

# Der CAN – Bus

## *Intelligente, dezentrale Datenkommunikation für den Praktiker (Teil 2)*

Nach der Beschreibung der Geschichte, der Normung und des grundsätzlichen Aufbaus des Controller Area Networks (besser bekannt als CAN-Bus) im ersten Teil steht der zweite Artikel ganz im Zeichen des Datenübertragungsprotokolls, das die Leistungsfähigkeit und Fehlersicherheit eines Feldbussystems wesentlich bestimmt.

Der wichtigste Teil der Busdefinition ist neben der Buskopplung (Physical Layer) das Protokoll zur Datenübertragung (Data Link Layer). Die grundlegende Bedeutung eines eindeutig genormten und anerkannten Datenübertragungsprotokolls soll an einem einfachen Beispiel klargestellt werden: Sie möchten Ihren Freund anrufen und ihm eine Nachricht übermitteln. Das "genormte, einheitliche" Datenübertragungsprotokoll für diese Aktion sieht bekanntermaßen wie folgt aus: Telefonhörer abnehmen und auf Freizeichen warten. Kein Freizeichen vorhanden: das Telefon (oder der Anschluß) ist defekt, eine Kommunikation kann nicht durchgeführt werden. Aufruf der "Fehlerbehandlungs-Routine", das heißt, Verständigung der Telefon-Service-Stelle. Wenn Freizeichen vorhanden: Nummer wählen und warten bis abgenommen wird. Wenn besetzt oder wenn nach mehrfachem Rufzeichen keine Reaktion am anderen Ende erfolgt: Hörer wieder auflegen und später erneut bei 1. beginnen. Wenn abgenommen wird, mit dem Freund sprechen und unbedingt beachten: immer nur abwechselnd sprechen und zuhören, sonst können keine Daten sinnvoll ausgetauscht werden. Bei Fehlern in der Datenübertragung (es wurde etwas nicht richtig verstanden), nachfragen und Daten (Nachricht) wiederholen lassen. Beendigung der Kommunikation durch Auflegen des Hörers.

Sie sehen hier schon, daß die Regeln für einen sinnvollen Kommunikationsablauf sehr komplex sein können (versuchen Sie das Obige jemandem zu erklären, der kein Telefon kennt) und daß Verstöße gegen diese Regeln (z.B. Wählen, ohne vorher den Hörer abgenommen zu haben) dazu führen, daß

keine Kommunikation zustande kommt.

Dem Datenübertragungsprotokoll muß also besondere Aufmerksamkeit geschenkt werden. Wir beginnen daher mit Erläuterungen zu einigen wichtigen Begriffen:

### *Nachrichtenaustausch*

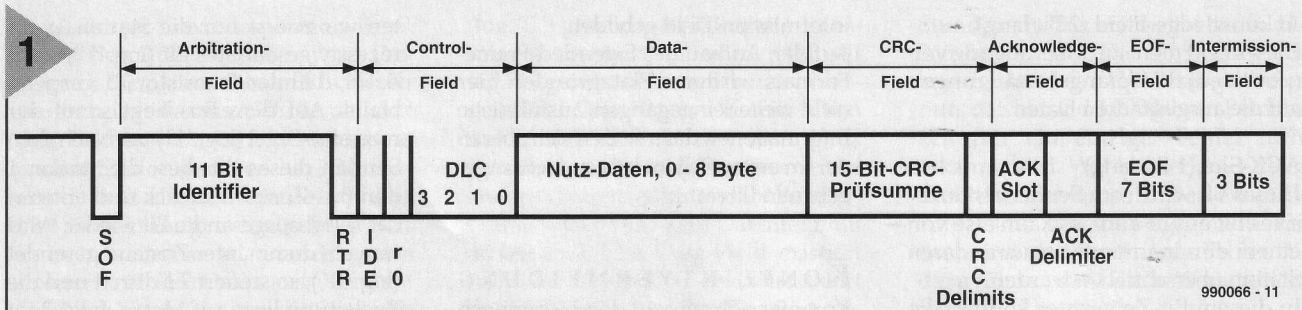
Der Austausch von Nachrichten zwischen den Busteilnehmern kann im wesentlichen auf zwei grundverschiedene Arten erfolgen:

### **Teilnehmer-orientierter Nachrichtenaustausch**

Hierbei spricht der Nachrichtensender den Nachrichtenempfänger ganz konkret mit seiner Empfangsadresse an, z.B.: "Station Nr. 25 sendet eine Nachricht an Station 37". Es wird also über den Bus eine "virtuelle" (scheinbare) Verbindung zwischen Sender und Empfänger aufgebaut, die Nachricht ist nur für eine einzige Station bestimmt. Im ausgesandten Datensatz stehen daher die Adresse des Empfängers und die Adresse des Absenders (37 und 25). Alle anderen Stationen am Bus ignorieren dieses Datenpaket vom Sender, da es nicht an sie adressiert ist. Der Empfänger wertet die Nachricht aus und quittiert im Normalfall den korrekten Empfang. Bei Fehlern während der Datenübertragung (negative Quittung von Seiten des Empfängers), wiederholt der Sender die Aussendung der Daten.

### **Objekt-orientierter Nachrichtenaustausch**

Der Nachrichtensender ordnet hierbei seiner Nachricht eine eindeutige Nachrichtennummer (Identifizier) zu und sendet Nachricht und Identifizier auf den Bus aus, z.B.: "Station A sendet einen Spannungsmesswert mit dem Identifizier 978 aus", Empfangs- und Absenderadresse werden nicht mit angegeben.



**Bild 1. Der Aufbau eines Data-Frames (Standard-Frame-Format gem. CAN 2.0A-Spezifikation).**

Diese Nachricht ist daher gleichzeitig für "jede Menge" von Empfängern bestimmt (Broadcasting-Prinzip) und das Motto der Sendestationen an die Empfänger lautet dementsprechend: "Nehmt euch vom Bus, was ihr braucht".

Die jeweils angeschlossenen Stationen müssen nun selbst (aufgrund der intern programmierten Software) entscheiden, ob diese Nachricht für sie relevant ist oder nicht.

#### Kommunikationsablauf

Der Ablauf der Kommunikation zwischen den einzelnen Stationen am CAN-Bus erfolgt nun in Form eines Broadcast-Austausches von ereignisgesteuerten, priorisierten (durchnummerierten) Botschaften bzw. "(Kommunikations)Objekten" zwischen den Busteilnehmern ( $\equiv$  objektorientierte Nachrichtenübertragung).

#### Dominante und rezessive Buszustände/Bits

Die eigentliche Datenübertragung auf dem Datenübertragungsmedium geschieht nun nicht, wie gewohnt, in Form von log. 0 - und log. 1-Bits, sondern durch dominante (überstimmende, überschreibende) und rezessive (nachgebende) Bits. Dabei charakterisiert rezessiv den einen Buszustand, der von dem zweiten, dominanten Buszustand überschrieben werden kann. Wenn also eine Station auf dem Bus ein rezessives Bit aussendet und eine andere Station gleichzeitig ein dominantes Bit sendet, so überschreibt das dominante Bit das rezessive, d.h. der dominante Zustand (das dominante Bit) setzt sich auf dem gesamten Bus durch. Die Zuordnung der logischen Zustände zu diesen Buszuständen geschieht im allgemeinen in der Art und Weise, daß eine log.0 dem dominanten und eine log.1 dem rezessiven Zustand entspricht.

Diese Festlegungen bilden einen wesentlichen Kernpunkt der CAN-Spezifikationen und werden nachfolgend noch näher erläutert.

#### Kommunikationsobjekte

Zum Austausch von Daten auf dem Bus benutzt man bei CAN vier Arten von Kommunikationsobjekten, die

auch Frames (Rahmen) genannt werden: Data-Frame, Remote-Frame, Error-Frame und Overload-Frame.

#### Data-Frame

Hiermit senden die CAN-Stationen ganz nach Belieben (entsprechend ihrer Software) ihre Daten aus. Den Aufbau eines solchen, aus einzelnen Feldern (Fields) bestehenden, Frames zeigt Bild 1 (Standard-Frame-Format gem. CAN 2.0A-Spezifikation). Es bedeuten:

**SOF** (Start of Frame-Bit, immer dominant (log. '0'))

Hiermit wird der Beginn eines Frames angezeigt und alle Busteilnehmer synchronisieren ihre interne Empfangsstufe mit der fallenden Flanke dieses Bits.

#### Arbitration-Field (12 Bit lang)

Hier sind die Informationen für den geregelten Buszugriff enthalten

#### 11 Bit Identifier

In diesem Teil steht der Identifier (ID) des ausgesandten Kommunikationsobjektes. Mit den 11 Bits lassen sich  $2^{11} = 2048$  unterschiedliche ID's bilden, wobei jedoch nur 2032 ID's frei verfügbar sind, da die restlichen 16 ID's für bestimmte Sonderfunktionen reserviert werden. Mit andern Worten: in einem einzigen CAN-Bus-System kann man mit 2032 verschiedenen Objekten (Meßwerte, Schalterstellungen, Lampenfunktionen, etc.) arbeiten. Das hört sich sehr viel an, in einer ganzen Reihe von Anwendungen ist das aber nicht ausreichend. Deshalb hat man ein Extended-Frame-Format mit 29 Identifier-Bits definiert (CAN 2.0B), bei dem man mit  $2^{29} \equiv 536.870.912$  verschiedenen Objekten arbeiten kann (Sie lesen richtig: fünfhundertsechunddreißig Millionen .....).

#### RTR

(Remote Transmission Request-Bit) Hiermit kann eine Station eine andere Station ganz gezielt auffordern, sofort und unmittelbar ihre Daten (Kommunikationsobjekte) auszusenden, weil diese z.B. irgendwo dringend benötigt werden (mehr dazu später). Bei einem Data-Frame

ist dieses Bit immer dominant (log. '0').

#### Control-Field (6 Bit lang)

Hier befinden sich Informationen über den Aufbau des Data-Frames.

#### IDE (Identifier Extension-Bit)

Mit diesem Bit wird angezeigt, ob ein Standard-Frame-Format mit 11-Bit-Identifier ausgesandt wird (IDE  $\equiv$  dominant, log. '0') oder ob ein Extended-Frame-Format mit einem 29-Bit-Identifier verwendet wird (IDE  $\equiv$  rezessiv, log. '1').

#### r0 (Reserve-Bit 0)

Dieses dominant gesendete Bit dient als Reserve-Bit für zukünftige Erweiterspezifikationen.

#### DLC (Data length Code, 4 Bit lang)

Mit diesen vier Bits wird angegeben, wie viele Datenbytes nachfolgend im Daten-Feld (Data-Field) übermittelt werden. Die CAN-Spezifikation läßt hierbei Datenfeldlängen von 0 bis zu 8 Bytes zu, d.h. in einem Data-Frame können maximal 8 Nutzdatenbytes übertragen werden.

#### Data-Field (0-8 Byte lang)

In diesem Feld stehen die zu übertragenden Nutzdatenbytes, 0 bis 8 Stück.

#### CRC-Feld (16 Bit lang)

Dieses Feld dient der Unterbringung von Zusatzinformationen zur Sicherung der zu übertragenden Daten gegenüber Störungen. Es wird dabei auf der Senderseite aus allen vorhergehenden Daten nach einer bestimmten Regel eine 15-Bit-CRC-Prüf(Check)-Summe gebildet, die in diesem CRC-Feld mit ausgesandt wird. Der Empfänger berechnet dann nach der gleichen Regel ebenfalls eine CRC-Prüfsumme aus den empfangenen Daten und vergleicht diese mit dem empfangenen CRC-Wert vom Sender. Sind beide Summen gleich, so ist (mit hoher Wahrscheinlichkeit) kein Datenübertragungsfehler aufgetreten. Bei Ungleichheit beider Werte ist ein Datenübertragungsfehler entstanden und die "Fehlerbehandlungsroutine" läuft ab (s. nachfolgend). Das CRC-Feld wird begrenzt durch das CRC-Delimiter-Bit, das immer rezessiv gesendet wird.

### Acknowledge-Field (2 Bit lang)

Dieses Feld dient zur Übertragung von (positiven) Empfangsbestätigungen auf die ausgesandten Daten.

### ACK-Slot (1 Bit lang)

Dieses Bit wird vom Sender als rezessives Bit ausgesandt, es kann also von einem dominanten Bit einer anderen Station überschrieben werden.

In diesem Bit-Zeitfenster können die am Bus angeschlossenen Empfänger eine positive Quittung aussenden, als Zeichen dafür, daß sie den Data-Frame korrekt, d.h. fehlerfrei, empfangen haben. Dieses Quittungssignal wird durch ein dominant gesendetes Bit dargestellt, das jeder

Intermission-Field gebildet.

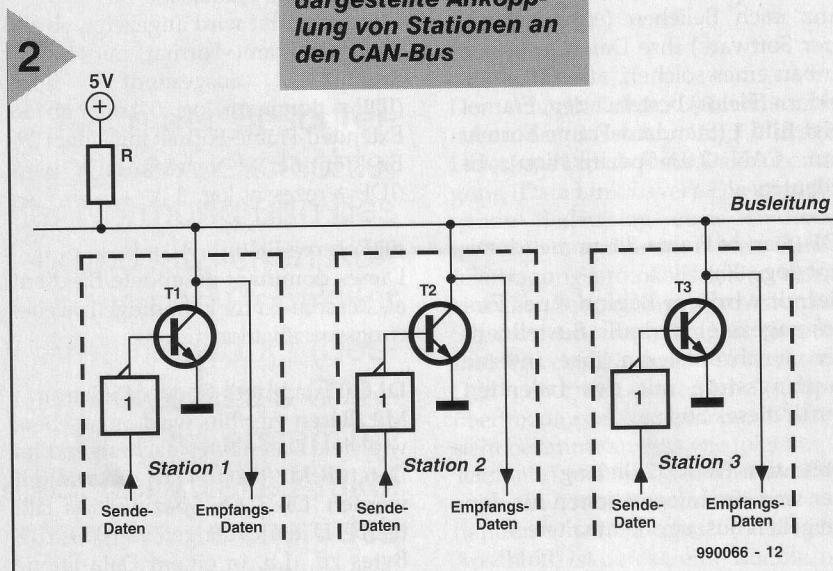
Auf den Aufbau des Extended-Frame-Formats wird aus Platzgründen hier nicht weiter eingegangen, ausführliche Informationen dazu finden sich aber in der im ersten Teil im letzten Heft angegebenen Literatur.

### KONFLIKTVERMEIDUNG

Kommen wir nun zu den bisher noch offenen zwei Kernproblemen beim CAN-Bus. Da alle CAN-Stationen am Bus gleichberechtigt sendefähig sind, fragen Sie sich bestimmt:

- Was passiert eigentlich, wenn mehrere Stationen gleichzeitig etwas aussenden wollen?

**Bild 2. Die vereinfacht dargestellte Ankopplung von Stationen an den CAN-Bus**



Empfänger bei fehlerfreiem Empfang aussendet und so das rezessive ACK-Slot-Bit des Senders überschreibt. Mit anderen Worten: empfängt der Sender während des ACK-Slot-Zeitfensters ein dominantes Bit (anstelle des von ihm gesendeten rezessiven Bits, so weiß er, daß mindestens eine Station seinen Data-Frame fehlerfrei empfangen hat. Das Acknowledge-Field wird von rezessiv gesendeten ACK-Delimiter-Bit begrenzt.

Der komplette Data-Frame wird nun abgeschlossen mit der EOF(End of Frame)-Bitkombination, die aus sieben rezessiv gesendeten Bits besteht.

Bevor ein nächster Frame gesendet werden kann, muß für die Empfänger eine "kleine Ruhepause" auf dem Bus eingelegt werden, damit sie die zuvor empfangenen Daten auch verarbeiten oder zumindest wegspeichern können. Diese Zeitverzögerung wird dadurch erreicht, daß nach dem Aussenden eines Frames ein rezessiver Bus-Zustand von mindestens 3 Bit-Zeiten eingehalten werden muß, bevor das nächste dominante Start-Bit (SOF) eines Frames gesendet wird. Diese minimale Wartezeit wird durch das

- Welche Station darf zuerst senden, welche Station muß warten?

Um diesen Konflikt aufzulösen gibt es beim CAN-Bus ein besonderes **Bus-Zugriffsverfahren (Bus-Arbitrierung)**, an das sich alle Stationen halten müssen wenn sie etwas aussenden wollen. Hierbei spielen nun die rezessiven und die dominanten Bits des Arbitration-Fields eine ganz besondere Rolle. Grundsätzlich gilt hier zunächst: Jeder Sender hört seine eigenen Aussendungen auf dem Bus mit: er sendet ein Bit aus, empfängt es wieder und vergleicht, ob beide Bits identisch sind. Ist das der Fall, so ist die Übertragung noch in Ordnung. Empfängt ein Sender jedoch einen anderen Bitzustand als denjenigen, den er gesandt hat, so liegt ein Problem vor. Wir wissen bereits, daß ein rezessives Bit (normalerweise log. '1') durch ein dominantes Bit (normalerweise log. '0') überschrieben werden kann. Schauen wir uns dazu einmal die in **Bild 2** vereinfacht dargestellten Buskoppelstufen der CAN-Stationen an. Es handelt handelt vom Prinzip her um Open-Collector-Ausgangsstufen, die eine Wired-And-Verknüpfung realisieren. Betracht

ten wir zuerst nur die Station 1: ein rezessiv gesendetes Bit (log. '1') sorgt dafür, daß der Transistor T1 gesperrt bleibt. Auf dem Bus liegt somit der rezessive Pegel (log. '1') an. Nach dem Senden dieses Bits liest die Station 1 den Bus-Zustand zurück und erkennt das selbst ausgesandte Bit wieder. Wird nun ein dominanter Zustand gesendet (log. '0'), so steuert T1 durch und die Busleitung liegt auf Masse. Jetzt liegt also der dominante Buszustand vor. Auch hierbei liest die Station 1 den gesandten Wert korrekt wieder zurück. Betrachten wir jetzt aber einmal alle drei Stationen am Bus, so kann man sehr leicht erkennen: sobald nur eine einzige Station ein dominantes Bit (log. '0') aussendet, wird die Busleitung fest auf den dominanten Pegel gezogen und alle anderen Stationen am Bus lesen diesen Pegel zurück. Nach dieser Betrachtung kann man nun am folgenden Beispiel nachvollziehen, wie der automatische Buszugriff, die Arbitrierung, beim CAN-Bus funktioniert. Die drei in **Bild 3** angegebenen Stationen wollen ihre Data-Frames mit den drei unterschiedlichen Identifiern aussenden:

- Station 1: Nachrichtenobjekt mit Identifier = 367
- Station 2: Nachrichtenobjekt mit Identifier = 232
- Station 3: Nachrichtenobjekt mit Identifier = 239.

Zum Bitzeitpunkt a wollen nun alle drei Stationen gleichzeitig auf den Bus zugreifen, um ihre Frames zu senden. Sie beginnen also alle mit der Arbitrierungs(Buszugriffs-)Phase, indem sie das SOF-Bit senden (siehe Bild 1). Diese SOF-Bit ist ein dominantes Bit und jede Station liest zunächst den richtigen (ihren richtigen) Wert wieder zurück. Nun werden von den Stationen die Identifier ausgesandt: Zum Zeitpunkt b senden alle ein dominantes Bit aus und alles ist noch in Ordnung. Im Zeitpunkt c gibt es auch noch kein Problem. Zum Zeitpunkt d sendet die Station 1 ein rezessives Bit aus, Station 3 und Station 2 jedoch jeweils ein dominantes Bit. Beim Rücklesen erkennt nun die Station 1, daß ihr rezessives Bit überschrieben worden ist, sie also den Buszugriff an mindestens eine andere Station verloren hat. Ab diesem Zeitpunkt (d) stellt die Station 1 ihren Sendebetrieb ein und geht in den Empfangszustand (Station 1 versucht zu einem späteren Zeitpunkt noch einmal, ihre Daten auszusenden). Die Stationen 2 und 3 dagegen senden weiter.

In den Zeitpunkten d bis i geben die Stationen 2 und 3 ihre Daten weiterhin parallel auf den Bus und nichts Besonderes passiert. Zum Zeitpunkt j aber sendet die Station 3 einen rezessiven