

Hochschule für Telekommunikation, Leipzig

Controller Area Network

Sebastian Lammermann

Controller Area Network

Projektarbeit

Verfasser:	Sebastian Lammermann Matrikelnummer 074111
Themenstellung:	„Erstellen Sie im Rahmen der Projektarbeit eine Unterlage, die auf anschauliche und verständliche Weise den Aufbau und die Funktionsweise des CAN-Busses beschreibt.“
Fertigstellung:	30. November 2008
Kontakt:	sebastian@lammermann.eu



Inhalt

1 Vorwort, Abgrenzung und Hinweise	1
2 Einleitung	1
3 Feldbusse	2
4 Entstehung des Controller Area Networks	3
5 Anforderungen an das CAN	4
6 Netzwerktopologie	6
7 Beschreibung der OSI-Schichten im CAN	7
7.1 Allgemein.....	7
7.2 CAN-Bitübertragungsschicht.....	7
7.2.1 Leitungscode.....	7
7.2.2 Bitstuffing.....	8
7.2.3 High-Speed-Modus.....	9
7.2.4 Low-Speed-Modus.....	9
7.3 CAN-Sicherungsschicht.....	10
7.3.1 Allgemein.....	10
7.3.2 Data Frame.....	11
7.3.3 Remote Frame.....	12
7.3.4 Error Frame.....	13
7.3.5 Overload Frame.....	15
7.3.6 Interframe Space.....	16
7.3.7 Erweitertes Rahmenformat.....	16
8 Buszugriffsverfahren	18
9 Echtzeiteigenschaften	20
10 Fazit	21

1 Vorwort, Abgrenzung und Hinweise

Die vorliegende Projektarbeit zum Thema *Controller Area Network (CAN)*¹ entstand im Rahmen des Masterprojektes im Wintersemester 2008/2009 an der Hochschule für Telekommunikation, Leipzig.

Ein CAN lässt sich mittlerweile nicht mehr nur über Kupferdoppelader, sondern auch über Glasfaser realisieren. Aufgrund der einerseits relativ geringen Verbreitung und andererseits derzeit noch mangelhaften Beschreibung in der Fachliteratur, befasst sich diese Arbeit ausschließlich mit der konventionellen kupferbasierten Variante.

Abkürzungen werden in dieser Projektarbeit in der Regel mit Artikel, sowie ggf. mit Genitiv-s versehen. Um einen besseren Lesefluss zu ermöglichen, wird eine eingeführte Abkürzung nicht durchgehend verwendet, sondern stellenweise die ausgeschriebene Variante bevorzugt.

Im vorliegenden Text wird sowohl für die Transkription als auch für die Translation des griechischen Buchstabens 'Φ', außer bei Namen, ausschließlich die offizielle neugriechische Übersetzung 'F' verwendet. Damit wird die Schreibweise des Dudens in einigen Fällen missachtet. Da dieser allerdings 1998 seine Monopolstellung bei der Rechtschreibung verloren hat und nun vielmehr Richtlinien beinhaltet, sei hiermit auf diese Hinwegsetzung aufmerksam gemacht.

2 Einleitung

Ziel dieser Projektarbeit ist die Erstellung eines Dokuments, welches auf anschauliche und verständliche Art den Aufbau und die Funktionsweise des CAN-Busses beschreibt. Sie dient daher als relativ leicht verständliche Einführung in die Thematik und kann, aufgrund ihres Umfangs, keinen Ersatz für Fachliteratur darstellen. Die im Quellenverzeichnis aufgeführten Dokumente sind tiefer gehend und ferner auch als Literaturempfehlung zu verstehen.

Diese Arbeit gliedert sich in insgesamt zehn Kapitel, in denen einerseits allgemeine Informationen, wie generelles über Feldbusse oder die Entstehungsgeschichte des CANs behandelt werden. Der Hauptteil des Textes beschäftigt sich jedoch mit den technischen Details des Controller Area Networks. Hier werden sowohl die Anforderungen an die Technologie und der Netzaufbau erläutert, als auch die Protokollschichten, das Buszugriffsverfahren und die Echtzeiteigenschaften des CANs beschrieben. Die Kapitel bieten im einzelnen:

¹ Engl.: „Steuerungsgebietsnetzwerk“.

- **Kapitel 3:** Informationen über Feldbusse allgemein
- **Kapitel 4:** Näheres zur Entstehung des CANs
- **Kapitel 5:** Eine Auflistung der Anforderungen an das CAN
- **Kapitel 6:** Eine Beschreibung der CAN-Netztopologie
- **Kapitel 7:** Eine detaillierte Beschreibung der im CAN-Standard beschriebenen OSI-Schichten. Dieses Kapitel entspricht dem Hauptteil der Projektarbeit
- **Kapitel 8:** Eine Erläuterung des beim CAN verwendeten Buszugriffsverfahrens
- **Kapitel 9:** Näheres zu Echtzeitsystemen im Allgemeinen, sowie eine Zusammenfassung aller für die Echtzeitkommunikation relevanten Eigenschaften des CANs
- **Kapitel 10:** Eine wertende Zusammenfassung dieser Arbeit

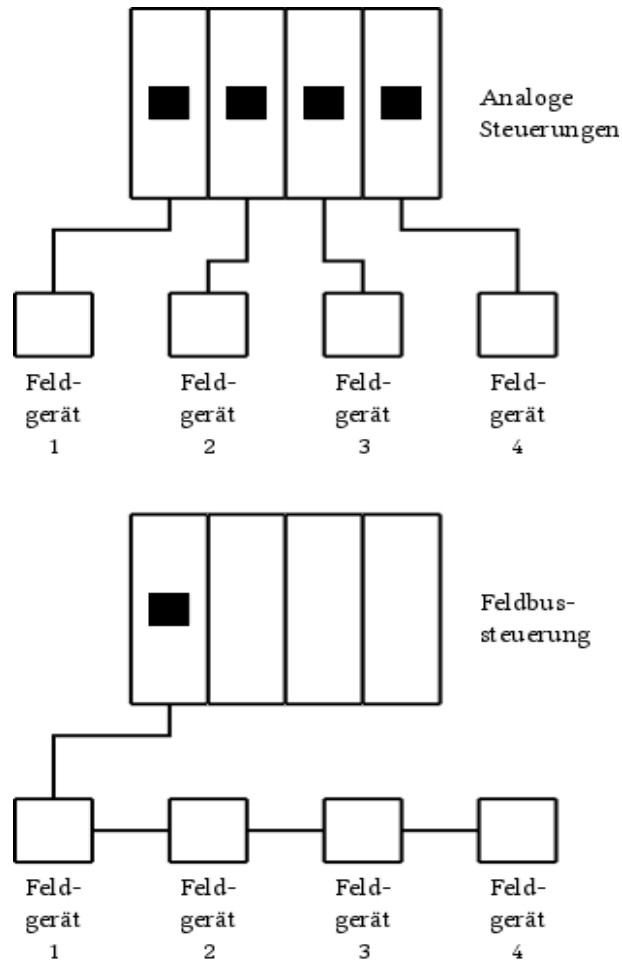
3 Feldbusse

Insbesondere in der Industrie werden seit der Einführung der Elektronik in diesem Bereich Bussysteme benötigt, die Sensoren, Antriebe und andere so genannte *Feldgeräte* miteinander vernetzen [Gruh01]. Bis in die 1980er Jahre war es üblich, jedes Gerät mit jedem anderen direkt zu verbinden (*Parallelverkabelung*), und dies oftmals noch über Analogtechnik. Da schließlich die Verkabelung immer komplexer wurde und größere räumliche Ausmaße annahm, begann man mit der Entwicklung von digitalen seriellen Bussystemen, den *Feldbussen*.

Im Unterschied zur Parallelverkabelung verfügt ein Feldbus in der Regel nur über eine einzige Leitung, die allerdings mit allen Feldgeräten verbunden ist. Hierdurch wird einerseits der Verkabelungsaufwand erheblich reduziert und andererseits ist es so möglich, alle Geräte mit einer gemeinsamen Steuerungseinheit zu regeln. In der Praxis bietet ein Feldbus gegenüber den konventionellen Systemen ferner die Vorteile, dass sich Inbetriebnahme und Wartung vereinfachen sowie Kosten senken lassen.

Feldbusse bauen in der Regel auf dem *OSI-Schichtenmodell* auf, beschreiben diese aber üblicherweise höchstens bis zur Vermittlungsschicht (Schicht 3) [Jans00]. Dadurch, dass der Bus seriell ist und die Kommunikation von mehr als zwei Geräten ermöglicht, ergibt sich darüber hinaus die Notwendigkeit, dass die Nachrichten adressierbar sind. Um in einer industriellen Umgebung eingesetzt werden zu können, muss ein Feldbus im Übrigen bestimmte Leistungsmerkmale aufweisen, wie z. B. *Echtzeiteigenschaften*, eine *hohe Zuverlässigkeit* und eine *geringe Störempfindlichkeit*. Anzustreben ist außerdem eine Verträglichkeit mit anderen Feldbussystemen (*Interoperabilität*), um verschiedene Technologien mit geringem Aufwand mitein-

ander zu verbinden.



Grafik 3.1: Vergleich von konventioneller paralleler (oben) und serieller Feldbusverkabelung (unten)

4 Entstehung des Controller Area Networks

Als Mitte des 20. Jahrhunderts zum ersten Mal Elektronik Einzug in den Kraftfahrzeugbau erhielt, wurden Sensoren, Steuereinheiten und andere – bis dahin ausschließlich analoge – Geräte zunächst direkt, also parallel, miteinander verkabelt. In den folgenden Jahrzehnten gewann die Bordelektronik mehr und mehr an Bedeutung, was letztlich dazu führte, dass die im Automobil verlegten Kabel Längen von bis zu 2 km und eine Masse von über 100 kg erreichten [Enge00].

1983 begann bei der Robert Bosch GmbH (BOSCH) in Stuttgart die Entwicklung des CAN-Busses. Ziel war zunächst die Entwicklung eines digitalen Feldbusses zur Steuerung des Antriebsstrangs², der den Anforderungen im Kraftfahrzeugbereich gerecht wird und gleich-

² Zum Antriebsstrang eines Kraftfahrzeugs zählen alle Komponenten, die für die Übertragung des Drehmoments vom Motor auf die Räder zuständig sind. Hierzu gehören z. B. Antriebswellen, Getriebe, Kupplungen etc.

zeitig einen Großteil der bisherigen Leitungen einspart. Zwei Jahre später wurde die erste CAN-Spezifikation vorgestellt und die Entwicklung von geeigneten Chips begann in Kooperation mit Intel. 1988 erschienen die ersten Serienchips auf dem Markt.

Die damalige Daimler-Benz AG war der erste Automobilkonzern, welcher mit der Entwicklung einer CAN-basierten Vernetzung der Fahrzeugelektronik begann. Binnen weniger Jahre setzte sich BOSCHs – komplett in Hardware implementiertes [Maye06] – Controller Area Network bei allen namhaften Kraftfahrzeugherstellern durch und verdrängte die bis dahin populäre Analogtechnik. 1993 wurde der Feldbus außerdem im ISO-Standard Nummer 11898 erstmals international genormt, welcher seitdem mehrfach aktualisiert worden ist.

Seit den 1990er Jahren findet das CAN auch vermehrt außerhalb des Kraftfahrzeugbereichs Anwendung. Zu nennen sind hier insbesondere die industrielle Automatisierung und die Medizintechnik, in der ähnliche Anforderungen an ein Bussystem herrschen wie im Fahrzeugbau.

5 Anforderungen an das CAN

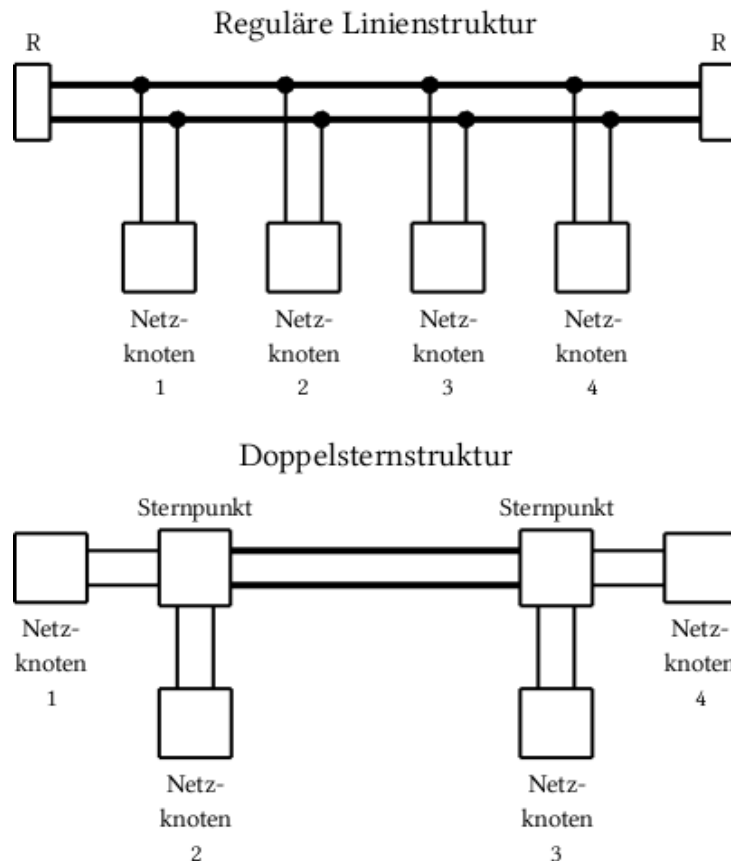
Feldbusse werden, unabhängig von ihrem Einsatzort, in der Regel immer in kritischen Umgebungen verwendet. Damit gelten auch für das Controller Area Network, welches insbesondere im Kraftfahrzeug- und Automatisierungsbereich eingesetzt wird, bestimmte technische Anforderungen, die oftmals über das Maß bei anderen Bussystemen hinausgehen. Konkret hat das CAN nachfolgende Voraussetzungen zu erfüllen. Über die konkrete Umsetzung dieser geben die anschließenden Kapitel Aufschluss.

- **Prioritätsbasierter Multimaster-Buszugriff:** Für das CAN muss das Prinzip des *Multimaster*-Buszugriffs gelten. Dies bedeutet, dass alle Netzknoten physikalisch zunächst gleichberechtigte Busteilnehmer sind und nicht von vornherein, wie z. B. beim *Master-Slave-Prinzip*, eine Hierarchie der Geräte existiert. Dennoch ist es notwendig die zu sendenden Nachrichten zu priorisieren.
- **Zerstörungsfreie und auf Konkurrenz basierte Arbitrierung:** Versuchen zwei oder mehr Netzknoten zur gleichen Zeit eine Nachricht zu versenden, kommt es zu einem Übertragungskonflikt. Das CAN muss in der Lage sein, diese Konflikte unter drei Bedingungen zu lösen: Zum einen darf es bei der Konfliktlösung weder zu Zeit- noch zu Datenverlusten kommen. Darüber hinaus muss ein Konflikt so bewältigt werden, dass die Nachricht mit der höchsten Priorität auf dem Bus verbleibt, während alle anderen zurückgestellt werden.

- **Multicast-Rahmenübertragung mit Akzeptanzfilter:** Wie bei anderen Bussystemen auch, muss beim CAN jeder Netzknoten in der Lage sein, jede auf dem Bus gesendete Nachricht zu empfangen. Um dies zu gewährleisten, ist das Gesamtsystem so zu dimensionieren, dass sich ein Bit auf dem ganzen Bus ausbreiten kann, bevor das nächste gesendet wird. Ferner müssen die Netzknoten über einen Akzeptanzfilter verfügen, also in der Lage sein zu erkennen, ob eine Nachricht an sie gerichtet ist oder nicht. Es sind einzig die Nachrichten zu verarbeiten, die für das jeweilige Gerät bestimmt sind.
- **Datenanforderung von anderen Netzknoten:** Benötigt ein Netzknoten Informationen von einem anderen, muss er in der Lage sein, diese anzufordern. Hierfür ist im CAN eine Möglichkeit vorzusehen. Dies gilt auch für Informationen, welche ein Knoten bereits angefordert hat, auf die er aber bisher vergeblich warten musste.
- **Flexibilität in der Netzkonfiguration:** Da es wünschenswert ist, das CAN in verschiedenen Umgebungen einzusetzen und unterschiedlich zu dimensionieren, muss der Standard flexibel sein. Rücksicht ist dabei sowohl auf verschiedene Buslängen und -topologien, sowie variierende Teilnehmerzahlen als auch auf die damit verbundene Übertragungsrate zu nehmen.
- **Systemweite Datenkonsistenz:** Bei der Dimensionierung des Netzwerks muss sicher gestellt sein, dass die übertragenen Daten im gesamten System konsistent, d. h. widerspruchsfrei, sind. An jedem Ort im Netzwerk müssen die gesendeten Daten also immer gleich interpretiert werden können.
- **Fehlererkennung und Fehlermeldung:** Ist ein Datenrahmen fehlerhaft, muss das Controller Area Network die Möglichkeit haben, diesen zu erkennen. Außerdem hat jeder Knoten, der Fehler erkennt, dies an alle anderen zu melden.
- **Automatisches erneutes Übertragen von Datenrahmen:** Ist ein Datenrahmen als fehlerhaft identifiziert worden, hat der sendende Netzknoten die Aufgabe, die Nachricht erneut zu verschicken. Zusätzlich muss jedes Gerät auf Datenanforderungen reagieren können.
- **Erkennung fehlerhafter Netzknoten:** Ist ein Netzknoten teilweise oder völlig defekt, muss das CAN in der Lage sein, das Gerät teilweise oder ganz vom Bus zu trennen. Hierdurch wird die Stabilität des Gesamtsystems gewährleistet.

6 Netzwerktopologie

Der CAN-Bus weist fysikalisch in der Regel eine mit Abschlusswiderständen³ terminierte Linienstruktur (*Bus*) auf. Es ist jedoch auch möglich, eine Doppelsterntopologie zu verwenden, bei denen die Netzknoten an Sternverbinder an den Endpunkten der Leitung angeschlossen sind. Als Medium dient üblicherweise eine verdrehte Kupferdoppelader (*Twisted Pair*).



Grafik 6.1: Vergleich von klassischer Linienstruktur (oben) und Doppelsternstruktur (unten)

Die möglichen Leitungslängen hängen vom Betriebsmodus des Netzes ab. Generell wird zwischen folgenden zwei Modi unterschieden:

- Der *High-Speed-Modus* bietet eine Übertragungsrate von bis zu 1 Mbit/s und ermöglicht den Anschluss von bis zu 30 Netzknoten. Die Leitungslänge ist in diesem Modus jedoch auf maximal 40 m beschränkt [ISO203]. Gedacht ist dieser Modus insbesondere für räumlich kleine Bereiche, in denen hohe Datenraten nötig sind, wie z. B. beim Motorraumbus.
- Im *Low-Speed-Modus* hingegen beträgt die maximale Übertragungsrate 125 kbit/s und es können ca. 20 Knoten mit dem Bus verbunden werden. Die maximale Buslänge beträgt hier ca. 500 m [ISO306]. Der große Vorteil des Low-Speed-Modus liegt in der Ausfallsicherheit,

³ Ein Widerstand von 120 Ω ist laut Standard vorgesehen. Die Integration des Widerstands in einen Netzknoten wird nicht empfohlen.

da auch im *Eindrahtmodus*⁴ – aufgrund des größeren Spannungshubs – dominante und rezessive Bits unterschieden werden können.

Um die Faktoren für den Aufbau eines CANs zu vereinheitlichen, lassen sich folgende Rahmenkriterien festlegen: Ein Netzwerk mit der maximalen Gesamtlänge von 40 m verbindet ca. 20 Knoten miteinander, wobei der maximale Abstand zweier benachbarter Knoten höchstens 20 m betragen darf.

7 Beschreibung der OSI-Schichten im CAN

7.1 Allgemein

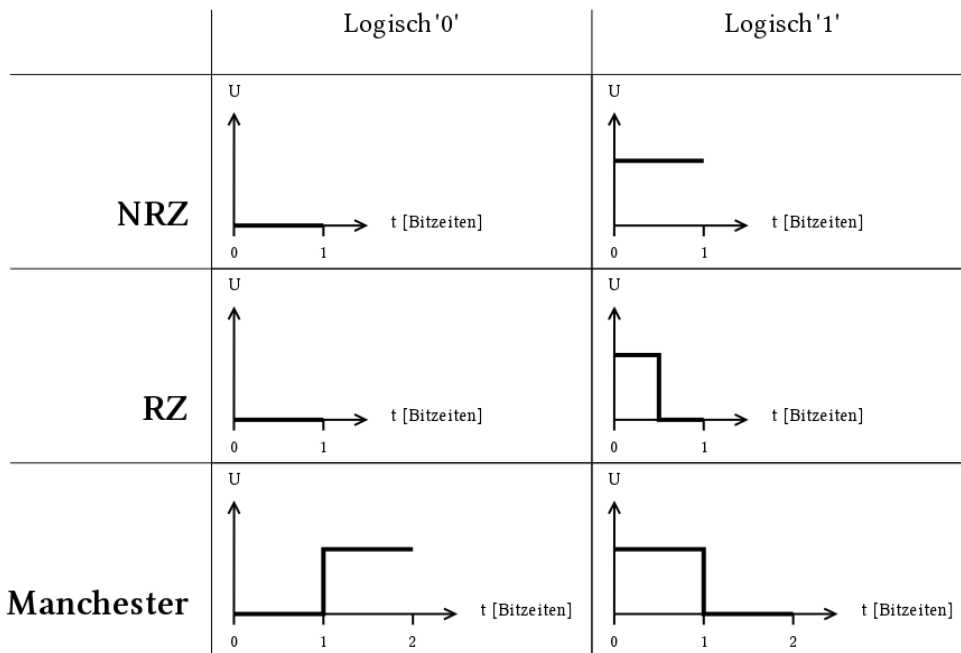
Wie bei anderen Feldbussen dient auch beim Controller Area Network das OSI-Schichtenmodell als Grundlage für die Protokollarchitektur. Beschrieben werden im Standard allerdings nur die Bitübertragungsschicht (Schicht 1) und die Datensicherungsschicht (Schicht 2) [ISO103]. Die nachfolgenden Abschnitte erläutern diese Schichten näher.

7.2 CAN-Bitübertragungsschicht

7.2.1 Leitungscode

Als Leitungscode verwendet der CAN-Bus *Non-Return to Zero (NRZ)*. Grundsätzlich sind bei diesem Code die Daten binär codiert und die beiden logischen Zustände durch unterschiedliche (und hier unipolare) Spannungswerte repräsentiert. Der Signalpegel bleibt während der gesamten Bitzeit konstant und fällt nicht auf einen Basiswert zurück, wie beispielsweise beim *Return-to-Zero-Code (RZ)*. Außerdem verfügt er über kein codiertes Taktsignal, wie z. B. der *Manchestercode*, wodurch für die Darstellung eines Bits lediglich ein Zeitabschnitt benötigt wird. Dies macht ihn zum einfachsten möglichen Leitungscode, allerdings wird eine externe Synchronisation vorausgesetzt.

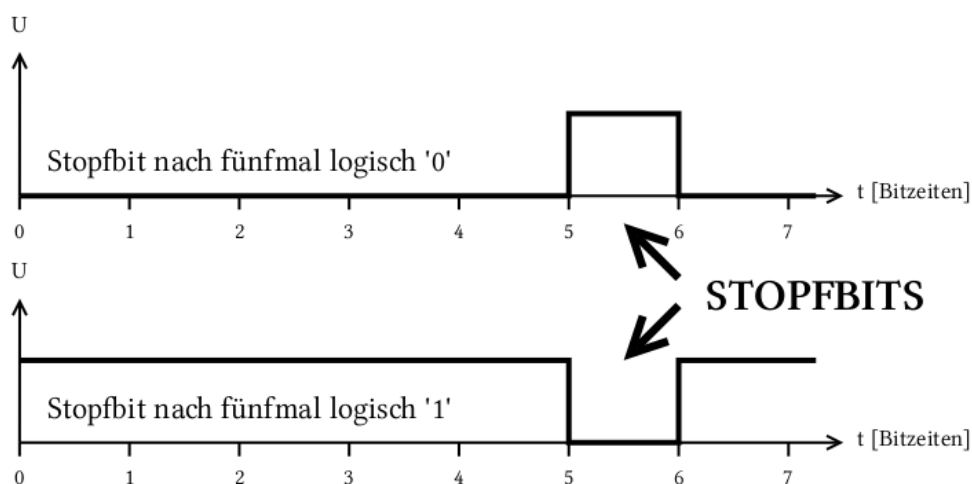
⁴ Als Eindrahtmodus gilt sowohl der Kurzschluss der Adern gegen Masse und untereinander, als auch mit der Versorgungsspannung.



Grafik 7.2.1.1: Vergleich von NRZ-, RZ- und Manchesterleitungscodierung

7.2.2 Bitstuffing

Damit sich empfangende Knoten auf das Signal des Senders synchronisieren können, macht das Controller Area Network vom Prinzip des *Bitstuffings*⁵ Gebrauch. Hierbei werden in regelmäßigen Abständen so genannte *Stopfbits* eingefügt, sofern sich der logische Pegel auf dem Bus für eine bestimmte Zeitspanne nicht verändert. Beim CAN geschieht dies nach fünf gleichwertigen Bits durch das Einfügen eines jeweils gegenteiligen Bits. Stopfbits werden nicht eingesetzt, wenn das Signal vorher den Pegel ändert.

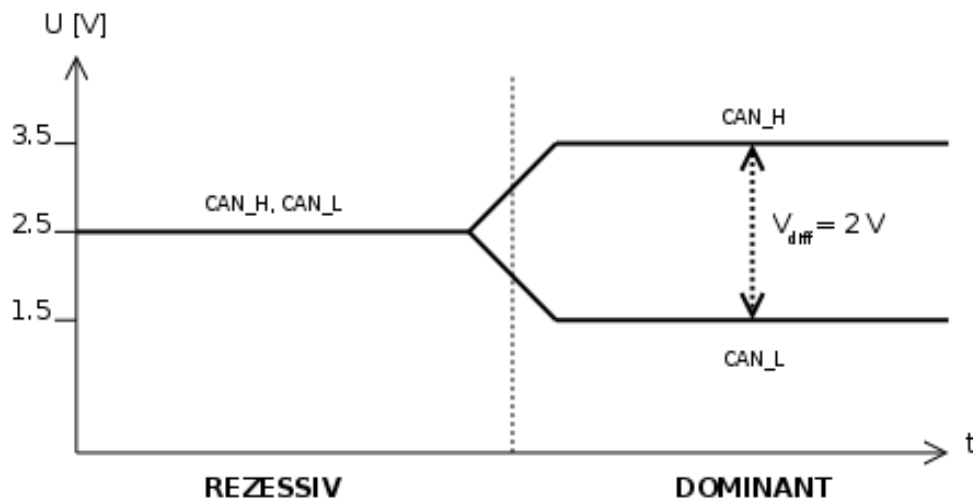


Grafik 7.2.2.1: Stopfbits nach fünf logischen Nullen (oben) bzw. fünf logischen Einsen (unten)

⁵ Engl.: „Bitstopfen“.

7.2.3 High-Speed-Modus

Der CAN-Bus verfügt über zwei logische Zustände: *rezessiv* und *dominant*. Im rezessiven Zustand, der außerdem der Normalzustand bei Übermittlungspausen ist, haben die Signalpegel der beiden Kupferadern einen festen Wert. Im *High-Speed-Modus* beträgt dieser im Idealfall bei beiden +2,5 V. Es existiert folglich in der Theorie keine Spannungsdifferenz und in der Praxis meist nur eine kleine. Ein dominanter Pegel wird in diesem Modus hingegen erreicht, indem die Spannungen auf den beiden Adern verändert werden. Der Standard empfiehlt hier die Absenkung der Spannung auf der *CAN-Low-Ader* (*CAN_L*) auf +1,5 V sowie die Erhöhung auf der *CAN-High-Ader* (*CAN_H*) auf +3,5 V. Es entsteht folglich eine Spannungsdifferenz von +2 V. Dieser Wert ist jedoch nicht bindend; wichtig ist einzig, dass die Spannungsdifferenz im dominanten Zustand größer ist als im rezessiven. Hierdurch ist sichergestellt, dass ein dominantes Bit ein rezessives überschreibt.



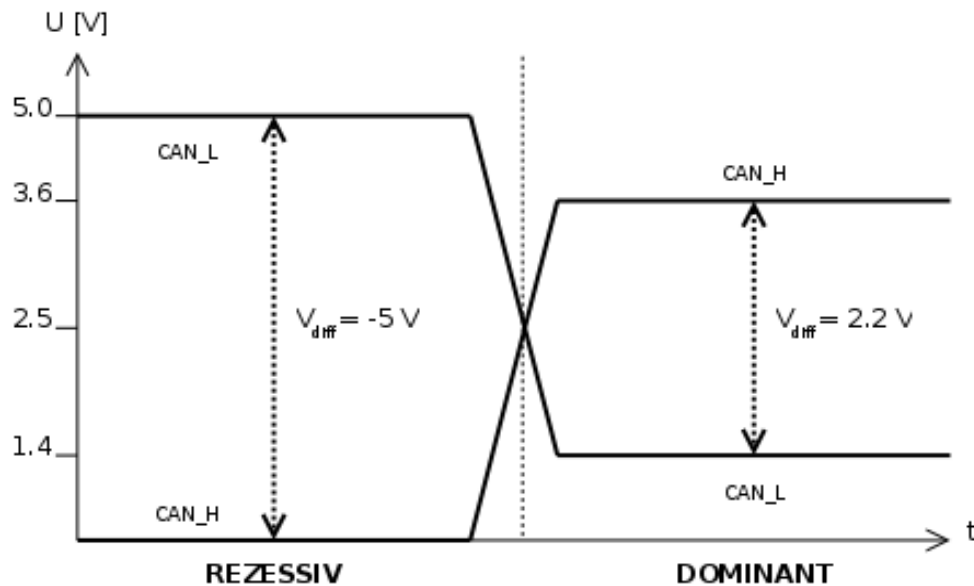
Grafik 7.2.3.1: Rezessiver und dominanter Pegel im High-Speed-Mode

Neben dem gewöhnlichen High-Speed-Modus existiert auch eine kompatible Variante, die mit einer niedrigeren Spannung arbeitet [ISO507]. Aus Gründen der besseren Übersichtlichkeit wird auf eine nähere Erläuterung dieser verzichtet.

7.2.4 Low-Speed-Modus

Der *Low-Speed-Modus* funktioniert etwas anders, auch wenn hier der rezessive und der dominante Zustand existieren. Im Idealfall besitzt *CAN_H* bei einem rezessiven Bit einen Pegel von 0 V und *CAN_L* einen Pegel von +5 V. Hieraus resultiert im Regelfall eine Spannungsdifferenz von -5 V. Im dominanten Zustand sinkt die Spannung von *CAN_L* auf +1,4

V ab, während die von CAN_H auf +3,6 V ansteigt. Die Spannungsdifferenz ist dementsprechend positiv und beträgt +2,2 V. Die Spannungsdifferenzen von rezessivem und dominantem Zustand unterscheiden sich demnach idealerweise um 7,2 V.



Grafik 7.2.4.1: Rezessiver und dominanter Pegel im Low-Speed-Mode

Aufgrund der unterschiedlichen betragsmäßigen Spannungspegel im Low-Speed-Modus beschreibt die Literatur ein dominantes Bit üblicherweise mit einer logischen 0, während ein rezessives Bit durch eine logische 1 dargestellt wird.

7.3 CAN-Sicherungsschicht

7.3.1 Allgemein

Nachrichten zwischen Netzknoten werden im CAN mittels eines MAC-Rahmens⁶ übertragen. Grundsätzlich werden hier fünf verschiedene Arten unterschieden:

- **Data Frame:** Dieses reguläre Datentelegramm überträgt Nachrichten von einem Sender zu mindestens einem Empfänger.
- **Remote Frame:** Dieses Datentelegramm wird von einem Empfänger genutzt, um Daten von einem Sender anzufordern.
- **Error Frame:** Erkennt ein Netzknoten einen Fehler, meldet das Gerät mittels dieses Fehlertelegramms die Störungsdetektion.
- **Overload Frame:** Dieses Überlasttelegramm erzeugt einen zusätzlichen zeitlichen Puffer

⁶ „Media Access Control“, engl.: „Medienzugriffssteuerung“. Zweitunterste OSI-Schicht, welche Protokolle enthält, die den Buszugriff regeln.

zwischen der vorangegangenen und der nachfolgenden Nachricht.

- **Interframe Space:** Ebenfalls als MAC-Rahmen kann der festgelegte Raum zwischen zwei Data Frames oder Remote Frames betrachtet werden.

7.3.2 Data Frame

Ein regulärer *Data Frame* besteht aus den folgenden Bitfeldern:

- *Start of Frame (SOF):* Das SOF-Bit bildet die Telegrammanfangskennung. Mit ihr signalisiert ein Netzknoten durch Aufschalten eines dominanten Pegels auf den bis dahin rezessiven Bus, dass eine Nachricht folgt. Alle anderen Geräte am Bus nutzen das SOF-Bit zur Synchronisation.
- *Nachrichtenidentifizier (ID):* Der Identifizier kennzeichnet eine Nachricht eindeutig und bestimmt deren Priorität. Im Standardformat ist die ID 11 Bit lang und ermöglicht somit die Unterscheidung von bis zu 2048 Nachrichten. Die Übertragung beginnt mit dem höchstwertigen Bit.
- *Remote-Transmission-Request-Bit (RTR):* Das RTR-Bit ermöglicht die Unterscheidung eines Data Frames von einem Remote Frame. Bei einem Data Frame ist dieses Bit immer dominant, was ihn damit automatisch höher priorisiert als einen Remote Frame mit der gleichen ID.
- *Control Field (CF):* In diesem 6 Bit langem Steuerfeld wird die Datenlänge der nachfolgenden Information übermittelt. Die vier niederwertigen Bits stellen binär die Anzahl der Bytes dar, die ein Empfänger im Datenfeld erwarten darf. Im Standardformat sind die beiden hochwertigen Bits für zukünftige Anwendungen reserviert und immer dominant.
- *Data Field (DF):* Dieses Datenfeld enthält die Nutzinformation der Nachricht und kann eine Länge von 0 bis 8 Byte annehmen. Von jedem Byte wird immer das höchstwertige Bit zuerst übertragen.
- *Prüfsumme (CRC):* Damit ein Empfänger prüfen kann, ob eine empfangene Nachricht durch Störungen verfälscht worden ist, enthalten die Datenrahmen eine auf der *zyklischen Redundanzprüfung (Cyclic Redundancy Check, CRC)* basierte Prüfsumme [Will93]. Diese Sequenz ist insgesamt 15 Bit lang und das Feld endet zusätzlich mit einem rezessiven Begrenzungsbit.
- *Acknowledge Field (ACK):* Das Acknowledge Field wird zur Bestätigung der fehlerfreien Nachrichtenübermittlung verwendet und besteht aus zwei Bits. Beim ersten Bit, dem so

genannten *Acknowledge Slot*, schaltet der Sender einen rezessiven Pegel auf den Bus. Alle Empfänger, die die übermittelte Nachricht nach dem Ende der CRC-Prüfung als fehlerfrei identifiziert haben, schalten hingegen einen dominanten Pegel. Gemäß der Regel „dominant überschreibt rezessiv“ nimmt der Bus einen dominanten Pegel an, sofern mindestens ein Empfänger die Nachricht korrekt erhalten hat. Der Sender erkennt dies durch den Vergleich seines gesendeten und des tatsächlich empfangenen Wertes. Ist das Bit dominant, kann zumindest sichergestellt werden, dass ein womöglich aufgetretener Fehler nicht auf den Sender zurückzuführen ist. Das zweite Bit, der *Acknowledge Delimiter*, dient der Feldbegrenzung und ist immer rezessiv.

- *End of Frame (EOF)*: Ein normaler Data Frame wird mit einem 7 Bit langen rezessiven Pegel beendet.

Anzahl bit	1	11	1	6	0...64	15	1	1	1	7
Zustand	D	?	D	?	?	?	r	?	r	r
Feld	Start of Frame	Nachrichtenidentifizier	Remote Transmission Request	Control Field	DATA FIELD	CRC Checksum	CRC-Begrenzungsbit	Acknowledge Slot	Acknowledge Delimiter	End of Fame

D = dominant, r = rezessiv, ? = beide Zustände möglich

Grafik 7.3.2.1: Aufbau des CAN Data Frames

Der Bereich vom SOF-Bit bis zur CRC-Prüfsumme kann darüber hinaus Stopfbits enthalten, was den Rahmen im ungünstigsten Fall um 20 Bits vergrößert (bei einem Verhältnis von 5:1). Ab dem CRC-Begrenzungsbit sind keine Stopfbits mehr vorgesehen.

7.3.3 Remote Frame

Neben dem normalen Data Frame existiert ferner der *Remote Frame*. Er wird von einem Datenempfänger an einen Datensender verschickt und enthält die Aufforderung, eine bestimmte Nachricht zu übermitteln. Diese ist durch den Nachrichtenidentifizier, welcher mit

dem der Antwort identisch ist, eindeutig spezifiziert.

Der Aufbau eines Remote Frames entspricht weitestgehend dem des Data Frames, allerdings weist er einige Besonderheiten auf. Der markanteste Unterschied ist der rezessive Zustand des RTR-Bits. Hierdurch unterliegt ein Remote Frame während der Arbitrierung⁷ automatisch einem Data Frame mit dem gleichen Identifier. Dadurch wird sichergestellt, dass die Antwort auf eine Nachrichtenanforderung gegenüber dessen Anfrage immer priorisiert wird und es somit nicht zu einer unnötigen Auslastung des Busses kommt.

Ein weiteres spezielles Merkmal ist das Control Field bzw. das Data Field. Das Data Field ist immer leer, enthält also 0 Byte. Der Wert des Control Fields hingegen entspricht der erwarteten Länge des Data Fields im Antwortpaket (!) und ist somit in der Regel größer als 0.

Anzahl bit	1	11	1	6	15	1	1	1	7
Zustand	D	?	r	?	?	r	?	r	r
Feld	Start of Frame	Nachrichtenidentifier	Remote Transmission Request	Control Field	CRC Checksum	CRC-Begrenzungsbit	Acknowledge Slot	Acknowledge Delimiter	End of Fame

D = dominant, r = rezessiv, ? = beide Zustände möglich

Grafik 7.3.3.1: Aufbau des CAN Remote Frames

Analog zum Data Frame können auch im Remote Frame im Bereich bis zur Prüfsummenbegrenzung Stopfbits eingefügt werden. Die maximale Anzahl beträgt hier 8 (bei einem Verhältnis von 5:1).

7.3.4 Error Frame

Ist ein Data Frame oder Remote Frame gestört⁸, erfolgt der Abbruch des Sendeprozesses mittels eines *Error Frames*. Dieses Fehlertelegramm erzeugt durch das Senden von sechs gleichen Bits

⁷ Siehe: 8 Buszugriffsverfahren.

⁸ Fehler treten bspw. auf, wenn die Prüfsumme oder das Rahmenformat inkorrekt sind, das Acknowledgement nicht gesendet oder die Bitcodierungsregel nicht eingehalten wird.

eine Codeverletzung, zu der es im fehlerfrei Betrieb nie kommen könnte.

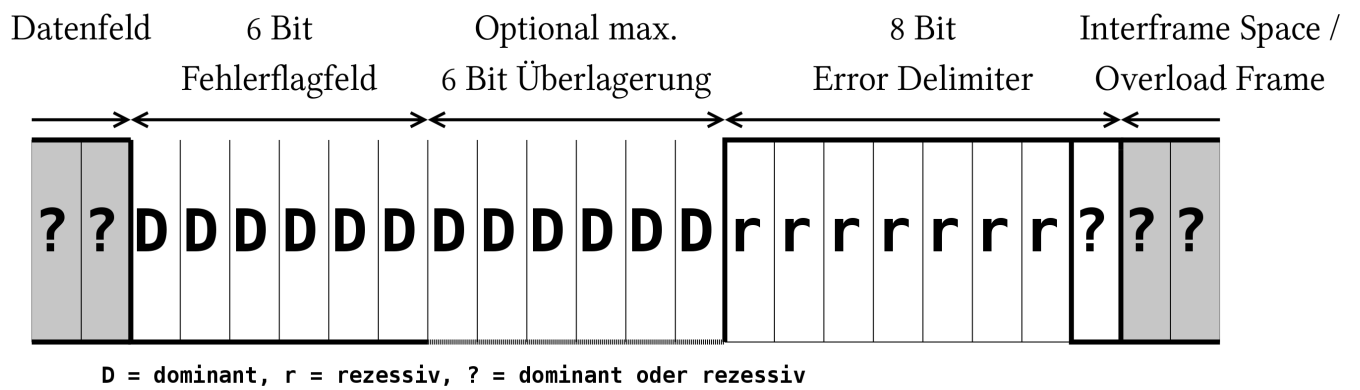
Ein Error Frame ist völlig anders aufgebaut als ein Data Frame und beinhaltet lediglich zwei Felder. Das erste ist das *Fehlerflagfeld* und es besteht aus 6 dominanten Bits, die ggf. auf dem Bus befindliche rezessive Pegel überschreiben. Falls mehrere Knoten gleichzeitig oder dicht aufeinander folgend einen Fehler melden, können sich deren Fehlerflagfelder auch überlagern, wodurch maximal 12 dominante Bits hintereinander gesendet werden. Da durch das Prinzip des Bitstuffings maximal fünf Bits gleicher Polarität aufeinander folgen dürfen⁹, bemerken alle teilnehmenden Netzknoten diese Codeverletzung und verwerfen den entsprechenden Datenrahmen.

Das zweite Feld des Error Frames ist der *Error Delimiter*. Er besteht aus einer Folge von 8 rezessiven Bits und bildet den Abschluss des Error Frames. Da es sein kann, dass mehrere Netzknoten kurz hintereinander Fehler melden, wird zunächst nur das erste rezessive Bit auf den Bus geschaltet. Bleibt der Bus dominant, weiß das Gerät, dass andere Knoten ebenfalls einen Fehler melden. So ist es möglich abzuschätzen, ob das Gerät den Fehler zuerst erkannt hat, was bei der Fehlereingrenzung behilflich sein kann. Sobald der Bus tatsächlich den rezessiven Zustand annimmt, werden die letzten 7 Bit gesendet.

Damit defekte Netzknoten den Bus nicht übermäßig mit Fehlermeldungen auslasten, verfügt das CAN über die Fähigkeit, defekte Netzknoten zu erkennen und diese vom Bus zu trennen. Jeder Netzknoten verfügt über einen Zähler für Sende- und Empfangsfehler. Meldet ein Knoten einen Fehler, erhöht sich der Wert des entsprechenden Zählers bei sekundärer Erkennung um eins. Stellt der Knoten fest, dass er als erstes einen Error Frame gesendet hat (die Störung also primär meldet), erhöht er den Zählerwert zudem um weitere sieben, da es wahrscheinlich ist, dass die Ursache für den Fehler bei ihm selber liegt. Erfolgt die Übertragung eines Rahmens fehlerfrei, wird der Zähler wieder dekrementiert. Aus dem Wert des Zählers wird der Zustand des Gerätes gefolgert:

- *Error Active (bis Zählerwert 127)*: Der Knoten nimmt normal an der Kommunikation teil und meldet ggf. auch Fehler.
- *Error Passive (ab Zählerwert 128)*: Der Netzknoten ist nicht voll kommunikationsfähig. Er darf zwar prinzipiell senden und empfangen, jedoch nicht mehr als erstes Fehlerflags verschicken. Außerdem muss er vor dem erneuten Senden einer Nachricht mindestens 8 Bitzeiten abwarten, um die Kommunikation der fehlerfreien Knoten nicht zu stören.
- *Bus Off (ab Zählerwert 256)*: Die Fehlerrate des Netzknotens ist so hoch, dass er komplett vom Bus getrennt wird.

⁹ Siehe: 7.2.2 Bitstuffing.

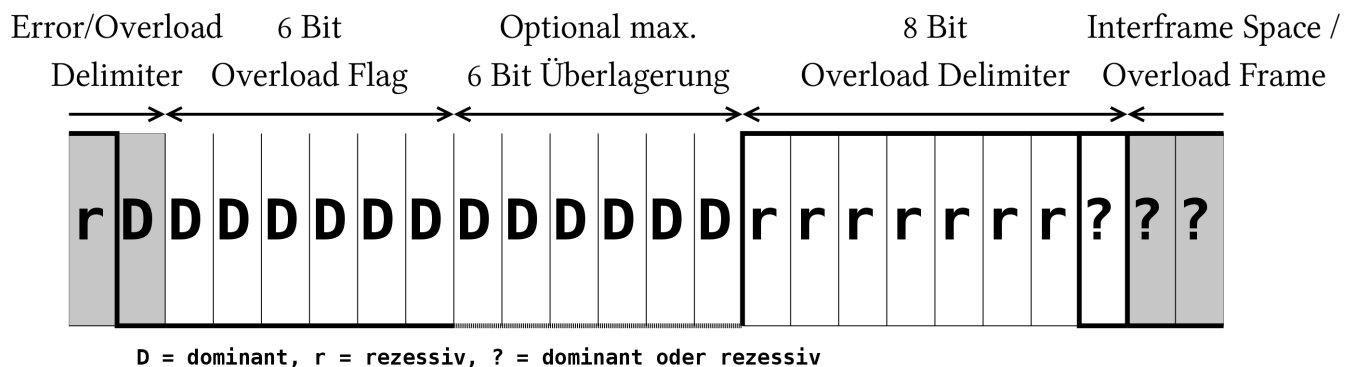


Grafik 7.3.4.1: Aufbau des CAN Error Frames

7.3.5 Overload Frame

Eine spezielle Art des Error Frames stellt der *Overload Frame* dar. Dieses Überlasttelegramm wird verschickt, wenn der nachfolgende Datenrahmen erst nach einer Verzögerung gesendet werden soll. Dies ist einerseits nötig, wenn während des Interframe Spaces ein dominantes Bit erkannt wird. Andererseits kann das letzte Bit eines vorhergehenden Error Frames oder Overload Frames dominant gesetzt sein, um das Übertragen des Rahmens zu forcieren. Es dürfen maximal zwei Overload Frames hintereinander gesendet werden.

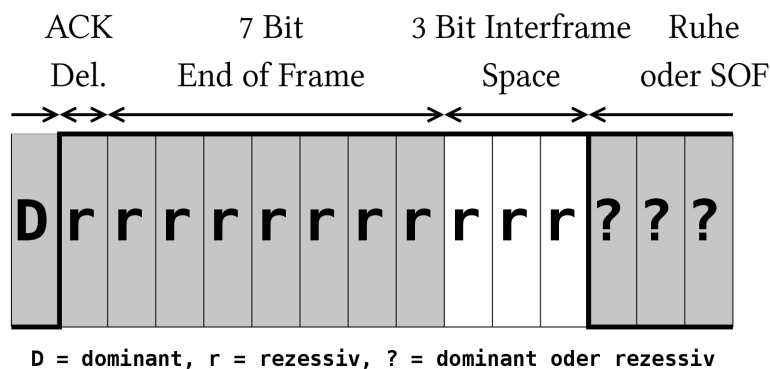
Der Overload Frame basiert auf dem Error Frame und er besteht, analog zu diesem, aus einem *Overload Flag* und einem *Overload Delimiter*. Das Overload Flag besteht aus 6 dominanten Bits und unterbricht damit zunächst die festgelegte Sendepause im Interframe Space. Alle anderen aktiven Netzknoten erkennen diesen Regelbruch und senden ihrerseits Overload Flags. Ist dies geschehen, endet der Overload Frame mit dem Overload Delimiter, welcher aus 8 rezessiven Bits besteht.



Grafik 7.3.5.1: Aufbau des CAN Overload Frames

7.3.6 Interframe Space

Der Standard des Controller Area Networks schreibt vor, dass es zwischen zwei Data Frames bzw. Remote Frames einen *Interframe Space* geben muss. Dieser besteht zunächst aus drei rezessiven Bits, die nur durch einen Error Frame oder Overload Frame überschrieben werden dürfen. Nach diesem Mindestabstand der Rahmen folgt entweder sofort der nächste Data Frame oder der Bus bleibt so lange im rezessiven Ruhezustand, bis wieder Daten gesendet werden müssen. Sofern keine Fehler auftreten, hat der Bus zwischen Acknowledge Slot und Start-of-Frame-Bit damit mindestens 11 Bitzeiten lang einen rezessiven Pegel. Wie bereits erwähnt verlängert sich diese Spanne um weitere 8 Bitzeiten, sofern sich der nächste sendende Netzknoten im fehlerpassiven Zustand befindet.



Grafik 7.3.6.1: Aufbau des CAN Interframe Spaces

7.3.7 Erweitertes Rahmenformat

Um eine weiterreichende Adressierbarkeit und damit bessere Interoperabilität zu erreichen, wurde der ursprüngliche CAN-Standard um das erweiterte und kompatible Format ergänzt. Der Nachrichtenidentifizier wurde hierfür vergrößert und hält statt 11 nun 29 Bits für die ID bereit, wodurch über 500 Mio. Nachrichten unterschieden werden können.

Ein erweiterter Data Frame besteht aus den folgenden Feldern:

- *Start of Frame (SOF)*: Wie beim Standardformat beginnt ein Data Frame mit einem dominanten SOF-Bit.
- *Basisidentifizier (BID)*: Der Nachrichtenidentifizier ist im erweiterten Format in zwei Teile geteilt. Zunächst wird, analog zum Standardrahmen, der 11 Bit lange Basisidentifizier gesendet.
- *Substitute Remote Request Bit (SRR)*: Um Rahmen im Standardformat während der Arbitrierung immer den Vortritt zu lassen, ist das grundsätzlich rezessive SRR-Bit an Stelle des RTR-Bits gesetzt.

- *Identifier Extension Bit (IDE)*: Als Flag zur Kennzeichnung des Rahmentyps wird das IDE-Bit benutzt. Es befindet sich an Stelle des ersten ungenutzten dominanten Bits im Control Field des Standardrahmens und ist immer rezessiv.
- *Erweiterter Identifier (EID)*: Es folgt der restliche, 18 Bit lange Teil des Identifiers.
- *Remote-Transmission-Request-Bit (RTR)*: Wie beim Standardformat ermöglicht das RTR-Bit die Unterscheidung eines Data Frames von einem Remote Frame.
- *Control Field (CF)*: Auch hier wird, wie im Standardformat, die Datenlänge der nachfolgenden Information übermittelt.
- *Data Field (DF)*: Die Nutzinformation der Nachricht kann im erweiterten Format ebenfalls eine Länge von 0 bis 8 Byte annehmen.
- *Prüfsumme (CRC)*: Analog zum Standardformat gibt es auch hier eine CRC-Prüfsumme.
- *Acknowledge Field (ACK)*: Das zweigeteilte Acknowledge Field existiert gleichfalls in beiden Formaten.
- *End of Frame (EOF)*: Auch hier wird ein normaler Data Frame mit einem 7 Bit langen rezessiven Pegel beendet.

Anzahl bit	1	11	1	1	18	1	6	0...64	15	1	1	1
Zustand	D	?	r	r	?	D	?	?	?	r	?	r
Feld	Start of Frame	Basisidentifier	Substitute Remote Request	Identifier Extension	Erweiterter Identifier	Remote Transmission Request	Control Field	DATA FIELD	CRC Checksum	CRC-Begrenzungsbit	Acknowledge Slot	Acknowledge Delimiter

D = dominant, r = rezessiv, ? = beide Zustände möglich

Grafik 7.3.7.1: Aufbau des erweiterten CAN Data Frames

Auch im erweiterten Rahmenformat können Stopfbits eingefügt werden. Bei einem Data Frame sind dies maximal 24 zusätzliche Bits (bei einem Verhältnis von 5:1). Analog zum Standardformat sind ab dem Prüfsummenbegrenzungsbit keine Stopfbits mehr vorgesehen.

Der Remote Frame existiert, analog zum Data Frame, ebenfalls im erweiterten Format.

8 Buszugriffsverfahren

Um den Zugriff auf das Medium zu steuern, verwendet das CAN einen relativ simplen Algorithmus namens *Carrier Sense Multiple Access with Collision Resolution (CSMA/CR)*¹⁰, welcher eine Modifikation von CSMA darstellt.

CSMA besteht aus zwei Verfahren: Das erste (*Carrier Sense, CD*) beschreibt, dass ein Netzknoten vor dem Senden von Daten prüft, ob ein Trägersignal verfügbar ist. Ist dieser Träger anwesend, ist das Gerät folglich an ein Netzwerk angeschlossen. Im nächsten Schritt wird nun überprüft, ob bereits eine Kommunikation auf dem Bus stattfindet, wodurch ein Senden der eigenen Nachricht nicht möglich wäre. Ist der Bus für eine bestimmte Periode (*Interframe Space*)¹¹ frei, kann mit der Datenübermittlung begonnen werden.

Das zweite Verfahren (*Multiple Access, MA*) beschreibt die Tatsache, dass das Netzwerk einen logischen Bus bildet. Dies bedeutet, dass jeder Netzknoten auf das Medium zugreifen kann und dass jede Nachricht von allen anderen mit dem Bus verbundenen Geräten empfangen wird. Weiterverarbeitet werden allerdings einzig die Daten, die an den eigenen Knoten adressiert sind.

Da die Datenübertragung beim CAN unkoordiniert und asynchron erfolgt, kann es passieren, dass zwei oder mehr Geräte gleichzeitig versuchen Daten zu senden. In diesem Fall kommt es zu einem Übertragungskonflikt bzw. zu einer Kollision. Um den CSMA-Algorithmus zu verbessern und die Effizienz des CANs zu steigern, wurde daher das zusätzliche Verfahren der Kollisionsauflösung (*Collision Resolution, CR*) eingeführt. Beginnen mehrere Knoten gleichzeitig mit dem Senden von Daten, wird während einer Auswahlphase (*Arbitrierung*) entschieden, welches Gerät sich durchsetzt und am Ende weiter senden darf.

Um die Kollision zu lösen, verfügt jede zu sendende Nachricht über ein *Nachrichtenarbitrierungsfeld*, bestehend aus Nachrichtenidentifizier sowie RTR-Bit, welches von jedem beteiligten Netzknoten bitweise auf den Bus geschaltet werden [Ets00]. Die Nachricht mit dem dominantesten Wert in diesen Feldern „gewinnt“ den Konflikt und der entsprechende Netzknoten sendet schließlich als einziger seine Nachricht. Das Procedere funktioniert wie folgt:

Starten mehrere Netzknoten auf einem bis dahin ruhigen Bus eine Datenübertragung, beginnen sie diese mit dem Aufschalten eines dominanten Pegels auf den Bus (Start-of-Frame-Bit). Während der Übertragung des Nachrichtenarbitrierungsfeldes und des RTR-Bits wechselt der Pegel meist zwischen dominant und rezessiv, je nachdem welche logische Bitfolge übertragen wird. Da beim CAN die Grundregel gilt, dass ein dominanter einen rezessiven Pegel

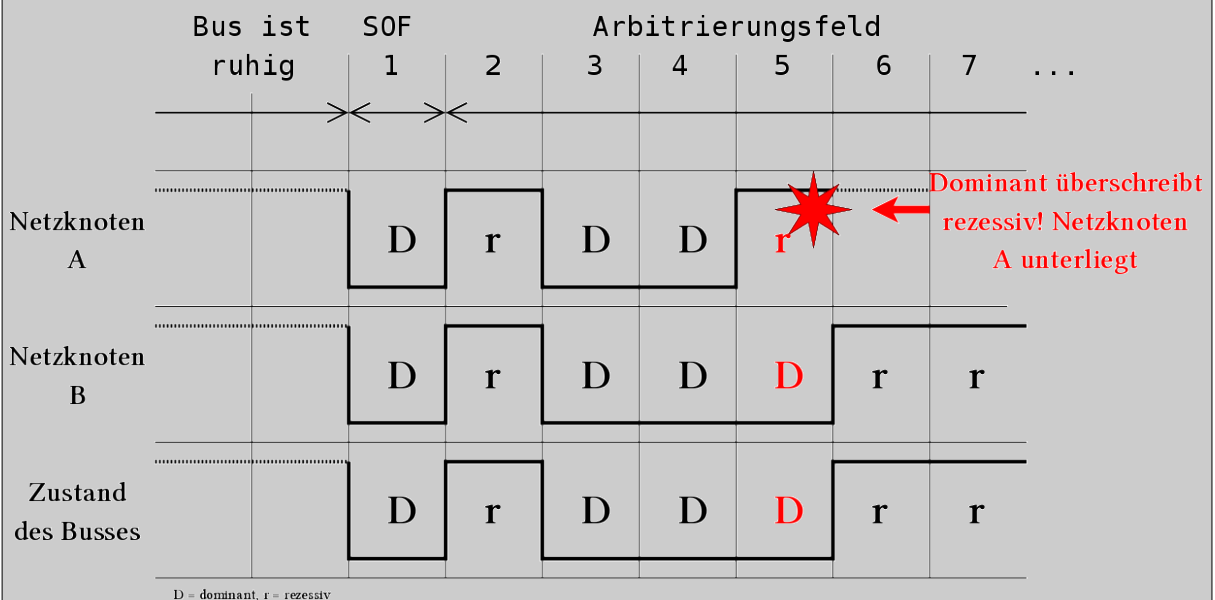
¹⁰ Engl. in etwa: „Trägererkennung und Mehrfachzugriff mit Kollisionsauflösung“.

¹¹ Engl.: „Zwischenrahmenraum“. Entspricht mindestens 11 Bitzeiten. Siehe: 7.3.6 *Interframe Space*.

überschreibt, ist irgendwann während der Übertragung der Zeitpunkt erreicht, an dem das rezessive Bit des einen durch das dominante Bit des anderen Netzknotens überlagert wird. Durch den ständigen Vergleich der gesendeten und empfangenen Daten erkennt der „unterlege“ Netzknoten, dass der von ihm übertragene rezessive Pegel nicht auf dem Bus anliegt und beendet seinen Übertragungsversuch.

Beispiel 8.1:

Zwei Netzknoten, A und B, greifen gleichzeitig auf den Bus zu. Sie signalisieren beide als erstes mit dem Aufschalten eines dominanten SOF-Bits, dass sie den Bus belegen. Die Bits 2 bis 4 sind bei A und B identisch: zunächst wird der Pegel rezessiv und dann zweimal dominant. Bis zu diesem Punkt gleichen sich die Rahmen der beiden Nachrichten und der Konflikt kann nicht gelöst werden.



Grafik 8.1: Netzknoten B gewinnt eine Kollision gegen Netzknoten A

Bei Bit 5 fällt die Entscheidung: Während Netzknoten A einen rezessiven Pegel sendet, ist der von B dominant. Gemäß der Regel „dominant überschreibt rezessiv“ nimmt der Bus einen dominanten Pegel an. Durch den Vergleich der gesendeten und der empfangenen Daten merkt A nun, dass diese nicht übereinstimmen. Eine „höherwertige“ Nachricht muss also ebenfalls gerade auf dem Bus übertragen werden. Daraufhin beendet Netzknoten A seinen Übertragungsversuch und kehrt in den Empfangsmodus zurück. Er wird zu einem späteren Zeitpunkt erneut versuchen, seine Nachricht zu senden. Netzknoten B sendet hingegen weiter seinen Datenrahmen.

Da einzig die „unterlegenen“ Netzknoten die Kollision bemerken, hat das CSMA/CR-

Verfahren den Vorteil, dass die Nachricht mit dem höchsten Wert im Arbitrierungsfeld zerstörungsfrei übertragen wird. Außerdem erlaubt dies eine Priorisierung der Nachrichten, was essenziell für den Echtzeitbetrieb ist.

9 Echtzeiteigenschaften

In der Informatik bezeichnet der Begriff „*Echtzeitsystem*“ ein Hard- oder Softwaresystem, welches *Echtzeitbedingungen* unterliegt [LiuJ00]. Dies bedeutet, dass das System in der Lage sein muss, in einer genau festgelegten Zeitspanne auf eine Eingabe zu reagieren. Auch wenn dies nicht heißt, dass ein Echtzeitsystem auf der Stelle antwortet, so ist doch garantiert, dass ein Vorgang in einem bestimmten Zeitintervall nach dessen Initiierung abgeschlossen ist (relatives System) oder zu festgelegten Zeitpunkten stattfindet (absolutes System) [HoHa91].

Im Gegensatz dazu arbeitet ein konventionelles System ohne diese *Deadlines* und liefert ein Resultat, sobald der Bearbeitungsprozess beendet ist. In so genannten *weichen Echtzeitsystemen* führt das Fehlen einer Deadline dazu, dass das Ergebnis am Ende wertlos ist, wenn der Vorgang zu lange dauert. Benötigt z. B. ein Telefon für die Verarbeitung der Sprachsignale mehrere Minuten, kann – aufgrund der Verzögerung – keine gewohnte Kommunikation über die Verbindung stattfinden.

Kritisch hingegen sind so genannte *harte Echtzeitsysteme*. Ist hier das Resultat nicht nach der definierten Zeitspanne verfügbar, bspw. die Aktivierung eines Airbags bei einem Autounfall, endet der Prozess in einem Desaster.

Verstehen wir Computernetzwerke als Hard- und Softwaresysteme, können Aspekte der Echtzeitkommunikation von Bedeutung sein. Ist dies der Fall, so muss garantiert werden, dass die zu sendenden Nachrichten innerhalb einer festgelegten Zeitspanne ihr Ziel erreichen und dort verarbeitet werden.

Das Controller Area Network bietet, bis zu einem gewissen Grad, die Grundlage für harte und weiche Echtzeitkommunikation in Netzwerken. Da das CAN jedoch einzig bis zur OSI-Schicht 2 beschrieben ist, muss die Steuerung des Echtzeitsystems auf den höheren Ebenen erfolgen [ISO404]. Konkret weist der Standard folgende Eigenschaften für den Betrieb eines Echtzeitsystems auf:

- **Ausfallsicherheit:** Generell sind auf physikalischer Ebene die Toleranzen für die Detektion der beiden logischen Zustände relativ hoch. Der Low-Speed-Modus wartet zudem mit der Besonderheit auf, dass sogar Kabelbrüche in einem bestimmten Umfang kompensiert werden können. So ist es mit großer Wahrscheinlichkeit möglich ein System weiter zu

betreiben, selbst wenn eine Kupferader¹² unterbrochen ist.

- **Priorisierbarkeit:** Das im CSMA/CR-Algorithmus enthaltene Arbitrierungsverfahren ermöglicht es, Nachrichten eine Priorität zu vergeben und sie somit gegenüber anderen zu bevorzugen oder zu benachteiligen. Dies erlaubt die Einrichtung eines *Schedules*¹³, welcher jedoch auf den höheren Schichten durch einen Algorithmus realisiert werden muss. Ferner ist die Lösung von Kollisionen auf dem Bus zerstörungsfrei realisiert, wodurch es nicht zu unnötigem Paketverlust kommt. Die relativ geringe *Payload*¹⁴ von maximal 64 Byte pro Frame verhindert zwar eine lange Auslastung durch nur einen einzigen Rahmen (wie bspw. beim *Ethernet* [Lamm08]), allerdings kann eine hohe Anzahl von Netzknoten bzw. eine zu geringe Übertragungsrate ebenfalls zu einer überhöhten Belastung des Busses führen. Daher ist bei der Dimensionierung des Netzwerks auf jeden Fall auf Leitungslänge und Knotenanzahl zu achten¹⁵.
- **Fehlererkennung:** Die Netzknoten im CAN besitzen die Fähigkeit fehlerhafte Nachrichten zu identifizieren und diese zu melden. Möglich ist dies einerseits über die in jedem Data Frame und Remote Frame enthaltene Prüfsumme, sowie bei Missachtung der Codierungsregel. In jedem Fall wird der Hinweis auf die Störung mittels eines Error Frames übertragen, welcher in der Lage ist, die fehlerhafte Kommunikation sofort zu unterbrechen. Damit defekte Netzknoten nicht ständig Fehlermeldungen versenden, verfügt das CAN darüber hinaus über die Möglichkeit, diesen Geräten eine eingeschränkte Konnektivität zu verleihen oder sie ganz vom Netz zu trennen¹⁶.

10 Fazit

Obwohl das Controller Area Network auf eine Entwicklung zurückgeht, die bei Erscheinen dieser Projektarbeit bereits über 20 Jahre zurückliegt, bietet das Bussystem immer noch erstaunlich gute Voraussetzungen für die heutige Datenkommunikation. Zwar ist das Verhältnis von Nutz- zu Steuerdaten in CAN Data Frames im Vergleich zu bspw. Ethernet relativ niedrig. Da die Einsatzgebiete, und damit auch die Anforderungen, jedoch völlig andere sind, ist dies in der Regel eher ein Vorteil. Denn die relativ kurzen Datenrahmen ermöglichen eine bessere Planbarkeit der Sendereihenfolge (Schedule).

12 Siehe: 6 *Netzwerktopologie*.

13 Engl. hier: „Ablaufplan“. Gemeint ist die Reihenfolge, in der Nachrichten verschickt werden müssen, damit möglichst alle Deadlines eingehalten werden können.

14 Engl.: „Nutzlast“. Gemeint ist der Teil eines Nachrichtenrahmens der Informationen der höheren Schichten enthält.

15 Siehe: 6 *Netzwerktopologie*.

16 Siehe: 7.3.4 *Error Frame*.

Was das CAN gegenüber anderen Netzwerktechnologien hervorhebt, ist seine Eignung für Echtzeitumgebungen. Auf der logischen Ebene ist der Standard so durchdacht, dass ein Schedulingalgorithmus, bei richtiger Implementierung, sogar die Bedingungen für harte Echtzeitsysteme einhalten kann. Dies wird insbesondere durch die verlustfreie Arbitrierung, aber auch durch das Fehlererkennungs- und -meldungssystem ermöglicht. Darüber hinaus bietet die physikalische Ebene ausreichend Spielraum, um Beschädigungen des Netzes bis zu einem gewissen Umfang zu kompensieren. Allerdings müssen bei der Dimensionierung des Systems relativ strikte Vorgaben zu Leitungslänge, Netzknotenanzahl und Übertragungsrate eingehalten werden, um einen ausreichend stabilen Betrieb des Busses zu gewährleisten. Nur so ist es überhaupt denkbar, einen digitalen Bus in einer so kritischen Umgebung wie dem Automobil einzusetzen.

Dennoch hat das Controller Area Network nicht nur Vorteile. Mittlerweile sind andere Netzwerktechnologien entwickelt worden, die deutlich höhere Übertragungsraten bieten (z. B. *FlexRay* [Joch07]). Außerdem enthält der Standard an vielen Stellen lediglich Richtwerte, wie z. B. bei den Spannungspegeln, und erlaubt gleichzeitig teilweise großzügige Abweichungen von diesen. Ferner weist das Bussystem von sich aus keinerlei Interoperabilität auf, welche erst durch Zusatzgeräte (*Bridges*) ermöglicht wird. Und letztlich bietet selbstverständlich auch das CAN keine vollständige Ausfallsicherheit, da nicht gewährleistet werden kann, dass bei hohen physikalischen Beschädigungen der Betrieb aufrecht erhalten bleibt. In sehr ungünstig konstruierten Szenarien würde auch ein CAN somit nicht das nötige Maß an Sicherheit bieten.

Zusammenfassend lässt sich sagen, dass das Controller Area Network eine durchdachte und gereifte Technologie ist, die völlig zu recht ihren Weg auch in andere Einsatzgebiete gefunden hat. Sie mag zwar nicht mehr dem neusten Stand der Technik entsprechen, bietet für die Automatisierung, für die Medizintechnik und auch für den Fahrzeugbau in der Regel aber dennoch eine mehr als ausreichende Grundlage für gesicherte Datenkommunikation. Hinzu kommt, dass die CAN-Hardware, aufgrund der großen Produktionsmenge und der langjährigen Verfügbarkeit, ausgereift und preisgünstig ist. Und, sofern die Entwicklung der Spezifikation wie in den letzten Jahren weiter betrieben wird, steht einer Zukunft des Controller Area Networks nichts im Wege.

Literaturverzeichnis

- [Enge00] Engels, H.: "CAN-Bus", Franzis, Poing, 2000, ISBN: 978-3772351464.
- [Etsco0] Etschberger, K.: "Controller-Area-Network", Hanser, München, 2000, ISBN: 978-3446194311.
- [Gruh01] Gruhler, G.: "Feldbusse und Geräte-Kommunikationssysteme", Franzis, Poing, 2001, ISBN: 978-3772357459.
- [HoHa91] Hoogeboom, B. / Halang, W.: "The Concept of Time in Software Engineering for Real Time Systems", University of Groningen, IEEE, 1991.
- [ISO103] International Organization for Standardization: "Road vehicles - Controller area network (CAN) - Part 1: Data link layer and physical signalling", ISO, Genève, 2003, Referenznummer: ISO 11898-1:2003(E).
- [ISO203] International Organization for Standardization: "Road vehicles - Controller area network (CAN) - Part 2: High-speed medium access unit", ISO, Genève, 2003, Referenznummer: ISO 11898-2:2003(E).
- [ISO306] International Organization for Standardization: "Road vehicles - Controller area network (CAN) - Part 3: Low-speed, fault-tolerant, medium-dependent interface", ISO, Genève, 2006, Referenznummer: ISO 11898-3:2006(E).
- [ISO404] International Organization for Standardization: "Road vehicles - Controller area network (CAN) - Part 4: Time-triggered communication", ISO, Genève, 2004, Referenznummer: ISO 11898-4:2004(E).
- [ISO507] International Organization for Standardization: "Road vehicles - Controller area network (CAN) - Part 5: High-speed medium access unit with low-power mode", ISO, Genève, 2007, Referenznummer: ISO 11898-5:2007(E).
- [Jans00] Jansen, W. / Phoenix Contact: "Grundkurs Feldbustechnik", Vogel, Würzburg, 2000, ISBN: 978-3802318139.
- [Joch07] Jochim, M.: "Zeitig steuern - Sichere Datenübertragung im Automobil", c't, Verlag Heinz Heise, Hannover, 02/2007, Seite 190 ff., ISSN: 0724-8679.
- [Lamm08] Lammermann, S.: "Ethernet as a Real-Time Technology", Hochschule für Telekommunikation, Leipzig, 17. Juni 2008, <http://www.lammermann.eu/wb/pages/arbeiten/ethernet-as-a-real-time-technology.php>.
- [LiuJ00] Liu, J.: "Real-Time Systems", Prentice Hall, Upper Saddle River, 2000, ISBN: 978-0130996510.
- [Maye06] Mayer, E.: "Datenkommunikation im Automobil - Teil 2: Sicherer Datenaustausch mit CAN", Elektronik automotive, WEKA Fachmedien, Poing, 08/2006, Seite 34 ff., ISSN: 1614-0125.
- [Will93] Williams, R.: "A Painless Guide to CRC Error Detection Algorithms", University of Adelaide, Adelaide, 19. August 1993, http://www.ross.net/crc/download/crc_v3.txt.

Lizenz

Dieses Dokument unterliegt einer Creative-Commons-Lizenz (BY-ND). Zusammenfassung:
Sie dürfen:

- Das Werk vervielfältigen, verbreiten und öffentlich zugänglich machen

Zu den folgenden Bedingungen:

- **Namensnennung.** Sie müssen den Namen des Autors/Rechteinhabers in der von ihm festgelegten Weise nennen (wodurch aber nicht der Eindruck entstehen darf, Sie oder die Nutzung des Werkes durch Sie würden entlohnt).
- **Keine Bearbeitung.** Dieses Werk darf nicht bearbeitet oder in anderer Weise verändert werden.
- Im Falle einer Verbreitung müssen Sie anderen die Lizenzbedingungen, unter welche dieses Werk fällt, mitteilen. Am Einfachsten ist es, den unten aufgeführten Hyperlink anzugeben.
- Jede der vorgenannten Bedingungen kann aufgehoben werden, sofern Sie die Einwilligung des Rechteinhabers dazu erhalten.
- Diese Lizenz lässt die Urheberpersönlichkeitsrechte unberührt.

Details der Lizenz im Internet: <http://creativecommons.org/licenses/by-nd/2.0/de/deed.de>.

Besuchen Sie <http://www.lammermann.eu> für weitere freie Dokumente.