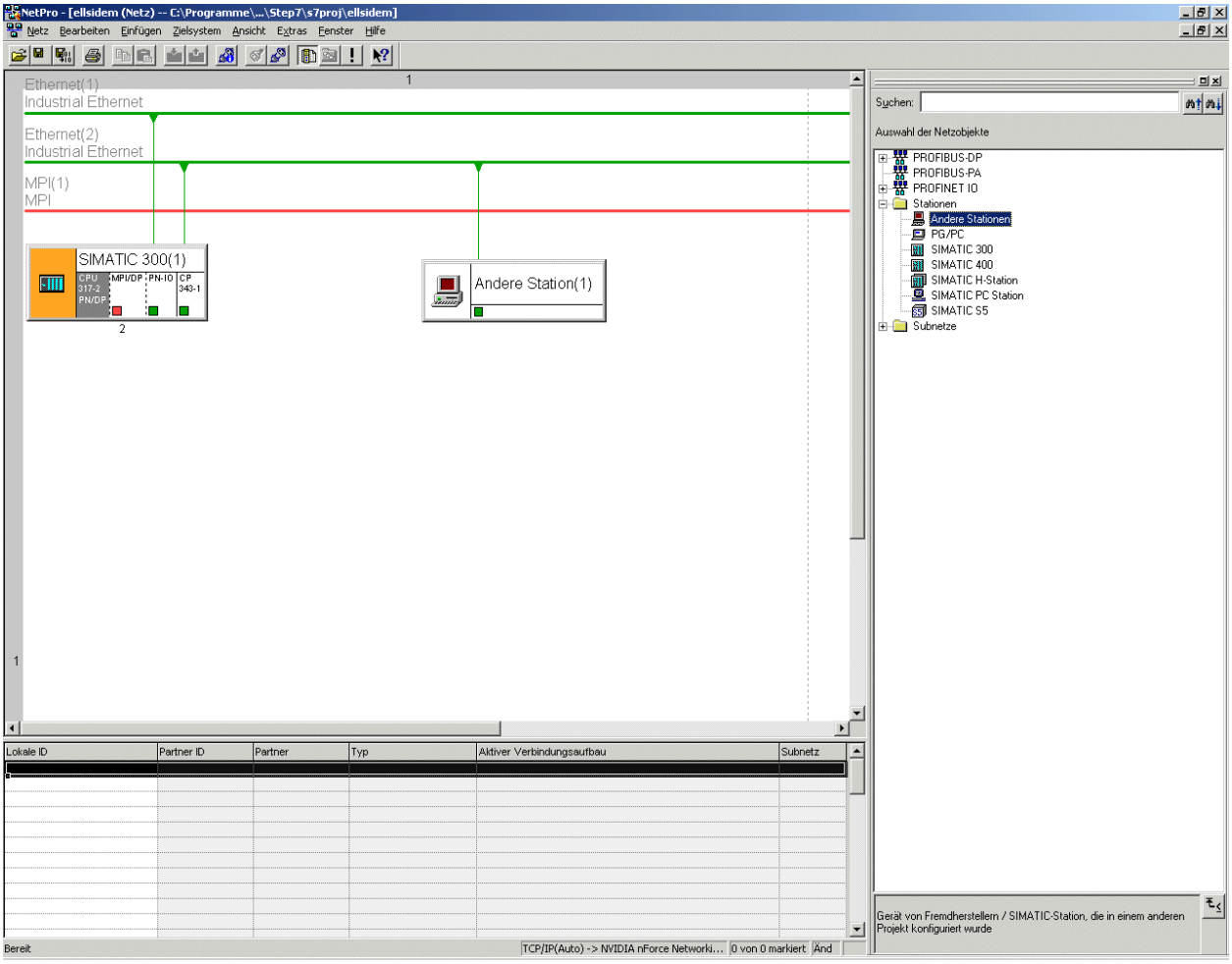
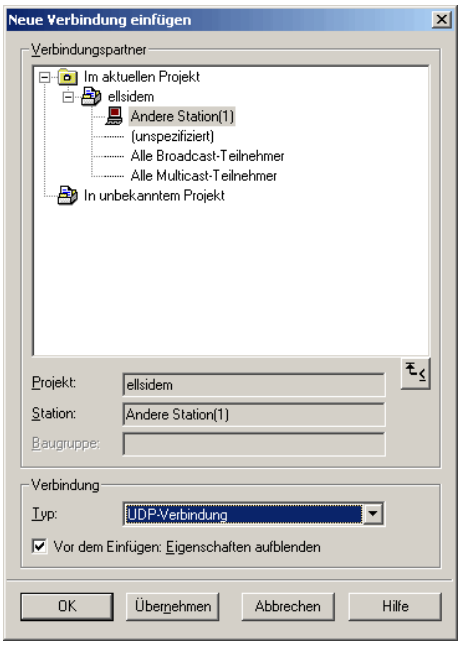


Abb. 11: Verbindungsfenster



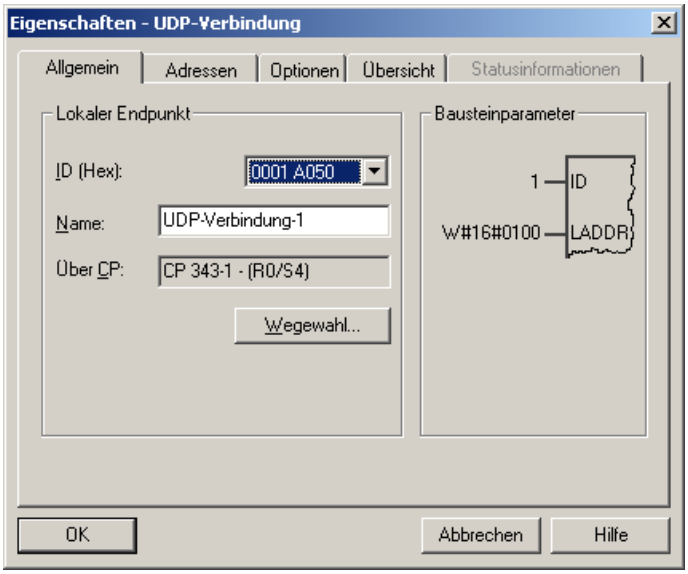
Wenn Sie im Hauptfenster das Symbol für die CPU der SIMATIC300 auswählen, wird im unteren Teil des Programmfensters ein Verbindungsfenster geöffnet. Klicken Sie jetzt mit der rechten Maustaste in die erste Reihe des Verbindungsfensters und wählen Sie in dem erscheinenden Kontextmenü *Neue Verbindung einfügen*.

Es öffnet sich das nebenstehende Fenster.

Wählen Sie in diesem Fenster *Andere Station(1)* als Verbindungspartner und als Verbindungs-Typ UDP-Verbindung. Die Bestätigung mit OK öffnet das Eigenschaftsfenster für die UDP-Verbindung (Abb. 13).

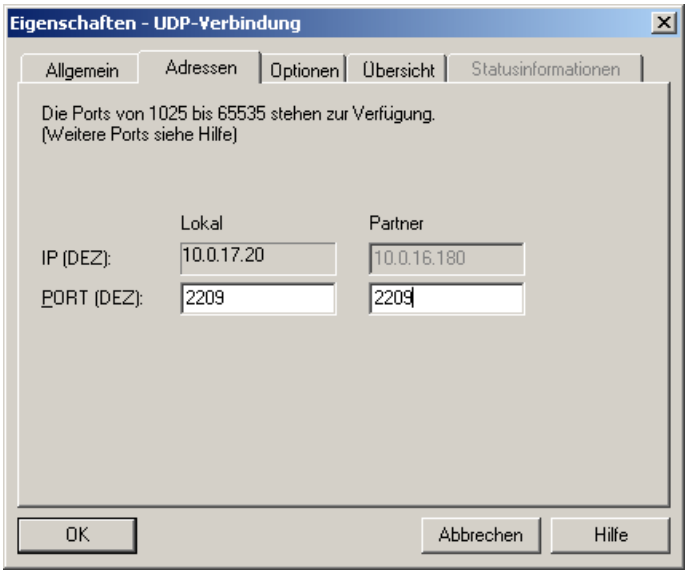
Abb. 12: Neue Verbindung

- Eigenschaften der UDP-Verbindung



Wechseln Sie auf den Reiter *Adressen*.

Abb. 13:
Eigenschaften UDP-Verbindung



Unter dem Reiter *Adressen* muss unbedingt unter *PORT (DEZ)* sowohl unter *Lokal* als auch unter *Partner* in beiden Feldern Port 2209 eintragen werden.

Abb. 14: Adressen

- Netzwerkkonfiguration speichern

Speichern Sie die neue Netzwerkkonfiguration.

2.4 Kopieren der Bausteine aus dem mitgelieferten Projekt

Kopieren Sie die folgenden Bausteine aus dem mitgeliefertem Projekt:

FB77, FC5, FC6, DB1, DB2, DB3, DB77, UDT1

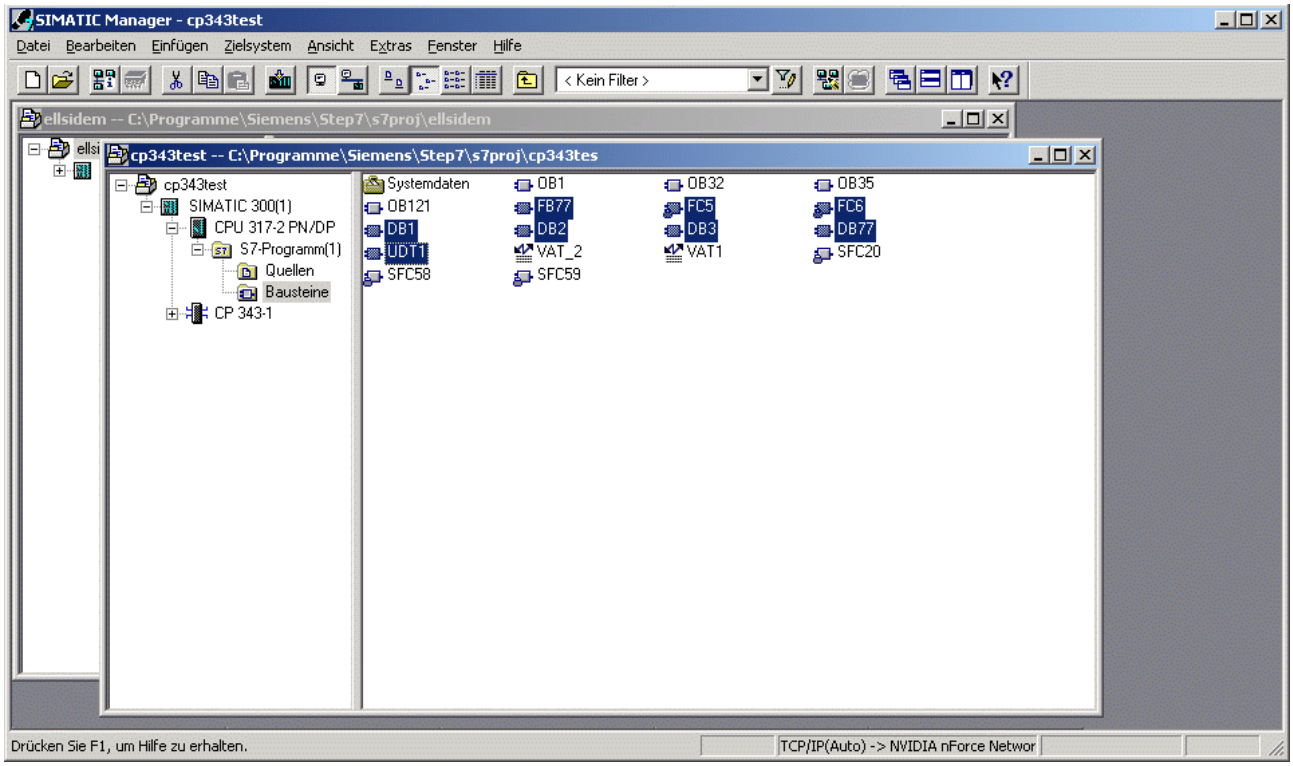


Abb. 15: Bausteine des mitgelieferten Projekts

FB77 ist der Funktionsbaustein zur Bearbeitung des ELLSI-Protokolls.

FC5, FC6 sind die Siemens-Funktionen AG_SEND und AG_RECV zum Handling von UDP-Paketen (können auch aus SIMATIC_NET_CP-Bibliothek übernommen werden).

DB1 liefert die Schnittstelle zum Versenden von CAN-Telegrammen (siehe Seite 43).

DB2 liefert die Schnittstelle zum Hinzufügen und Entfernen von zu empfangenden CAN-Identifiern. (Die Standard 11-Bit-Identifier 0..7FF$_h$ werden vom FB77 beim Initialisieren automatisch freigeschaltet. Zur Performance-Steigerung oder zur Vereinfachung der Auswertung können CAN-Identifier wieder aus der Empfangsmaske entfernt werden).

DB3 hier können empfangene CAN-Telegramme abgelegt werden.

DB77 ist der Instanz-DB zu FB77, die Beschreibung der Übergabeparameter findet sich in den Kommentaren.

UDT1 hier ist die sogenannte CMSG, die esd-Struktur zur Übergabe von CAN-Telegrammen, die an mehreren Stellen (in DB1, DB3 und FB77/DB77) benutzt wird, definiert (s. Kapitel 4.2).

3. CAN

3.1 Grundlagen der CAN-Kommunikation

“Controller Area Network” (CAN) ist nach dem ISO-OSI Schichtenmodell ein Layer 2 Protokoll (Data Link Layer), das in ISO 11898-1 international genormt ist.

Der Text *1) dieses Kapitels gibt eine kurze Einführung in die CAN-Bus-Technologie.

CAN-Kommunikation

Bei CAN werden gleichberechtigte Stationen (Steuergeräte, Sensoren und Aktoren), im folgenden auch CAN-Knoten genannt, über einen seriellen Bus miteinander verbunden.
Die Busleitung selbst ist eine symmetrische oder unsymmetrische Zweidrahtleitung, die je nach Anforderung geschirmt oder ungeschirmt ausgelegt wird.
Die elektrischen Parameter der physikalischen Übertragung sind ebenfalls in ISO 11898 festgelegt.

Das CAN-Protokoll, das im ISO/OSI-Referenzmodell der Datensicherungsschicht (Data Link Layer) entspricht, genügt Echtzeit-Anforderungen.
Im Unterschied zum Kabelbaum erkennt und korrigiert das Netzprotokoll Übertragungsfehler, die durch elektromagnetische Einstrahlung verursacht werden.
Weitere Vorteile einer solchen Vernetzung liegen in der leichten Konfigurierbarkeit des Gesamtsystems und der Möglichkeit einer zentralen Diagnose. CAN wird mit dem Ziel eingesetzt, dass jede Station mit jeder anderen kommunizieren kann, ohne die Steuerung dadurch besonders zu belasten.

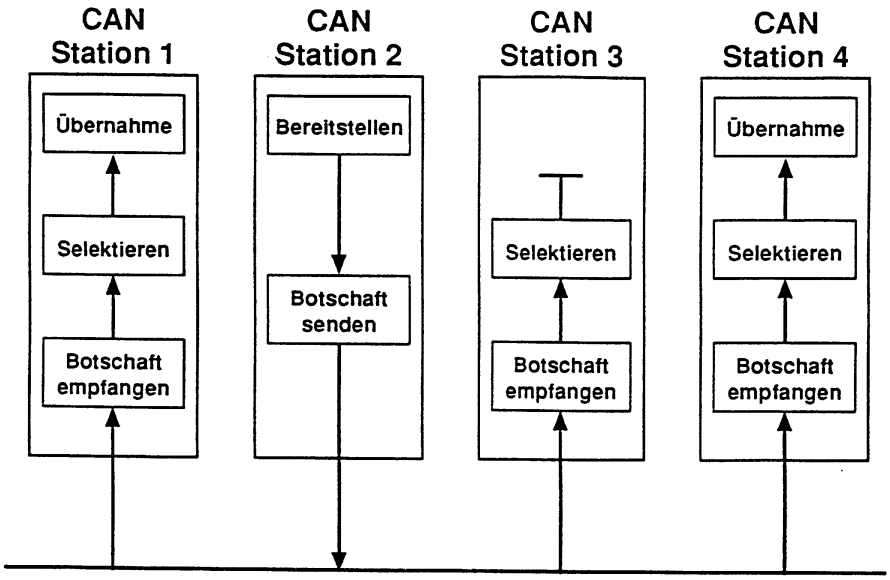
Prinzip des Datenaustausches

Bei der CAN-Datenübertragung werden keine Stationen adressiert, sondern der Inhalt einer Nachricht (z. B. Drehzahl oder Motortemperatur) wird durch einen netzweit eindeutigen Identifier gekennzeichnet.
Neben der Inhaltskennzeichnung legt der Identifier auch die Priorität der Nachricht fest. Dies ist für die Buszuteilung entscheidend, wenn mehrere Stationen um das Buszugriffsrecht konkurrieren.

Möchte die CPU einer beliebigen Station eine Nachricht einer oder mehreren Stationen senden, so übergibt sie die zu übertragenden Daten und deren Identifier mit der Übertragungsaufforderung an den zugeordneten CAN-Baustein (“Bereitstellen”).
Damit ist die Aufgabe der CPU zur Initiierung des Datenaustausches abgeschlossen. Die Bildung und Übertragung der Nachricht übernimmt der CAN-Controller Baustein.

*1)... Dieser Text basiert auf der Einführung: “Controller Area Network, CAN, Ein serielles Bussystem nicht nur für Kraftfahrzeuge” vom internationalen Anwender- und Herstellerverein CiA (CAN in Automation e.V.).

Sobald der CAN-Baustein die Buszuteilung bekommt (“Botschaft senden”), werden alle anderen Stationen des CAN-Netzes zu Empfängern dieser Nachricht (“Botschaft empfangen”). Mit der dann folgenden Akzeptanzprüfung stellen alle Stationen im CAN-Netz nach korrektem Empfang der Nachricht anhand des Identifier fest, ob die empfangenen Daten für sie relevant sind oder nicht (”Selektieren”). Sind die Daten für die Station von Bedeutung, so werden sie weiterverarbeitet (“Übernahme”), ansonsten einfach ignoriert.



Broadcast-Übertragung und Akzeptanzprüfung durch die CAN-Knoten

Durch die beschriebene inhaltsbezogene Adressierung wird eine hohe System- und Konfigurationsflexibilität erreicht. Es lassen sich sehr einfach Stationen zum bestehenden CAN-Netz hinzufügen, ohne dass bei den vorhandenen Stationen Software- oder Hardware-Änderungen vorgenommen werden müssen, wenn die neuen Stationen ausschließlich Empfänger sind.

Da von Seiten des Datenübertragungsprotokolls keine physikalischen Zieladressen für die einzelnen Komponenten vorgeschrieben sind, wird das Konzept der modularen Elektronik ebenso unterstützt wie die Möglichkeit des Mehrfachempfanges (broad/multi-cast) und der Synchronisation von verteilten Prozessen: Messgrößen, die von mehreren Steuergeräten als Information benötigt werden, können über das CAN-Netz so verteilt werden, dass nicht jedes Steuergerät einen eigenen Sensor benötigt.

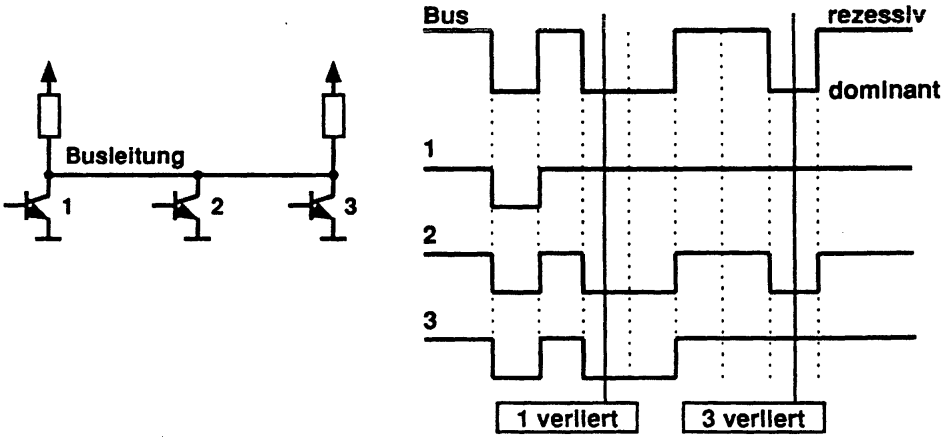
Zerstörungsfreie bitweise Arbitrierung

Um Daten in Echtzeit verarbeiten zu können, müssen sie schnell übertragen werden. Dies setzt jedoch nicht nur eine physikalische Datenübertragungsstrecke mit bis zu 1 Mbit/s voraus, sondern verlangt auch eine rasche Buszuteilung, sofern mehrere Stationen gleichzeitig Nachrichten übertragen wollen.

Bei der Echtzeit-Verarbeitung können die Dringlichkeiten von Informationen, die über den Bus ausgetauscht werden, sehr unterschiedlich sein: eine sich schnell ändernde Größe muss häufiger und deshalb mit geringeren Verzögerungen übertragen werden als andere Größen, die sich nur verhältnismäßig langsam ändern.

Die Priorität, mit der eine Botschaft vor einer anderen, weniger dringlichen Botschaft übertragen wird, ist durch den Identifier der jeweiligen Botschaft festgelegt.
Die Prioritäten werden beim Systementwurf durch entsprechende Binärwerte vergeben und sind nicht dynamisch veränderbar. Der Identifier mit der niedrigsten Binärzahl hat die höchste Priorität.

Der Buszugriffskonflikt wird mittels bitweiser Arbitrierung über die jeweiligen Identifier gelöst, indem jede Station Bit für Bit den Buspegel beobachtet. Entsprechend dem “Wired-and”-Mechanismus, bei dem der dominante Zustand (logisch 0) den rezessiven Zustand (logisch 1) überschreibt, verlieren all diejenigen Stationen den Wettstreit um die Buszuteilung, die rezessiv senden, aber den Bus dominant beobachten.
Alle “Verlierer” werden automatisch zu Empfängern der Nachricht mit der höchsten Priorität und versuchen erst dann wieder zu senden, wenn der Bus frei wird.



Prinzip der zerstörungsfreien, bitweisen Arbirierung

Effizienz der Busvergabe

Die Effizienz des Buszuteilungsverfahrens bestimmt im wesentlichen die möglichen Einsatzgebiete eines seriellen Bussystems. Um möglichst einfach beurteilen zu können, welche Bussysteme für welche Einsatzgebiete geeignet erscheinen, wird in der Literatur ein Schema zur Klassifikation von Buszuteilungsverfahren angegeben.

Generell unterscheidet man dabei zwischen folgenden Klassen:

- Zuteilung mit festem Zeitraster
  Die Zuteilung erfolgt sequentiell an jeden Teilnehmer für eine maximale Zeitspanne, unabhängig davon, ob dieser Teilnehmer in diesem Augenblick den Bus benötigt oder nicht. (Beispiele: Token Slot oder Token Passing)

- Bedarfsabhängige Buszuteilung
  Die Zuteilung erfolgt gemäß vorliegender Übertragungswünsche an einen Teilnehmer, d. h. nur Teilnehmer mit Sendewunsch werden bei der Zuteilung berücksichtigt. (Beispiele: CSMA, GSMA/ CD, Flying-Master, Round Robin oder bitweise Arbitrierung)

Bei CAN wird die Buszuteilung ausschließlich unter den zur Übertragung anstehenden Botschaften ausgehandelt. Daraus folgt, dass das bei CAN spezifizierte Verfahren in den Bereich der bedarfsabhängigen Buszuteilung fällt. Eine andere Klassifikation für die Beurteilung der Effizienz von Busvergabeverfahren ist die Art des Buszugriffs:

- Zerstörungsfreier Buszugriff
  Bei solchen Verfahren wird der Bus sofort oder nach einer spezifizierten Zeit nach Beginn eines einmaligen Buszugriffs (einer oder mehrerer Stationen gleichzeitig) genau einer Station eindeutig zugeteilt. Damit ist sichergestellt, dass jeder Buszugriff von einer oder mehreren Stationen immer zu einer eindeutigen Busvergabe führt. (Beispiele: Token Slot, Token Passing, Round Robin, bitweise Arbitrierung)

- Nicht zerstörungsfreier Buszugriff
  Ein gleichzeitiger Buszugriff von mehreren Stationen führt zum Abbruch der Sendeversuche und damit zu keiner erfolgreichen Buszuteilung. Um überhaupt irgendeiner Station den Bus zuzuteilen, können mehrere Buszugriffe notwendig werden, wobei die Anzahl der Versuche bis zum Erfolg nur eine statistische Größe ist. (Beispiele: GSMA/CD, Ethernet)

Um alle Übertragungsanforderungen eines CAN-Netzes unter Einhaltung der Latenzzeit-Bedingungen bei möglichst geringer Datenübertragungsrate abarbeiten zu können, muss das CAN-Protokoll ein Buszuteilungsverfahren realisieren, das garantiert, dass auch gleichzeitige Buszugriffe mehrerer Stationen immer zu einer eindeutigen Busvergabe führen.

Durch das Verfahren der bitweisen Arbitrierung über die Identifier der zur Übertragung anstehenden Botschaften wird jede Kollision von mehreren sendewilligen Stationen eindeutig aufgelöst, und zwar spätestens nach 13 (Standard-Format) bzw. 33 Bitzeiten (Erweitertes Format) jedes zeitlich beliebigen Buszugriffs.
Im Gegensatz zur nachrichtenweisen Arbitrierung des CSMA/CD-Verfahrens wird mit dieser zerstörungsfreien Kollisionsauflösung gewährleistet, dass in keinem Fall Buskapazität benötigt wird, ohne dabei auch Nutzinformationen zu übertragen.

Auch in Situationen der Busüberlastung erweist sich die Anbindung der Buszugriffspriorität an den Inhalt der Botschaft als vorteilhafte Systemeigenschaft gegenüber existierenden CSMA/CD- oder Token-Verfahren:
Alle aufgelaufenen Übertragungsanforderungen werden trotz der zu geringen Bustransportkapazität in der Reihenfolge der Wichtigkeit für das Gesamtsystem (entsprechend der Botschaftspriorität) abgearbeitet.
Dabei wird die vorhandene Übertragungskapazität für die zu übertragenden Nutzdaten in effizienter Weise genutzt, da zeitliche Lücken bei der Busvergabe sehr klein gehalten werden können.

Ein Kollaps des gesamten Übertragungssystems aufgrund von Überlastsituationen, der beim CSMA/CD-Verfahren auftreten kann, ist bei CAN ausgeschlossen. CAN erlaubt somit die Realisierung eines schnellen bedarfsabhängigen Buszugriffs, der aber aufgrund der bitweisen Arbitrierung über die Botschaftspriorität zerstörungsfrei vonstatten geht.

Beim zerstörungsfreien Buszugriff kann weiter unterschieden werden zwischen
- zentraler Buszugriffskontrolle
- dezentraler Buszugriffskontrolle

je nachdem, ob die Kontrollmechanismen einmal (zentral) oder mehrfach im System (dezentral) vorhanden sind.

Ein Kommunikationssystem mit einer ausgezeichneten Station (u.a. auch für die zentrale Buszugriffskontrolle) muss eine Strategie bereitstellen, die im Falle eines Ausfalls der Leitstation zum Tragen kommt.
Dieses Konzept hat den Nachteil, dass zum einen die Strategie der Ausfallbeherrschung aufwendig zu realisieren ist, und zum anderen die Übernahme der zentralen Station durch eine redundante Station sehr zeitaufwendig sein kann.

Aus den genannten Gründen, und um das Problem der Zuverlässigkeit der Leitstation (und somit auch des gesamten Kommunikationssystems) zu umgehen, realisiert das CAN-Protokoll eine dezentrale Bus-Zugriffskontrolle. Alle für die Kommunikation wesentlichen Mechanismen, auch die Buszugriffskontrolle, werden mehrfach im Netz implementiert. Denn nur dadurch können die hohen Anforderungen an die Verfügbarkeit des Kommunikationssystems erfüllt werden.

Zusammenfassend ist festzustellen, dass CAN ein bedarfsabhängiges Buszuteilungsverfahren realisiert, das über einen zerstörungsfreien Buszugriff mit dezentraler Buszugriffskontrolle eine hohe Nutzdatenrate bei möglichst niedriger Busdatenrate, bezogen auf die Busbelegungszeit aller Stationen, erlaubt.
Die Effizienz des Buszuteilungsverfahrens wird dadurch erhöht, dass ausschließlich die Stationen den Bus belegen, bei denen Übertragungsanforderungen anstehen.

Diese Anforderungen werden entsprechend der Wichtigkeit von Botschaften für das Gesamtsystem abgearbeitet. Dies erweist sich insbesondere in Überlastsituationen als günstig. Da der Buszugang botschaftsbezogen priorisiert ist, können auch niedrige individuelle Latenzzeiten in Echtzeitsystemen garantiert werden.

Formate der Botschaftsrahmen

Das CAN Protokoll unterstützt zwei Formate von Botschaftsrahmen, die sich im wesentlichen nur in der Länge der **Identifier (ID)** unterscheiden. Die Länge des ID beträgt im Standard-Format 11 Bits und im Erweiterten-Format 29 Bits. Der Botschaftsrahmen zur Übertragung von Nachrichten auf dem Bus besteht aus sieben Kennfeldern.

Eine Botschaft im Standard-Format beginnt mit dem Startbit **"Start of Frame"**, dem sich das **"Arbitration Field"** anschließt; dieses Feld enthält den Identifier und das **"RTR"**-Bit (Remote Transmission Request), das kennzeichnet, ob es sich um einen Datenrahmen oder einen Anforderungsrahmen ohne Datenbytes (Remote Frame) handelt.

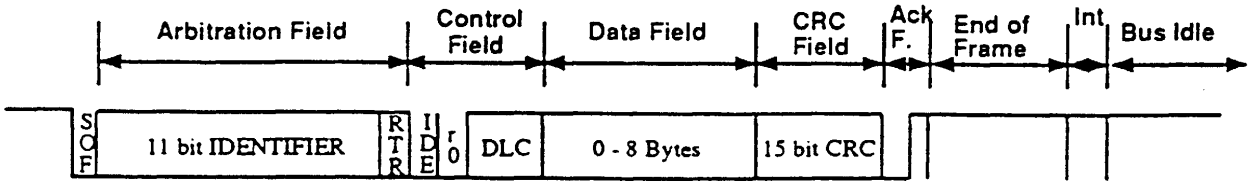
Das **"Control Field"** enthält zur Unterscheidung von Standard- und Erweitertem Format das **IDE**-Bit (Identifier Extension Bit), ein reserviertes Bit für zukünftige Erweiterungen und in den letzten 4 Bits die Anzahl der im Datenfeld enthaltenen Datenbytes.

Dem **"Data Field"**, das eine Länge von 0 bis 8 Byte aufweisen kann, folgt
das **"CRC Field"**, das als Rahmensicherung zur Erkennung von Bitfehlern dient.

Das **"ACK Field"** umfaßt sowohl den ACK-Slot (1 Bit) als auch den ACK-Delimiter (1 rezessives Bit). Das Bit im ACK-Slot wird rezessiv gesendet und von denjenigen Empfängern dominant überschrieben, die die Daten bis zu diesem Zeitpunkt korrekt empfangen haben (positives Acknowledgement). Dabei wird die Bestätigung korrekter Botschaften unabhängig von dem Ergebnis der Akzeptanzprüfung in den Empfängern wahrgenommen.

Mit dem **"End of Frame"** wird das Ende der Botschaft gekennzeichnet. **"Intermission"** ist die minimale Anzahl von Bitzeiten, die aufeinanderfolgende Botschaften trennen. Erfolgt danach kein weiterer Buszugriff durch eine beliebige Station, so bleibt der Bus im Ruhezustand (**"Bus Idle"**).

Standard Format



Botschaftsrahmen im Standard-Format (CAN-Spezifikation 2.0A)

Erkennen und Signalisieren von Fehlern

Das CAN-Protokoll verwendet im Gegensatz zu anderen Bussystemen keine Quittungen, sondern signalisiert eventuelle aufgetretene Fehler. Um Fehler zu erkennen, sind im CAN-Protokoll drei Mechanismen auf Botschaftsebene implementiert:

- Cyclic Redundancy Check (CRC)

Der CRC sichert die Information des Rahmens, indem sendeseitig redundante Prüfbits hinzugefügt werden. Empfangsseitig werden diese Prüfbits aus den empfangenen Bits neu berechnet und mit den empfangenen Prüfbits verglichen. Bei Nichtübereinstimmung liegt ein CRC-Fehler vor.

- Frame-Check

Dieser Mechanismus verifiziert die Struktur des übertragenen Rahmens, indem die Bitfelder mit dem vorgegebenen festen Format sowie die Rahmenlänge überprüft werden. Die durch Frame-Checks erkannten Fehler werden als Formatfehler bezeichnet.

- ACK-Fehler

Wie bereits erwähnt, werden empfangene Rahmen von allen Empfängern durch positives Acknowledgement quittiert. Wird am Sender kein Acknowledgement erkannt (ACK-Fehler), so deutet dies auf einen möglicherweise nur von den Empfängern erkannten Übertragungsfehler, auf eine Verfälschung des ACK-Feldes oder auf nicht vorhandene Empfänger hin.

Außerdem sind im CAN-Protokoll zwei Mechanismen zur Fehlererkennung auf der Bitebene implementiert:

- Monitoring

Die Fähigkeit des Senders, Fehler zu erkennen, basiert auf dem Überwachen der Bussignale: jeder Knoten, der sendet, beobachtet gleichzeitig den Buspegel und erkennt dabei Differenzen zwischen gesendetem und empfangenen Bit. Dadurch können alle globalen Fehler und lokal am Sender auftretenden Bitfehler sicher erkannt werden.

- Bit-Stuffing

Auf der Bitebene wird die Codierung der Einzelbits überprüft. Als Bitrepräsentation verwendet das CAN-Protokoll die NRZ-Codierung (Non-Return-to-Zero), die eine maximale Effizienz bei der Bitcodierung gewährleistet. Dabei werden die Synchronisationsflanken nach der Methode des Bit-Stuffings erzeugt, indem vom Sender nach fünf aufeinanderfolgenden gleichwertigen Bits ein Stuff-Bit mit komplementärem Wert in den Bitstrom eingefügt wird, welches die Empfänger automatisch wieder entfernen. Der Code-Check beschränkt sich auf die Überprüfung der Stuffing-Regel.

Werden ein oder mehrere Fehler mit Hilfe der oben beschriebenen Mechanismen von mindestens einer beliebigen Station entdeckt, so wird die laufende Übertragung durch Senden eines “Error Flag” abgebrochen. Dadurch wird die Annahme der übertragenen Nachricht durch andere Stationen verhindert und somit die netzweite Datenkonsistenz sichergestellt.

Nach Abbruch der Übertragung einer fehlerhaften Botschaft beginnt der Sender automatisch, seine Nachricht erneut zu senden (Automatic Repeat Request). Dabei können wiederum mehrere Stationen um die Buszuteilung konkurrieren. Das erneute Senden einer Botschaft kann in der Regel spätestens 23 Bitzeiten nach der Fehlererkennung begonnen werden, in Sonderfällen beträgt die Systemerholzeit 31 Bitzeiten.

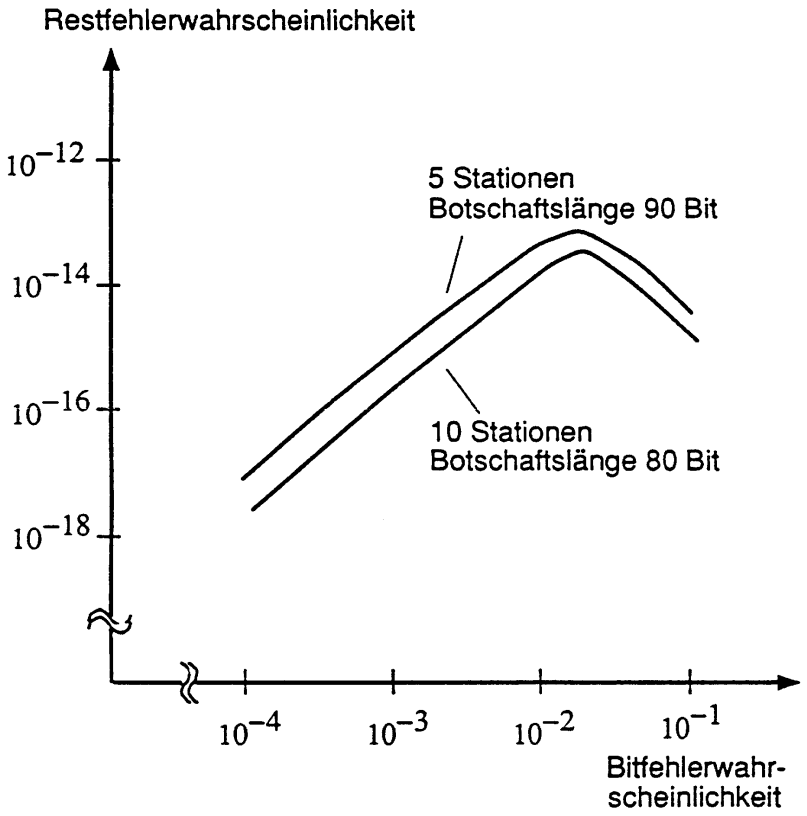
So effektiv und effizient die beschriebene Methode der Fehlerbehandlung mittels Error-Flag sein mag, sie könnte im Falle einer defekten Station zum Abbruch aller (auch korrekter) Botschaften und damit zur Blockierung des Bussystems führen, sofern keine Maßnahmen zur Selbstüberwachung getroffen worden sind. Das CAN-Protokoll verfügt deshalb über einen Mechanismus, der gelegentlich auftretende von anhaftenden Störungen unterscheidet und Stationsausfälle lokalisiert (Fault Confinement).

Dies geschieht durch statistische Bewertung von stationseigenen Fehlersituationen mit dem Ziel, eigene Defekte zu erkennen und eventuell in einen Betriebszustand überzugehen, bei dem das übrige CAN-Netz nicht beeinträchtigt wird.
Dies kann bis zur Selbstabschaltung der Station führen, damit vermeintlich inkorrekte Botschaften nicht mehr von einer solchen Station abgebrochen werden können.

Datensicherheft des CAN-Protokolls

Im Zusammenhang mit Bussystemen wird unter Datensicherheit die Eigenschaft verstanden, durch Übertragungsstörungen verfälschte Daten identifizieren zu können.
Die Restfehler-Wahrscheinlichkeit ist ein statistisches Maß für die Verletzung der Datensicherheit: sie gibt an, mit welcher Wahrscheinlichkeit Daten verfälscht werden und diese Verfälschungen unerkannt bleiben.

Die Restfehler-Wahrscheinlichkeit sollte so niedrig sein, dass während der gesamten Betriebsdauer eines Systems im Mittel keine verfälschten Daten unerkannt bleiben.
Die Berechnung der Restfehler- Wahrscheinlichkeit setzt voraus, dass die auftretenden Fehler klassifiziert werden und die gestörte physikalische Übertragungsstrecke durch ein Modell beschrieben wird.



Restfehler-Wahrscheinlichkeit als Funktion der Bitfehler-Wahrscheinlichkeit

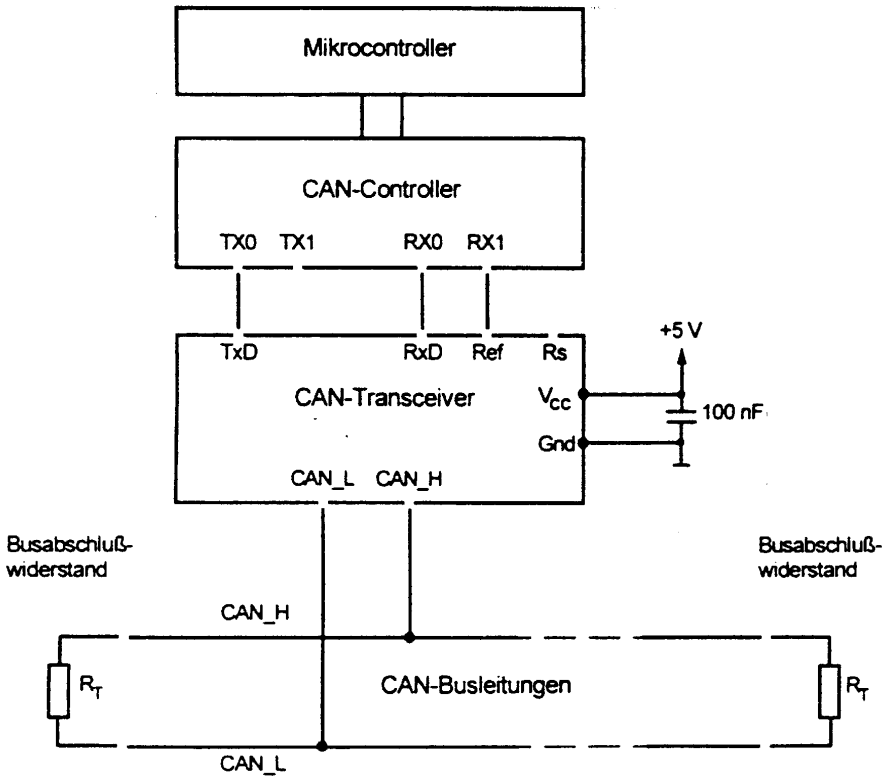
Bestimmt man die Restfehler-Wahrscheinlichkeit von CAN in Abhängigkeit von der Bitfehler-Wahrscheinlichkeit für Botschaftslängen von 80 bis 90 Bits, für Systemkonfigurationen mit z.B. fünf oder zehn Knoten und bei einer Fehlerrate von 1/1000 (jede tausendste Botschaft ist gestört), so nimmt sie bei einer Bitfehler-Wahrscheinlichkeit von ungefähr 0,02 ihren Maximalwert an, der in der Größenordnung von 10^{-13} liegt. Geht man von diesem Maximalwert aus, so läßt sich daraus für ein gegebenes CAN-Netz die maximale Anzahl von nicht erkennbaren Fehlern berechnen.

Wird beispielsweise ein CAN-Netz mit einer Datenrate von 1 MBit/s betrieben und beträgt die mittlere Busauslastung 50 Prozent, die Gesamtdauer des Betriebes 4000 Stunden und sind die Botschaften im Mittel 80 Bit lang, so berechnet sich die Gesamtzahl übertragener Botschaften zu 9 x 1010.
Die statistische Anzahl unerkannter Übertragungsfehler während der gesamten Betriebsdauer liegt damit in der Größenordnung kleiner 10^{-2}. Oder anders ausgedrückt, bei einer Betriebszeit von acht Stunden pro Tag an 365 Tagen im Jahr und einer Fehlerrate von 0,7 s tritt im statistischen Mittel in 1000 Jahren ein unentdeckter Fehler auf.

Physikalische Schnittstelle

Aufgrund der Datenraten (bis zu 1 MBit/s) sind entsprechende Flankensteilheiten erforderlich, die sich nur mit Leistungsbauelementen realisieren lassen.
Prinzipiell sind verschiedene physikalische Schnittstellen möglich. Die Anwender- und Herstellervereinigung “CAN in Automation” (CiA) empfiehlt jedoch die Verwendung von Treiberschaltungen entsprechend ISO 11898.

Integrierte Treiberchips nach ISO 11898 sind von mehreren Herstellern verfügbar (Bosch, Philips, Siliconix und Texas Instruments). Darüber hinaus spezifiziert die CiA verschiedene mechanische Schnittstellen (Kabel und Steckverbinder).



Physikalische CAN-Schnittstelle nach ISO 11898

CAN-Botschaften im Erweiterten-Format

Das CAN-Protokoll erlaubt die Verwendung zweier Botschaftsformate:

Standard-CAN (Version 2.0 A) und
Extended-CAN (Version 2.0 B).

Für CAN-Botschaften im erweiterten Format wird das CAN-Protokoll durch Einführung eines 29-Bit-ldentifiers erweitert. Dieser 29-Bit-Identifier setzt sich aus dem existierenden 11-Bit-Identifier (Base-ID) und einer 18-Bit-Erweiterung (ID-Extension) zusammen.

Da beide Formate auf einem Bus koexistieren müssen, ist festgelegt, welche Botschaft bei Buszugriffskollisionen mit unterschiedlichem Format und gleichem Basis-Identifier die höhere Priorität auf dem Bus hat. Danach hat grundsätzlich die Botschaft im Standard eine höhere Priorität als die Botschaft im Erweiterten-Format.
Außerdem ist gewährleistet, dass CAN-Controller, die das Erweiterte Format unterstützen, auch Botschaften im Standard-Format senden und empfangen können.

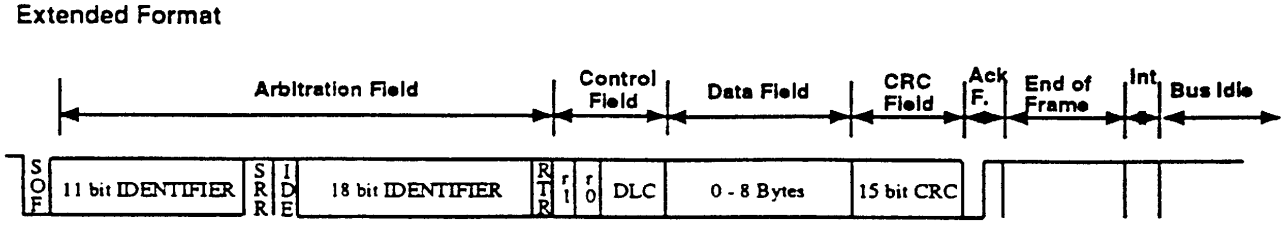
Wenn in einem Netz CAN-Controller verwendet werden, die nur das Standard Format beherrschen (Version 2.0 A), dürfen im gesamten Netz nur Botschaften im Standard-Format übertragen werden. Botschaften im Erweiterten-Format würden missverstanden werden.

Es gibt aber auch CAN-Controller, die nur das Standard Format unterstützen, aber Botschaften im Erweiterten-Format erkennen können und dann ignorieren (Version 2.0 B passiv).
Die Unterscheidung zwischen dem Standard-Format und dem Erweiterten-Format erfolgt mit Hilfe des **IDE-Bit** (Identifier Extension Bit), das im Falle eines Rahmens im Standard-Format dominant gesendet wird. Bei Rahmen im Erweiterten-Format ist es rezessiv.

Das **RTR-Bit** wird dominant oder rezessiv gesendet, je nachdem, ob Daten gesendet werden oder von einer Station eine bestimmte Botschaft angefordert wird.

An Stelle des RTR-Bit im Standard-Format wird beim Rahmen mit Erweitertem ID das **SRR-Bit** (Substitute Remote Request) übertragen.
Das SRR-Bit wird ausschließlich rezessiv gesendet, um sicherzustellen, dass im Falle der Arbitrierung der Standard-Rahmen immer die Buszuteilung gegenüber einem erweiterten Rahmen erhält, wenn beide Botschaften den gleichen Basis-Identifier haben.

Im Unterschied zum Standard-Rahmen folgt beim Erweiterten-Format nach dem IDE-Bit die **18-Bit-ID-Erweiterung** sowie das **RTR-Bit** und ein reserviertes Bit (r1). Alle folgenden Felder sind identisch mit dem Standard-Format. Die Konformität der beiden Formate wird dadurch gewährleistet, dass die CAN-Controller, die das Erweiterte Format unterstützen, ebenso im Standard-Format kommunizieren können.



Botschaftsrahmen im Erweiterten-Format (CAN-Spezifikation 2.0B)

3.2 CAN-Fehler und -Fehlereingrenzung

Da CAN-Knoten in der Lage sind zwischen kurzen Störungen und permanentem Ausfall zu unterscheiden, können Fehler eingegrenzt werden.
Innerhalb jedes CAN-Knotens werden je ein 8-Bit Sende-Fehlerzähler und ein 8-Bit Empfangs-Fehlerzähler verwendet. Tritt einer der Fehlertypen CRC-Error, Stuff-Error, Form-Error, Bit-Error oder Acknowledgement (ACK) Error auf, wird der entsprechende Fehler-Zähler hochgezählt.
Wird dagegen das Senden oder Empfangen erfolgreich abgeschlossen, so wird der entsprechende Fehler-Zähler dekrementiert. Dauerhafte Ausfälle führen damit zu hohen Fehlerzahlen, kurze Störungen dagegen nur zu kleinen Fehlerzahlen, die im laufenden System wieder reduziert werden.

Abhängig vom Wert der Fehlerzähler nimmt der Knoten einen der drei Zustände *error active*, *error passive* oder *bus off* an.

error active der normale Betriebszustand des Knotens, beide Fehlerzähler sind kleiner als 128. In diesem Zustands nimmt der Knoten normal an der Kommunikation teil. Werden während der Kommunikation Fehler erkannt, wird ein ERROR ACTIVE FLAG, das aus 6 dominanten Bits besteht, gesendet. Dadurch wird die aktuelle Übertragung blockiert.

error passive einer der beiden Fehlerzähler übersteigt den Wert 127 und zeigt damit eine erhöhte Fehlerzahl für den Knoten an. Der Knoten nimmt weiterhin an der Kommunikation teil, muss aber nach dem Senden einer Nachricht etwas länger warten, bevor er eine neue Nachricht senden kann. Diese zusätzliche Zeitverzögerung im *error passive* Modus wird *Suspended Transmission* genannt und wird durch das Senden von 8 zusätzlichen rezessiven Bits am Ende des Frames erreicht. Das bedeutet, ein Knoten im Zustand *error passive* verliert die Buskontrolle gegenüber Knoten im *error active* Modus unabhängig von der Rangfolge von deren IDs.
Entdeckt ein *error passive* Knoten während der Kommunikation einen Fehler, wird dieser über das Senden eines ERROR PASSIVE FLAG angezeigt. Dieses Flag besteht aus 6 rezessiven Bits, die die aktuelle Übertragung nicht beeinflussen (vorausgesetzt ein anderer Knoten ist der Sender) wenn sich herausstellt, dass sich der Fehler lokal auf dem *error passive* Knoten befindet.

bus off der Sende-Fehlerzähler hat den Wert 255 überstiegen und zeigt damit an, dass an dem Knoten beim Senden über längere Zeit Fehler aufgetreten sind.
In diesem Zustand schaltet der Knoten seine Bus-Treiber aus und hat damit keinen Einfluss mehr auf den Bus. Am Knoten wird das Senden wieder aktiviert und der Knoten wird wieder *error active*, wenn 128 mal das Auftreten von 11 aufeinanderfolgenden rezessiven Bits, auf dem Bus festgestellt wurde.

4. Funktionsbeschreibung

4.1 Funktionsbaustein FB77

Der Funktionsbaustein FB77 übernimmt die Kommunikation über das nach Kapitel 2 konfigurierte EtherCAN-S7-Modul mit dem angeschlossenen CAN-Bus. Die notwendige Konfiguration der Bitrate auf dem CAN-Bus, die Vorbereitung des Empfangs von CAN-Nachrichten und der sogenannte ELLSI-Heartbeat zur Überwachung der Verbindung zwischen Siemens-SPS S7 und EtherCAN-S7 erfolgt selbständig im FB77.
Bei einem Aufruf des FB77 können bis zu 7 CAN-Telegramme verschickt und empfangen werden.

Der FB77 verwendet die im Folgenden aufgelisteten IN-Parameter und OUT-Parameter.

IN-Parameter

Über die IN-Parameter ENABLE, CONN_ID, MOD_ADDR und BAUD konfiguriert der FB77 das EtherCAN-S7-Modul.
Die IN-Parameter SEND_INIT, SEND_LEN und SEND_DATA initiieren das Versenden von CAN-Telegrammen.

IN-Parameter	Datentyp	Beschreibung
ENABLE	BOOL	FALSE -> reset FB77, TRUE -> run
CONN_ID	integer	ID der UDP-Verbindung (siehe Eigenschaften der UDP-Verbindung in NetPro, Abb.16, Seite 34)
MOD_ADDR	WORD	Modul-Adresse der CP in der Hardware-Konfiguration (siehe Abb.16) Beispiel : Die E-/A-Adressen der CP werden vom SIMATIC-Manager (siehe Abb.4, Seite 9) auf Steckplatz 4 angezeigt
BAUD	DWORD	CAN Bitrate (siehe Kapitel 4.1.2)
SEND_INIT	BOOL	steigende Flanke -> sende SEND_DATA (Länge SEND_LEN Bytes)
SEND_LEN	DWORD	Länge der Daten in SEND_DATA
SEND_DATA	BLOCK_DB	Daten, die gesendet werden sollen (entweder DB1 zum Senden von CAN-Frames oder DB2 zum Aktivieren/Deaktivieren von CAN-Identifiern)

OUT-Parameter

Die OUT-Parameter CONNECT, ERROR und STATUS dienen der Überwachung des CAN-Bus.
Eine Rückmeldung, ob das Versenden von CAN-Telegrammen beendet wurde, erfolgt über SEND_DONE.
Zur Übergabe empfangener CAN-Telegramme werden die OUT-Parameter REC_NEW, REC_LEN und REC_DATA genutzt.

OUT-Parameter	**Datentyp**	**Beschreibung**
CONNECT	BOOL	TRUE -> aktive Verbindung zum EtherCAN-S7 FALSE -> keine aktive Verbindung zum EtherCAN-S7
ERROR	BOOL	TRUE -> Error (für weitere Details siehe STATUS) FALSE -> kein Fehler
STATUS	WORD	Rückgabewert von AG_SEND/RECV (Eine Auflistung der Rückgabewerte befindet sich im Anhang, Kapitel 6.2)
SEND_DONE	BOOL	TRUE -> vorheriges Datenpaket wurde gesendet FALSE -> es wurde noch nichts gesendet oder es wurde noch nichts beauftragt
REC_NEW	BOOL	TRUE -> es gibt neue Daten FALSE -> es gibt keine neuen Daten
REC_LEN	DWORD	Länge der empfangenen Daten in REC_DATA, wenn es neue Daten gibt (REC_NEW -> True)
REC_DATA	ARRAY 1..7 of UDT1	empfangene Daten (siehe Kapitel 4.2)

4.1.1 Parameter CONN_ID und MOD_ADDR

Bei der Konfiguration der Verbindung des Netzwerkes mit NetPro, wie in Kapitel 2 beschrieben, werden die *Bausteinparameter* CONN_ID und MOD_ADDR im Fenster *Eigenschaften* der UDP-Adresse angezeigt.

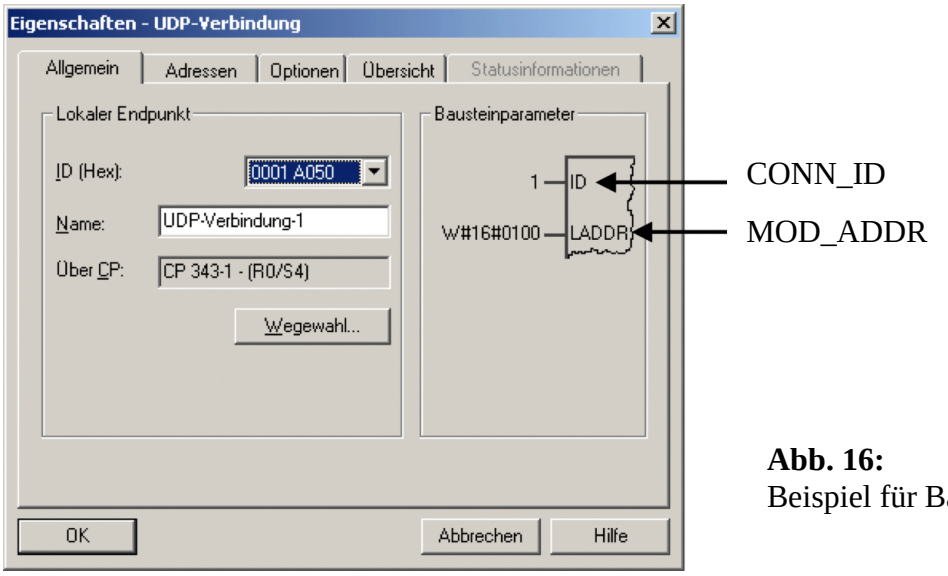


Abb. 16:
Beispiel für Bausteinparameter

Diese Parameter müssen in die IN-Parameter CONN_ID und MOD_ADDR eingetragen werden.

4.1.2 Konfiguration der CAN-Bitrate mit BAUD

Bevor Daten gesendet oder empfangen werden können, muss die Baudrate der CAN-Schnittstelle einmalig initialisiert werden. Diese Initialisierung ist dann für diese logische ID der UDP-Verbindung und somit für die zugeordnete physikalische Schnittstelle gültig.
Die Baudrate wird bei einer Flanke ENABLE FALSE →TRUE aus dem Parameter **BAUD** übernommen und eingestellt.

Die Struktur des 32-Bit IN-Parameters **BAUD** ist abhängig vom Wert des 'User Bit Rate' Flag (UBR) und wird im Folgenden definiert:

Bit-Nr.:	**31**	**30**	**29... ...24**	**23... ...16**	**15... ...8**	**7... ...0**
Belegung:	*UBR* = 0	*LOM*	Reserviert			*Baudrate index*
	UBR = 1	*LOM*	Reserviert		*BTR0*	*BTR1*

Tabelle 1: Struktur des IN-Parameters **BAUD**

Bit(s)	**Werte**	**Beschreibung**
UBR	0	Verwenden Sie die vordefinierte Bitratentabelle (*Table index*)
	1	Setzen Sie das Bitratenregister des CAN-Controllers direkt (*BTR0/BTR1*)
LOM		Listen-Only Mode:
	0	Konfigurieren Sie die Bitrate im 'active Modus' (Normal Operation)
	1	Konfigurieren Sie die Bitrate im Listen-Only-Modus
Baudrate index	x	Verwenden Sie die Bitrate der vordefinierten Tabelle von Seite 36
BTR0/BTR1	x	Bit-Timing-Register des CAN-Controllers

Tabelle 2: Beschreibung des IN-Parameters **BAUD**

UBR

Wird das '**User Bit Rate' Flag (UBR) auf '0'** gesetzt, werden die Bits 0...7 als Index einer vordefinierten Bitratentabelle ausgewertet. So können gebräuchliche CAN-Bitraten ohne Wissen über Hardware-Details des CAN-Controllers konfiguriert werden.

Baudrate index [hex]	Bitrate *1) [kBit/s]
0	1000
1	666.6
2	500
3	333.3
4	250
5	166
6	125
7	100
8	66.6
9	50
A	33.3
B	20
C	12.5
D	10

Tabelle 3:
Vordefinierte Bitratentabelle

*1) Die Konfiguration der vordefinierten Bitraten entspricht den Empfehlungen der CiA (CAN in Automation e.V.).

Wird das **UBR Flag auf '1'** gesetzt, werden die Bits 0...15 verwendet um das Bitratenregister des CAN-Controllers direkt mit den vorgegebenen Werten zu konfigurieren.
Bei der direkten Definition des Bitratenregisters ist die eingesetzte Hardware, (CAN-Controller, Taktfrequenz) zu berücksichtigen. Siehe Anhang 'Bit-Timing Werte (Beispiele)'.

LOM

Wird das '**Listen-Only-Modus' (LOM)-Flag auf '0'** gesetzt, arbeitet der CAN-Controller im normalen Active-Modus mit dieser Bitrate, was bedeutet, dass Nachrichten empfangen und gesendet werden können.

Wird das **LOM Flag auf '1'** gesetzt, arbeitet der CAN-Controller im Listen-Only-Modus und kann ausschließlich Nachrichten empfangen.
Es wird kein CAN-ACK generiert (siehe Seite 23). Auch das Senden von CAN-Nachrichten ist mit gesetztem LOM-Flag nicht möglich. Dieser Modus kann im Wesentlichen zu Diagnosezwecken eines CAN-Netzes eingesetzt werden.

4.2 Datentyp UDT1

Der Datentyp **UDT1** dient als Schnittstelle zum Versenden und Empfangen von CAN-Telegrammen. Er enthält die Information über den Identifier, die Länge und die Daten des CAN-Telegramms. Außerdem werden beim Empfang auch noch die Anzahl eventuell im EtherCAN-S7 verloren gegangener CAN-Telegramme und auch generelle Störungen des CAN-Busses (siehe Kapitel 4.2.2) angezeigt. Bei den CAN-Telegrammen lässt sich dafür zwischen CAN-Nachrichten (CMSG) und CAN-Ereignissen (EVMSG), die durch den Event-Identifier gekennzeichnet sind, unterscheiden.

Die Datenstruktur der CAN-Nachrichten (CMSG) ist in Kapitel 4.2.1 detailliert beschrieben.
Die Datenstruktur der CAN-Ereignisse (EVMSG) ist in Kapitel 4.2.2 detailliert beschrieben.

4.2.1 UDT1 bei CAN-Nachrichten (CMSG)

Dieses Kapitel enthält eine ausführliche Beschreibung des Datentyps **UDT1 (CMSG)**:

UDT1	Datentyp	Beschreibung
ID	DWORD	CAN-Identifier, 11- oder 29-bit CAN ID: Daten werden darauf empfangen oder Daten sollen darauf gesendet werden
len	BYTE	0..8 (+ b#16#10, entsprechend $+10_h$, für RTR), Bit 0-3: Anzahl der CAN Datenbytes [0..8] Bit 4: RTR (Remote Transmission Request) Das RTR-Bit kennzeichnet, ob es sich um einen Datenrahmen oder einen Anforderungsrahmen ohne Datenbytes (Remote Request) handelt. Bit 5-7: Reserviert
msg_lost	BYTE	nur für Empfang, Zähler für verloren gegangene CAN Empfangs Nachrichten. Ermöglicht es dem Nutzer Datenverluste im EtherCAN-S7 zu erfassen: msg_lost = 0 : keine verlorenen Nachrichten 0 < msg_lost < 255 : Anzahl verlorener Nachrichten = Wert von msg_lost msg_lost = 255 : Anzahl verlorener Nachrichten ≥ 255
reserved	WORD	reserviert, Rückgabe von ‘0’
data	Array 1..8 of Byte	CAN-Datenbytes
timestamp1	DWORD	reserviert für zukünftige Anwendungen, Rückgabe von ‘0’
timestamp2	DWORD	reserviert für zukünftige Anwendungen, Rückgabe von ‘0’

Tabelle 4: UDT1 (CMSG)

CAN-Message (CMSG) Datenstruktur

ID: Um eine Nachricht mit einem 29-Bit-Identifier zu senden, muss das Bit 29 des CAN-Identifier-Parameters **ID** in der Struktur CMSG zusätzlich zum CAN-Identifier gesetzt werden. Wird eine Nachricht mit einem 29-Bit-Identifier empfangen, so ist das Bit 29 des CAN-Identifier-Parameters **ID** in der Struktur CMSG zusätzlich zum CAN-Identifier gesetzt.

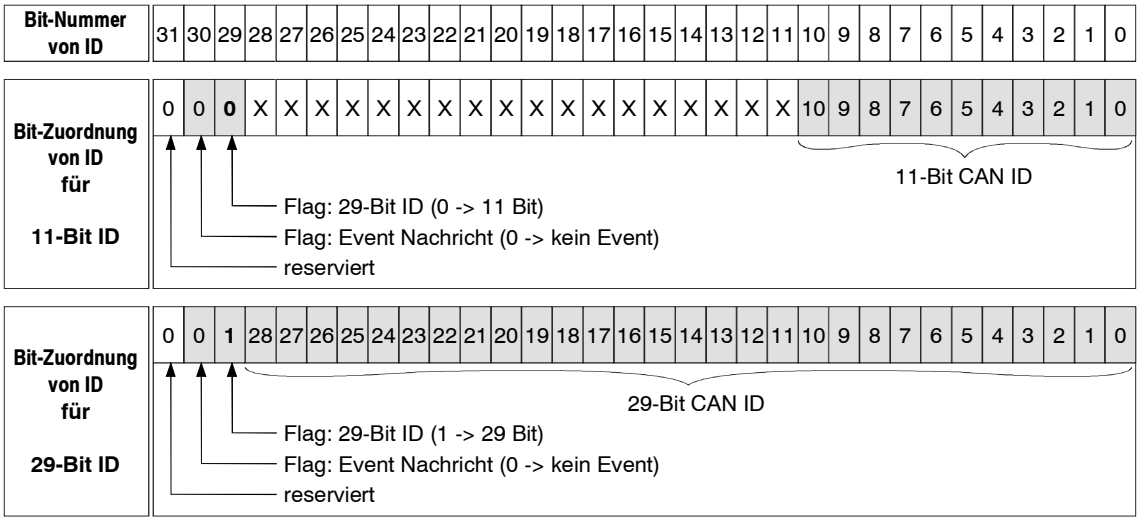


Tabelle 5: Kodierung von **ID** in CMSG

Für CAN-Nachrichten ist Bit 30 des CAN-Identifier-Parameters **ID** immer auf ‘0’ gesetzt.

len:

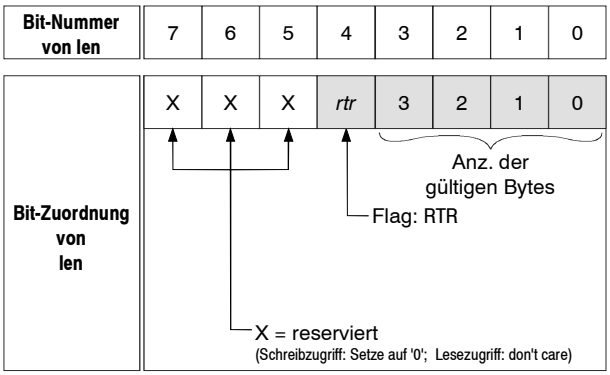


Tabelle 6: Kodierung von **len** in CSMG Datenstruktur

Die Bits 0 bis 3 des Längenfeldes **len** der Struktur CMSG werden verwendet, um die Zahl der gültigen Bytes im Datenfeld des empfangenen oder zu sendenden CAN-Datentelegramms entsprechend nachfolgender Tabelle anzuzeigen.

Wert von **len** [binär] Bit7... ...Bit0	Anzahl der Datenbytes [Bytes]
xxxx 0000	0
xxxx 0001	1
xxxx 0010	2
xxxx 0011	3
xxxx 0100	4
xxxx 0101	5
xxxx 0110	6
xxxx 0111	7
xxxx 1000	8
xxxx 1xxx	8*

Tabelle 7: Kodierung der Länge

* Gemäß CiA Standard sind für **len** Werte > 8 erlaubt

Bit 4 (RTR) des Längenfeldes **len** der Struktur CMSG wird verwendet, um beim Senden und Empfangen des CAN-Telegramms ein Datentelegramm (Data Frame) von einem Datenanforderungstelegramm (Remote Frame), zu unterscheiden:

Wert des Bits **RTR** [binär]	Funktion
0	Datentelegramm senden oder empfangen
1	Datenanforderungstelegramm (RTR) senden oder empfangen

Tabelle 8: Funktion des Bits **RTR**

Im Falle eines Datenanforderungstelegramms repräsentieren die Bits 0 bis 3 des Längenfeldes den Datenlängen-Code entsprechend der Tabelle 5 auf Seite 38. Die Daten **data** der Struktur CMSG sind ungültig.
Die Bits 5, 6 und 7 sind für zukünftige Anwendungen reserviert. Bei Leseoperationen ist der Zustand dieser Bits undefiniert. Bei Schreiboperationen sollten diese Bits immer auf ‘0’ gesetzt werden.

msg_lost:

Ist das Receive-FIFO des EtherCAN-S7 voll und neue Nachrichten treffen ein, so werden alte Nachrichten überschrieben und der **msg_lost**-Zähler wird hochgezählt.
Der Anwender kann mit Hilfe des **msg_lost**-Zählers einen Datenüberlauf erkennen. Ein ansteigender Zählerwert signalisiert, dass das Anwendungsprogramm den CAN-Datenstrom langsamer liest, als der Treiber neue Daten bereitstellt.

Wert des Message-Lost-Zählers	Bedeutung
msg_lost = 0	keine verlorenen Nachrichten
0 < **msg_lost** < 255	Anzahl verlorener Nachrichten = Wert von msg_lost
msg_lost = 255	Anzahl verlorener Nachrichten ≥ 255

Tabelle 9: Wertebereich von **msg_lost**

4.2.2 UDT1 bei CAN-Ereignissen (EVMSG)

Dieses Kapitel enthält eine ausführliche Beschreibung von UDT1 bei Ereignissen (Events) auf dem CAN-Bus (EVMSG).

Neben den Fehler-Rückgabewerten kann der Funktionsbaustein asynchron über Events signalisieren, dass ein Fehler und/oder eine Statusänderung, wie z.B. ein CAN-Bus-Off aufgetreten ist.

Datentyp **UDT1 (EVMSG)**:

UDT1	Datentyp	Beschreibung
ID	DWORD	Event-Identifier
len	BYTE	Bit 0-3: Anzahl der CAN Datenbytes [0..8] Bit 4-7: Reserviert
msg_lost	BYTE	nur für Empfang, Zähler für verloren gegangene CAN Empfangs-Nachrichten; Ermöglicht es dem Nutzer Datenverluste zu erfassen: msg_lost = 0 : keine verlorenen Nachrichten 0 < msg_lost < 255 : Anzahl verlorener Nachrichten = Wert von msg_lost msg_lost = 255 : Anzahl verlorener Nachrichten ≥ 255
reserved	WORD	reserviert, Rückgabe von ‘0’
data	Array 1..8 of Byte	Event-Daten
timestamp1	DWORD	reserviert für zukünftige Anwendungen, Rückgabe von ‘0’
timestamp2	DWORD	reserviert für zukünftige Anwendungen, Rückgabe von ‘0’

Tabelle 10: UDT1 (EVMSG)

Event Message (EVMSG) Datenstruktur

Die Größe in Bytes der Struktur EVMSG ist die gleiche wie die Größe der Struktur CMSG.

ID: Events werden immer durch das auf ‘1’ gesetzte Bit 30 des Parameters **ID** gekennzeichnet. So können sie von den vorausgehend beschriebenen CAN 2.0A und CAN 2.0B Nachrichten der CMSG-Struktur unterschieden werden, wenn sie mit einem Standard Empfangs-I/O-Aufruf zusammen mit CAN-Nachrichten zurückgegeben werden.

Für die 8-Bit **Event-ID** wird derzeit nur die Event-Nummer 4000 0000_h verwendet.

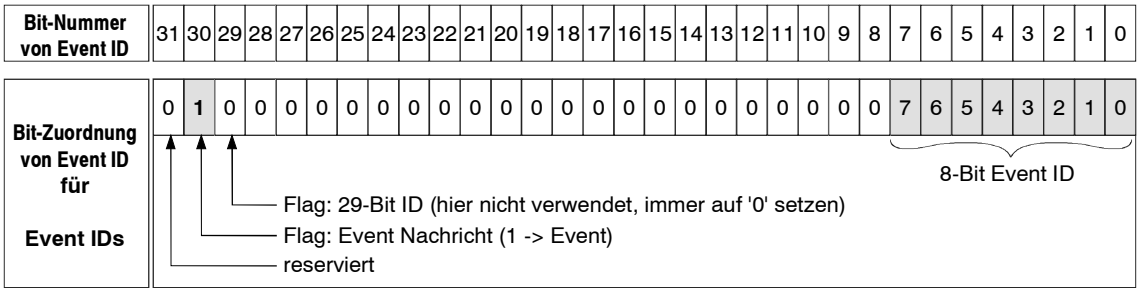


Tabelle 11: Kodierung von **ID** in EVMSG

len:

Wert von **len** [binär] Bit7... ...Bit0	Anzahl der Datenbytes [Bytes]
xxxx 0000	0
xxxx 0001	1
xxxx 0010	2
xxxx 0011	3
xxxx 0100	4
xxxx 0101	5
xxxx 0110	6
xxxx 0111	7
xxxx 1000	8

Tabelle 12: Kodierung der Länge **len** in EVMSG Datenstruktur

Die Bits 7...4 sind für zukünftige Applikationen reserviert. Für den Lesezugriff sind die Werte dieser Bits undefiniert. Für den Schreibzugriff müssen diese Bits auf ‘0’ gesetzt werden.

data: Das Datenfeld **data** enthält in der EVMSG-Datenstruktur die Event-Daten.
Derzeit gibt es das folgende CAN-Event:

EV_CAN_ERROR

Event-ID: EV_CAN_ERROR
Event-Nr.: 4000 0000_h
Daten-Länge: 8 Bytes

Daten-Byte-Nr.:	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
Belegung:	reserviert (Rückgabe von '0')	*Error*	reserviert (Rückgabe von '0')	*LOST_ MESSAGE*	reserviert (Rückgabe von '0')	*LOST_ FIFO*	reserviert (Rückgabe von '0')	reserviert (Rückgabe von '0')

Tabelle 13: Belegung des Events EV_CAN_ERROR

Das Event wird erzeugt, wenn sich einer der Fehler-Parameter, die von diesem Event abgedeckt werden, ändert.

Parameter-Wertebereiche:

ERROR
- 00_h - CAN-Controller-Status ist *ErrorActive* (BUSSTATE_OK)
- 40_h - CAN-Controller-Status ist *Error Warn* (BUSSTATE_WARN)
  Einer der Fehler-Zähler (Rx oder Tx) ist gößer als 95.
- 80_h - CAN-Controller-Status ist *Error Passive* (BUSSTATE_ERRPASSIVE)
  Einer der Fehler-Zähler (Rx oder Tx) ist gößer als 127.
- $C0_h$ - CAN-Controller-Status ist *Bus Off* (BUSSTATE_BUSOFF)
  Einer der Fehler-Zähler (Rx oder Tx) ist gößer als 255

LOST_MESSAGE Zähler für verlorengegangene Nachrichten des CAN-Controllers
Dieser Zähler wird von einem Fehler-Ausgang des CAN-Controllers gesetzt. Er zeigt die Anzahl der verlorengegangenen CAN-Frames an (Empfangs- oder Sende-Nachrichten).

LOST_FIFO Zähler für verlorengegangene Nachrichten des FIFOs
Dieser Zähler wird erhöht, wenn Nachrichten aufgrund eines FIFO-Überlaufs verloren gehen (FIFO full).

4.3 Datenbaustein DB1

Aufbau des Datenbausteins **DB1** zum Versenden von CAN-Telegrammen:

DB1	Datentyp	Beschreibung
command	BYTE	immer 1 zum Senden von CAN-Nachrichten
sendData	ARRAY 1..7 of UDT1	1..7 CAN-Nachrichten

Tabelle 14: Datenbaustein **DB1**

4.4 Datenbaustein DB2

Zur applikations-spezifischen Selektion der empfangenen Nachrichten auf Basis der CAN-Identifier kann ein individueller Filter für 11-Bit CAN-Identifier (Standard-Format) konfiguriert werden.
Der Funktionsbaustein FB77 öffnet bei der Initialisierung einen Filter für alle 11-Bit CAN-Identifier. Einzelne Identifier oder Bereiche können unter Verwendung des Datenbausteins DB2 ausgeblendet werden.
Die CAN-Identifier der zu sendenden Nachrichten müssen zuvor nicht explizit bekannt gemacht werden.

Aufbau des Datenbausteins **DB2** zum Freigeben/Sperren von CAN-Identifiern für den Empfang:

DB2	**Datentyp**	**Beschreibung**	
command	BYTE	= 2 CAN_ID_ADD (Freigeben von CAN-Identifiern) = 3 CAN_ID_DELETE (Sperren von CAN-Identifiern)	
CanIdRange	ARRAY 1..21 of rangeStart (DWORD), rangeEnd (DWORD)	rangeStart (DWORD)	DW#16#0 .. DW#16#7FF (entsprechend 0...+7FF$_h$) für den Anfang des Bereichs
		rangeEnd (DWORD)	DW#16#0 .. DW#16#7FF (entsprechend 0...+7FF$_h$) für das Ende des Bereichs

Tabelle 15: Datenbaustein **DB2**

4.4.1 CanIdRange

Variable	Datentyp	Beschreibung
rangeStart	DWORD	Beginn des Bereichs der CAN ID(s), die für den Empfang freigegeben werden / gesperrt werden
rangeEnd	DWORD	Ende des Bereichs der CAN ID(s), die für den Empfang freigegeben werden / gesperrt werden

Tabelle 16: Bereich der CAN ID(s)

Der gesamte Bereich, einschließlich **rangeStart** und **rangeEnd**, wird in Abhängigkeit von **command** freigegeben oder gesperrt.
Ist **rangeEnd** kleiner oder gleich **rangeStart**, wird nur die CAN ID freigegeben oder gesperrt, die durch **rangeStart** festgelegt ist.

5. Überwachung der Ethernet Kommunikation mit Heartbeat

Dieser Dienst (ELLSI_CMD_HEARTBEAT) dient ausschließlich der Überwachung der Verbindung zwischen Siemens-SPS S7 und EtherCAN-S7 und ist nicht mit einem Heartbeat oder Guarding der CAN- oder CANopen-Baugruppen zu verwechseln.
Er wird vom FB77 automatisch bearbeitet und führt bei Störung zu CONNECT = FALSE.

Siemens-SPS S7 (ELLSI-Client) und EtherCAN-S7 (ELLSI-Server) senden in regelmäßigen Intervallen Heartbeat-Nachrichten, wenn kein Datenaustausch stattfindet. Derzeit ist dieses Heartbeat-Intervall auf 2500 ms festgelegt.

Hat der Client (Siemens-SPS S7) innerhalb eines Zeitraumes von 7500 ms weder Daten noch Heartbeats vom Server (EtherCAN-S7) erhalten, schließt der Client daraus, dass der Server nicht mehr vorhanden ist. Beispielsweise wurde die Netzwerkverbindung unterbrochen oder es wurde ein Reset auf dem EtherCAN-S7 ausgeführt, etc. Resultierend daraus versucht der Client (Siemens-SPS S7) nun erneut, sich wieder mit dem Server zu verbinden.

Empfängt der Server (EtherCAN-S7) innerhalb eines Zeitraumes von 7500 ms weder Daten noch Heartbeats vom Client (Siemens-SPS S7), schließt er daraus, dass der Client nicht mehr vorhanden ist. Der Server sendet dann keine Daten und Heartbeats an den Client mehr. Erst wenn sich der Client (Siemens-SPS S7) wieder mit dem Server verbunden hat und eine Heartbeat-Nachricht aussendet, beginnt der Server wieder Heartbeats zu senden.

6. Anhang

6.1 Bit-Timing-Werte (Beispiele)

Über die Bestimmung des Bit-Timings und der Baudrate mit Hilfe der Registerwerte informieren Sie sich bitte in dem Handbuch des Controllers SJA1000 (NXP).

Das SJA1000-Handbuch kann aus dem Internet heruntergeladen werden, zum Beispiel von der NXP Homepage unter:

`http://www.nxp.com/acrobat_download/datasheets/SJA1000_3.pdf`

Hinweis: Bitte beachten Sie, dass für ein lauffähiges CAN-Netzwerk nicht nur alle CAN-Teilnehmer über die selbe Bitrate verfügen müssen, sondern auch die anderen Timing-Parameter identisch sein müssen!

Baudrate Buslänge	Nominelle Bit-Time t_B	Position des Sample Point *) [% of t_B]	Setzen der Register BTR0 und BTR1 SJA1000/20 MHz [HEX]
1 Mbit/s 25 m	1 µs	75 %	00 16
800 kBit/s 50 m	1,25 µs	80 %	00 18
500 kBit/s 100 m	2 µs	87,5 %	00 2F
250 kBit/s 250 m	4 µs	87,5 %	01 2F
125 kBit/s 500 m	8 µs	87,5 %	04 1C
100 kBit/s 650 m	10 µs	87,5 %	04 2F
50 kBit/s 1 km	20 µs	87,5 %	09 2F
20 kBit/s 2,5 km	50 µs	87,5 %	18 2F
10 kBit/s 5 km	100 µs	87,5 %	31 2F

*) Die ‘Position des Sampling Point’ ist entsprechend der CiA Empfehlungen gewählt worden.

Tabelle: Bit-Timing Werte

6.2 S7-Rückgabewerte

Die Rückgabewerte wurden aus der Siemens-SPS S7-Online Dokumentation übernommen.

FC5 (AG_SEND):
Die folgende Tabelle informiert über die vom Anwenderprogramm auszuwertende Anzeige, gebildet aus DONE, ERROR und STATUS.

DONE	ERROR	STATUS	Bedeutung
1	0	0000H	Auftrag fertig ohne Fehler.
0	0	0000H	Kein Auftrag in Bearbeitung.
0	0	8181H	Auftrag läuft.
0	1	7000H	Die Anzeige ist nur bei S7-400 möglich: Der FC wurde mit ACT=0 aufgerufen; der Auftrag wird jedoch nicht bearbeitet.
0	1	8183H	Die Projektierung fehlt oder der Dienst im Ethernet-CP ist noch nicht gestartet.
0	1	8184H	Unzulässiger Datentyp für den Parameter SEND angegeben. Systemfehler (Der Quelldatenbereich ist fehlerhaft).
0	1	8185H	Parameter LEN größer als Quell-Bereich SEND.
0	1	8186H	Parameter ID ungültig. ID != 1,2....64.
0	1	8302H	keine Empfangsresourcen bei Ziel-Station, Empfänger-Station kann empfangene Daten nicht schnell genug verarbeiten bzw. hat kein Empfangsresourcen bereitgestellt.
0	1	8304H	Die Verbindung ist nicht aufgebaut. Der Sendeauftrag sollte erst nach einer Wartezeit >100 ms erneut abgesetzt werden.
0	1	8311H	Zielstation ist unter der angegebenen Ethernet-Adresse nicht erreichbar.
0	1	8312H	Ethernet-Fehler im CP.
0	1	8F22H	Quell-Bereich ungültig. z.B.: Bereich im DB nicht vorhanden Parameter LEN < 0
0	1	8F24H	Bereichsfehler beim Lesen eines Parameters.
0	1	8F28H	Ausrichtungsfehler beim Lesen eines Parameters.
0	1	8F32H	Parameter enthält zu große DB-Nummer.
0	1	8F33H	DB-Nummer Fehler.
0	1	8F3AH	Bereich nicht geladen (DB).
0	1	8F42H	Quittungsverzug beim Lesen eines Parameters aus dem Peripheriebereich.
0	1	8F44H	Der Zugriff auf einen in der Bausteinbearbeitung zu lesenden Parameter ist gesperrt.
0	1	8F7FH	Interner Fehler. z.B. unzulässige ANY-Referenz z.B. Parameter LEN = 0 .
0	1	8090H	Baugruppe mit dieser Baugruppen-Anfangsadresse nicht vorhanden; Der verwendete FC passt nicht zur verwendeten Systemfamilie (es sind unterschiedliche FCs für S7-300 und S7-400 zu verwenden).
0	1	8091H	Baugruppen-Anfangsadresse nicht auf Doppel-Wort-Raster.
0	1	8092H	In ANY-Referenz ist eine Typangabe ungleich BYTE angegeben. (nur bei S7-400)
0	1	80A4H	Die K-Busverbindung zwischen CPU und CP ist nicht aufgebaut. (bei neueren CPU-Ausgabeständen)
0	1	80B0H	Baugruppe kennt den Datensatz nicht.
0	1	80B1H	Die Längenangabe (im Parameter LEN) ist falsch.
0	1	80B2H	Die K-Busverbindung zwischen CPU und CP ist nicht aufgebaut.
0	1	80C0H	Datensatz kann nicht gelesen werden.
0	1	80C1H	Der angegebene Datensatz ist gerade in Bearbeitung.
0	1	80C2H	Es liegt ein Auftragsstau vor.
0	1	80C3H	Die Betriebsmittel (Speicher) der CPU sind temporär belegt.
0	1	80C4H	Kommunikationsfehler (tritt temporär auf; daher ist eine Wiederholung im Anwenderprogramm sinnvoll.)
0	1	80D2H	Baugruppen-Anfangsadresse ist falsch.

FC6 (AG_RECV):

Die folgende Tabelle informiert über die vom Anwenderprogramm auszuwertende Anzeige, gebildet aus NDR, ERROR und STATUS.

NDR	ERROR	STATUS	Bedeutung
1	0	0000H	Neue Daten übernommen.
0	0	8180H	Es liegen noch keine Daten vor.
0	0	8181H	Auftrag läuft.
0	1	8183H	Die Projektierung fehlt oder der ISO-Transport-Dienst im Ethernet-CP ist noch nicht gestartet.
0	1	8184H	Unzulässiger Datentyp für den Parameter RECV angegeben. Systemfehler.
0	1	8185H	Ziel-Puffer (RECV) ist zu klein.
0	1	8186H	Parameter ID ungültig. ID != 1,2....16 (S7-300). ID != 1,2....64.(S7-400)
0	1	8304H	Die Verbindung ist nicht aufgebaut. Der Empfangsauftrag sollte erst nach einer Wartezeit >100 ms erneut abgesetzt werden.
0	1	8F23H	Quell-Bereich ungültig. z.B.: Bereich im DB nicht vorhanden.
0	1	8F25H	Bereichsfehler beim Schreiben eines Parameters.
0	1	8F29H	Ausrichtungfehler beim Schreiben eines Parameters
0	1	8F30H	Parameter liegt im schreibgeschützten 1. akt. Datenbaustein.
0	1	8F31H	Parameter liegt im schreibgeschützten 2. akt. Datenbaustein.
0	1	8F32H	Parameter enthält zu große DB-Nummer.
0	1	8F33H	DB-Nummer Fehler.
0	1	8F3AH	Zielbereich nicht geladen (DB).
0	1	8F43H	Quittungsverzug beim Schreiben eines Parameters in den Peripheriebereich.
0	1	8F45H	Der Zugriff auf einen in der Bausteinbearbeitung zu schreibenden Parameter ist gesperrt.
0	1	8F7FH	Interner Fehler. z.B. unzulässige ANY-Referenz.
0	1	8090H	Baugruppe mit dieser Baugruppen-Anfangsadresse nicht vorhanden oder CPU in STOP. Der verwendete FC passt nicht zur verwendeten Systemfamilie (es sind unterschiedliche FCs für S7-300 und S7-400 zu verwenden)
0	1	8091H	Baugruppen-Anfangsadresse nicht auf Doppel-Wort-Raster.
0	1	8092H	In ANY-Referenz ist eine Typangabe ungleich BYTE angegeben. (nur bei S7-400)
0	1	80A0H	Negative Quittung beim Lesen von Baugruppe.
0	1	80A4H	Die K-Busverbindung zwischen CPU und CP ist nicht aufgebaut.
0	1	80B0H	Baugruppe kennt den Datensatz nicht.
0	1	80B1H	Der Zielbereich ist ungültig oder zu klein.
0	1	80B2H	Die K-Busverbindung zwischen CPU und CP ist nicht aufgebaut.
0	1	80C0H	Datensatz kann nicht gelesen werden.
0	1	80C1H	Der angegebene Datensatz ist gerade in Bearbeitung.
0	1	80C2H	Es liegt ein Auftragsstau vor.
0	1	80C3H	Die Betriebsmittel (Speicher) der CPU sind temporär belegt.
0	1	80C4H	Kommunikationsfehler (tritt temporär auf; daher ist eine Wiederholung im Anwenderprogramm sinnvoll.)
0	1	80D2H	Baugruppen-Anfangsadresse ist falsch.