Datenkommunikation

CAN
(Controller Area Network)
Kommunikation im Automobil

Anton Federl
8T, 14072300

Serein Pfeiffer
8T, 09

_____
Anton Federl

_____
Serein Pfeiffer

Inhalt

1. Überblick, Motivation

1.1. Motivation

In den letzten Jahren hat die Elektronik in Automobilen immer mehr an Bedeutung gewonnen. So sind heute in einem Mittelklassewagen bis zu 70 Steuergeräte verbaut, die die verschiedensten Funktionen übernehmen (Motorsteuerung, Automatikgetriebesteuerung, elektrische Fensterheber, Klimaanlage und vieles mehr). Dadurch entsteht (mit konventioneller Verkabelung) ein hoher Verkabelungsaufwand mit vielen Steckverbindungen (etwa 1km Kabel und rund 1000 Steckverbindungen sind in einem aktuellen Fahrzeug verbaut). Vor Allem durch die vielen Steckverbindungen ergibt sich eine hohe Fehleranfälligkeit (Korrosion, Lockerung der Verbindungen, …), aber auch Kabelbruch ist ein großes Thema. Die Fehlersuche wird dabei extrem aufwendig.
Als Alternative bietet sich hier ein Bussystem an, das aber gewissen Anforderungen genügen muss (Fehler- und Ausfallsicherheit, Echtzeitfähigkeit, Flexibilität, …).

BLD001: Kabelbaum

1.2. Bussysteme im Automobil

In Automobilen werden verschiedene Bussysteme (je nach Anforderungen und entstehenden Kosten) verbaut. Hier die wichtigsten:

- *FlexRay* (bitorientiert, voll echtzeitfähig, 10 Mbits/s) für besonders sicherheitsrelevante Anwendungen.
- *Interbus*-S (Sensor-/Aktorbus) für einfache Sensoren und Aktoren bei zeitkritischen Anwendungen.
- *MOST* (Media Oriented Systems Transport Bus, hohe Übertragungsrate: 24.8 Mbits/s) für Multimedia Anwendungen (Radio, CD-Wechsler, Telefon, Fernsehdisplay, …).
- ***CAN* (Controller Area Network, ereignisgesteuert)** für Steuerungs- und Regelungsfunktionen sowie für Komfortfunktionen (siehe Praxisbeispiele).
- ***TTCAN* (Time Triggered CAN)** für Sicherheitsrelevante Anwendungen. Es wird nicht wie bei CAN ereignisgesteuert sondern mit „Zeitfenstern“ gearbeitet, so dass jedes Steuergerät eine garantierte Zeit zur Verfügung hat. Es können freie Zeitfenster eingesetzt werden, so dass das ereignisgesteuerte CAN Modell eingebunden werden kann. Somit ist TTCAN eine voll Echtzeitfähige Erweiterung von CAN.

1.3. Grundsätzliches zu CAN

Geschichte:

Der CAN Bus wurde Anfang der 80er gemeinsam von Bosch und Intel entwickelt und wurde schließlich 1985 vorgestellt.
Anfang der 90er fand CAN dann bereits Einsatz in Serienautomobilen. So wurde 1992 die S-Klasse von Mercedes-Benz zum Ersten Mal mit dieser Technologie ausgestattet. Damals beschränkte sich der Einsatz jedoch vor Allem auf Motor- und Getriebesteuerung sowie der Schaltung der Instrumente und Anzeigen am/im Armaturenbrett.
Seit etwa 2000 wird CAN sogar in Autos der unteren Mittelklasse eingesetzt. Mittlerweile wird CAN von fast allen Automobilherstellern verwendet und findet auch in vielen anderen Gebieten Anwendung (siehe Praxisbeispiele).

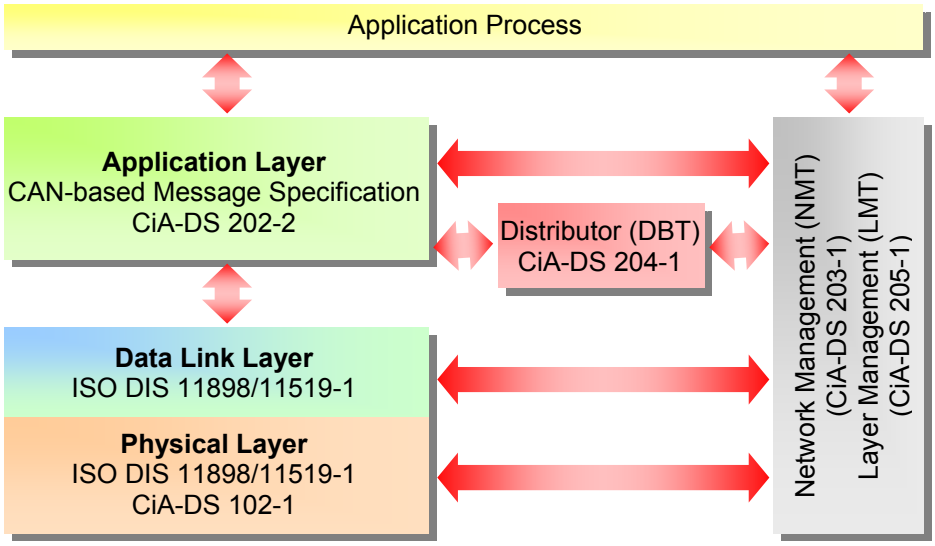
Einordnung:

Der CAN Bus ist im Wesentlichen ein Sensor-/Aktorbus, zählt aber auch zum Teil zu den Feldbussen. Er ist vorrangig für Echtzeitmeldungen konzipiert.

Standardisierung:

Die physikalischen Eigenschaften sowie das CAN-Protokoll wurden in ISO DIS 11898/11519-1 festgelegt.
Es ist zudem der Nutzerverein CiA (CAN in Automation) entstanden, welcher zusätzliche/einschränkende eigene Normierungen für fehlende Normen festlegt.

1.4. OSI Schichtenmodell



- *OSI Schicht 1:* Physical Layer: ISO DIS 11898/11519-1, CiA-DS 102-1
- *OSI Schicht 2:* Data Link Layer: ISO DIS 11898/11519-1
  - *2a:* MAC (Medium Access Control Zugriffsverfahren)
  - *2b*: LCC (Logical Link Control Sicherungsverfahren)
- OSI Schicht 7: Application Layer: CiA-DS 202-2, noch nicht international standardisiert.
-

Weitere Schichten sind bisher nicht international standardisiert.

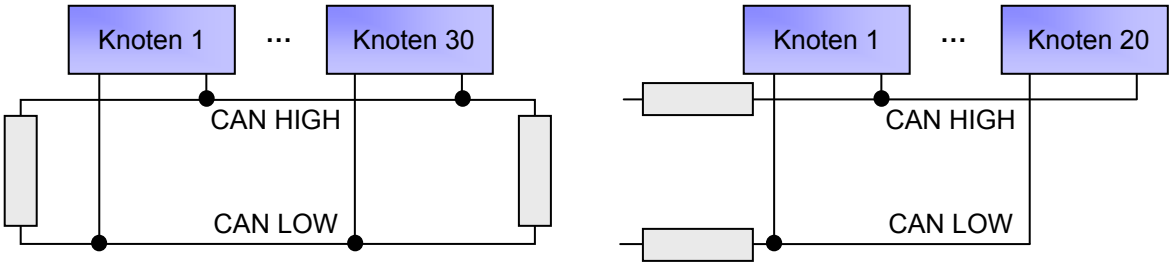
2. Hardware

2.1. Medium Dependend Interface

Es sind grundsätzlich zwei verschiedene Realisierungsmöglichkeiten der Hardware des CAN-Busses:

CAN-High-Speed (ISO-DIS 11898)
- Datenrate 125kBit/s bis 1Mbit.
- 2 bis 30 Knoten pro Netzwert.
- Kurzschlussfestigkeit im Bereich -3 bis 16 Volt.
- Sendeausgangsstrom > 25mA.
- Typische Leitungsimpedanz 120Ω

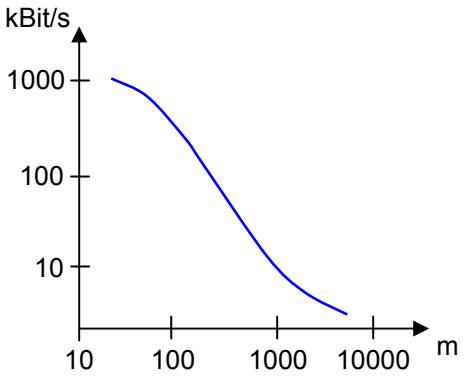
CAN-Low-Speed (ISO-DIS 11898)
- Datenrate bis 125kBit/s
- 2 bis 20 Knoten pro Netzwerk.
- Kurzschlussfestigkeit im Bereich -6 bis 16 Volt.
- Sendeausgangsstrom > 1mA.
- Maximale Leitungslänge ist auch von der kapazitiven Belastung der Busleitung abhängig.



Für beide Varianten gilt:
- 5 Volt Stromversorgung.
- Common Mode Spannungsbereich[1] -2 bis 7 Volt.
- Symmetrische Signalübertragung über Zweidrahtleitung mit Rückleitung.
- Terminierung mit 120Ω an beiden Enden, um Reflexionen zu vermeiden. Eine Terminierung ist immer notwendig, da der Widersand gleichzeitig als Pullup- bzw. Pulldownwiderstand eingesetzt wird (siehe Aufbau der Busteilnehmer). In der Praxis reicht es oft aus, nur Ende zu terminieren.
- Eine Schirmung ist nicht vorgeschrieben, kann aber eingesetzt werden.
- Linienförmige Topologie, bei kurzen Leitungen auch sternförmig möglich.

2.2. Übertragungsraten und Kabellänge

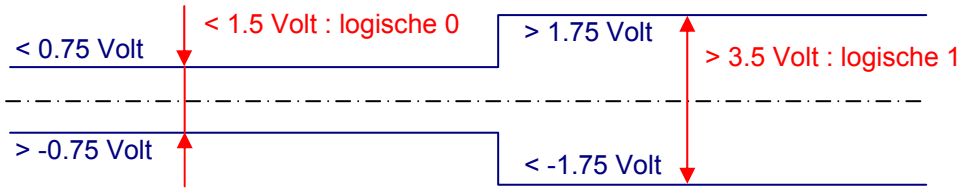
Übertragungs rate	Leitungslänge (beidseitig terminiert)
10 kBit/s	6,7 km
20 kBit/s	3,3 km
50 kBit/s	1,3 km
125 kBit/s	530 m
250 kBit/s	270 m
500 kBit/s	130 m
1 MBit/s	40 m



[1] *Common Mode Spannungsbereich:* Spannungsbereich des differentiellen Signals.

2.3. Physikalisches Signal

CAN-LOW führt immer die negative Spannung zu CAN-HIGH.
Eine logische 0 oder 1 entsteht durch Differenzspannungsmessung (es handelt sich um ein differentielles Signal).
Ist die Spannungsdifferenz zwischen CAN-LOW und CAN-HIGH kleiner als 0.75 Volt, so liegt eine logische 0 vor. Ist die Differenz größer als 3.5 Volt, so handelt es sich um eine logische 1. Eine Spannungsdifferenz zwischen 1.5 und 3.5 Volt wird als Fehler erkannt.

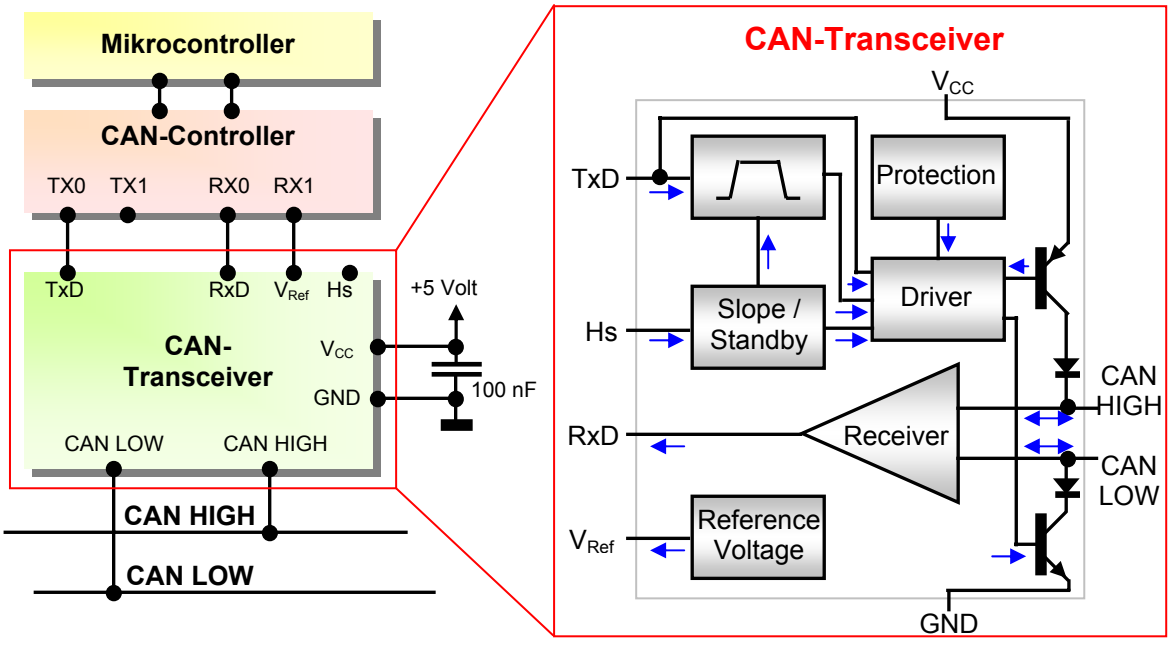


Signalcodierung: NRZ (Non Return to Zero) mit Bitstuffing (Weite 5).
Synchronisation bei verschiedenen Busgeschwindigkeiten sowie Resynchronisation ist auf Grund eines Synchronisationssegments zu Beginn jeder Nachricht möglich.
Die Verwendung eines differentiellen Signals hat den Vorteil der Gleichtaktunterdrückung (CMRR) im Falle von Störsignalen (Störung wirkt auf beiden Leitungen in dieselbe Richtung und im selben Maße, wodurch die Spannungsdifferenz annähernd gleich bleibt).

2.4. Aufbau der Busteilnehmer und Hardwareschnittstelle

Aufbau (Schematisch):

Busteilnehmer (Mikrocontroller) werden mittels eines CAN-Controllers und eines CAN-Transceivers an die CAN-Busleitung angeschlossen.



CAN HIGH	CAN LOW	Differenz	Logisch	Zustand
Transistor geöffnet	Transistor geschlossen (zieht auf GND)	5 Volt	**0**	dominant
Transistor geschlossen (zieht auf V_{CC})	Transistor geöffnet	0 Volt	**1**	rezessiv
Transistor geöffnet	Transistor geöffnet		Tristate	floating

Durch die Schaltung des CAN-Tansceivers können mehrere Teilnehmer gleichzeitig auf den Bus zugreifen, ohne dass ein Kurzschluss entsteht. Zudem ergibt sich, dass eine logische 0 dominant und eine logische 1 rezessiv ist (eine 0 überschreibt eine 1). Diese Tatsache wird vor Allem bei der Bus Arbitrierung ausgenutzt.

Kabel und Steckverbindungen:

Zur Verkabelung reicht ein dreipoliges Kabel (CAN HIGH, CAN LOW und Masse) aus. Es gibt 3, 4, 5 9 und 10polige Stecker (ab 4polig wird die Versorgungsspannung mitgeführt).
Durchgesetzt hat sich vor Allem der von der Organisation CiA vorgeschlagene 9polige Sub-D Stecker. Dieser hat folgende Belegung:

Pin	Signal
1	
2	CAN LOW
3	CAN GND
4	
5	CAN SHLD (optionale Schirmung)
6	GND (Gerätemasse)
7	CAN HIGH
8	
9	V_{CC} (Versorgungsspannung)

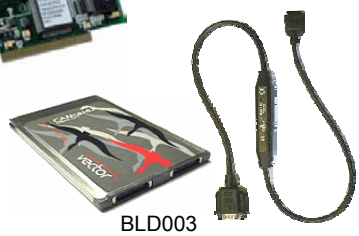
Einbindung eines Computers:

Um einen Rechner mit in den CAN Bus einzubinden gibt es verschiedene Hardware, welche jeweils einen CAN Transceiver und einen CAN Controller enthalten. Meist sind die beiden Einheiten sogar zweimal vorhanden.
Hier einige Beispiele:

- ISA bzw. PCI Steckkarten für Laborrechner.


BLD002

- PCMCIA Karten für mobile Rechner.



BLD003

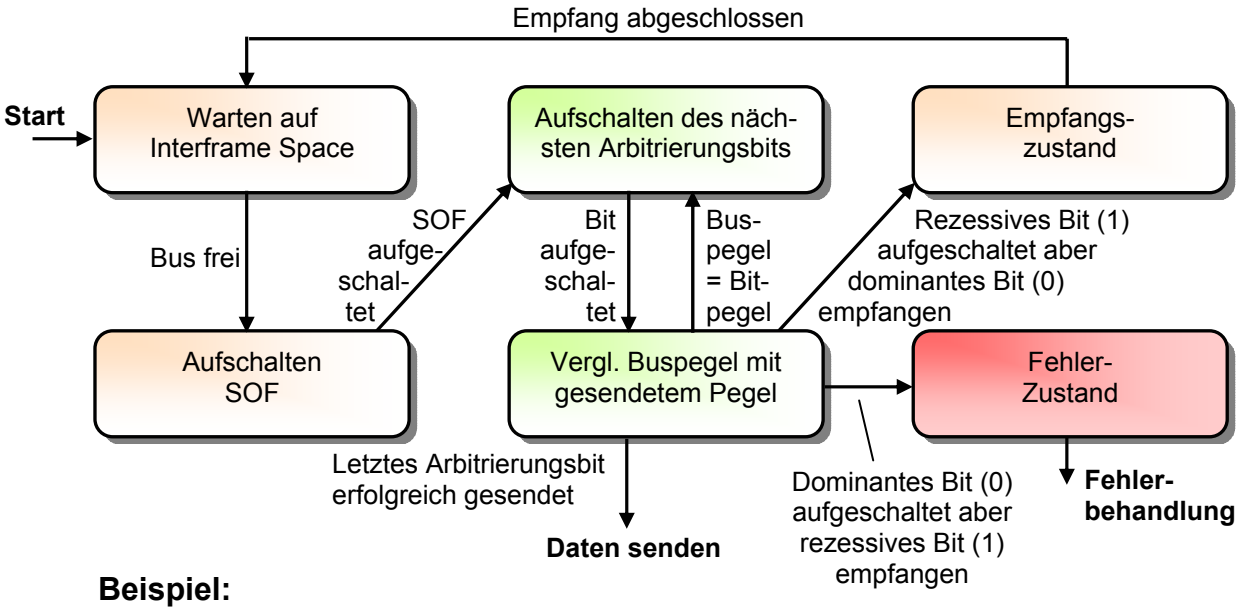
- USB Adapter für flexiblen Einsatz.

BLD004

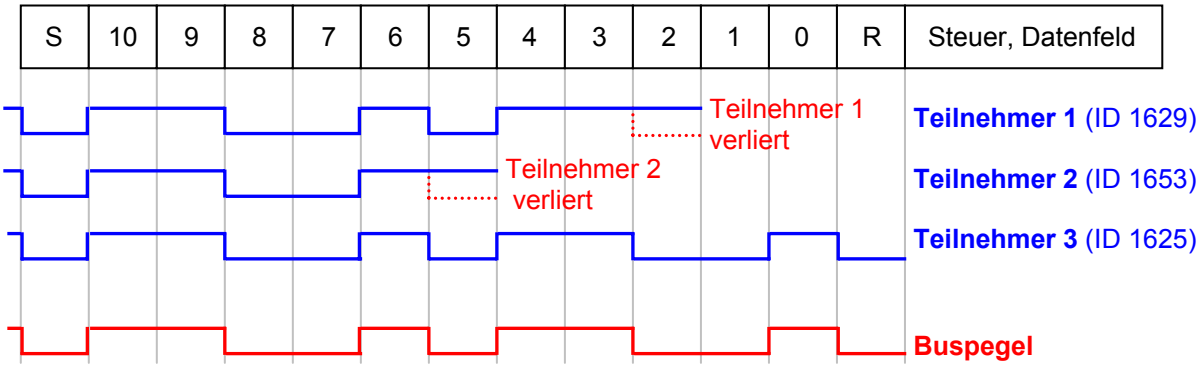
2.5. Bus Arbitrierung

Alle Busteilnehmer arbeiten im Multi-Master Betrieb. Deshalb ist eine spezielle, bitweise Arbitrierung erforderlich. Dadurch, dass eine logische 0 dominant ist (eine logische 1 eines anderen Teilnehmers überschreibt), kann gleichzeitig eine Priorisierung angewendet werden, wobei die Nachricht mit der höchsten Priorität ohne Zeitverzögerung gesendet wird.
Jeder Teilnehmer, der senden will, wartet zunächst, bis der Bus frei ist (mindestes 10 Bitzeiten rezessiv). Dann sendet er zunächst ein Startbit und dann den Identifier der entsprechenden Nachricht, der gleichzeitig die Priorität darstellt (je niedriger die ID, desto höher die Priorität).
Beim Senden des Identifiers empfängt der Teilnehmer gleichzeitig. Solange genau der Pegel auf dem Bus anliegt (logische 1 oder 0), die auch von ihm gesendet wurde, wird weiter gesendet (wenn ein anderer Teilnehmer gleichzeitig sendet, so hat er bislang auch die gleichen Bits gesendet). Wird jedoch eine logische 1 gesendet und eine logische 0 empfangen, so hat der Teilnehmer verloren (ein anderer Teilnehmer mit höherer Priorität bzw. kleinerem Identifier hat den Bus auf eine logische 0 gezogen). Alternativ könnte auch ein Leitungsfehler aufgetreten sein.
Konnte ein Teilnehmer die ID seiner Nachricht vollständig senden und empfangen, so har er nun Zugriff auf den Bus (gleiche IDs für verschiedene Nachrichten dürfen nicht vergeben werden.

Zustandsautomat für die Arbitrierung:



Beispiel:



3. Protokoll

3.1. Nachrichtenformat/Frames

Für den CAN Bus wird ein ereignisgesteuertes (bedarfsorientiertes) Protokoll verwendet, bei dem sog. *Botschaften* verschickt werden. Diese werden auch häufig als Nachrichten oder Telegramme bezeichnet. Die Botschaften werden mittels eines Frames verschickt.
Es gibt zwei verschiedene Spezifikationen für die Frames:

CAN2.0A (Standard Frame):

Start 1 Bit	Identifier 11 Bit	RTR 1 Bit	IDE 1 Bit	r_0 1 Bit	DLC 4 Bit	Daten 0 bis 8 Byte	CRC 15+1 Bit	ACK 2 Bit	EOF 7 Bit

CAN2.0B (Extended Frame):

Start 1 Bit	Identifier 11 + 18 Bit SRR 1 Bit / IDE 1 Bit	RTR 1 Bit	r_1 1 Bit	r_0 1 Bit	DLC 4 Bit	Daten 0 bis 8 Byte	CRC 15+1 Bit	ACK 2 Bit	EOF 7 Bit

Beschreibung der Felder:

- *Start:* Dominantes Bit (logische 0), dient zur Synchronisation.
- *Identifier:* ID der Botschaft und gleichzeitig Priorität (siehe Arbitrierung).
- *RTR:* (Remote Transmission Request) Unterscheidung zwischen Datentelegram (logische 0, dominant) und Datenanforderungstelegramm (logische 1, rezessiv).
- *SSR:* Ersetzt das RTR-Bit des Standard Frame (immer rezessiv).
- *IDE:* Identifier Extension, zeigt an, dass noch weitere 18 Bits folgen. Somit kann der Extended Frame vom Standard Frame unterschieden werden (dominant: Standardframe).
- r_0 , r_1: Reserviert.
- *DLC:* (Data Length Code) Längeninformation für den Datenblock.
- *Daten:* Datenblock, kann 0 bis 64 Bit lang sein.
- *CRC:* (Cyclic Redundancy Check) CRC Prüfsumme + ein rezessives Begrenzungsbit (logiche 1).
- *ACK:* (Acknowledge) Rückmeldung der anderen Teilnehmer + 1 rezessives Begrenzungsbit (logische 1).
- *EOF:* (End Of Frame) Sieben rezessive Bits (logische 1, ohne Bitstuffing).

Nach einem kompletten Rahmen folgen immer noch drei rezessive Bits (Interframe Space), bis der nächste Block beginnen darf.

Für die beiden Spezifikationen gibt es verschiedene Controller. Ältere CAN2.0A Controller können nicht in einem CAN2.0B Bus verwendet werden. Es gibt jedoch neuere CAN2.0A Controller, die den Extended Frame erkennen und passiv bleiben (Frame ignorieren, keinen Fehler melden). Diese können dann in einem solchen Bus gleichzeitig betrieben werden.
Die Unterscheidung in eines Standard Frames von einem Extended Frame geschieht durch das IDE Bit (ist es rezessiv, so handelt es sich um einen Extended Frame.
Priorität: Bei CAN2.0B ist das Bit, das bei CAN2.0A das RTR Bit repräsentiert (SRR) immer rezessiv (hat nie Vorrang). Da danach das IDE Bit bei CAN2.0A dominant ist gewinnt immer ein kurzer Identifier (CAN2.0A) gegenüber einem CAN2.0B Identifier, egal ob Datenanforderungs- oder Datenframe.

Frame-Formate:

- *Datenframe:*
  Sender sendet Daten (von sich aus). Das *RTR*-Bit ist auf 0 gesetzt und ist somit dominant (gewinnt bei der Arbitrierung gegenüber einem Datenanforderungsframe mit gleicher ID.
- *Datenanforderungsframe:*
  Ein Empfänger fordert Daten mit dem entsprechenden *Identifier* an. Das *RTR*-Bit ist auf 1 gesetzt (rezessiv) und ist somit rezessiv.
  Ein Datentelegramm mit gleicher ID hat somit Vorrang.
- *Fehlerframe:*
  Signalisierung eines erkannten Fehlers durch den Sender oder Empfänger. Der Frame wird direkt nach dem Datenfeld des gesendeten Datenframes gesendet und beginnt sofort mit 6 bis 12 dominanten Bits (logische 0, kein Startbit, kein Bitstuffing). Dann wird der Frame mit 8 rezessiven Bits beendet. Das führt zu einem negativen Acknowledge. Ein Fehlerframe ist immer dominant (kann nicht überschrieben werden).
- *Überlastframe:*
  Zur Realisierung einer Verzögerung zwischen Datenanforderungs-frames kann dieser Frametyp verwendet werden. Es werden 6 bis 12 dominante Bits (logische 0) nach Ende eines Datenframes (an Stelle des Interfame Space) gesendet und dann wird der Frame mit 8 rezessiven Bits beendet. Eine Verzögerung kann sinnvoll sein um Pufferüberläufe zu verhindern oder das Senden einer hoch prioren Nachricht zu ermöglichen, die noch nicht bereit ist (wird eine nieder priore Nachricht gerade gesendet, so kann sie nicht unterbrochen werden, was ggf. zu längeren Latenzzeiten führen kann.

3.2. Fehlererkennung/-behandlung

Fehlererkennung auf Bit-Ebene:

- *Bitstuffing:*
  Nach fünf aufeinander folgenden Bits wird ein invertiertes Bit eingefügt. Dies wird von den Empfängern überwacht und sie entfernen das Bit wieder. Bitstuffing wird in Sonderfällen nicht eingesetzt (siehe oben).
- *Bit Monitoring:*
  Während der Sender sendet empfängt er gleichzeitig. falls die empfangenen Bits nicht mit den gesendeten übereinstimmen, so wird ein Fehler erkannt (außer bei der Arbitrierung).

Fehlererkennung auf höherer Ebene (Protokoll):

- *Message Frame Check:*
  Überprüfung der Empfänger, ob die Frames richtig aufgebaut sind. Dies kann mit den rezessiven Begrenzungsbits festgestellt werden.
- *CRC-Prüfung:*
  Überprüfung der CRC-Checksumme.
  Generatorpolynom: $x^{15} + x^{14} + x^{10} + x^8 + x^7 + x^4 + x^3 + 1$.
- *Acknowledge-Prüfung:*
  Der Sender Prüft, ob ein dominantes Bit (logische 0) in das ACK-Feld geschrieben wurde.

Durch diese Mechanismen kann eine sehr geringe Fehlerwahrscheinlichkeit erreicht werden (Hammingdistanz 6, Restfehlerwahrscheinlichkeit 10^{-13}). Fehlerkorrektur mit CRC wird bei CAN nicht durchgeführt.

Durch ein aufgesetztes Netzwerkmanagement kann eine Fehlerquelle (häufiges Auftreten von Fehlermeldungen eines Controllers) eingegrenzt werden und der betroffene Controller kann ggf. automatisch abgeschaltet werden.

3.3. Latenzzeiten

Die Systemerholung nach einem Fehler erfordert im Regelfall 17 bis 23 Bitzeiten, in Sonderfällen 29 Bitzeiten (bei 500 KBit/s etwa 40 bis 60µs), bis wieder eine Nachricht gesendet bzw. die fehlerhafte Nachricht wiederholt werden kann.

Die maximale Reaktionszeit der höchst prioren Nachricht kann berechnet werden (Beispiel für den Standard-Frame unter der Annahme, dass keine Fehler auftreten und nur CAN2.0A Controller existieren):

Bis zu 130 Bitzeiten, bis der Bus frei ist (aktuell gesendete Nachricht).
66 Bitzeiten für ein Datenanforderungstelegramm und 130 Bitzeiten für das Senden der Nachricht.
Insgesamt ergeben sich maximal 326 Bitzeiten (entspricht 0.65 ms bei 500 KBit/s).

Für niedriger priorisierte Nachrichten kann keine allgemeine Angabe gemacht werden. Im Extremfall werden sie durch viele hochpriore Nachrichten (hohe Busauslastung) ggf. komplett verdrängt.

3.4. DBC (CAN DataBase Container)

In der Industrie werden oft DBC-Dateien verwendet, um die Botschaften zu spezifizieren und ohne Konvertierungsaufwand auf die Controller zu übertragen.

```
VERSION "DNNBNNNYNN…NNNNNNNNN/4/%%%/4/'%**4NNN///"
```
} Versionsstring

```
NS_ :
      NS_DESC_
      CM_
      BA_DEF_
      …
```
} Sektion mit zusätzlichen Schlüsselwörtern

```
BS_:
```
} Beginn der Botschaftssektion

```
BU_: DME  DGS DIS …
```
} Sender- und Empfängerdeklaration:
DME: Digitale Motorelektronik
DGS: Digitale Getriebesteuerung
DIS: Anzeigegerät (Armaturenbrett)

Identifier/Priorität | Botschaftsname | Anzahl der Sendebytes | Sender

```
BO_ 1288 DME_1: 8 DME
```
} Botschaftsdefinition

SG_ Signalname : Startbit|Länge@ns (Faktor, Offset) [Min|Max] “Einheit“ EMPFÄNGERSET

n = 1 → Intel Format
n = 0 → Motorola Format
s = + → Unsigned
s = – → Signed

```
SG_ N_MOT : 0|16@1+ (0.25,0) [0|32767] "1/min"  DGS DIS
SG_ N_AB : 0|16@1+ (0.5,0) [0|32767] "1/min"  DGS DIS
```
} Signaldef.

```
BO_ 1198 DGS_1: 8 DGS
 SG_ Z_GANG : 0|8@1- (1,0) [-1|7] "" DIS
 SG_ RES : 8|24@1+ (1,0) [0|0] ""

…
```

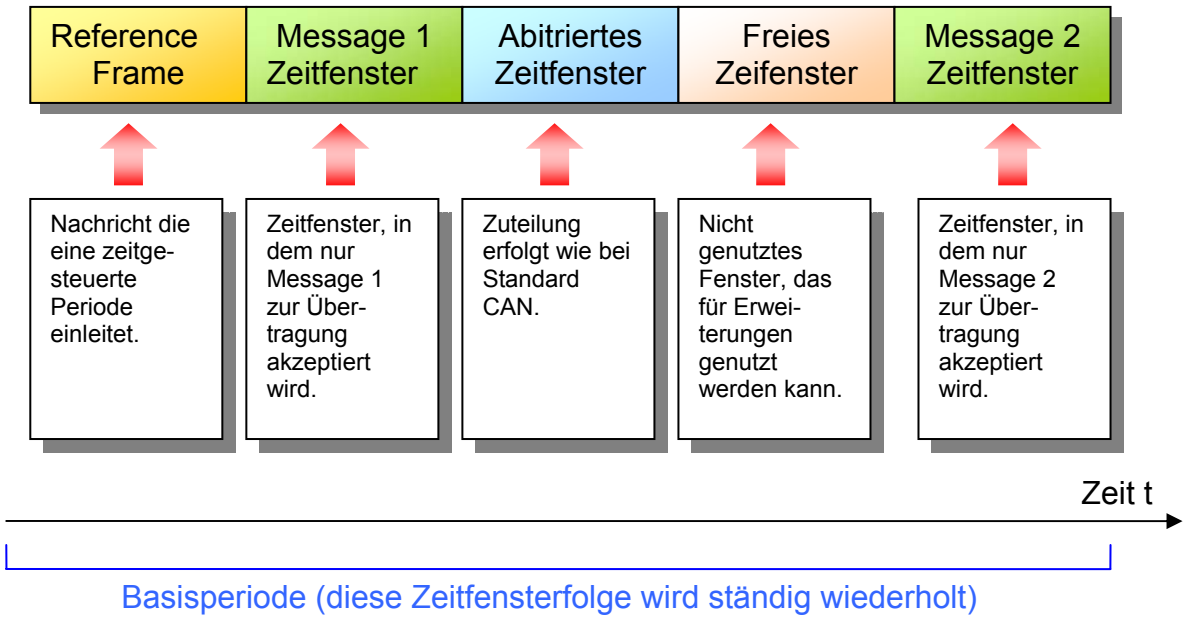
Diese Dateien müssen nicht per Hand erstellt werden. Es gibt Leistungsfähige Tools, die auch Eingabefehler erkennen können.
Die Dateien können dann auch für Diagnosezwecke eingesetzt werden (Visualisierung von CAN-Daten auf dem PC mit physikalischen Einheiten).

4. TTCAN (Time Triggered CAN)

Unterschiede (Erweiterungen) zu CAN:

Die momentane Entwicklung der x-ByWire Technologie, wie sie zum Beispiel bei der entkoppelten Lenkung (Stichwort DriveByWire) realisiert werden soll, macht eine echtzeitfähige Bus-basierte Kommunikation unabdingbar. Es muss also bei diesem hochsicherheitsrelevanten Bereich der Datenkommunikation ziemlich genau vorausgesagt werden können, wann spätestens eine Nachricht am Bus durchsetzen wird (harte Echtzeitbedingungen).
Da aber eine verbindliche Aussage im Allgemeinen (außer für die höchst priore Nachricht) nicht möglich ist (etwa bei stark ausgelastetem Bus), wurde TTCAN (Time Triggered CAN) entwickelt. Dieser realisiert garantierte Übertragungs-und Reaktionszeiten über ein Zeitfenstersystem. Das bedeute, dass für bestimmte Nachrichten spezifisch Zeitintervalle zugeteilt werden. Diese Zeitintervalle werden vor der eigentlichen Inbetriebnahme des Bus-Systems festgelegt (Offline Scheduling).
Neben exklusiv reservierten Zeitrahmen gibt es auch freie Rahmen, in denen die Standard-CAN Funktionalität greift, d.h. jede beliebige Nachricht kann an den Bus angelegt werden, eine Priorisierung wird über die bekannte Bitabetrierung erreicht (Arbitrierungsfenster). Um den BUS aber auch flexibel für zukünftige Erweiterungen (neue Bus-Teilnehmer, höhere Bandbreiten-Anforderungen für vorhandene Knoten) gestalten zu können, sind auch sogenannte Free Windows (freie Zeitfenster), die gar nicht genutzt werden, in dieser CAN Erweiterung vorgesehen.

Basiszyklus eines Beispielschedulings für TTCAN:



5. Vor- und Nachteile

Bussysteme (insbesondere CAN) haben im Automobil vor allem Vorteile gegenüber der konventionellen Verkabelung. Es gibt allerdings auch einige kleinere Nachteile, die aber wesentlich schwächer wiegen.

5.1. Vorteile

- Sehr hohe Zuverlässigkeit (sehr geringe Fehlerwahrscheinlichkeit, Kurzschlusssicherheit, robuste Hardware, automatisches Abschalten von Fehlerhaften Controllern).
- Kosteneffektiv (günstige Verkabelung, Controller sind Massenware, einfache Signalspezifikation, gute Simulationsmöglichkeiten).
- Einfache Fertigung, Installation, Erweiterung (Verlegung nur eines Kabels bzw. weniger Kabel im Fahrzeug bei der Produktion, bei Sonderausstattung keine zusätzliche Verkabelung nötig).
- Verbesserte Diagnosefähigkeit (einheitliche Diagnoseschnittstelle im Fahrzeug).
- Flexibilität (z.B. Änderung von Kennfeldern, einfaches Hinzufügen bzw. Entfernen von Komponenten, welche theoretisch sofort mit allen anderen Komponenten kommunizieren).
- Mechanismen für Netzwerkmanagement (Erkennung fehlerhafter Controller und ggf. Abschaltung dieser).
- Komponenten unterschiedlicher Hersteller einsetzbar (offenes Bussystem, Spezifikationen generell verfügbar).
- Übertragung physikalischer Werte (können numerisch ausgelesen und verwertet werden, eine genauere Signalspezifikation ist nicht nötig).
- Gewichtsersparnis (in einem durchschnittlichen Fahrzeug werden etwa 80 kg eingespart, was Verbrauchs- und fahrdynamische Vorteile mit sich bringt).

5.2. Nachteile

- Feste Übertragungsrate.
- Jeder Sensor benötigt einen Controller.
- Nur eingeschränkte Echtzeitfähigkeit, sicherheitsrelevante Anwendungen erlangen nur schwer eine TÜV-Genehmigung.
- Keine Verschlüsselung vorgesehen (bedenklich, sobald Funk-verbindungen zum Übertragen von Nachrichten auf den CAN verwendet werden).

6. Praxiseinsatz

Jede einzelne Funktion kann mit konventioneller Verkabelung realisiert werden, aber die Vielzahl von Funktionen kann nur mit einem Bussystem realisiert werden.

6.1. Einsatz in Automobilen

Blinker- und Beleuchtungssteuerung:

Bisher wurden alle Leuchten einzeln verkabelt. (Scheinwerfer, Standlichter links und rechts, Blinker, Rückfahrscheinwerfer, Bremslichter…).
Verschiedene Anwendungen (z.B, amerikanische Blinkerversion mit ständig eingeschaltetem Blinker) erforderten zusätzliche Hardware und waren im Allgemeinen nicht auf die europäische Version umschaltbar.
Mit der Ansteuerung über den CAN Bus müssen nur noch Signale gesendet werden wie „Blinker links an“, „Blinker links aus“, … Eine Einheit am Blinker kann überprüfen, ob der Blinker tatsächlich funktioniert und ggf. eine Fehlermeldung senden, die dann im Armaturenbrett angezeigt wird.
Es lassen sich (per Software) viele verschiedene Funktionen implementieren (wie z.B. Warnblinker, bremskraftabhängige Bremslichter …), ohne die Hardware zu ändern (es entstehen praktisch unbegrenzte Entwicklungs-möglichkeiten und es können Fahrzeuge mit Software Updates nachgerüstet werden).
Es gibt bereits Anhänger-Verbindungen, die CAN-Leitungen besitzen. Somit kann die Anhängerbeleuchtung auch mit eingebunden werden.

Motor- und Getriebesteuerung:

Durch die hohe Geschwindigkeit des CAN-Busses können sogar die Einspritzzeitpunkte des Motors koordiniert werden. Interessant sind vor Allem Anwendungen, bei denen Automaik- und Variable Getriebe einbezogen werden. Es können zum Beispiel die Schaltpunkte des Getriebes heruntergesetzt werden (bei geringerer Drehzahl hochschalten), so lange der Motor noch kalt ist. Um die Funktion zu realisieren muss nur im Getriebe das Temperatursignal vom Motor angefordert und ausgewertet werden. Besondere Vorzüge bietet die Zusammenarbeit mit Variablen Getrieben. Der Motor kann immer auf der günstigsten Drehzahl gefahren werden, unabhängig von der Abtriebsdrehzahl. Des Weiteren können beliebige weitere Funktionen per Software realisiert werden (Bergaberkennung, Übersetzungskennlinien, …).
Es ist auch möglich, Getriebefunktionen während der Fahrt umzustellen (dynamisch, Kraftstoff sparend, Bergabfahrt, Anhängerfahrt, virtuelle Gänge).
Derartige Anwendungen sind erst durch eine flexible Kommunikation möglich, wie sie durch CAN geboten wird.

ABS, ESP/DCS:

Brems- und Stabilisationssysteme können in Motor- und Getriebefunktionen eingreifen und Daten für einzelne Räder verarbeiten. Eine große Datenmenge kann ausgewertet werden und die Systeme können wieder durch Software Updates verbessert werden.

Komfortfunktion:
Auf einem getrennten 125 KBit/s Bus werden häufig Funktionen wie Fensterheber, Spiegelverstellung, Radio- und CD-Wechsler- und Autotelefonsteuerung, PDC (Park Distance Control) und Multifunktionslenkrad umgesetzt. Die Anbringung von Schaltern ist dabei an jeder erdenklichen Stelle möglich und es können für jede Funktion beliebig viele Schalter eingesetzt werden (ein elektrisches Schiebedach könnte dann beispielsweise den Schalter in der Tür habe, falls das sinnvoller erscheint.) Man braucht sich um die Verkabelung kaum mehr Gedanken machen. Die Ergonomie kann somit auch gefördert werden.

Schlüsselerkennung:
Um bei verschiedenen Fahrern das Auto automatisch nach den individuellen Bedürfnissen einzustellen wird heutzutage in den teureren Fahrzeugen eine Schlüsselerkennung eingesetzt. Dabei erhält jeder Fahrer einen Schlüssel mit einem eigenen Code. Dieser Code kann dann über den CAN gesendet werden. Jeder Controller kann den Code lesen und wenn er die Schlüsselerkennung unterstützt, so kann er die Daten, die beim letzten Mal gespeichert wurden verwenden, um die Komponente (elektrische Sitzverstellung, Klimaanlage, Spiegel, Radio, Navigationssystem, weiter Präferenzen des aktuellen Fahrers) automatisch einzustellen. Dabei kann eine neue Komponente hinzugefügt werden, ohne dass sich für das System etwas ändert (keine neuen Nachrichten). Der neue Controller muss nur die jeweils empfangene Schlüsselnummer auswerten.
Beim Verlassen des Fahrzeugs wird eine andere Nachricht mit der Schlüsselnummer gesendet, die die Controllern dazu veranlasst, ihre Daten zu speichern.

Adaptive Cruise Control:
Unter Adaptive Cruise Control versteht man, dass sich Fahrzeuge am Vordermann orientieren können (Radarmessungen, automatisches Beschleunigen und Bremsen). Da hier viele Komponenten beteiligt sind, die die gleichen Daten auswerten müssen, bietet sich wieder ein System wie der CAN Bus an. Wieder bestehen beliebige Erweiterungsmöglichkeiten.

Diagnose:
Durch die einheitliche Schnittstelle kann theoretisch in jeder Werkstatt und mit jedem mobilen Rechner mit CAN-Karte und entsprechender Software der Fehlerspeicher ausgelesen werden und es

6.2. Weitere Einsatzgebiete

- Medizintechnik
- Landwirtschaftstechnik
- CNC Maschinen
- Robotik
- Gebäudetechnik (Beleuchtungs-/Belüftungssteuerung…)
- Aufzug- und Rolltreppensteuerung
- Textilmaschinen
- Waschstraßen
- Fern- und Nahverkehrstechnik (Weichen-/Signalschaltungen)

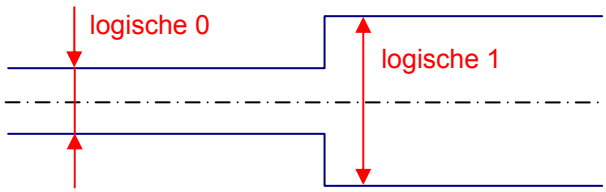
7. Prüfungsfragen und Antworten

Welche grundlegenden Vorteile haben Bussysteme gegenüber konventioneller Verkabelung in Automobilen?

- *Enorme Gewichtsersparnis (vor allem Verbrauchsvorteile)*
- *Geringer Aufwand bei der Produktion verschiedener Ausstattungsvarianten (keine zusätzliche Verkabelung).*
- *Verbesserte Diagnosefähigkeit (einheitliche Schnittstelle).*
- *Einfachere Fehlersuche (im Wesentlichen nur ein Kabel, das überprüft werden muss).*

Beschreiben Sie, was man unter einem differentiellen Signal versteht und nennen Sie den wesentlichen Vorteil gegenüber einem „normalen" Signal.

Es werden zwei Leitungen (und eine Masseleitung) für die Signalübertragung verwendet, wobei auf einer Leitung genau das invertierte Signal zur zweiten Leitung gesendet wird. Die Information steckt in der Differenz der beiden Signale. Das Beispiel zeigt die Unterscheidung einer logischen 0 von einer logischen 1 bei digitalen Signalen.



Der Vorteil liegt darin, dass eine Störung auf beide Leitungen gleich stark und in dieselbe Richtung wirkt. Die Differenz bleibt dadurch nahezu erhalten und physikalische Störungen werden somit minimiert.

Wie kann bei einem Bus mit dominanten und rezessiven Pegeln eine verlustfreie priorisierte Arbitrierung realisiert werden?

Der Buspegel wird standardmäßig auf dem rezessiven Pegel gehalten. Soll eine Nachricht gesendet werden, so wird zuerst ein dominantes Bit und dann die Priorität (bei CAN der Identifier) gesendet. Rezessive Bits verlieren gegen dominante Bits. Der Sender hört gleichzeitig den Bus ab und bricht das Senden ab, sobald er ein dominantes Bit empfängt, obwohl er ein rezessives Bit gesendet hat (ein anderer Teilnehmer, der gleichzeitig sendet, hat höhere Priorität). Der Sender mit der höchsten Priorität kommt sofort durch (Wiederholungen bei Kollisionen sind nicht nötig).

Beschreiben Sie, wie durch „Bit Monitoring" Fehler erkannt werden können.

Jeder Sender empfängt gleichzeitig (liest den Buspegel mit). Wird ein anderer Pegel empfangen, als der, der gesendet wurde, so liegt ggf. ein Fehler vor.

Was unterscheidet das TTCAN Protokoll vom Standard CAN Protokoll und inwiefern wird dadurch die Echtzeitfähigkeit von CAN beeinflusst? Erläutern Sie kurz.

Statt einer reinen Prioritätssteuerung mittels Bitarbitrierung (Standard CAN) werden Nachrichten zusätzlich feste Zeitintervalle, in denen sie exklusiv das Recht zur Bus-Belegung haben, zugeteilt. Durch diesen Mechanismus kann das System auf Echtzeitfähigkeit hin überprüft werden, sprich ein Echtzeitnachweis wird ermöglicht (bei Standard CAN im Allgemeinen nicht möglich).

8. Quellen

Internet:

- http://www.kfz-tech.de/CAN-Bus.htm, CAN-BusZ.htm
- http://www.can.bosch.com/doku/can.pdf
- http://www.me-systeme.de/canbus.html
- http://www.esd-electronics.com/german/PDF-file/CAN/Deutsch/intro-d.pdf
- http://thomas.dohmke.de/dokumente/bussysteme_folien.pdf
- http://www.rcs.ei.tum.de/courses/seminar/fieldbus/node59.html
- http://www.can-cia.de/can/ttcan/fuehrer.pdf
- http://www.cosy.sbg.ac.at/~held/teaching/wiss_arbeiten/slides_03/XVIII_CAN_WAP_ENDVERSION.pdf

Sonstige:

- Vorlesungsskript Fahrzeugtechnik, FB 03 Fahrzeugtechnik, FH München
- Vorlesungsskript Feldbussysteme, Wolfgang Kastner, TU Wien

Bilderverzeichnis:

- BLD001: http://www.kfz-tech.de/CAN-Bus.htm
- BLD002: http://www.vector-informatik.de/english/products/index.html??canboardxl.php
- BLD003: http://www.vector-japan.co.jp/products/cancardx.html
- BLD004: www.dspaceinc.com/shared/data/pdf/katalog2004/dspace_catalog2004_dci-can.pdf