Workshop

„Bussysteme im Automobil”

ECT 2002

Augsburg,

04. - 06. Juni 2002

Copyright © 2002 by Michael Randt

Haftungsausschluss

1. Intro

1.1 Konzept der Dokumentation und des Workshops

Die hier im Mai 2002 gesammelte und aufbereitete Information über Bussysteme im Automobil hat eine kurze Halbwertszeit - viele aktuelle Systeme / Entwicklungen werden in einigen Jahren nur noch historischen Wert haben, andere Konzepte werden sich als Standard etablieren.

Solch ein Workshop inklusive der erstellten Dokumentation kann somit nur eine Momentaufnahe der dynamischen Entwicklung sein. Um eine Aktualisierung, Erweiterung und Pflege der Daten entsprechend der fortschreitenden technischen Entwicklung zu ermöglichen, wurde daher ein assoziierter Webserver implementiert. Die doch recht voluminösen Datenmengen sind zudem mit einem Browser deutlich besser zu managen als in Papierform - und bedingt durch den Anpruch, möglichst alle Bussysteme abzudecken, wird für den individuellen Interessenten ja auch typisch nur eine Teilmenge der Informationen interessant sein. Weiterhin ist der Verweis auf ebenfalls online verfügbare Dokumente natürlich wertvoll bei der Vertiefung.

Diese - nichtkommerzielle - Site ist zu finden unter:

www.carbussystems.com

www.carbussystems.de

www.carbussystems.net

(Website ist trotz differierender Suffixes / tlds identisch).

Bedingt durch den begrenzten zeitlichen Rahmen können in einem solchen Workshop viele Punkte nur oberflächlich beleuchtet werden. Andererseits ist das Ziel, mehr als nur eine Klassifizierung duchzuführen. Daher sollen einige Bussysteme, die aktuelle im Einsatz sind oder dabei sind, sich zu etablieren, näher beleuchtet werden.

Der Workshop ist daher wie folgt aufgebaut:

Ungefähr eine Stunde des Workshops wird verwendet für die Grundlagen.

- Einführung
- Historie
- „Primer“: Phys. Layer, Codierung, Datensicherung, Buszugriff...
- Kriterien für die Auswahl
- Klassifizierung

Nachfolgend werden einzelne Bussysteme näher betrachtet.

- LIN

  - als Beispiel für eine aktuelle Subbus-Entwicklung)

- CAN

  - als „Universalbus“, dargestellt in seinen Varianten

- TTP, Flexray

  - zwei konkurrierende Konzepte eines streng deterministischen „Echtzeitbusses“ für die kommenden x-by-wire Applikationen

- MOST, IEEE1394

  - für Multimedia im Fahrzeug

Abgeschlossen wird der Workshop mit Thesen und einer Diskussion zur Systemarchitektur des Kommunikationssystems des Fahrzeuges der Zukunft. Hier bleibt sicherlich Raum für Spekulation.

1.2 Warum Bussysteme? (..wenn es doch ohne ging...)

Die Motivation zur Einführung von Bussystemen lag zunächst in der Einsparung von Verdrahtungsaufwand - die Verringerung der Herstellungskosten ist ja bekanntlich der universelle Antrieb im Bereich Automotive. Der amerikanische Sprachgebrauch „Multiplex Wiring“ zeigt, dass zunächst das Umschalten eines Signalpfades für unterschiedliche Zwecke der pragmatische Ansatz war. Ein Draht weniger - 50 cent gespart pro Serienfahrzeug...

Kosten sind weiterhin die Triebfeder und der universelle Maßstab für jedwede Entwicklung. Und es gilt: If it is stupid but it works - it’s not stupid. Eine bestehende, funktionierende Lösung wird also schwerlich ersetzt durch eine innovative, elegante - wenn diese teurer ist.

Was also sind die Megatrends, die den Einsatz von standardisierten Bussystemen vielleicht schneller treiben, als viele wahr haben wollen?

- Neue Ansprüche an Reduzierung des Verbrauchs und der Emission bei gleichzeitig höheren Ansprüchen an Komfort und Sicherheit zwingen zum verstärkten Einsatz elektronischer Sensoren und elektromechanischer Aktoren im Fahrzeug. Diese müssen miteinander kommunizieren.

- In diesem Kontext werden auch bestehende (z.B. mechanische, hydraulische) Systeme durch die sogenannten x-by-wire Funktionen abgelöst.

- Um den Verbrauch reduzieren zu können, muss das Gewicht des Fahrzeuges verringert werden - zumindest darf es trotz zusätzlicher Funktionalität nicht weiter ansteigen.

- Der Umbau des Fahrzeuges zum „mobile office“ bzw. zur multimedialen Infotainmenteinheit mit Koppelung an Kommunikations- und Telematiksysteme.

- Die Vielzahl der installierten dezentralen Systeme erfordert permanente Diagnosefunktionen, die - den Aufwand der Realisierung betrachtet - die eigentliche Funktionalität oft in den Hintergrund treten lassen.

- Unter produktionstechnischen Aspekten ist es erforderlich, die Fahrzeuge im laufenden Herstellungsprozess zu programmieren und mit Parametern zu versehen. Auch hier ist ein Zugriff auf die Subsysteme nötig, um nötigenfalls aktualisierte Firmware in tief eingebettete Controller „flashen“ zu können.

- Schließlich erfordert die Anpassung der Automobilindustrie an den Rhythmus der Mikroelektronik die Einführung von herstellerübergreifenden Standards - andernfalls können die nötigen Stückzahlen an Bauelementen nicht erreicht werden bzw. die langfristige Verfügbarkeit ist nicht zu gewährleisten.

Eines der Grundgesetze der Mikroelektronik ist „Moore’s law“ - die Verdoppelung der Rechenleistung alle 18 Monate. Dieses Gesetz hat schon seit 30 Jahren Bestand. Die permanente Weiterentwicklung der Mikroelektronik mit der Bereitstellung bestimmter Funktionen für immer weniger Kosten treibt die Entwicklung - die Mechanik wird nämlich - trotz Einführung von Plattformkonzepten - nur unwesentlich billiger.

2. Historie

Am Anfang war das Kabel, wenn man einmal von den Rohrverbindungen absieht, die Karbid für die Beleuchtung übertragen oder Druckflüssigkeit für die Bremsen. Mit der Elektrik im Fahrzeug, im Anfang bei 6 Volt, kam das Kabel: jede Komponente wurde mit einer separaten Leitung mit ihren versorgenden und steuernden Elementen verbunden. So entstand bald im wahrsten Sinne des Wortes ein „Kabelbaum“, die Steuerleitungen für Solid State Relais wurden immer mehr, die Leistungskabel von der Sicherung zum Verbraucher immer dicker und schwerer. Bald entstand unter Kosten- und Funktionalitätsgesichtspunkten der Wunsch, die steuernden Leitungen in einem Bussystem zusammenzufassen.

Eine geschickte Leistungsverdrahtung, sei sie stern- oder ringförmig, war bald Gang und Gebe. Auch die Einführung von 12 V war eine gewisse Optimierung. Aber spätestens mit der Einführung von Mikrocomputern in das Fahrzeug wurden auch die Steuerleitungen revolutioniert. Mit den neuen Rechnerstrukturen hielten digitale Bussysteme im Kfz Einzug.

Die Entwicklung der folgenden Jahre führte zu einer Vielzahl von Bussystemen abhängig von Anwendungsgebiet, Hersteller und Region. Im Vergleich zu einem Fahrzeug mit herkömmlicher Verdrahtung und vergleichbarer Elektronik konnten durch den Einsatz dieser Bussysteme mehr als 1000 Meter Verdrahtung eingespart werden. Das sparte Gewicht und Kosten. Dennoch entwickelte jeder Automobilhersteller und/oder Zulieferer im Geheimen seine Insellösung, die sich schließlich auch vom Wettbewerb abheben sollte.

Doch das Blatt hat sich gewendet. Durch a) einen enormen Kostendruck in der Automobilindustrie und b) durch die Notwendigkeit, Systeme von sich wettbewerbenden Zulieferern austauschen zu können, haben sich gewisse Standards herauskristallisiert. Zwar gibt es immer noch nicht ***den*** einen Universalbus für das Auto, der alle Applikationen abdeckt. Zu unterschiedlich sind die Anforderungen in Bezug auf Übertragungsgeschwindigkeit, Buslänge, Fehlertoleranz, Kosten, etc. Aber man bedenke: auch in der Industrie-, Computer- oder Unterhaltungselektronik existieren für jede Applikation spezielle Übertragungssysteme.

Je nach benötigtem Datendurchsatz und Sicherheitsanforderungen werden verschiedene Bussysteme gewählt: Im Karosseriebereich (Bodyelectronic) reichen Übertragungsraten von 10-20kbit/s aus, insbesondere dann, wenn Teilbereiche als Sub-Bus-System aufgebaut werden. Die niedrigen Übertragungsraten und das vereinfachte Busprotokoll machen die Realisierung dieser Busse technisch einfacher und damit preiswerter. Eine einfache Leitung reicht zur Datenübertragung aus. Typische Vertreter dieser Busse sind J1850, ISO 9141, K-Bus, LIN Bus, etc. Diese Sub-Bus-Systeme oder Karosseriebusse werden entweder isoliert betrieben oder über „Gateways“ mit dem wesentlich schnelleren, sicheren aber auch aufwendigeren CAN bzw. VAN Bus kombiniert und so mit den grundsätzlichen Funktionen der Motor-, Karosserie- oder Dashboard - Elektronik verbunden. Bei Übertragungsraten von 125 kb/s bis 1 Mb/s wird für CAN oder VAN Busse eine symmetrische Übertragung mittels Zweidrahtleitung genutzt. Dies bietet Vorteile durch die

Einstrahlungsfestigkeit und die geringere Störaussendung. Durch schaltungstechnische Maßnahmen kann selbst bei Bruch einer Leitung eine Diagnose erfolgen sowie eine eingeschränkte Datenübertragung aufrechterhalten werden. Allerdings sind auch die Kosten durch die aufwendigeren. Bei höheren Übertragungsgeschwindigkeiten (z. B. für Multimedia-Anwendungen, Navigations-rechnern, Digitales Radio und Fernsehen) halten zur Zeit auch Optische Medien wie D2B, MOST, etc. Einzug.

Die Zukunft hat erst begonnen, …

Literatur

/1/ Vernetzung im Fahrzeug – von der Idee zum Silizium: K-Bus, LIN, CAN, VAN, byteflight, L. Krücke, Dr. J. Gondermann, H. Pera, W. Wetzel, ELMOS Semiconductor AG, Dortmund

3. „Primer“: Phys. Layer, Kodierung, Datensicherung, Buszugriff, Grundprinzipien

Wird nachgereicht.

4. Kriterien für die Auswahl

Bussysteme im Automobil dienen der Reduzierung des Verkabelungsaufwandes innerhalb eines Fahrzeugs und stellen so einen wichtigen Faktor zur Minimierung der Herstellungskosten und auch zur Gewichtsreduzierung der Fahrzeuge dar. Für die Auswahl von Bussystemen im Automobil existieren technische Randbedingungen, die kaufmännischen Randbedingungen gegenüber stehen. Darüber hinaus existieren automobilspezifische Randbedingungen. So ist bei der Auswahl eines Bussystems für den Automobilbereich ein Kompromiss zwischen technischen Randbedingungen, kaufmännischen Randbedingungen und den automobilspezifischen Anforderungen zu finden. Neben reinen Bussystemen von einem Typ, können auch lokale Bussysteme durch übergeordnete Bussysteme vernetzt werden. Technische Auswahlkriterien für Bussysteme im Automobil sind:

- Bandbreite
- Störsicherheit
- Echtzeitfähigkeit
- Zahl der in adressierbaren Knoten

Die Art der Anwendung bestimmt die benötigte Bandbreite. Zur Übertragung von Kommandos zum Verstellen von Klappen in Klimaanlagen wird nur eine geringe Bandbreite benötigt, die Zahl der adressierbaren Motoren kann jedoch verhältnismäßig hoch sein. Die Anforderungen, die an die Störsicherheit eines Bussystemsgestellt werden hängen vom Grad der Sicherheitsrelevanz der zu steuernden Einheiten ab. An Einheiten, die ausschließlich Komfortfunktionen dienen, werden geringere Anforderungen gestellt, als an Einheiten, die einen direkten Einfluss auf den Fahrvorgang haben, wie etwa ABS oder Antischlupfregelungssysteme. Durch kryptographische Verfahren zur Abhörsicherheit von Bussystemen kommen im Automobil derzeit nicht zum Einsatz. Die wichtigsten kaufmännischen Auswahlkriterien für Bussysteme sind:

- Leitungskosten
- Komponentenkosten

In die Leitungskosten sind neben den Materialkosten für Verbindungskabel auch Handhabungskosten beim Einbau zu berücksichtigen. Bei den Komponentenkosten sind weiter Kosten für zusätzliche Hardware oder Software gegebenenfalls zu berücksichtigen.

Neben den oben aufgeführten technischen sowie kaufmännischen Auswahlkriterien für Bussysteme, die ebenso gut als Auswahlkriterien für Bussysteme bei Anwendungen in anderem Kontext dienen können, sind für den Automobilbereich weitere, charakteristische Auswahlkriterien zwingend, die dagegen in anderem Anwendungskontext von untergeordneter Bedeutung sein können. Automobilspezifische Kriterien sind:

- Elektromagnetische Verträglichkeit (EMV)
- Elektromagnetische Abstrahlung

- Versorgungsspannungstoleranz
- Topologie der Verkabelung

Anwendungsabhängige Alternativen könnten exemplarisch folgende Bussysteme sein:

- LIN vs. CAN
- TTP vs. FlexRay
- MOST vs. IEEE1934

Ist die benötigte Bandbreite gering und sind nur wenige Knoten zu adressieren, so kann LIN verwendet werden. Alternativ kann CAN anstelle von LIN verwendet werden. CAN ist bezüglich zur Verfügung stehender Bandbreite und Adressraum weitaus leistungsfähiger als LIN. Dagegen sind CAN-Komponenten im allgemeinen teurer als Komponenten für LIN. TTP und FlexRay sind Echtzeitfähige Feldbussysteme für den Automobilbereich. Da die Echtzeitfähigkeit von CAN nicht gewährleistet ist, gibt es eine Variante von CAN namens TTCAN, welche in Bezug auf Echtzeitanforderungen erweitert ist.

MOST ist ein Bussystem für Multimedia-Anwendungen im Automobil. Die kostensensible Implementierung von Feldbussystemen im Automobil wirft jedoch die Frage auf, in wieweit auf Busstandards und kostengünstige Komponenten, die in den Massenmärkten “Consumer” und ”Office Automation” verbreitet sind, zurück gegriffen werden kann. Als Alternative zu MOST ist hier der IEEE1394-Standard zu nennen, auch bekannt unter den Namen FireWire (Apple Computer, Inc.) oder i.Link (Sony Corporation).

Andererseits haben sich Bussysteme, wie der CAN – Bus, der von der Firma Bosch zunächst für den Automobilbereich entwickelt wurde, für zahlreiche Anwendungen im industriellen Bereich durchgesetzt.

Ohne den Anspruch auf Vollständigkeit erheben zu wollen, sei an dieser Stelle zunächst einmal eine Liste von verschiedenen Bussystemen gegeben, die derzeit im Automobil verwendet werden. Die Liste umfasst aber auch solche Bussysteme, die teilweise nicht mehr oder nur von einigen Automobilherstellern verwendet werden sowie solche Bussysteme, die möglicherweise zukünftig im Automobil zum Einsatz kommen werden.

General Purpose

- A-BUS
- CAN
- SAE J1567 (C^2D)
- SAE J1850 (PWM)
- SAE J1850 (VPWM)
- USB 1.1
- VAN

High Speed / Real Time

- ByteFlight

- FlexRay
- SAE J2106
- TTCAN
- TTP
- Multimedia
- D2B
- DVI
- GigaStar
- IEEE1394 (FireWire (Apple), i.Link (Sony))
- MOST
- USB 2.0

SubBus

- BSD
- Ford UBP
- GM Single-Wire-CAN
- I^2C
- K-Line / L-Line / ISO9141 / KWP2000
- LIN
- RS-232, RS485
- SAE J2058
- SPI

Wireless

- Bluetooth

5. Klassifizierung, Positionierung

Durch die steigenden Anforderungen und Anzahl der Funktionen (Sensoren und Aktoren) in der Automobilentwicklung, werden immer mehr elektronische Systeme statt mechanischer Komponenten eingesetzt. Dabei kommen häufig verschiedene Bussysteme, die sich hinsichtlich Anforderungen, Leistungsfähigkeit, Kosten und Echtzeitfähigkeit unterscheiden, zum Einsatz. Mit X-By-Wire und Telematik-Systemen wird zudem eine neue Generation an Netzwerken notwendig werden. Unterschieden werden Busse in Übertragungsgeschwindigkeit, Buslänge, Fehlertoleranz, Kosten, etc. Daher werden Busse im KFZ durch die SAE (Society of Automotive Engineering) in drei Klassen eingeteilt:

Class A:
bis 10 kBit/s
LIN, TTP/A und J-1850 gehören dazu, mechatronische Anwendungen wie smart sensors und actuators. Diese Busse haben wesentlich dazu beigetragen, den Kabelbaum im Fahrzeug zu reduzieren. Preis eines Class A Knotens zur Zeit: ca. US$ 4,-.

Class B:
10 kBit/s … 100 kBit/s
Sicherheitsrelevante Applikationen mit Fehlertoleranz, z. B. in Chassis, Powertrain und x-by-wire – Anwendungen. Damit wird die Anzahl redundanter Sensoren reduziert.
Beispiele:
TTP/B, Byteflight, TT-CAN, etc. Auch J-1850 reicht bis in Class B hinein. Preis eines Class B Knotens zur Zeit: ca. US$ 5,-.

Class C:
100 kBit/s … 1 MBit/s
Verteilte Echtzeit-Systeme, Multimedia, Echtzeit-Datenverarbeitung, Beispiele: MOST, D2B, auch Kombinationen mit drahtlosen Anwendungen wie Bluetooth. Preis eines Class C Knotens zur Zeit: ca. US$ 10,-.

Busse, die mehrere Klassen abdecken, werden mit dem entsprechenden Suffix „/x“ gekennzeichnet. Beispiel: “CAN/B” für Low Speed CAN und “CAN/C” für High Speed CAN.

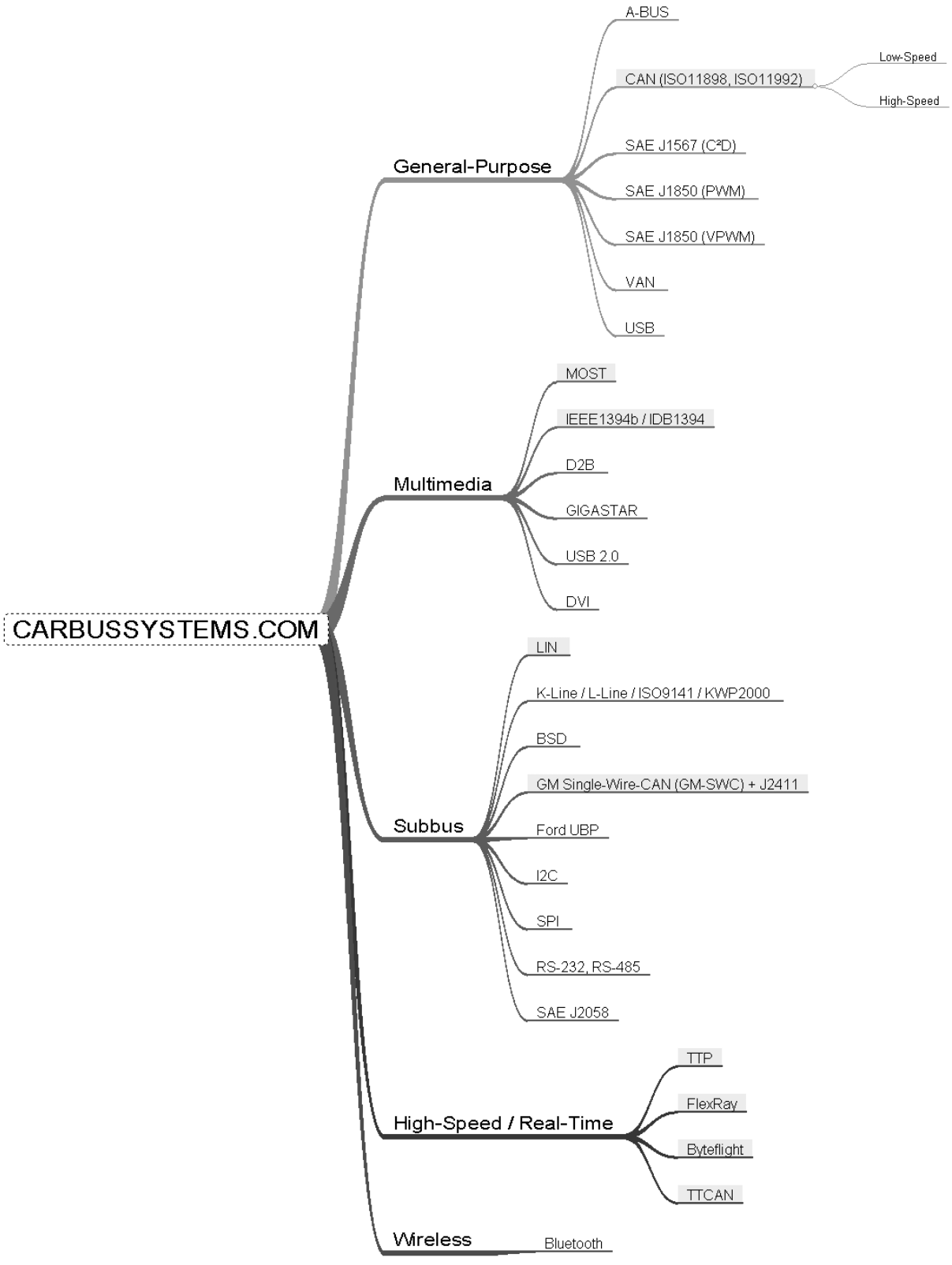


Abb. 5-1: Klassifizierung, Positionierung

6. Ausgewählte Bussysteme

6.1 LIN

6.1.1 Einleitung

Der LIN- Bus (Local Interconnect Network) ist das jüngste und universellste serielle Low-Cost-Kommunikationssystem im Fahrzeug. Er wurde ins Leben gerufen, um einen offenen Standard “unterhalb” von CAN zu generieren, dort, wo CAN zu aufwändig und zu teuer ist. Federführend bei der Spezifikation war ein Konsortium von Motorola, Audi, BMW, DaimlerChrysler, Volcano, VW und Volvo. 22 weitere Firmen sind als assoziierte Mitglieder angeschlossen (Adam Opel, Alcatel, Atmel, Elmos, Infineon, Philips, STM, Visteon, ZMD, und andere).
Hintergrund war, in der Klasse A einen ersten einheitlichen Standard zu schaffen im Bezug auf Systemkonfiguration, Signalübertragung, Software-Programmierung, etc. LIN ermöglicht eine kostengünstige Kommunikation für intelligente Sensoren und Aktuatoren, bei denen die Bandbreite und Flexibilität des CAN nicht erforderlich ist. Das Datenformat basiert auf SCI (UART), einem Single-Master/Multiple-Slave-Konzept. Physikalisch liegt dem LIN ein Single-Wire 12V-Bus zugrunde, eine stabilisierte Zeitbasis für eine Clock-Synchronisation ist nicht nötig.

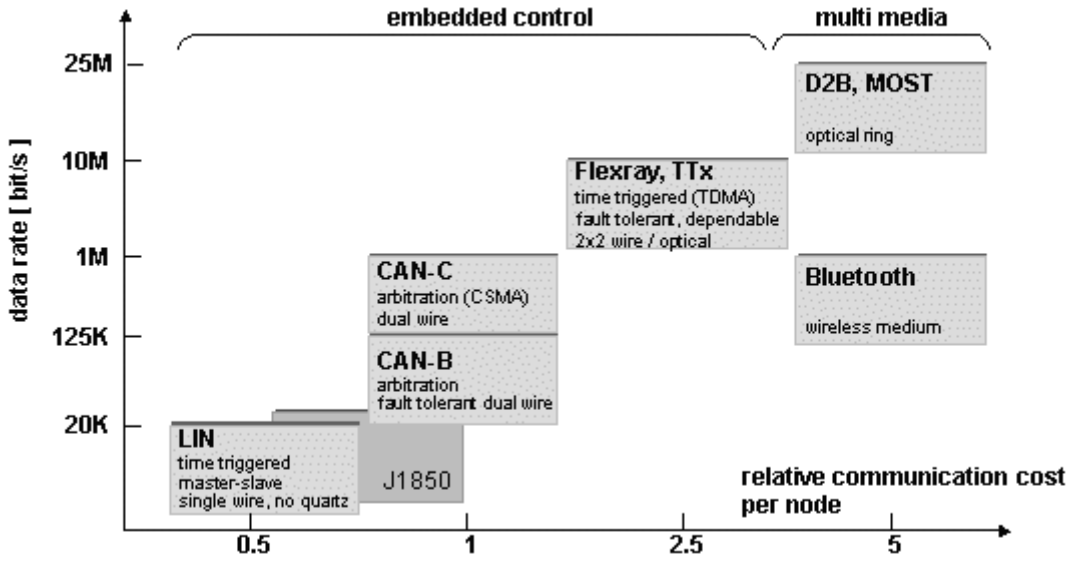


Abb. 6-1 : Einordnung des LIN in die Bussystemwelt im Kfz

LIN ist ein ganzheitliches Entwicklungskonzept. Die Spezifikation besteht nicht nur aus dem Übertragungsprotokoll, sondern umfasst gleichermaßen den Physical Layer, die Anwendungssoftware, sowie die Interfaces zu den Entwicklungs-Tools. Es beschleunigt daher die Konfiguration und Entwicklung des LIN-Netzes.
Typische Kandidaten für eine Ansteuerung über LIN sind Türmodule (Fensterheber, Schlossverriegelung, Spiegeleinstellung), Schiebedach, Steuerungen am Lenkrad

(Radio, Telefon, …), Sitzsteuerung, Heizung und Klimaregelung, Smart Scheibenwischer, Licht, Regensensoren, Starter/Generator u. v. m.

6.1.2 Das LIN Konzept

LIN ist ein Ein-Draht-Kommunikations-Protokoll, basierend auf dem genormten SCI (UART) 8-Bit Interface. UART Interfaces sind überall verfügbar, sei es als preiswerte Module auf fast allen Mikrokontrollern, sei es in Software oder Firmware, sei es als reine State-Machines in ASICs integriert. Die Arbitrierung geschieht über einen Bus-Master, so dass für für alle Slave-Knoten kein Kollisionsmanagement mehr nötig ist. Damit wird auch die maximale Übertragungszeit definiert und garantiert.

Eine Besonderheit des LIN ist der Synchronisations-Mechanismus. Er passt die Taktrate des Slave-Knotens and den Master ohne Quarz bzw. Keramik-Resonator an. Der Bus-Treiber besteht im Wesentlichen aus dem ISO 9141 Eindraht Transceiver mit wenigen Modifikationen. Mehr als 20 kbit/s werden nicht benötigt, dabei sind EMV-Aspekte (Elektromagnetische Verträglichkeit) und und Taktsynchro-nisation noch gut handhabbar.

Ein Knoten im Netzwerk muss nicht die Systemkonfiguration kennen, außer dem Master. Es können daher Knoten ohne Software- oder Hardwareänderungen der bestehenden Slaves hinzugefügt oder herausgenommen werden.

Alle diese Fakten (automatische Taktsynchronisation, die Einfachheit der UART-Kommunikation, die Ein-Draht-Verkabelung, etc.) machen den LIN zu einem kostengünstigen Medium.

6.1.3 Das Schichtenmodell

Das ISO/OSI Modell teilt Netzwerkverbindungen sieben Schichten auf. Je höher die Schicht im Modell angesiedelt ist, desto abstrakter sind ihre Funktionen.

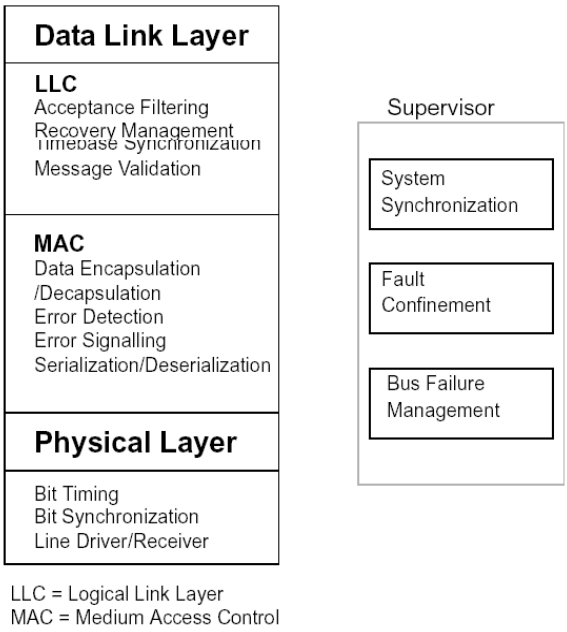


Abb. 6-2 : Einordnung des LIN in das OSI-Schichtenmodell

Der Data Link Layer entspricht der Schicht 2 (mit dem Logic Link Layer 2b und dem Medium Access Layer 2a). Darüber stehen die Vermittlungsschicht 3 und Transportschicht 4. Die obersten Schichten sind reine Anwendungsschichten. Die unterste Schicht 1 entspricht dem Physical Layer und der Leitung.

6.1.4 Der Data Link Layer

Das Netzwerk besteht aus einem Master und einen der mehreren Slave-Knoten. Jeder Slave-Knoten besteht aus einem Sende- und einem Empfangs-Teil, den sogenannten „Tasks“. Der Master-Knoten besitzt zusätzlich einen Master-Sende-Task.

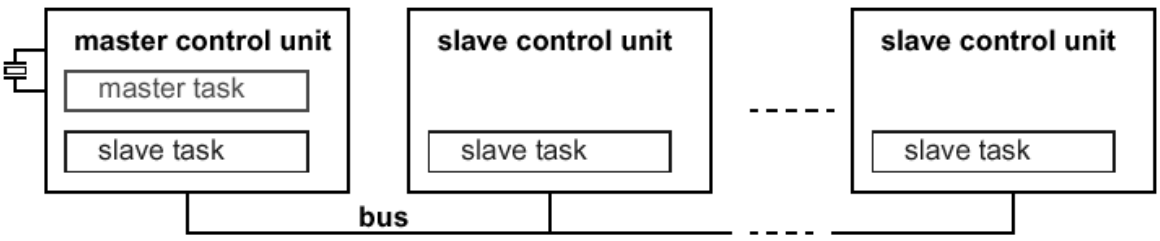


Abb. 6-3 : Master ↔ Slave Architektur

Die Übertragung im LIN-Netz wird immer vom Master mit einer Headernachricht initiiert. Dieser Message-Header besteht aus einem Synchronisations-Start (Sync Break), ein Synchronisations-Byte, sowie einem 6 Bit langen Identifier (+2 Check Bits).

Der Identifier beinhaltet Informationen des Absenders, den Empfänger, die Absicht und die Anzahl der zu übertragenen Bytes.

Der Nachrichten-Identifier aktiviert den passenden Slave Task und gibt die Anzahl der zu übertragenden Bytes an.

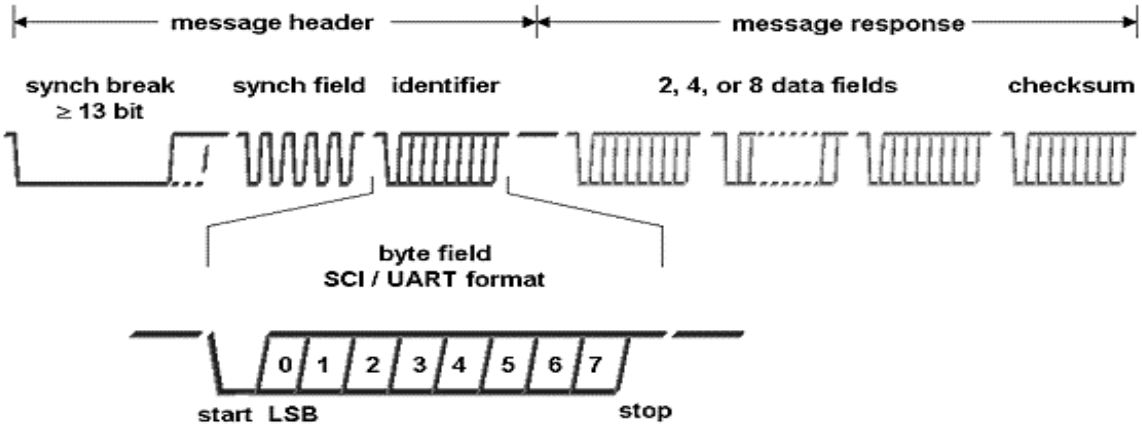


Abb. 6-4 : Codierung des Frame Headers

Der Slave seinerseits wartet auf einen „Sync Break“, anschließend wird er über das „Sync Field“ synchronisiert. Nach Auswertung der Slave Adresse und Befehl, welche sich im „Identifier Field“ befinden, wird entschieden, ob Daten zu senden oder zu empfangen, sind. Die nachfolgenden Daten bestehen aus zwei, vier oder acht Bytes plus einem Byte Checksumme. Alles zusammen umfasst den sog. Message Frame.

6.1.5 Fehlererkennung

In der Master-Nachricht schützen 2 ID Parity Check Bits das sensitive Identifier Feld:

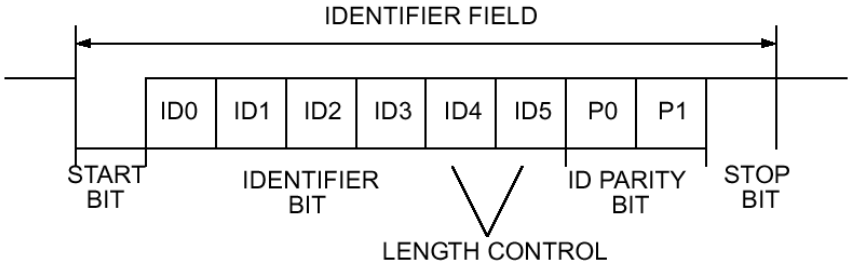


Abb. 6-5 : Codierung des Identifier-Felds

In der Slave-Antwort befindet eine 8-Bit-Wort Checksumme zur Fehlererkennung:

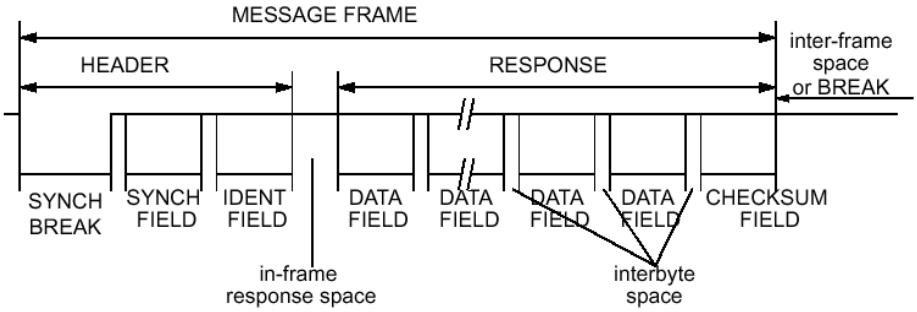


Abb. 6-6 : Codierung des gesamten Message Frames

6.1.6 Der LIN Physical Layer

Der Physical Layer ist ein Ein-Draht-Übertrager auf Basis der Kfz-Batteriespannung U_{batt}, basierend auf dem ISO9141-Standard.

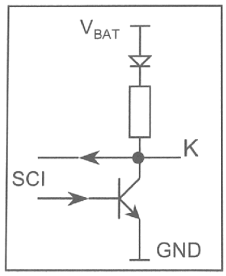


Abb. 6-7 : Prinzipschaltbild

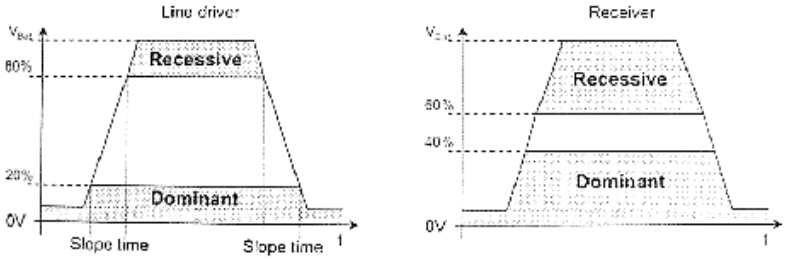


Abb. 6-8 : Sende- bzw Empfangstoleranzen

Sendeseitig wird eine Spannung von mindestens 80% U_{batt} als „high“ definiert, ein Wert unter 20% von U_{batt} gilt als „low“. Empfangseitig ist > 60% von U_{batt} = „high“, < 40% = „low“. Damit ist eine gewisse Sicherheit gegenüber Spannungsabfällen über Leitungslängen gegeben. Die Steigrate wird über den Pull-Up-Widerstand geregelt und beträgt 1-2 V / µs.

6.1.7 Software-Entwicklungsumgebung

Wir schon erwähnt, gehört zu dem Ganzheitlichen LIN-Konzept auch die entsprechende Software-Entwicklungsumgebung. Dazu gibt es im wesentlichen 4 Tools, die u. a. von dem Consortium-Mitglied Volcano Communication Technologies AB (VCT) in Göteborg entwickelt wurden.

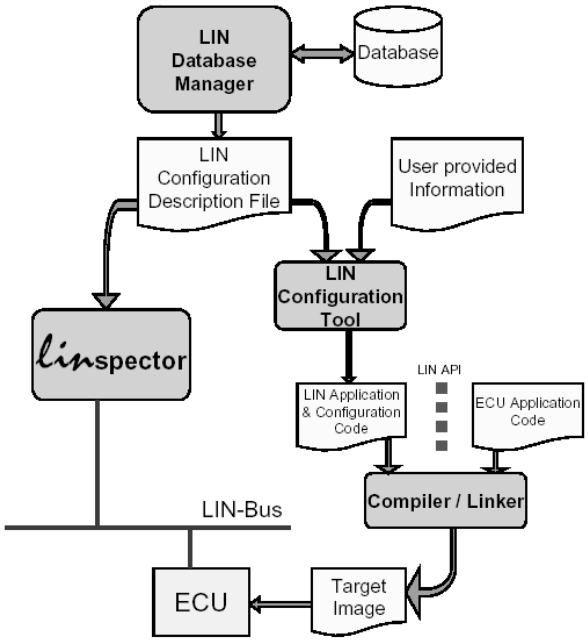


Abb. 6-9 : LIN - Entwicklungsumgebung

Mit dem LIN Database Manager (LDM) lassen sich LIN-Systeme auf einer hohen Abstraktions-Ebene logisch beschreiben und konfigurieren. Es läuft offline auf jedem PC, und zeichnet sich durch eine benutzerfreundliche Oberfläche aus.
Das LIN Application Programmers Interface (API) erlaubt es, auf einer relativ abstrakten Ebene zu entwickeln, ohne auf „Bits und Bytes“ des Datentransfers eingehen zu müssen. Zusammen mit dem LIN Configuration Tool (LCFG) und erstsprechenden Software-Tools erhält der Benutzer sowohl einerseits Flexibilität als auch andererseits die Sicherheit der Korrektheit.

Der LINspector schließlich ist ein flexibles Test- und Verifikationswerkzeug.

6.1.8 Zusammenfassung

Bussysteme unterhalb der Performance von CAN und höherwertigen Systemen stellen heute im Kraftfahrzeug die Mehrheit. Und der Markt wächst signifikant weiter! Mit LIN ist es gelungen, ein einheitliches Buskonzept bis zu einer Datenrate von ca. 20 kBit/s zu entwickeln. Überall, wo bisher konkurrierende Systeme (ISO9141, J1850, …) oder eigenentwickelte Insellösungen eingesetzt wurden, ist ein ganzheitliches System vorhanden. Entsprechende Hardware von verschiedenen Herstellern ist genauso verfügbar wie passende Software und geeignete Entwicklungs- und Test-Tools.
Beim LIN handelt es sich um ein offenes System, das einem Quasi-Standard gleichkommt. Eine formale Standardisierung ist sicher nur eine Frage der Zeit.

Literatur

LIN Specifiacation Package Revision 1.2, Nov. 2000, Issued by the LIN Consortium, Contact: Hans-Chr. V. d. Wense, Motorola, Munich

Introduction to Local Interconnect Network (LIN), SAE 2000,
Hans-Chr. V. d. Wense, Motorola, Munich

LIN-Protocol, Development Tools, and Software Interfaces for Local Interconnect Vehicles, 9th International Conference on Electronic Systems for Vehicles, Baden-Baden, Oct. 5&6, 2000, Dr.-Ing J. Will Specks, Motorola, Munich and Antal Rajnák, Volcano Communication Technologies, Gothenburg

SAE Vehicle Networks for Multiplexing and Data Communications Standards Committee, SAE J1850 Standard, “Class B Data Communications Network Interface”, Rev. May 1994

Vernetzung im Fahrzeug – von der Idee zum Silizium: K-Bus, LIN, CAN, VAN, byteflight, L. Krücke, Dr. J. Gondermann, H. Pera, W. Wetzel, ELMOS Semiconductor AG, Dortmund

6.2 CAN (Controller Area Network)

6.2.1 Wesentliche Eigenschaften

- Priorisierung von Nachrichten
- Garantierte Verzögerungszeiten
- Flexible Konfiguration von Systemen
- Multicast-Empfang durch mehrere Empfänger mit Zeitsynchronisation
- System-weite Datenkonsistenz
- Multimaster
- Fehlerdetektion und Fehlersignalisierung
- Automatische Neuübertragung, sobald der Bus wieder frei ist
- Unterscheidung temporärer Fehler und dauerhafter Fehlerbedingungen einzelner Knoten mit autonomer Abschaltung defekter Knoten
- Baudraten bis 1Mbaud – bei einer Auslastung von 50% und einem Protokolloverhead von 50% stehen in Multimastersystemen ca. 25Kbyte/s an Nutzdatenrate zur Verfügung

6.2.2 Herkunft, Anwendungen

Der CAN-Bus ist durch die Firma Bosch spezifiziert worden und wurde entwickelt für Anwendungen im Automobil. In Europa hat sich CAN auch durchgesetzt in Bereichen der Anlagenautomatisierung, z.B. der Kommunikation von Einheiten untereinander in größeren Geräten, wie z.B. Testgeräte in der Halbleiterindustrie, Laborautomatisierung, Vernetzung antriebstechnischer Einheiten.
Die CAN-Spezifikation umfasst dabei nur die physikalische Ebene, die Bittransfercodierung sowie die Übermittlung von Telegrammen und die Fehlerbehandlung. Weiterhin sind Dokumente zur angrenzenden Themen verfügbar. Strenge Auflagen zur Kompatibilität sichern einwandfreie Funktion, so sind z.B. CAN-Implementationen von Bosch in C und VHDL erhältlich.

6.2.3 Standardisierung von Funktionen: CAN-Open

Mit der sog. CAN-Open-Spezifikation steht ein Vorschlag zur Verfügung, wie Geräte aus unterschiedlichen Bereichen herstellerunabhängig mit definierten Telegrammen parametriert werden. Prinzipiell ist dadurch keine Anpassung der Systemsoftware mehr an die aktuell verwendete Hardware notwendig. Praktisch bedeutet CAN-Open jedoch einen Software-Overhead von ca. 16 Kbyte für ein typisches Gerät, sowie Anforderungen an Prozessorleistung und RAM. Da nicht alle herstellerspezifischen Features abgedeckt werden können, ist für jede Geräteklasse ein gewisser Funktionsumfang definiert, von dem meist nur ein Subset implementiert wird. Dadurch wird die Kompatibilität in vielen Fällen wieder erheblich eingeschränkt.

Applikationsebene
Objektebene Nachrichtenfilterung Nachrichten- und Statusauswertung
Transferebene Fehlerabschaltung Fehlerdetektion und –Signalisierung Validierung der Nachrichten Bestätigung Arbitration Nachrichtenkapselung in Telegramme Übertragungsrate und –Timing
Physikalische Ebene Signalpegel und Bitrepräsentation Übertragungsmedium

Abb. 6-10 : Umfang der CAN-Spezifikation

6.2.4 CAN2.0A vs. 2.0B

In der ersten CAN-Spec 2.0A wurde ein Identifier von 11 Bit vorgesehen. Da jedoch jedes Peripheriegerät eine Reihe von Identifiern nutzen kann, und der Wunsch besteht, Identifier anhand der Nachrichtentypen unabhängig von der jeweiligen Kombination von Teilnehmern zu vergeben, wurde in der Spezifikation 2.0B der Identifier auf 21 Bit erweitert. Nahezu alle auf dem Markt verfügbaren Bausteine unterstützen 2.0B, bzw. können in Netzwerken mit 2.0B Nachrichten verwendet werden, wobei sie selbst nur 2.0A Telegramme verarbeiten (2.0B passive Bausteine).

6.2.5 Telegramtypen und deren Aufbau

CAN definiert DATA-FRAMES für die Übertragung von Daten, REMOTE-FRAMES zur Anforderung von Daten mit der selben ID, ERROR-FRAMES zur Signalisierung von Übertragungsfehlern, sowie OVERLOAD-FRAMES zur Einführung von Verzögerungen.

Die Abbildung zeigt den Aufbau von DATA-FRAMES. Das Datenfeld kann eine Länge von 0-8 Bytes haben. 0-Byte Telegramme dienen dabei beispielsweise dem auslösen Vorprogrammierter Ereignisse, oder REMOTE-FRAMES haben ebenfalls die gleiche Struktur, jedoch kein Datenfeld.
Das 15-Bit CRC erlaubt die sichere Erkennung von bis zu 5 Einzel-Bit-Fehlern sowie die Erkennung von Fehlerburst bis zu 14 Bit. Eine Fehlerkorrektur ist nicht vorgesehen.
Der / die Empfänger bestätigen den korrekten Empfang einer Nachricht direkt im Acknowledge-Feld des Telegramms durch einen dominanten Pegel.

Detektiert eine Station einen Fehler in einer Übertragung, kann sie durch einen ERROR-FRAME dies allen Stationen übermitteln. Der Error-Frame besteht dabei in einer Verletzung des Bit-Stuffing-Protokolls, d.h. dass mehr als 5 dominante Bits aufeinander folgen. Diese Bedingung ist daher in jedem Fall feststellbar und sichert die Konsistenz bei Multicasts.

Standard-Datenformat (2.0A)

FS	Identifier		DLC	Data	CRC		Ack		EoF
1	11	3	4	0..8 Byte	15	1	1	1	7

Abb. 6-11 : Erweitertes Datenformat (2.0B)

FS	Identifier		Identifier		DLC	Data	CRC		Ack		EoF
1	11	2	18	3	4	0..8 Byte	15	1	1	1	7

Abb. 6-12 : Aufbau der Telegramme bei CAN2.0A und CAN2.0B

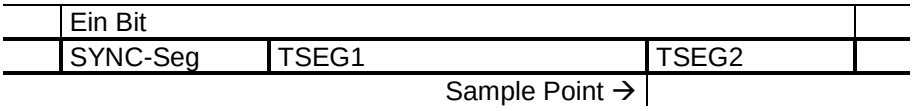


Abb. 6-13 : Aufbau der einzelnen Datenzellen

6.2.6 Arbitration mit Priorisierung

Prinzipiell kann ein Sender nur senden, wenn der Bus frei ist (CSMA/CD-Prinzip). Der CAN-Bus nutzt sowohl für die Auflösung von Kollisionen bei der Arbitration ein logisches Verordern der Informationen auf dem Bus, als auch für die Herstellung von Datenkonsistenz bei der Bestätigung von Multicast-Nachrichten. Dieses Konzept ist auch z.B. vom Bussystem I²C her bekannt. Es bedingt aber gleichzeitig, dass die Datenlaufzeit nicht höher sein darf als ein Bruchteil der Länge einer Bit-Zelle. Dadurch sind Größe des Bussystems und maximale Datenrate eng miteinander verbunden. Jeder Baustein kann beginnen zu senden, sobald der Bus frei ist. Dabei folgt nach einem Startbit, dass auch der Synchronisation der Sender untereinander dient, die Übertragung des Nachrichtenidentifiers, der u.A. den/die Zielknoten identifiziert. Sollten zwei Sender zugleich eine Übertragung starten, z.B. weil der Bus gerade frei geworden ist, so verliert derjenige den Masterzugriff, der als erstes einen „rezessiven“ (=passiven, d.h. durch die Abschlusswiderstände hergestellten) Pegel in seinem Telegramm enthält, der durch einen „dominanten“ Pegel des anderen Senders überdeckt wird. Der Sender, der die Arbitrationsphase verloren hat bricht dann seine Übertragung ab und startet erneut, sobald der Bus wieder frei ist. Auf den ersten Blick scheint die statische Kopplung der Prioritäten an die Nachrichten-IDs nachteilig zu sein. Sie kann jedoch aufgrund des großen Adressraums aufgelöst werden, indem die obersten Adressbits alleine zur Codierung der Priorität genutzt werden, während die niederwertigen Adressbits den Empfänger codieren.

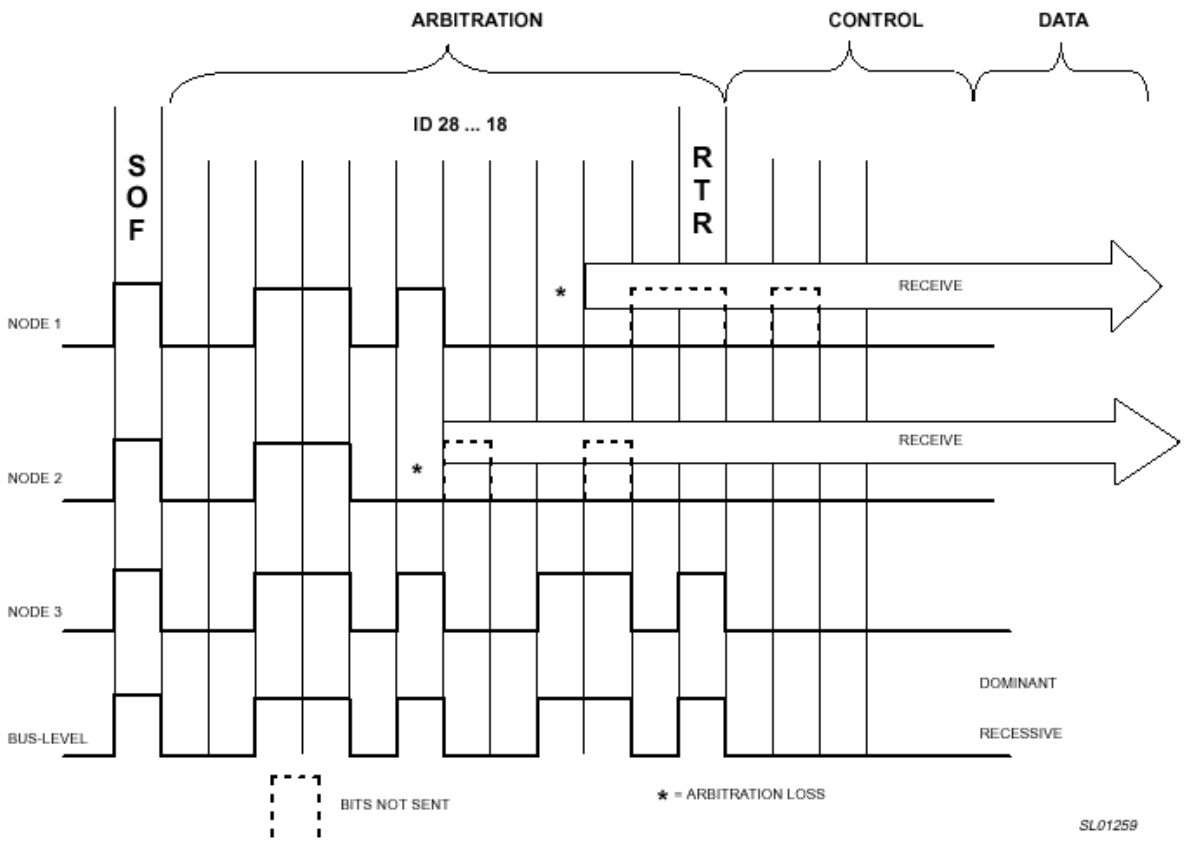


Abb. 6-14 : Arbitration durch logisches Verodern der Nachrichten-IDs auf dem Bus aus [1]

6.2.7 Codierung

CAN verwendet eine direkte Codierung der Bits (NRZ), d.h. stabiler Logikpegel während der gesamten Bitübertragung. Dabei wird zu Synchronisationszwecken nach je 5 gleichen Bits ein Pegelwechsel erzwungen, indem ein Bit der gegenüberliegenden Priorität eingeschoben wird. Spezielle Frames, wie der Error-Frame können jederzeit erkannt werden, indem sie gegen diese Konvention verstoßen.

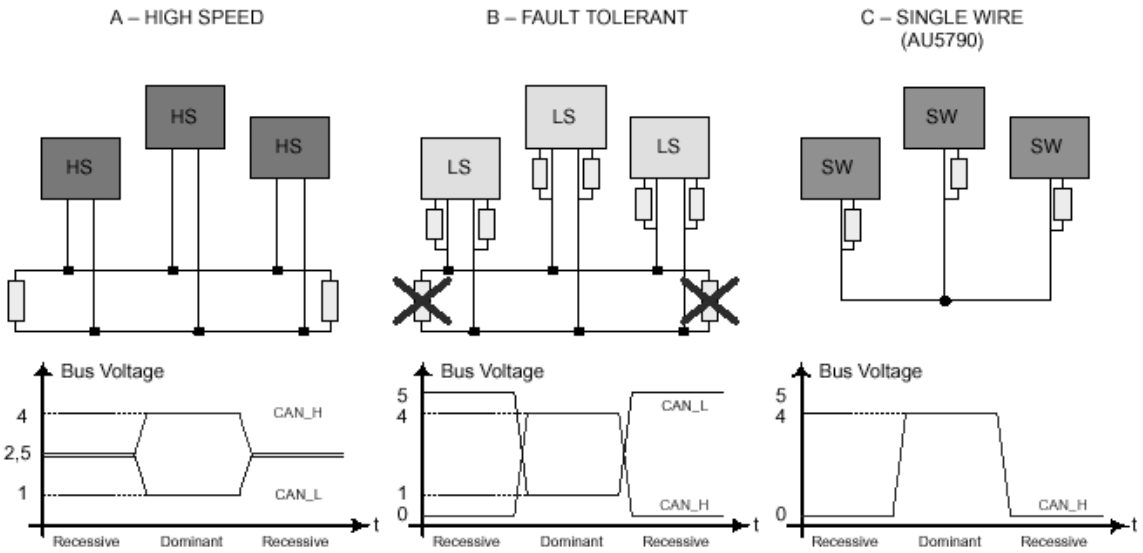


Abb. 6-15 : Physikalische Medien für A: Standard-CAN, B: Low-Speed, Fault-tolerant, C: Single-Wire [2]

6.3 Erweiterung von CAN zu TTCAN (Time Triggered CAN)

6.3.1 Hintergrund: Warum TTCAN?

Um Anforderungen an Echtzeitbedingungen gerecht zu werden, reicht das Verfahren der Nachrichtenpriorisierung in vielen Fällen nicht aus, da ein hochpriorer Sender alle anderen blockieren kann. TTCAN setzt zu diesem Zweck eine Zeitsteuerung auf das vorhandene CAN auf.
Aufbau der Zeitslots
Der Anwender kann beliebige Zeitslots definieren, die jeweils einem Master zur Verfügung stehen, bzw. in denen auch mehrere Master nach dem CAN-Arbitrationsprinzip Nachrichten übertragen können. So können Echtzeitbedingungen und maximale Datentransferrate gegeneinander abgewogen werden.

→						→		
Referenz-Nachricht	Exklusives Fenster	Exclusives Fenster	Arbitrietes Fenster	Exclusives Fenster	Freies Fenster	**Referenz-Nachricht**	Exclusives Fenster	...

Abb. 6-16 : Aufbau der Zeitslots bei TTCAN

6.3.2 Synchronisation der Uhren zwischen den Teilnehmern durch einen Master

Es gibt in jedem System je einen Zeitmaster, der regelmäßig Synchronisationsframes sendet. Um auch hier das System der Redundanz nicht zu verletzen, werden alternative Master definiert, die diese Aufgabe übernehmen, wenn der Ausfall des Zeitmasters detektiert wird. Die Zeitabstände zwischen einzelnen

Synchronisationsereignissen können unabhängig von der Anzahl der Zeitslots gewählt werden.

6.3.3 Implementation in Hardware

Ein Zeitslotverfahren kann prinzipiell in Software realisiert werden, wird jedoch in TTCAN sehr eng spezifiziert, um maximalen Durchsatz zu garantieren. Dies kann sinnvoll nur in Hardware realisiert werden, wobei ein solcher TTCAN-Controller keinen wesentlich höheren Logikaufwand erfordert als ein normaler CAN-Adapter, so dass, wenn sich dieses System durchsetzt, nichts dagegen spricht, auch in herkömmlichen Systemen einen solchen Controller einzusetzen.

6.4 VAN (Vehicle Area Network)

6.4.1 Wesentliche Unterschiede zu CAN

- In-frame response: Antworttelegramm direkt im Anfragetelegramm (wie ACK-Bit bei CAN)
- 12 Bit Identifier
- Die Arbitration ist nicht auf das Adressfeld beschränkt, sie setzt sich im Datenfeld fort
- Broadcast-Empfang ist möglich, die Bestätigung ist jedoch dabei nicht Protokollbestandteil
- Nutzt Mix aus NRZ-Codierung und Manchester-Codierung (3 Bit direkt codiert (NRZ), 1 Bit mit Pegelwechsel (Manchester)) statt Bit-Stuffing
- +/- 3% Abweichung der Taktfrequenz sind möglich durch Ausmessen des Start-of-Frame

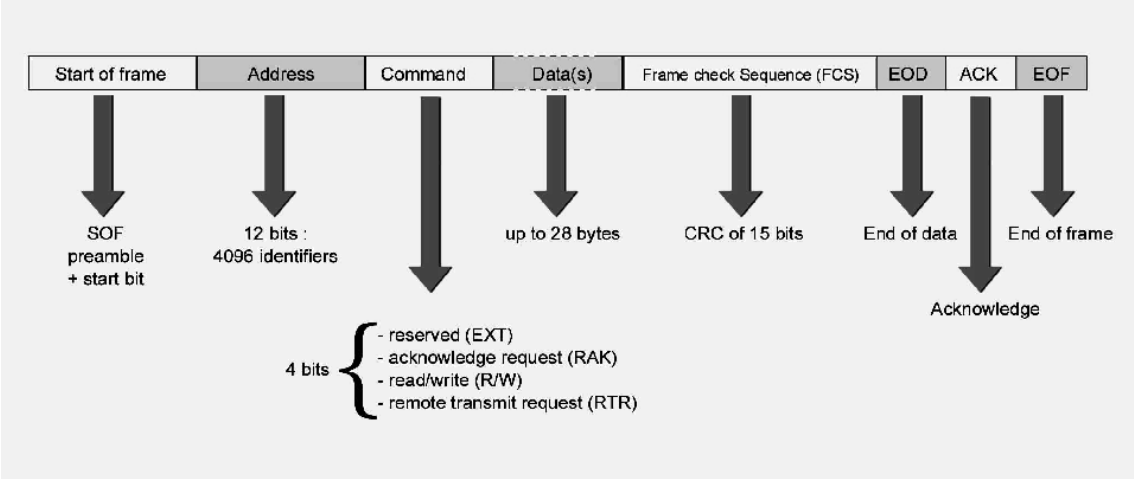


Abb. 6-17 : Aufbau des VAN-Telegramms aus [1]

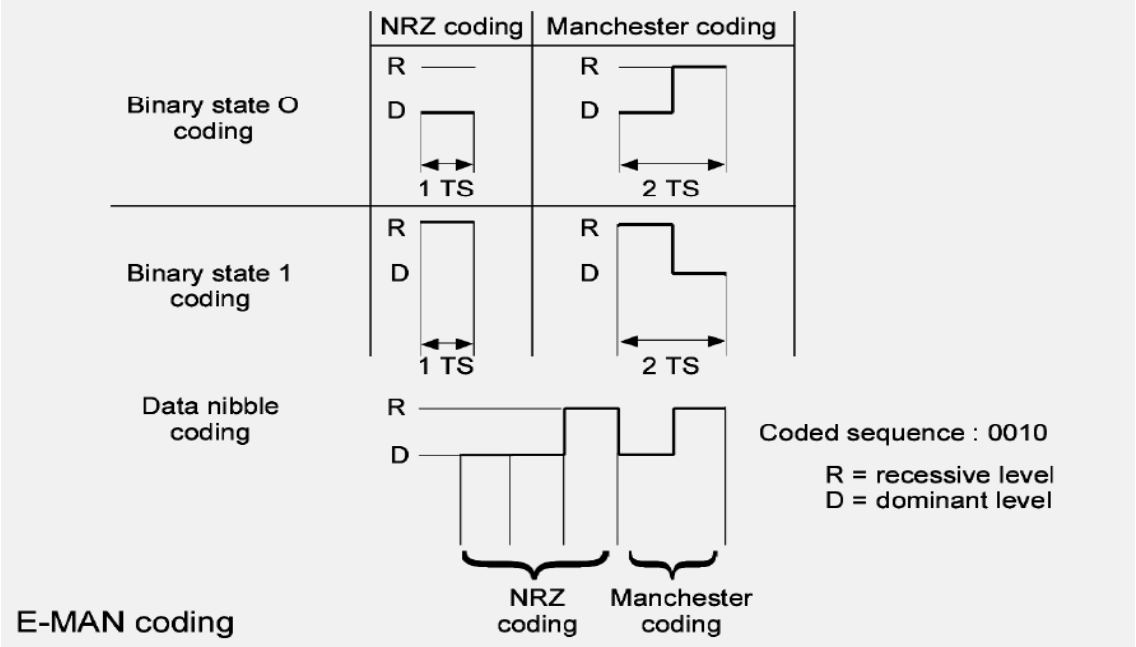


Abb. 6-18 : „Enhanced Manchester“ Codierung bei VAN aus [1]

Literatur

[1] nach VAN in Details von VAN-MUX.org

[2] Philips AN2005 zum AU5790 single wire CAN tranceiver

6.5 TTP (Time Triggered Protocol)

TTP/C ist ein zeitgesteuertes Kommunikationsprotokoll, das speziell für den Einsatz in sicherheitskritischen Anwendungen, wie z.B. im Flugzeugbereich oder für X-by-Wire-Anwendungen im Automobil geeignet ist. Mit diesem Protokoll können die höchsten Sicherheitsanforderungen erfüllt werden. Zudem wird es mit diesem Protokoll möglich, Komponenten eines Systems einzeln zu entwickeln und zu testen, und dann zusammenzufügen.

6.5.1 Grundprinzipien der Kommunikation

Das Protokoll arbeitet nach dem TDMA-Verfahren. Die Kommunikation des Hostprozessors mit dem Bus findet ausschließlich über den TTP/C-Communication-Controller (Abb. 6-19 : Überblick über einen Knoten [4]) statt. Im Gegensatz zu klassischen, ereignisgesteuerten Bussystemen wie z.B. CAN, kommunizieren beim TTP-Protokoll alle angeschlossenen Knoten ununterbrochen in vordefinierten Zeitabständen.

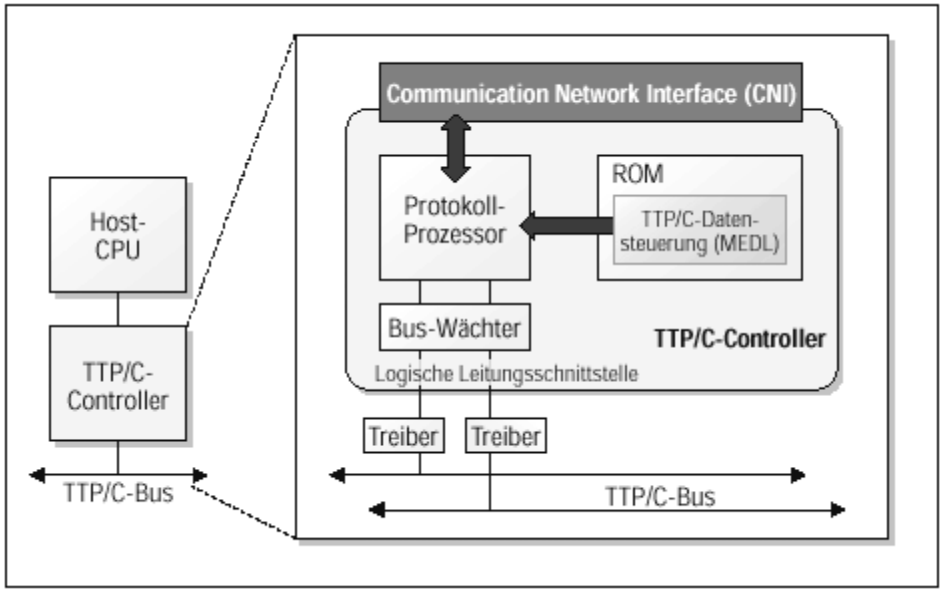


Abb. 6-19 : Überblick über einen Knoten [4]

6.5.2 MEDL

Während der Initialisierungsphase werden die notwendigen Kontrollinformationen in der lokalen *Message Descriptor List* (MEDL) des Controllers gespeichert [6]. Somit

kann der TTP/C-Controller nach der Initialisierungsphase völlig selbständig arbeiten, ohne Kontrollsignale vom Host zu erhalten.

In der MEDL sind die folgenden Daten enthalten:

- für zu sendende Nachrichten ist der Sendezeitpunkt (Sendeinstanz) und die Adresse in der CNI enthalten, wo die Daten abgeholt werden müssen.
- für zu empfangene Nachrichten ist der Empfangszeitpunkt (Empfangsinstanz) und die Adresse in der CNI enthalten, wo die Daten abgelegt werden müssen.
- zusätzliche Informationen für den Protokoll-Kontrollfluss.[2]

Alle Aktionen, die in dem System durchgeführt werden sollen, sind in der MEDL gespeichert. In der MEDL ist sogar gespeichert, wann welche Daten zu senden sind. Für unterschiedliche Modi gibt es dabei auch verschiedene MEDLs.

6.5.3 Clock-Synchronisation

Um den richtigen Sendezeitpunkt ermitteln zu können, ist eine verteilte Zeitbasis mit regelmäßiger Uhrensynchronisation notwendig. Statt eines Busmasters für die Zeitsynchronisation wird ein dezentraler Algorithmus verwendet. Der TTP/C-Controller führt diese Uhrensynchronisation autonom durch. Um die benötigte Genauigkeit der Zeitbasis zu erreichen, werden die bekannten Sendezeiten aus der MEDL verwendet. Da jeder Busteilnehmer weiß, wann er Nachrichten empfangen soll, kann er ein Empfangsfenster um diesen Zeitraum spannen. Aus der Differenz von erwartetem und tatsächlichem Empfangszeitpunkt kann die für die Generierung der Zeitbasis notwendige Korrektur errechnet werden.

6.5.4 Start der Kommunikation

Beim Hochfahren des Systems gibt es noch keine globale Zeit, da nicht alle Knoten zur selben Zeit aktiv sind. Jedem Knoten ist eine bestimmte Zeit bis zum Time-out in der MEDL zugewiesen worden. Auf dem Bus findet zunächst keine Kommunikation statt, bis bei dem ersten Knoten ein Time-out gemeldet wird. Dieser Knoten startet dann die Kommunikation durch Senden eines I-Frames. Die Startphase ist der einzige Zeitpunkt, in den Kollisionen auf dem Bus auftreten können [5].

6.5.5 Frames

Es gibt zwei verschiedene Arten von Frames: Den I-Frame, der zur Initialisierung und Resynchronisation von Controllern verwendet wird, und den N-Frame zum Senden der Daten.

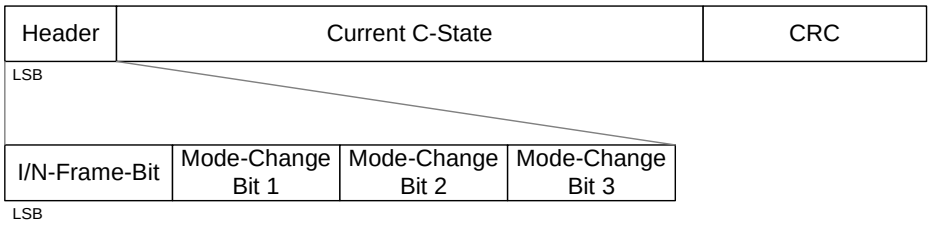


Abb. 6-20 : I-Frame nach [3]

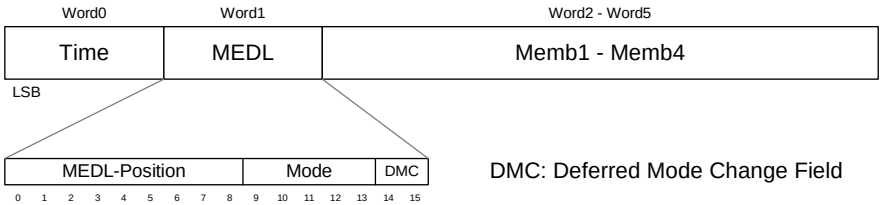


Abb. 6-21: C-State nach [3]

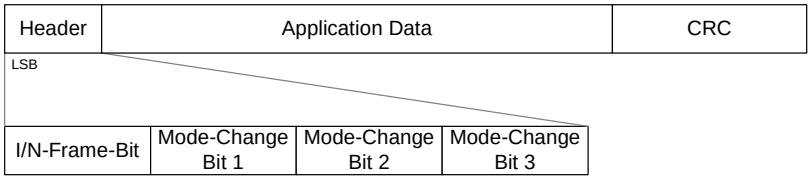


Abb. 6-22: N-Frame nach [3]

6.5.6 Media Access

Eine TDMA-Round besteht aus so vielen Zeitslots, wie SRUs im System vorhanden sind. Während des einer SRU zugewiesenen Zeitslots kann die SRU die geforderten Daten senden. Im System beinhaltet ein Cluster-Cycle alle zu sendenden Nachrichten.

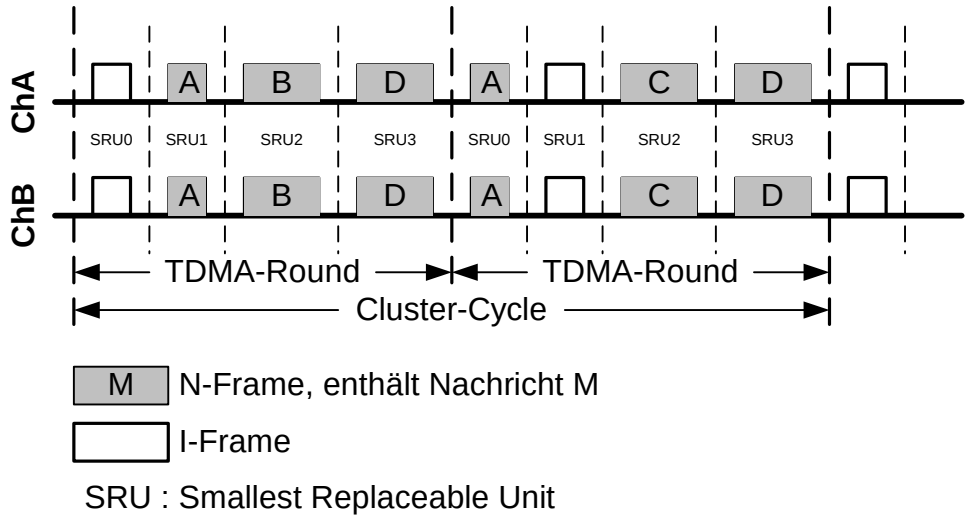


Abb. 6-23: Aufteilung eines Cluster Cycle in TDMA Rounds und SRU Slots nach [3]

6.5.7 Communication Network Interface (CNI)

Das CNI ist die Kommunikationsschnittstelle zwischen dem TTP/C-Controller und dem Hostprozessor. Es handelt sich dabei um eine reine Datenschnittstelle, über die keine Steuersignale gesendet werden: sie wirkt wie eine „temporäre Firewall“ [1]. Der Hostprozessor hat keine Möglichkeit, das zeitliche Verhalten des Kommunikationssystems zu beeinflussen.

6.5.8 Buswächter

Der Buszugriff erfolgt im TDMA-Verfahren. Somit ist sichergestellt, dass jeder TTP/C-Controller nur in dem ihm zugewiesenen Intervall Nachrichten auf den Bus senden kann. Ein im TTP/C-Controller enthaltener Bus-Wächter (Abb. 6-24 : Funktionsprinzip des Buswächters [1]) stellt diese Übertragung sicher. Somit wird auch eine Überlastung des Busses durch einen „Babbling Idiot“ vermieden.

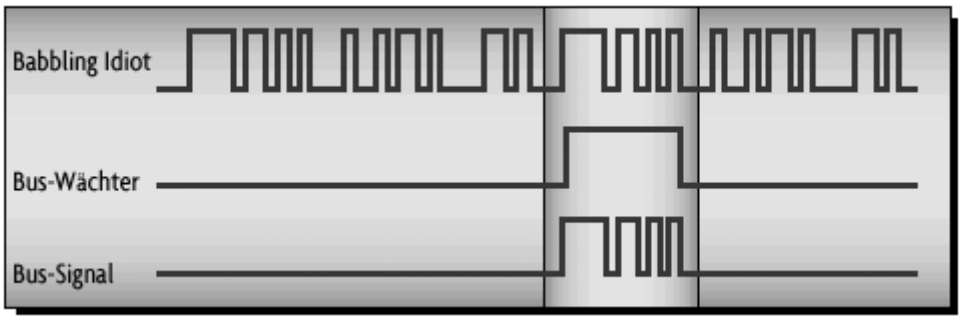


Abb. 6-24 : Funktionsprinzip des Buswächters [1]

6.5.9 Sicherheit und Fehlertoleranz

Um die Forderung nach Fehlersicherheit zu erfüllen, können zwei TTP/C-Controller, A und B parallel betrieben werden. Die beiden Controller bilden zusammen eine sogenannte fehlertolerante Einheit (FTU). Da die Nachrichtenübertragung zudem über zwei redundante Busse erfolgt, ist diese somit vierfach redundant vorhanden.

Jeder Busteilnehmer eines TTP/C-Netzwerks hat einen lokalen Membership-Vektor mit der *Membership*-Information für jeden anderen Busteilnehmer. Dadurch ist Fehlererkennung sowohl im Sender als auch im Empfänger möglich, weil jedem Knoten alle Empfangszeitpunkte bekannt sind. Kommt eine Nachricht nicht in dem angegebenen Zeitfenster an, so wird ein Fehler erkannt [3]. Alle Komponenten überprüfen kontinuierlich ihren Zeit- und Wertebereich selbst. Erkennt diese Komponente dabei eine Abweichung vom vorgegebenem Ablauf, so muss sich diese Komponente nach außen still verhalten (*Fail-Silent*) [2].

Fehlererkennung im Wertebereich wird durch die CRC-Kalkulation über den C-State, den Nachrichten-Header und die Nachricht selbst realisiert.

6.5.10 System Konfiguration

Es gibt zwei verschiedene mögliche Systemkonfigurationen:

6.5.11 Bus Konfiguration

Bei Verwendung der Bus Konfiguration hat jeder Knoten seinen eigenen Buswächter, so dass ein Schreibzugriff auf den Bus nur zu in der MEDL definierten Zeitpunkten möglich ist. Diese Konfiguration garantiert eine kollisionsfreie Kommunikation.

6.5.12 Stern Konfiguration

Bei der Stern-Konfiguration wird ein zentraler Buswächter verwendet. Jeder der zwei Übertragungskanäle hat dabei einen zentralen Buswächter. Durch die Stern-Konfiguration kann eine Aufbereitung der Bussignale durchgeführt werden. Ein weiterer Vorteil der Stern-Konfiguration ist, dass bei Zerstörung eines Knotens, z.B. durch Feuer, der Buswächter nicht betroffen ist, da dieser sich dezentral an einem anderen Ort befindet und somit den defekten Knoten abschalten kann [6].

6.5.13 Zusammensetzbarkeit (Composability)

Ein weiterer wichtiger Aspekt ist die Zusammensetzbarkeit von verschiedenen Knoten. Ist Zusammensetzbarkeit gegeben, so können die einzelnen Komponenten des Systems unabhängig voneinander entwickelt und getestet werden. Zudem können an einem Steuergerät Änderungen durchgeführt werden, ohne dass diese einen Einfluss auf die anderen Komponenten des Systems haben.

Mit TTP/C wird es ermöglicht, jeden Knoten einzeln gegen das CNI zu testen. Zusammensetzbarkeit wird garantiert und ermöglicht es so, Zeitaufwand und Kosten für Test- und Systemintegration drastisch zu senken.

6.5.14 Asynchroner Modus

TTP/C unterstützt auch das Senden von ereignisgesteuerten Nachrichten. Dafür ist eine vorher spezifizierte Anzahl von Bytes in einer Nachricht für ereignisgesteuerte Nachrichten vorzusehen. Mit diesen reservierten Bytes kann ein ereignisgesteuertes Protokoll, wie z.B. CAN, so verwendet werden, dass der Knoten, der auf den Bus senden kann, Nachrichten nach CAN-Standard übermitteln kann. Die CAN-Software kann dabei mit nur geringen Änderungen verwendet werden. Dieser Modus stört dabei das TTP/C-Protokoll in keiner Weise [6].

6.5.15 Robustheit

Die Robustheit des TTP/C-Netzwerkes wird durch eine Verkabelung entsprechend der CAN-Spezifikation , durch Manchester Codierung und durch spezielle Hardware, wie z.B. High Speed CAN-Treiber mit integrierter Fehlertoleranz [3].

6.5.16 TTP/A

TTP/A ist die kostengünstige Variante des TTP/C, erfüllt aber auch nicht die harten Echtzeitbedingen der Class C. TTP/A ist ein Master-Slave-UART-basiertes Protokoll in dem der Master- der zugleich ein Teilnehmer am TTP/C-Bus ist- den Zeittakt vorgibt [1]. Der Master spricht über Polling die einzelnen Slaves an. Die Slaves antworten nur, wenn das von Ihnen gefordert wird. Die Kommunikation zwischen den Slaves kann nur über den Master stattfinden.

6.5.17 TTP/C vs. Flexray

Der FlexRay-Bus bietet, ebenso wie TTP eine deterministische Nachrichtenübertragung, wie auch eine Uhren-Synchronisation, aber der Controller bewirkt keine Nachrichten-Konsistenz, die dafür sorgt, dass alle Knoten wissen, ob Nachrichten angekommen sind oder nicht. Dieser sogenannte „Membership-Service“ muss bei FlexRay per Middleware vom Anwender programmiert werden [7]. Da davon aber die Sicherheit des gesamten Systems abhängt, ist es gefährlich, diesen Service für jede Anwendung neu zu entwickeln.

Literatur

[1] Poledna, S.; Stöger, G.; Schlatterbeck, R.; Niedersüß, M.: Sicherheit auf vier Rädern; Elektronik 10/2001

[2] Dilger, E.; Führer, T.; Müller, B.; Poledna, S.; Thurner, T.: X-by-Wire: Design von verteilten, fehlertoleranten und sicherheitskritischen Anwendungen in modernen Kraftfahrzeugen; www.vmars.tuwien.ac.at/projects/xbywire/projects/new-vdifinal.html

[3] Specification of the TTP/C Protocol; www.tttech.com/specrequest.shtml

[4] Poledna, S.; Kroiss, G.: TTP: “Drive by Wire” in greifbarer Nähe; Elektronik 14/99

[5] Poledna, S.; Ettlmayr, W.; Novak, M.: Communication Bus for Automotive Applications

[6] Kopetz, H.: A Comparison of TTP/C and FlexRay; TU Wien Research Report 2001/10

[7] Klasche, G: TTP und FlexRay: Wann kommt die Einigung? Elektronik Automotive September 2001

6.6 FlexRay

6.6.1 Einleitung

Mit der zunehmenden Menge von Datenkommunikation zwischen den elektronischen Steuereinheit (ECUs) des Fahrzeugs, ist es wichtig eine hohe Datenrate zu erzielen. FlexRay lässt eine Datenrate von ungefähr 10Mbit/sec zu. das Design des Protokolls erlaubt jedoch, dass viel höhere Datenraten erzielt werden können.
FlexRay, als skalierbares Kommunikationssystem, erlaubt die synchrone und asynchrone Übertragung von Daten. Abhängig von den Anforderungen der jeweiligen Applikation, kann die Kommunikationsschleife synchron, asynchron oder eine Mischung von beiden sein. Die synchrone Datenübertragung ermöglicht die zeitgesteuerten Kommunikation, die den Anforderungen von sicherheitsrelevanten Systemen entspricht. Die asynchrone Übertragung basiert auf den Grundlagen des Byteflight™ Protokolls, und lässt jeden Knotenpunkt die volle Bandweite für Event gesteuerte Kommunikationen verwenden.
FlexRays synchrone, deterministische Datenübertragung hat eine garantierte minimale Meldungslatenzzeit und Meldungschwankung. FlexRay unterstützt Redundanz und eine fehlertolerante verteilte Uhrensynchronisierung für eine globale Zeitbasis. Somit werden die Zeitpläne aller Netzknoten innerhalb eines vorbestimmten engen Präzisionsfenster gehalten.

6.6.2 Geschichte

FlexRay ist ein herstellerübergreifendes Bussystem für Highspeed-Anwendungen.
Das FlexRay Protokoll ist eine Kombination von dem Byteflight™ (BMW1999) Protokoll und TTP/C. Byteflight™ wurde ursprünglich für passive Sicherheitssysteme wie Airbags entwickelt, in denen kurze Reaktionszeiten gefordert werden. Es ist jedoch nicht für aktive Kontrollsysteme geeignet da eine Fehlertoleranz nicht unterstützt wird. Für X-by-Wire Systeme ist die Fehlertoleranz eines Systems jedoch eine wichtige Vorraussetzung.
Das FlexRay Konsortium zwischen BMW und DaimlerChrysler wurde gegründet um das Byteflight™ Protokoll von BMW dahingehend weiterzuentwickeln, um es für X-by-wire Systeme tauglich zu machen. FlexRay zielt mit seiner Datenrate mit bis zu 10Mbits/s auf Applikationen wie X-by-Wire oder Powertrain ab, die ein deterministisches und fehlertolerantes Kommunikationssystem benötigen und wird frei verfügbar sein.

6.6.3 Systemstruktur

Auf der Hardwareseite lassen sich bis 64 Knoten verbinden. Jeder Knoten besitzt einen Host mit einem Kommunikations-Controller. Ein oder zwei Bustreiber und Buswächters können an diese angeschlossen werden und alle Komponenten sind über eine Spannungsversorgung (Fahrzeugbatterie) gekoppelt. Der Kommunikations-Controller ist das Herzstück von FlexRay. Er regelt den Datenfluss zwischen Host und Bus nach dem FlexRay Protokoll und erzeugt die globale Uhrzeit.

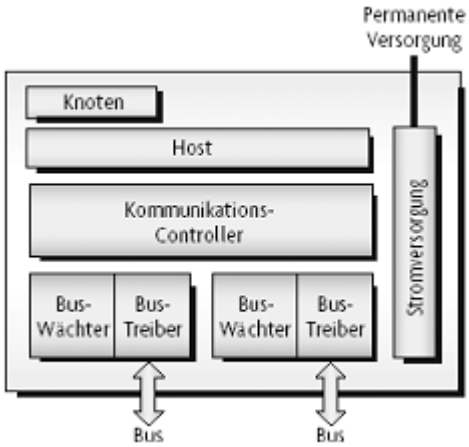


Abb. 6-25: Abbildung eines Knotens

Der Buswächter überwacht die Zeitslots und andere sicherheitsrelevante Funktionen. Er wird vom Controller gesteuert während der Bustreiber die Spannungsversorgung kontrolliert. Der Buswächter ist demnach nicht unabhängig vom Controller. Aus veröffentlichten Diagrammen ist jedoch ersichtlich, dass alle Komponenten eigene Uhrenoszillatoren besitzen, so dass der Kontroller und die Buswächter als eigenständige FCUs (Fault Containment Units) angesehene werden können. FlexRay unterstützt passiv Bus- und active Star- oder Multi-Star-Topologien, welche für sicherheitsrelevante Konfigurationen empfohlen werden. In beiden Fällen ist die Verdoppelung der Verbindungen optional. Diese aktiven Sterne enthalten jedoch keinen Buswächter.

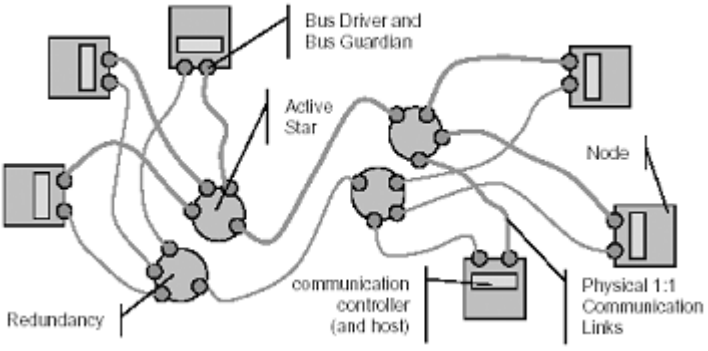


Abb. 6-26 : Netzkonfiguration mit aktiven Stern

Der Kommunikations-Controller kann entweder über zwei redundante Kanäle Daten senden und empfangen oder nur mit einem physikalischen Kanal verbunden sein (Abb. 6-20). Die Kommunikationsschnittstelle zwischen dem Kommunikations-Controller und dem Host Computer ist das Kommunikations- Network Interface (CNI). Jedes zeitgesteuerte Protokoll, wie FlexRay, benötigt dynamische Konfigurationsdaten, die vor Inbetriebnahme, in den Kommunikationscontroller geschrieben werden müssen. FlexRay minimiert diese Daten, indem viele der Parameter vorab im Protokoll festgesetzt sind. Es gibt drei Schnittstellentypen: Steuerparameter, Online-Steuerung und Online-Daten.

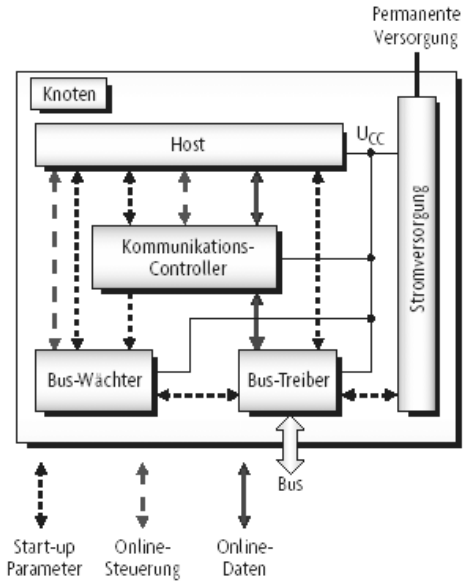


Abb. 6-27 : Schnittstellen zwischen den einzelnen Bausteinen [5]

6.6.4 Kommunikationsverfahren

Die Kommunikation läuft im Rahmen eines Kommunikationszyklus. FlexRay teilt die Zeit in zwei parallele wiederkehrende Intervalle, einen synchronen statischen Kanal für die zeitgesteuerten Nahrichten und einen asynchronen dynamischen Kanal für die Event-gesteuerten Nachrichten. Die Anteile dieser können flexibel festgelegt werden, wobei auch ein Teil leer sein kann. Somit gibt es drei mögliche Zykluskonfigurationen (statisch, gemischt (min. 2 statische Slots) und dynamisch).
Der Kommunikationszyklus beginnt mit einem SYNCSymbol. Die Zeit-Slots werden durch die ID-Nummern identifiziert, die sowohl im statischen wie auch im dynamischen Teil benutzt werden.

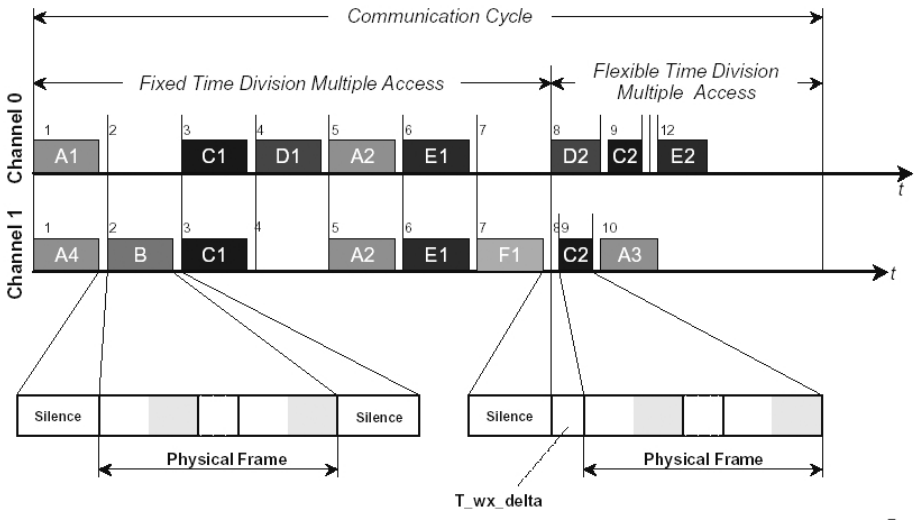


Abb. 6-28 : Kommunikationszyklus [10]

6.6.5 Statischer Teil

Die statischen Slots sind für Nachrichten hoher Priorität reserviert. Der Buszugriff erfolgt nach dem TDMA Verfahren (Time Division Multiple Access) für die Übertragung von zeitgesteuerten Nachrichten. Der Übertragungszyklus besteht aus einer frei konfigurierbaren Anzahl von Sende-Slots identischer Länge. Netzknoten die mit beiden Kanälen verbunden sind senden ihre Nachrichten synchron auf beiden Kanälen. Wird eine Nachricht nicht gesendet, verstreicht die entsprechende Zeit ungenutzt. Die Länge und der Inhalt der einzelnen Nachrichten können sich unterscheiden.
Jeder Teilnehmer erhält dieselbe Priorität, woraus sich der Vorteil ergibt, dass berechenbar und vorhersagbar ist, wann eine Nachricht gesendet oder empfangen werden kann. Dabei weiß jeder Busteilnehmer a priori die Intervalle, die er nutzen darf, wodurch eine Kollision automatisch ausgeschlossen wird.
Das Protokoll enthält eine fehlertolerante Uhrensynchronisation und schütz die Kommunikationskanäle vor „Babbling Idiots“ über einen Buswächter. Die Nachsynchronisation der Uhren ist Watchdog-überwacht. FlexRay bietet seit neuestem einen skalierbaren, erweiterten statischen Datenframe mit bis zu 254 Bytes.

6.6.6 Dynamischer Teil

Weniger wichtige „asynchrone“ Nachrichten werden im dynamische Teil gemäss der Byteflight™ Spezifikation übertragen. Seine Slots sind variabel und es kann Unterschiede in den Zeit-Slots auf den zwei Kanälen geben.
Diese Flexibilität erhöht die Übertragungsrate von FlexRay beträchtlich. Die Slot Zähler werden synchron hochgezählt und der Buszugriff folgt über das FTDMA Verfahren (Flexible Time Division Multiple Access), bei der die Einzelkanäle innerhalb einer verfügbaren Bandbreite auf einen Teilbereich dieser Bandbreite zugreifen. Bei einer Datenübertragung werden für die Dauer der Übertragung die

Slotzähler auf dem aktuellen Wert gestoppt. Kollisionsfreie Zugriffmechanismen steuern den Sendevorgang.

6.6.7 Nachrichtenformat

Das Nachrichtenformat ist im statischen wie im dynamischen Teil identisch.

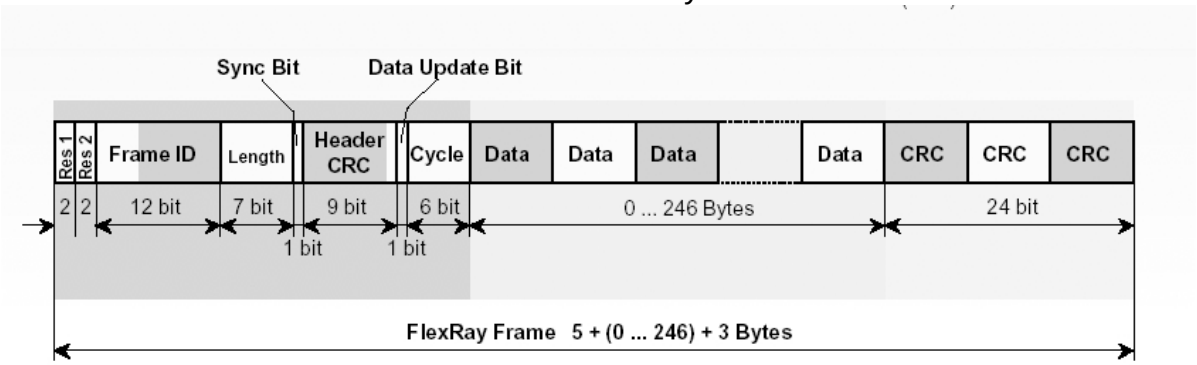


Abb. 6-29 : Frame Format mit Cycle Counter[10]

Frame ID:	Identifier, 10 Bit, Wertebereich: (110 ... 102310), definiert die Slotposition im statischen Teil und die Priorität im dynamischen Teil. Ein kleinerer Identifier bestimmt eine höhere Priorität. ID = 0 ist für das SYNC-Symbol reserviert. Ein Identifier darf in einem Netzwerk nur einmal verwendet werden. Jeder Knoten kann einen oder mehrere Identifier – sowohl im statischen als auch im dynamischen Teil – verwenden.
SYNC:	SYNC-Feld, 1 Bit. Dieses Bit zeigt an, ob die Nachricht zur Uhrensynchronisierung verwendet wird und ob das erste Datenbyte den Zykluszähler enthält. (SYNC = “1”: Botschaft mit Frame-Counter und Uhrensynchronisation, SYNC = “0”: Botschaft ohne Frame-Counter)
LENGTH:	Längen-Feld, 4 Bit, Anzahl der Datenbytes (010 ... 1210). Ein Wert größer 12 wird als LEN=12 interpretiert. Bei Nutzung des Zykluszählers (SYNC=1) wird bei einem Wert größer 11 LEN=11 gesetzt.
CYCLE:	Das CYCLE-Feld kann als Zykluszähler oder als erstes Datenbyte genutzt werden. Der Zykluszähler wird am Anfang eines jeden Kommunikationszyklus in allen Kommunikationscontrollern synchron hochgezählt.
D0 … D246:	Daten Bytes, 0 – 246 Bytes
CRC:	24 Bit Cyclic Redundancy Check.

Tabelle 6-1 : Erläuterung zum Frame Format

6.6.8 Uhrensynchronisation

Um den richtigen Sendepunkt ermitteln zu können, ist eine verteilte Zeitbasis mit regelmäßiger Uhrensynchronisation durch die Kommunikations-Controller der einzelnen Knoten notwendig. Um die Uhrensynchronisation zu garantieren, werden hierfür jedoch nur Nachrichten von Netzknoten verwendet die an beiden Kanälen angeschlossen sind. Zusätzlich werden nur ausgesuchte Nachrichten aus dem statischen Teil verwendet. Die Uhrensynchronisation ist aktiv, sobald mindestens zwei Knoten aktiv sind.

6.6.9 Sicherheit und Fehlertoleranz

In den veröffentlichten FlexRay Dokumenten ist keine Fehler-Hypothese oder Never-give-up Strategy spezifiziert. Im Falle eines physikalischen Fehlers eines Knotens ist durch die fehlende Nachrichtenkonsistenz, die dafür Sorgen würde dass alle Knoten den Systemstatus kennen, kein Vorgehensweise spezifiziert worden. Der Buswächter befindet sich schließlich in dem fehlerhaften Knoten. FlexRay benutz den Welch-Lyn Algorithmus, bei dem Zeitunterschiede wahrscheinlich in der Art und Weise von TTA detektiert werden. Da das FlexRay Netzwerk nicht über verteilte Membership-Informationen im Protokoll verfügt, erlangen fehlerhafte Knoten eine übermäßige Wichtigkeit in dem Algorithmus der Uhrensynchronisation. Es ist nicht bekannt wie das FlexRay-Protokoll sicherheitskritische Ereignisse behandelt. Eine mögliche Lösung dieses Problems, wäre die Einführung eines Membership Protokolls auf Applikationsebene, die jedoch individuell vom Anwender programmiert werden müsste und somit eine umfangreichere HostSoftware zur Folge hätte.

6.6.10 Zusammensetzbarkeit

Ein wichtiger Aspekt ist die Zusammensetzbarkeit verschiedener Knoten. Durch das Byteflight™ Protokoll ist es unmöglich die temporären Eigenschaften von Verbindungskomponenten unabhängig von dem globalen Inhalt der Applikation zu definieren. Zusammensetzbarkeit ist demnach nicht gegeben.

Literatur

[1] FlexRay Overview, Advanced Automotive Communication System
[2] Klasche, G: TTP und FlexRay: Wann kommt die Einigung? Elektronik Automotive September 2001
[3] Kopetz, H.: A Comparison of TTP/C and FlexRay; TU Wien Research Report 2001/10
[4] FlexRay-Homepage, www.flexray-group.com
[5] FlexRay für verteilte Anwendungen im Fahrzeug, Elektronik Automotive, Mai 2001

[6] Dr. Schedl, A., Lohrmann, P. :Anforderungen an ein zukünftiges Bussystem für fehlertolerante Anwendungen aus Sicht Kfz-Hersteller, VDI 17.10.2000
[7] Wengenmayr, R. : FlexRay – ein neuer Datenbus für Autos entsteht, Technische Rundschau Nr.18 2001
[8] Weiter verbessert, Design&Elektronik 03/2002
[9] Fuchs, E. : FlexRay™ Requirements Specification, Version 1.9.7, 7.Sept. 2001
[10] Bogenberger, F. : FlexRay, and where it stands today (02AE-41), SAE World Congress März 2002

6.7 MOST

6.7.1 Einleitung

MOST steht für *Media Oriented Systems Transport* und wurde als Standard von der MOST Cooperation definiert, mit dem Ziel, ein kostengünstig zu realisierendes Peer-to-Peer Netzwerk zusammen mit einem geeigneten Protokoll zur Übertragung von Multimediadaten im Automobil zu spezifizieren, wobei die Datenübertragung auch ohne PC oder andere zentrale Steuereinheiten erfolgen können soll. Multimedia umfasst im Sinne von MOST dabei nicht nur Video und Audio, sondern auch Telekommunikation, allgemeine Daten sowie Steuerdaten. Die *MOST Corporation* wurde 1998 von zwecks Standardisierung der MOST Technology von *BMW*, *Daimler-Chrysler*, *Becker Radio* und *OASIS Silicon Systems* gegründet.

MOST eignet sich sowohl für Single-Master Anwendungen, aber auch für Anwendungen im Multi-Master-Betrieb. MOST lässt bis zu 64 Netzwerkknoten zu, mit der Möglichkeit von Plug-and-Play für alle Elemente. Die Bitrate ist auf 24,8 Mbps beschränkt, entsprechend 3,1 MByte/s. Damit ist MOST zur Übertragung von komprimierten Videodatenströmen geeignet (MPEG-1, MPEG-2) sowie zur Übertragung von unkomprimierten Videodatenströmen, sofern die benötigte Bandbreite 24,8 Mbps nicht übersteigt. Die Übertragung von hochauflösenden, unkomprimierten Videodatenströmen ist also mit MOST nicht möglich. MOST unterstützt nach dem vorliegenden Standard [1,2] bis zu 15 unkomprimierte Stereo-Audio-Kanäle in CD-Qualität, bis zu 15 MPEG-1-Kanäle zur Video-Audio-Übertragung, oder einige MPEG-2-Kanäle zur Video-Audio-Übertragung. Weitere Bandbereite steht für Steuerungsaufgaben, Kommunikation sowie für asynchrone Aufgaben ständig zur Verfügung.
Bezüglich der Netzwerktopologie ist MOST flexibel. Möglich sind sowohl Ring, Doppel-Ring, Stern, Baum aber auch Kombinationen aus diesen verschiedenen Topologien. Neben Controller ICs für MOST existieren, Transceiver ICs zur einfachen Anbindung an analoge und digitale Endgeräte sowie auch zur direkten Verbindung zu integrierten AD-Wandlern und DA-Wandlern für Audioanwendungen, wie I^2S, S/PDIF, AES/EBU, I^2C, SPI.

6.7.2 Physikalische Schicht

Die physikalische Schicht [3] von MOST spezifiziert die bit-serielle, optische Übertragung von Datenströmen. Die Verbindung zwischen zwei MOST Geräten ist eine Punkt-zu-Punkt - Verbindung. Als Übertragungsmedium wird *Plastic Optical Fiber* (POF) verwendet. Eine als *Electrical Optical Converter* (EOC) bezeichnete Einheit wandelt elektrische Signale vom *MOST Network Transceiver* in optische Signale, die über eine polymere Lichtleiterverbindung zu einem *MOST Network Transceiver* übertragen werden. Dort wandelt ein als *Optical Electrical Converter* (OEC) die empfangenen optischen Signale in elektrische Signale, die dem nachgeschalteten MOST Network Transceiver zugeführt werden.

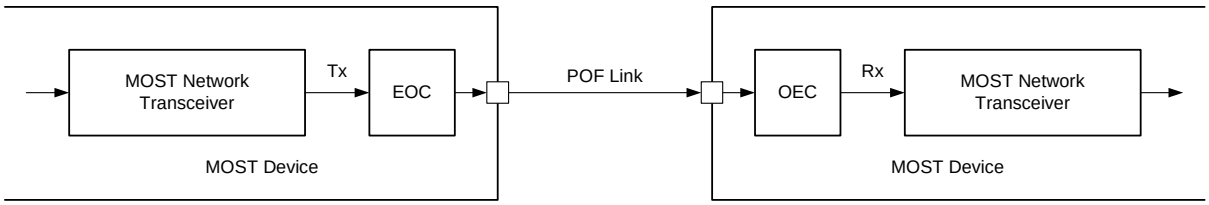


Abb. 6-30 : MOST Physical Layer

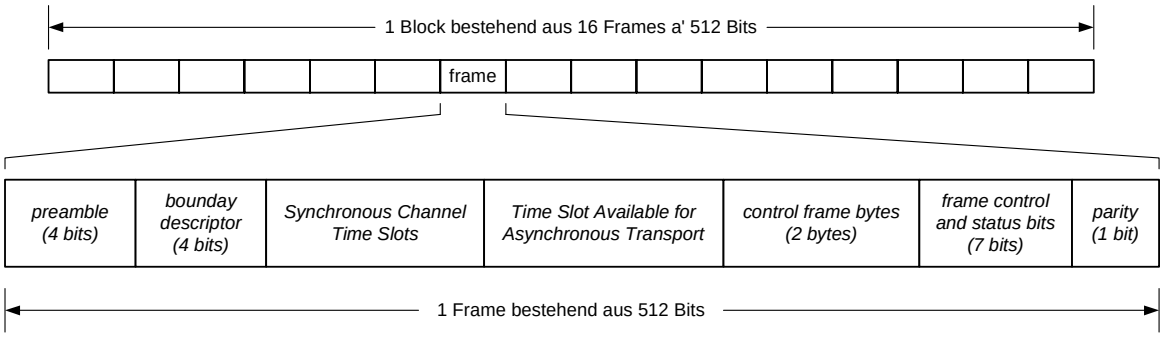


Abb. 6-31 : MOST Datenübertragung

6.7.3 Datenübertragung

Die Datenübertragung ist in Blöcken, bestehend aus 16 Frames zu je 512 Bits organisiert und trägt Charakteristika von Datenflussrichtung und benötigter Bandbreiten bei Multimedia-Anwendungen Rechnung. Dabei werden in der Regel von einer Quelle – z.B. CD-Player oder DVD-Player –Daten an eine oder mehrere Senken – z.B. (Aktiv-)Lautsprecher oder TFT-Display –mit hoher Bandbreite übertragen. Für Rückkanäle, etwa zur Bedienung – Start / Stop, Lautstärke, etc. – der Datenquellen, wird dagegen nur eine vergleichsweise geringer Bandbreite benötigt. Steuerkommandos treten aber asynchron zum Multimedia-Datenstrom auf. So enthält jeder Frame einen Zeit-Slot für die isochrone Übertragung von Multimedia-Daten sowie einen Zeit-Slot für den asynchronen Transport von Daten.

6.7.4 Schlussbetrachtung

Die optische Datenübertragung von MOST bietet in Bezug auf elektromagnetische Verträglichkeit und elektromagnetische Abstrahlung Vorteile gegenüber elektrischen Verbindungen. Kosten und Gewicht der polymeren optischen Datenkabel sprechen ebenfalls für MOST. Zwar trägt das Protokoll typischen Audi- und Video-Anwendungen Rechnung, die Bandbreite von nur 24,8 Mbit / s scheint jedoch weniger zukunftsweisend zu sein, insbesondere, da im PC Bereich zunehmend schnelle und kostengünstige Alternativen für die schnelle Datenübertragung zur

Verfügung stehen. So stellt sich die Frage, ob verfügbare Standards, wie IEEE1394 oder USB 2.0, aus dem PC Bereich – gegebenenfalls mit speziellen Anpassungen an die Anforderungen im Automobilbereich – auf lange Sicht nicht besser geeignet sind.

Literatur

[1] MOST Specification Framework Rev. 1.1., Version 1.1-07, MOST Cooperation, P.O. Box 4327, D – 76028 Karlsruhe, Germany, 1999, http://www.mostnet.de/

[2] MOST Specification, Rev. 2.1, 02/2001, Version 2.1-00, MOST Cooperation, P.O. Box 4327, D – 76028 Karlsruhe, Germany, 1999, http://www.mostnet.de/

[3] MOST Specification of Physical Layer, Rev. 1.0, 02/2001, Version 1.0-00, MOST Cooperation, P.O. Box 4327, D – 76028 Karlsruhe, Germany, 1999, http://www.mostnet.de/

6.8 IEEE1394

6.8.1 Einleitung

Erste Entwicklungen wurden bereits 1988 bei Apple gestartet. Damals war das Ziel SCSI durch einen seriellen Hochgeschwindigkeitsbus zu ersetzen. Das IEEE stellte 1995 den IEEE 1394-1995 [5] Standard vor. Dieser Standard basierte wesentlich auf den Entwicklungen von Apple. Heute wird der Standard auch unter dem Namen i.LINK (Sony) bzw. Firewire (Apple) genutzt.

6.8.2 Architektur

IEEE 1394 unterhält immer Punkt-zu-Punkt Verbindungen zwischen den Knoten (Nodes). Alle Knoten sind bei IEE1394 gleichberechtigt und können auch untereinander kommunizieren. Das Besondere an IEEE1394 ist, das Knoten auch mehrere Ports besitzen können. Dies führt dazu, dass Knoten auch als Repeater arbeiten können, d.h. ein Port empfängt ein Paket, synchronisiert es und sendet es über die anderen Ports des Knotens wieder auf den Bus.

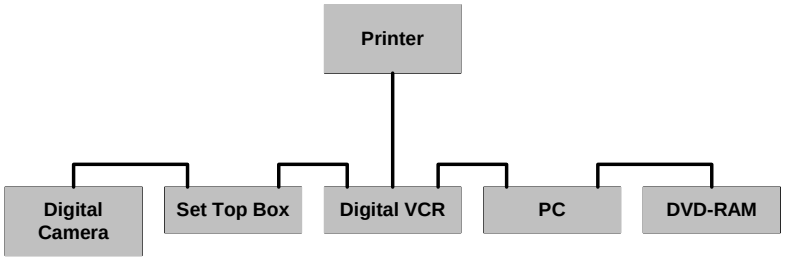


Abb. 6-32 : IEEE1394 Architektur nach [2]

Zur Adressierung stehen 64-bit Adressen zur Verfügung, aufgeteilt in 10-bit für die Netzwerk (BUS) ID, 6-bit für die Knoten ID und 48-bit für die Speicheradresse. Als Resultat ergibt sich die Adressierbarkeit von 1023 Netzwerken mit je 63 Knoten, wobei in jedem Knoten 256Terra Byte Speicher adressiert werden können.

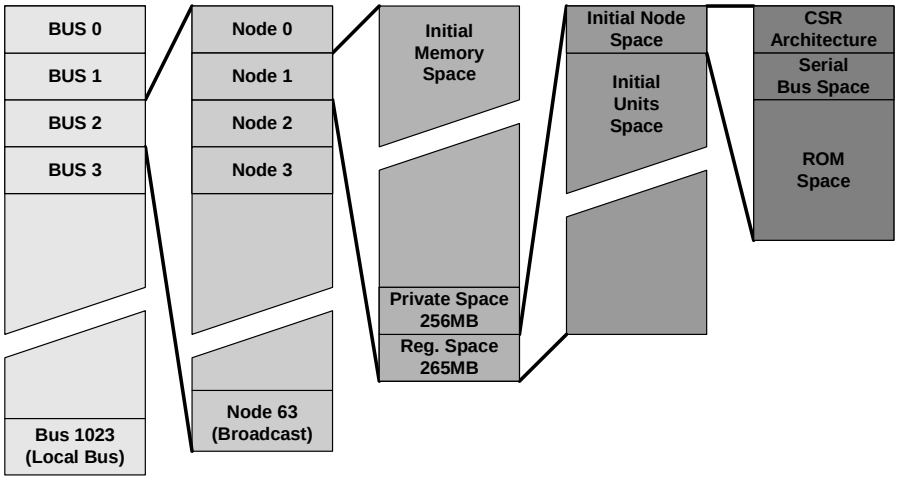


Abb. 6-33: IEEE 1394 Bus Adressraum nach [2]

6.8.3 Transfers und Transaktionen

Das 1394 Protokoll unterstützt asynchrone und isynchrone Datentransfers.
Der asynchrone Datentransfer umfasst die Übertragung von Nutzdaten in Paketen variabler Länge. Ein asynchroner Transfer zielt auf einen bestimmten Knoten auf dem Bus mit Hilfe einer eindeutigen 64-bit langen Knotenadresse. Asynchrone Transfers sind zuverlässige Verbindungen, die den ordnungsgemäßen Empfang eines Datenpaktes sicherstellen. Dazu werden CRC und Antwort-Codes eingesetzt und im Falle einer fehlerhaften Übermittlung wird die Übertragung unter Software-Kontrolle wiederholt.

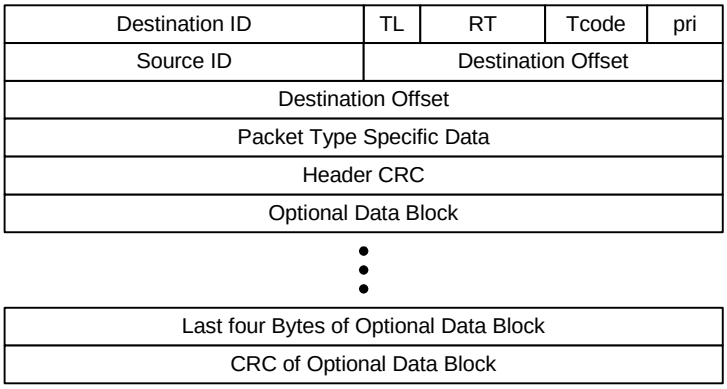


Abb. 6-34: Allgemeines Format Asynchroner Datenpakete nach [2]

Feldname	Beschreibung
Destination ID	Kombination der Busadresse und der physischen Bezeichnung des Knotens
TL	Transaction Label: Ein Bezeichner, festgelegt vom Urheber, der auch in der Antwort verwendet wird.
RT	Retry Code: Zeigt ob es sich bei diesem Paket um den ersten oder wiederholten Übertragungsversuch handelt.
Tcode	Transaction Code: Legt den Typ der Transaktion fest (Lese Anforderung, Lese Antwort, Acknowledge, usw.)
Pri	Priorität: Wir nur in Backplane Varianten benutzt.
Source ID	Absender dieses Pakets
Destination Offset	Gibt die Adresse im Adressraum des Zielknotens an.
Packet Type Specific Data	Kann genutzt werden, um die Länge der Daten anzuzeigen
Header CRC	CRC für die Headerdaten des Pakets.
Optional Data	Daten die zum Zielpunkt transferiert werden sollen.
Optional Data CRC	CRC für die Daten.

Tabelle 6-2 : Erläuterung zur allgemeinen Struktur asynchroner Datenpakete nach [2]

Bei isynchronen Transfers steht keine Fehlerkorrektur oder erneute Anforderung der Daten zur Verfügung. Außerdem wird keine eindeutige Adresse wie beim asynchronen Transfer verwendet, sondern nur eine 6-bit Kanalnummer. Das isynchrone Protokoll ist ganz besonders für zeitkritische Daten geeignet, da

mindestens 80% der Bandbreite garantiert sind. Video- oder Audiodaten zählen zum Beispiel zu den zeitkritischen Daten. Isynchrone Datenpakete enthalten u.a. Channel-Nummer, die Daten und eine CRC.

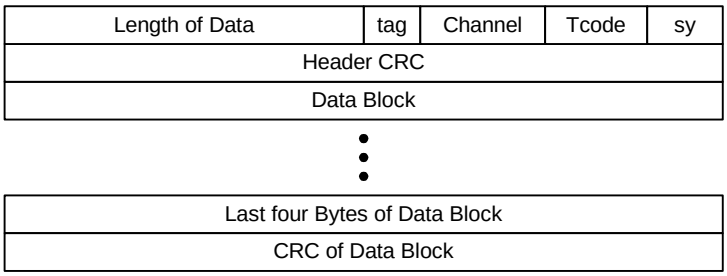


Abb. 6-35 : Allgemeines Format Isynchroner Datenpakete

Feldname	Beschreibung
Length of Data	Länge der Daten
tag	Isochrones Datenformat Tag. Der Wert 00_2 zeigt an, dass die isochronen Daten nicht formatiert sind. Alle anderen Werte sind reserviert.
Channel	Zielpunkt Adresse.
Tcode	Transaction Code: Legt den Typ der Transaktion fest. Eine isochrone Transaktion ist immer A_{16}
sy	Anwendungsspezifisches Feld
Header CRC	CRC für die Headerdaten des Pakets.
Data Block	Daten die zum Zielpunkt transferiert werden sollen.
Data CRC	CRC für die Daten.

Tabelle 6-3: Erläuterung zur allgemeinen Struktur isynchroner Datenpakete

6.8.4 IEEE Kommunikationsschichten

In der IEEE1394 Spezifikation werden vier Schichten definiert:

- Physikalische Schicht
- Sicherungsschicht
- Vermittlungsschicht (Transaction Layer)
- Transportschicht

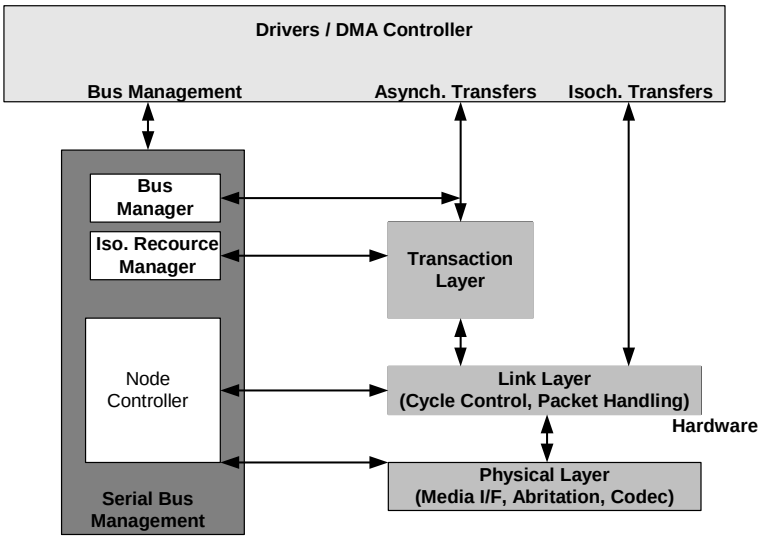


Abb. 6-36: IEEE 1394 Protokoll Layers nach [2]

6.8.5 Physikalische Schicht

Die Physikalische Schicht definiert die Eigenschaften und Kodierung der elektrischen Signale. Die Daten werden gemäß NRZ mit einem Strobe Signal und Daten RxD/ TxD übertragen. Das Verwenden eines Strobe Signals hat gegenüber der klassischen Variante mit Daten/Clock Vorteile in der Taktzurückgewinnung und dem Jitterverhalten.

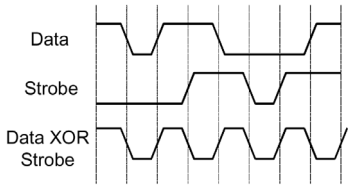


Abb. 6-37: Data Strobe Encoding nach [2]

Neben der elektrischen Eigenschaften werden außerdem auch die mechanischen Abmessungen der Steckverbinder und Kabel definiert. Zur Übertragung der Daten wird eine abgeschirmte vier oder sechs-polige Leitung benutzt. Zwei differentielle Paare dienen zur Übertragung der Daten, ein weiteres optionales Paar dient zur Energieversorgung der Knoten.

6.8.6 Identifikation der Baumstruktur

Nach jedem Reset verliert jeder Node das Wissen über die Bus Topologie. Während des Identifikationsprozesses wird die Bustopologie wieder neu definiert. Auch das entfernen oder anschließen von Geräten löst einen Reset aus. Im folgenden Beispiel wird anhand der folgenden Struktur der Prozess der Identifikation aufgezeigt.

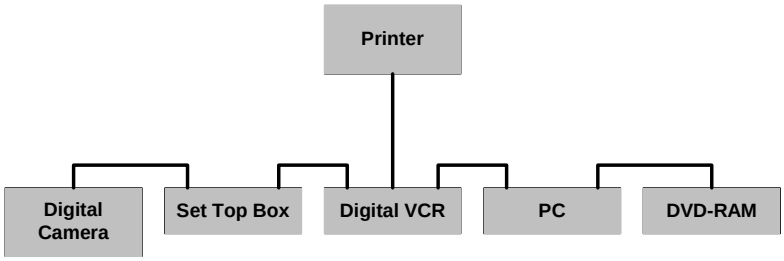


Abb. 6-38: IEEE1394 Architektur nach [2]

Nachdem Reset signalisieren alle einfachen Knoten (Knoten an denen nur ein Gerät angeschlossen ist) über ihre Daten und Strobe Leitungen ein Parent_Notify Signal. In unserem Beispiel signalisiert die Digitalkamera der Set-Top Box, der Drucker dem digitalen VCR und das DVD-RAM dem PC das *Parent_Notify* Signal.
Die verzeigten Knoten (mehrere angeschlossene Geräte) empfangen das *Parent_Notify* der einfachen Knoten und markieren diesen Port als Child und geben ein *Child_Notify* Signal an die einfachen Knoten aus. Verzweigte Knoten werden als Parent bezeichnet. Mit dem *Child_Notify* Signal nehmen die einfachen Knoten das *Parent_Notify* Signal wieder zurück, gleichzeitig dient dies auch zur Betätigung an den verzeigten Knoten. Anschließend werden die Knoten willkürlich durchnummeriert.

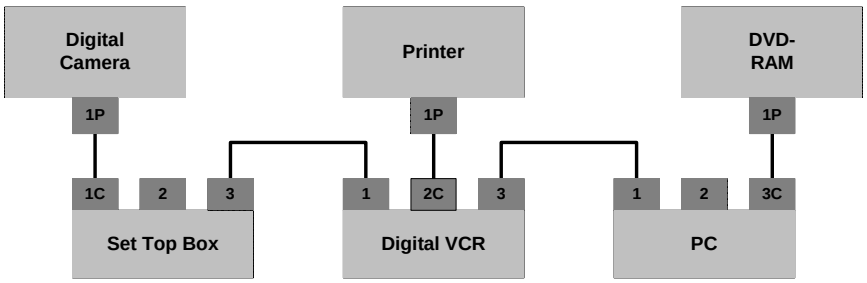


Abb. 6-39: Bus nach Knotenidentifikation nach [2]

Zum jetzigen Zeitpunkt sind die einfachen Knoten identifiziert. Aber der digitale VCR hat von zwei Ports noch kein Parent_Notify Signal erhalten, die Set-Top Box und PC haben von einem Port noch keine Antwort bekommen. In diesem Fall senden die Set-Top Box und der PC ein *Parent_Notify* Signal am den Port der noch nicht beantwortet wurde. Der digitale VCR bestätigt den Empfang der *Parent_Notify* Zustände mit dem *Child_Notify* Signal. Der VCR hat jetzt alle Ports als Children markiert und wird dadurch zum Root Node.

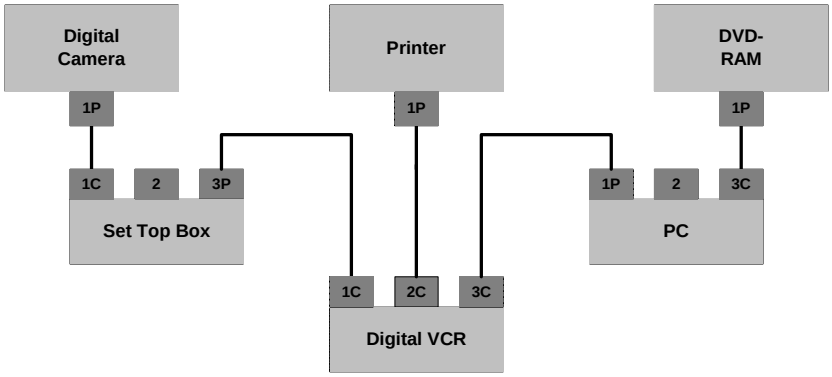


Abb. 6-40: Bus nach Knotenidentifikation 2 nach [2]

Natürlich ist es möglich das zwei Knoten zu Root-Knoten werden. Für diesem Fall wird sich für einen Root-Knoten entschieden.

6.8.7 Selbstidentifikation

Während der Selbstidentifikation werden allen Knoten physikalische Adressen zugewiesen, außerdem wird mit Nachbarknoten Informationen über Geschwindigkeit ausgetauscht.
Der Root-Knoten sendet ein *Arbitration Grant Signal* an den Port mit der kleinsten Nummer. Bei unserem Beispiel sendet der digitale VCR an die Set-Top Box. Da die Set-Top Box ein verzweigter Knoten ist leitet sie das *Arbitration Grant Signal* an einen ihrer Ports mit der kleinsten Nummer weiter. Hier an die Digitalkamera. Da die Digitalkamera ein einfacher Knoten ist, nimmt sie die physikalische Adresse 0 an and sendet eine *Self-ID* per Broadcast zurück. Des weiteren wird ein *SELF ID Done* Signal nur an die nächst höhere Hierarchie (hier: Set-Top Box) geschickt.
Nach Empfang dieser *Self-ID* inkrementieren alle Knoten ihren ID Counter. Der Root Knoten sendet ein weiteres Arbitration Grant Signal an den Port mit der kleinsten Nummer. In diesem Fall wird die Set-Top Box angesprochen, welche die Adresse 1 annimmt und anschließend *Self-ID* und *SELF ID Done* sendet.
Die Vorgang wird solange wiederholt bis an allen Ports des Root Knotens ein *SELF ID Done* Zustand detektiert wurde. Als nächsten Schritt gibt sich der Root Knoten die nächst höhere ID.
Für unser Beispiel wurden folgende IDs vergeben: Digitalkamera = 0, Set-Top Box = 1, Drucker = 2, DVD-RAM = 3, PC = 4 and VCR (Root Knoten) = 5.

6.8.8 Busarbitration

Das IEEE 1394 Protokoll wird in Zeitscheiben zu jeweils 125µs eingeteilt. Zu Beginn einer neuen Zeitscheibe sendet der Root Knoten ein Cycle Start Paket, mit dem sich alle anderen Knoten Synchronisieren.
In dieser Zeitscheibe gibt es jeweils für isynchrone und asynchrone Transaktionen Zeitspannen die Kennzeichnen, dass der Bus frei ist. Die eigentliche Arbitrationsregel ist trivial: Sind zwei Knoten gleichzeitig sendebereit, so beginnt der, der die höhere ID hat. Bei asynchronen Transfers kann es passieren das immer der Knoten mit der höchsten ID die Arbitration gewinnt, um dies zu verhindern wird zur Arbitration ein „Fairness Intervall“ verwendet. Innerhalb dessen, wird jedem Knoten, der eine asynchrone Transaktion durchführen will, genau einmal die Erlaubnis erteilt, ein Datenpaket zu senden.

Um eine Bandbreite von mindestens 80% für isynchrone Datentransfers zu garantieren, werden in einer Zeitscheibe mindestens 80% (100 µs) für den isynchronen Datentransfer zur Verfügung gestellt.

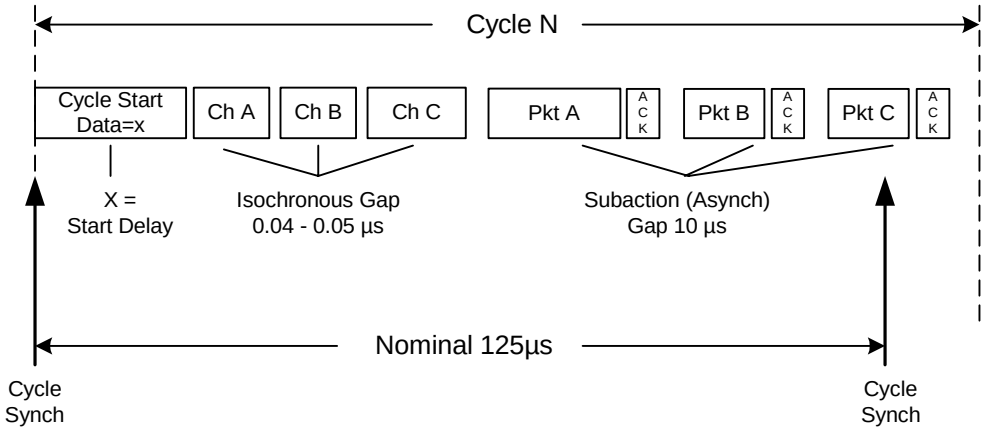


Abb. 6-41: Typischer IEEE1394 Zyklus

6.8.9 Transaction Layer

Der Transaction Layer wird für asynchrone Transfers genutzt. Es gibt fünf verschiedene Typen für asynchrone Transfers:

- Vier Byte Lesen
- Vier Byte schreiben
- Lesen von Bytes variabler Länge
- Schreiben von Bytes variabler Länge
- Swap und Compare Operationen

Asynchrone Datenpakete besitzen einen festen Headerteil und einen variablen Datenteil:

6.8.10 Weiterentwicklungen IEEE 1394

Die Automobil- sowie Systemhersteller wünschen sich einen offenen Standard, bei dem sich die unterschiedlichen Applikationen unabhängig vom Übertragungsmedium vernetzen lassen.
Im Jahre 1995 wurde von der Trade Association der Standard IEEE 1394 verabschiedet und im Jahre 2000 in einigen Punkten verbessert (IEEE1394.a). Mittlerweile liegt der Entwurf für eine erneute Erweiterung auf IEEE1394.b [6] bzw. IDB 1394 [7] vor. Dieser soll der fortschreitenden technischen Entwicklung Rechnung tragen.
Die ursprüngliche Begrenzung der Übertragungslänge wurde von etwa 4,5m bis 10m mit der Einführung einer neuen Signalcodierung (8B/10B) auf 100m ausgeweitet.
Als Übertragungsmedium stehen jetzt neben Kupfer auch Plastic Optical Fiber (POF) und Glasfaser (GOF) zur Verfügung.
Die bestehenden Geschwindigkeitsklasse von bis zu 400MBit/s wurde auf zurzeit auf 800Mbit/s erweitert. Darüber hinaus sind 1.6 bzw. 3.2 Gbit/s vorgesehen und bereits im Standard implementiert. Bisher ist im Bereich der Unterhaltungselektronik noch kein Gerät absehbar, welches die gesamte Bandbreite einer 3.2 Gbit Verbindung ausnutzen könnte. Die Entwicklungsziele waren eher an einer Zukunftssicheren Lösung orientiert, wie z.B. ein Backbone für Heimnetzwerke.

Eine weitere zukünftige Funktion ist die Segmentierung des Busses über Brücken. Mit Brücken wird es möglich sein Bussegmente unabhängig voneinander zu nutzen. Zur Zeit belegt ein Datentransfer noch den gesamten Bus.

Literatur

[1] Fachzeitschrift Elektronik: Mehr Bandbreite, größere Entfernungen; Joachim Kroll

[2] Questra Consulting: Fundamentals of Firewire; John Canosa

[3] keft Network: http://www.kefk.net/PDA/Technologie/Schnittstellen/Kabelgebunden/IEEE1394/index.html

[4] Fachzeitschrift Design&Elektronik: 1394 gibt Gas; Georg Haubner

[5] IEEE 1394-1995 Standard: http://www.ieee.org

[6] IEEE 1394.b Standard (Draft version): http://www.zayante.com/p1394b

[7] IDT 1394 Automotive Network: http://www.ami-c.org/

7. Weitere Bussysteme

7.1 SAE J1567 (C^2d) Chrysler Collision Detection

Das Ziel bei C^2D ist es, ein einfaches und zuverlässiges Protokoll für den Einsatz in verteilten Systemen im Automobil zu entwickeln. Das Protokoll ist dabei ausgelegt auf minimalen Software-Overhead.

7.1.1 Physical Layer

Die Datenübertragung erfolgt bei C^2D differenziell über ein konventionelles 120 Ohm twisted-pair-Kabel.

7.1.2 Nachrichtenformat

Die Daten werden im 10-Bit-NRZ-Format gesendet und haben die folgende Form:

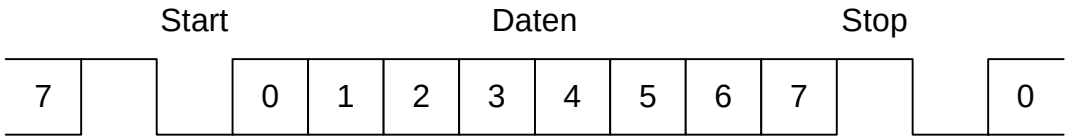


Abb. 7-1 : 10-Bit-NRZ Datagramm

Durch das verwendete Nachrichtenformat ist es möglich, verschiedene Protokolle auf höherem Level durchzuführen. Nach jedem Datenbyte können Idle-States eingefügt werden, z.B. Interbyte Separations (IBS), um so die Verwendung von Firmware und direkten Verbindungen zum asynchronen seriellen I/O-port des Microcontrollers zu unterstützen. Das Nachrichtenformat ist:

Abb. 7-2 : Nachrichtenformat

SOM: definiert den Start der Nachricht
id 1 : Schema für die 8-Bit Firmware Adressierung
id 2 : optionaler Identifier
data : alle möglichen Formen von Daten, wie z.B. Anwendungsdaten, CRC, Checksumme, Anzahl der Bytes in einer Nachricht, Bestätigung, usw.
EOM: definiert das Ende der Nachricht

Der Nachrichtenoverhead beträgt 34 Bits bei einer Datenlänge von bis zu 6 Byte. Overhead = 4 Startbits, 4 Stopbits, 8 Bits für die ID, 8 Bits für die Checksumme und 10 Bits für das Ende der Nachricht.

7.1.3 Funktionsweise

Jede einfache serielle SCI-Schnittstelle eines Microcontrollers kann verwendet werden. Über diese wird eine Kommunikation zum Businterface hergestellt. Der Netzwerkzugriff erfolgt über bit-by-bit Arbitration.

Das Businterface besteht aus folgenden Komponenten:

- differenzieller Transceiver
- Kollisions-Detektor
- Arbitration-Einheit
- digitales Filter für EMV
- Bus-Idle-Detektor
- Timing Synchronisation

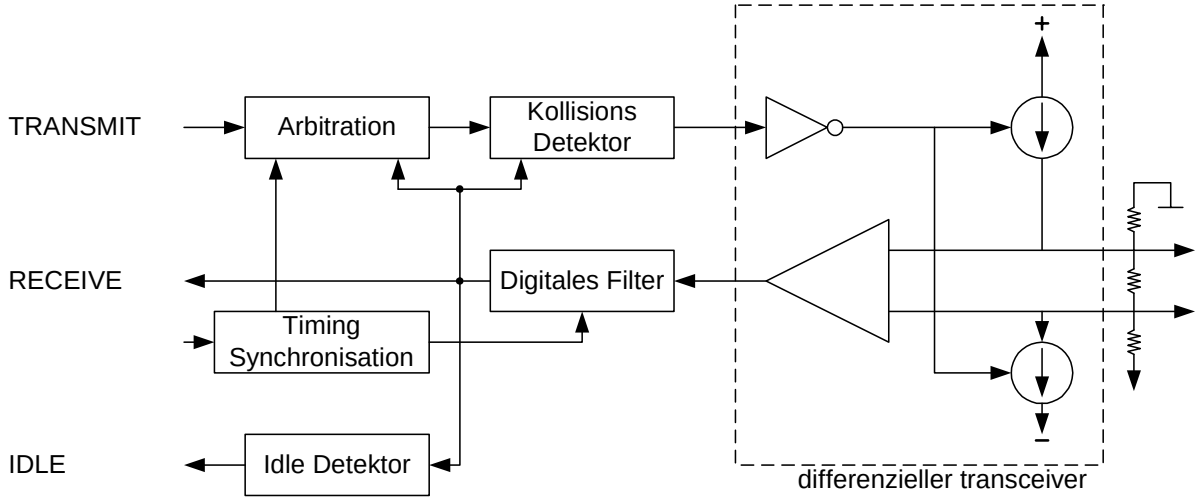


Abb. 7-3 : Vereinfachtes Blockdiagramm des Businterface

Um Kollisionen erkennen zu können, werden vom Kollisionsdetektor alle gesendeten und empfangenen Nachrichten aufgezeichnet. Das Timing für die Aufzeichnung der Daten wird von der Timing-Synchronisation vorgegeben. Die Nachricht mit der höchsten Priorität wird jeweils weitergeleitet.
Die Arbitration-Einheit arbeitet zusammen mit der Timing Synchronisation und entscheidet, wann Nachrichten gesendet und empfangen werden dürfen. Solange ein Knoten Nachrichten sendet, verhindert die Arbitration-Einheit das Senden von Nachrichten durch andere Knoten. Erst wenn der Idle-Zustand wieder erreicht ist, darf der nächste Knoten senden.

7.1.4 Fehlertoleranz

Wenn bei einem Knoten ein Fehler auftritt, so kann das System dennoch weiterarbeiten.

Literatur

Jurgen, R. K.: Automotive Electronics Handbook; McGraw-Hill-Handbooks, Second Edition, 1999

7.2 SAE J1850

Ist/war der Low End Bus in USA. Hat den Sprung nach Europa nie so richtig geschafft. Selbst Ford benutzt zwar in USA den J1850, aber für seine EU-Modelle den ISO9141 SABD Bus.

Attributes/Features

- Class B In-Vehicle network
- open architecture
- master-less
- single level
- 2 alternatives: 41.6 kBit/s PWM (2 wire) or 10.4 kBit/s VPW (single wire)
- for diagnosis and data sharing purposes
- applications: engine management, transmission, ABS, instrumentation
- asynchronous messages
- CSMA/CR
    - CS: carrier sense (listen before transmitting)
    - MA: multiple access
    - CR: collision resolution

7.3 USB (Universal Serial Bus) 1.1 / 2.0

7.3.1 Einleitung

Der USB Bus Standard 1.1 wurde 1995 eingeführt. Ziel war es damals ein Plug&Play fähiges Bussystem zu entwickeln. Anfänglich wurde USB vorrangig eingesetzt, wenn es darum ging Peripherie Geräte mit dem Computer zu verbinden. Doch immer komplexere Applikationen, wie z.B. Übertragung von Video und Multimedia Daten ließen den USB Bus, aufgrund seiner geringen Übertragungsrate von 10Mbit/s, schnell an seine Grenzen stoßen.
Im Jahre 2000 wurde mit einer Erweiterung der Spezifikation auf USB 2.0 dieser Nachteil ausgeglichen.

7.3.2 Unterschied USB 1.1 und USB 2.0

Die wesentliche Neuerung der zweiten Auflage ist die vierzigmal schnellere Datenübertragung als bei Version 1.1. USB 2.0 ist trotzdem vollständig abwärts-kompatibel zum bestehenden Standard. Vorhandene Kabel und Geräte können weiterhin genutzt werden.
Die USB-2.0-Spezifikation sieht vor, die Timeframes im Millisekundenbereich (USB 1.1) in jeweils acht Timeframes zu je 125 Mikrosekunden zu unterteilen. Somit wird die vierzigfache höhere Datenrate von 480 MBit/s erreicht.

Beim Einstecken eines USB-Gerätes schaltet der USB-2.0-Controller automatisch auf die jeweils geforderte Übertragungsgeschwindigkeit um. Dank der Abwärtskompatibilität können sämtliche Datenraten ohne Geschwindigkeitsverlust gleichzeitig genutzt werden.
Ein USB-2.0-Gerät, angeschlossen an einem USB-1.1-Controller kann allerdings nur die maximale Übertragungsrate von 10 MBit/s nutzen.

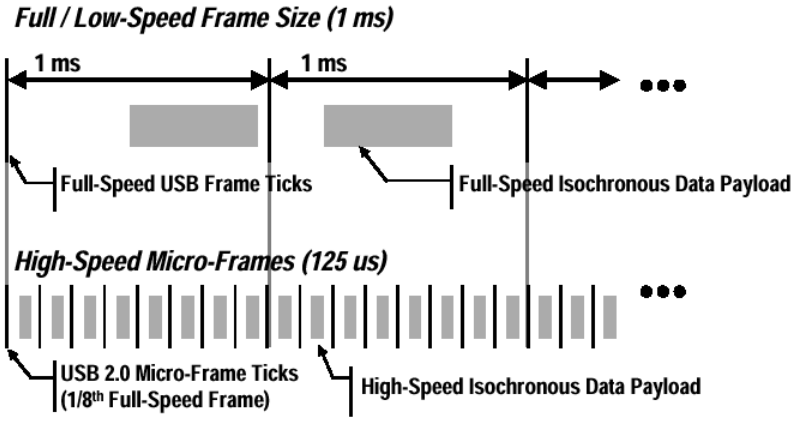


Abb. 7-4 : Frames und Mikroframes

7.3.3 Architektur

Ein USB System besteht immer aus einem Host (in der Regel ein PC) und bis zu 127 Peripherie Geräten (Devices). Die Devices sind am Host sternförmig, u.U. über einen Hub angeschlossen. Hubs dienen zur Erweiterung der Anzahl der verfügbaren. Es dürfen jedoch nur maximal fünf Hubs zwischen Host und Device eingesetzt werden.

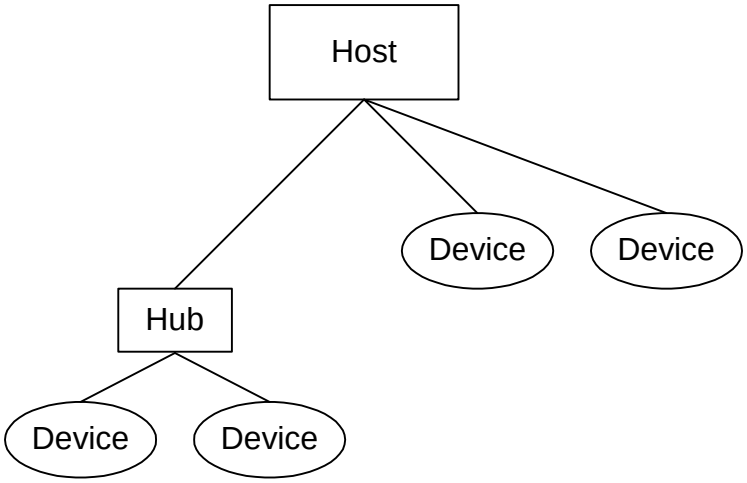


Abb. 7-5 : USB Busarchitektur

7.3.4 Protokoll

Host und Devices kommunizieren über Pipes (logische Kanäle) miteinander. Für jede Pipe sind Endpunkte im Host bzw. Device vorhanden. Je nach Übertragungsgeschwindigkeit steht eine unterschiedliche Anzahl von adressierbaren Endpunkten zur Verfügung, wobei alle Devices den Endpunkt 0 unterstützen.

Die Übertragung der Daten erfolgt asynchron oder isynchron (konstante Bandbreite) mit verschiedenen Transfers.

Control Transfers für die Konfiguration von Devices. Paketgröße ist maximal 64 Bytes. Control Transfer sind als einzige Transfervariante Bidirektional.

Isochronous Transfers für die periodische und kontinuierliche Übertragung von zeitkritischen Daten wie z. B. Audio oder Video. Paketgröße ist maximal 1023 Bytes.

Interrupt Transfers für kleine Datenpakete geringer Häufigkeit (z.B. Maus). Paketgröße maximal 8 Bytes für low-speed und 64 Bytes für full-speed Devices.
Bulk Transfers für große Datenmengen und nicht-periodische Übertragung. Dieser Transfer wird für Anwendungen verwendet, die keine bestimmte Bandbreite benötigen und deren Daten auch verzögert gesendet werden können (z.B. Drucker). Paketgröße ist maximal 64 Bytes.

Beim Anschluss neuer Geräte oder beim Reset werden alle Endpunkte auf die default Adresse 0 gesetzt. Erst danach beginnt der Host mit der Konfiguration der

Devices, bei der die Betriebsparameter wie z.B. Anzahl der Endpunkte, Device Adresse, Datenaufkommen, Art der Stromversorgung usw. festgelegt werden. Danach kann ein Device auf Requests vom Host reagieren.

7.3.5 Protokoll Overhead

Neben den zu übertragenden Daten muss jedes Datenpaket aufgrund von Synchronisations- und Fehlererkennungszwecken mit einem Overhead versehen werden. Je nach Übertragungsgeschwindigkeit und der Transfervariante ist die Anzahl, der für den Overhead verwendet Bytes, sowie die Struktur der Datenpakete, variabel.

7.3.6 Folgende Felder werden verwendet:

- Sync Feld enthält Synchronisationsinformationen für den Empfänger.
- Packet Identifier Feld (PID) beinhaltet Informationen über die verwendete Fehlererkennung und Typ des Datenpakets.
- Adress- und Endpunkt-Feld enthalten Adressen des Empfängers oder Senders.
- CRCs enthält die CRC über die Daten oder Adress und Endpunkt Feld.

Beispiele (Sync Felder weggelassen)

Der Start eines neuen Frames wird durch das Start-of-Frame gekennzeichnet:

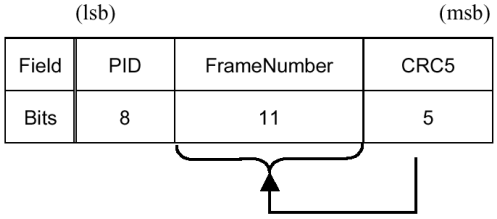


Abb. 7-6 : Start-Of-Frame

Die zu übertragenden Daten befinden sich in einem Datenframe:

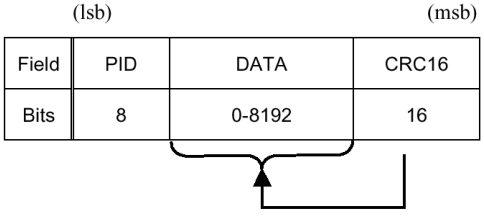


Abb. 7-7 : Datenframe

Literatur

[1] Universal Serial Bus Specification, Revision 2.0 April 27, 2000 http://www.usb.org/
[2] Universal Serial Bus Specification, Revision 1.1 September 23, 1998 http://www.usb.org/

8. Multimedia

8.1 Bussysteme für Multimedia

Die Übertragung von bewegten Videobildern im Automobil dient entweder der Unterhaltung oder direkt dem Fahren. Beispiele aus dem Bereich der Unterhaltung sind etwa der Fernsehempfang oder die Möglichkeit Videofilme zu sehen. Elektronische Rückspiegel, Seitenspiegel oder Frontsichtsysteme für Nachtfahrten auf Basis von Videokameras sind Beispiele für Anwendungen, die direkt dem Fahren dienen.

Die Übertragung von bewegten Videobildern kann grundsätzlich analog oder digital erfolgen, wobei die analoge Videoübertragung stark auf die Bildübertragung fixiert ist. Die digitale Breitbandübertragung kann neben der Videoübertragung ebenso gut für die Übertragung anderer digitaler Informationen genutzt werden.

Je nachdem, ob Videobildsequenzen nur dargestellt oder weiterverarbeiten werden sollen, ergeben sich sehr unterschiedliche Anforderungsprofile an die benötigten Bussysteme. Die Darstellung von Videofilmen beschränkt sich im wesentlichen auf die Übertragung und Dekodierung von im MPEG-2-Format komprimierten Datenströmen mit Datenraten im Bereich einiger MBit/s. Dabei ist vom verwendeten Bussystem eine garantierte untere Bandbreite sicherzustellen, um eine ruckelfreie Darstellung zu gewährleisten.

Sollen von Kameras stammende Videobildsequenzen in Echtzeit weiterverarbeitet werden, so müssen diese unkomprimiert Übertragen werden. Die MPEG-2-Kompression ist ein sehr stark asymmetrisches Verfahren: Für die Kodierung eines MPEG-2-Videodatenstroms wird etwa die hundertfache Rechenleistung wie für die Dekodierung eines MPEG-2-Videodatenstroms benötigt. Zwar kann die MPEG-2-Kompression mit leistungsfähigen Prozessoren oder speziellen Chips mittlerweile in Echtzeit erfolgen, die Benötigte Verarbeitungskapazität liegt dabei in der Größenordnung von Milliarden Operationen pro Sekunde. Ein unkomprimierter Videodatenstrom (PAL-Farbfernsehen, 25 Bilder / s mit ca. 800x600 Bildpunkten und 3 Byte (RGB, TrueColor) pro Bildpunkt) benötigt eine Bandbreite von 36 MByte / s also in der Größenordnung von 300 MBit/s.

Als Multimediale Standards stehen derzeit im wesentlichen folgende zu Verfügung:

- IEEE1394 (Firewire, I.Link)
- GigaStar
- MOST
- D2B

8.2 D2B

D2B steht für Digital Data Bus und wurde von Philips für Audio- und Videokommunikation insbesondere auch für Anwendungen im Automobilbereich vorgesehen. Die Signalübertragung erfolgt über Twisted Pair Leitungen. Die maximal mögliche Datenrate beträgt 1 MBit/s. Die Datenbits werden puls-weiten-moduliert (pwm) und setzen sich aus vier Bereichen zusammen: Vorbereitungsperiode, Synchronisationsperiode, Datenperiode und Stopperiode. Die Dauer der Perioden und Bits des D2B Busses wird während einer Verbindung bestimmt. Dabei stehen drei mögliche Übertragungsgeschwindigkeiten zur Auswahl.

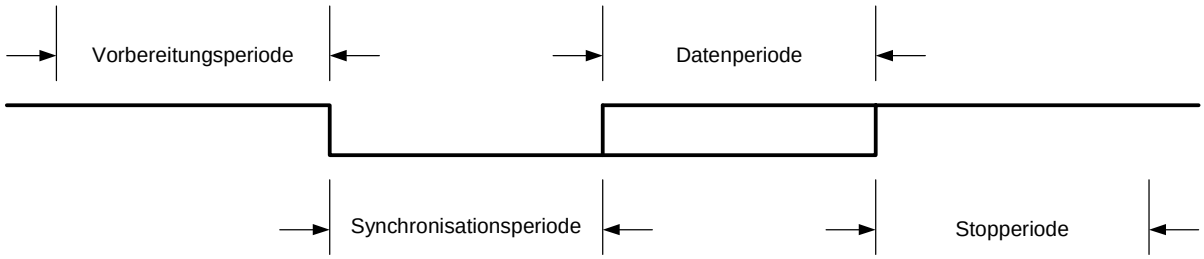


Abb. 8-1 : Bereiche der Datagramme

Die Arbitrierung wird über nichtdestruktive, bitweise Priorisierung gesteuert. Gleichzeitig sendende Netzwerkknoten entscheiden zuerst anhand des Modusfelds, in welchem Mode der Netzwerkknoten innerhalb eines Drei-Bit-Feldes arbeiten soll. Low Pegel sind dominant. Der Modus bestimmt die Übertragungsgeschwindigkeit. Die übertragene Datenmenge innerhalb eines Zeitfensters hängt von der während der Arbitrierung bestimmten Übertragungsgeschwindigkeit ab. Ein Datenframe besteht aus sechs Feldern. Je ein Paritätsbit (P) wird hinter Masterfeld, Slavefeld, Kontrollfeld sowie an das Ende der Datenbytes angehängt. Die maximale Länge eines Datenframes beträgt 47 Bits.

Abb. 8-2 : Struktur der Datagramme

Das Handshaking erfolgt durch positive Quittierung (Acknowledgement Bit A) innerhalb übertragener Daten. Eine fehlende Quittierung seitens eines Slaves wird als negative Quittierung interpretiert. Der Master kann dann erneut versuchen, Datenfelder zu übertragen. Während der Übertragung eines Datenframes haben mehrfach Quittierungen nach verschiedenen Frame-Segmenten zu erfolgen. Ein Master kann einen Slave auf seine Adresse fixieren, wodurch die Kommunikation des betreffenden Slaves mit anderen Mastern unterbunden werden kann. Dies erfolgt, wenn eine Datenübertragung das vorgesehene Zeitfenster überschreitet und der Master erneut eine Arbitrierung für den Bus durch führen muss, um die Datenübertragung zu vervollständigen. Fehlererkennung erfolg anhand der Paritätsbits (P). Ein Acknowledgement Bit (A) wird nicht durch einen adressierten

Slave weitergeleitet, wenn ein Paritätsfehler festgestellt wird, die selektierte Übertragungsgeschwindigkeit zu hoch ist, ein Timing-Fehler aufgetreten ist, ein Slave einem anderen Masterzugeordnet ist, oder wenn der Empfangspuffer voll ist.

Literatur

[1] Ronald K. Jurgen, Automotive Electronics, McGraw-Hill, 1999

8.3 GIGASTAR

GigaSTAR (Gigabit / s Tansmitter and Receiver) ist ein proprietärer Standard der *Firma Inova Semiconductors*. GigaSTAR überträgt 36 Bit Datenworte bit-seriell mit 1,3 G Bit / s von einem Sender auf einen Empfänger. Die Distanz zwischen Sender und Empfänger darf bis zu 50 m betragen [1] bzw. bis zu 100 m mit Shielded Twisted Pair Kabeln laut [2]. Nach Außen ist die GigaSTAR bezüglich der Datenübertragung transparent – 36 parallel am Transmitter anliegende Signale werden mit 33 MHz abgetastet und stehen als 36 parallele Signale an den Ausgängen des Receivers zur Verfügung. Mit 36 parallel zur Verfügung stehenden Bits können digitalisierte Videosignale zusammen mit Audiosignalen und Steuersignalen übertragen werden. Die Verbindung ist jedoch unidirektional, so dass für solche Steuersignale, die zurück an eine Datenquelle übertragen werden sollen, gegebenenfalls ein zusätzlicher Übertragungskanal erforderlich ist. Aufgrund der Transparenz der Datenübertragung ist GigaSTAR für solche Anwendungen geeignet, bei denen ein Protokoll-Overhead vermieden werden soll.

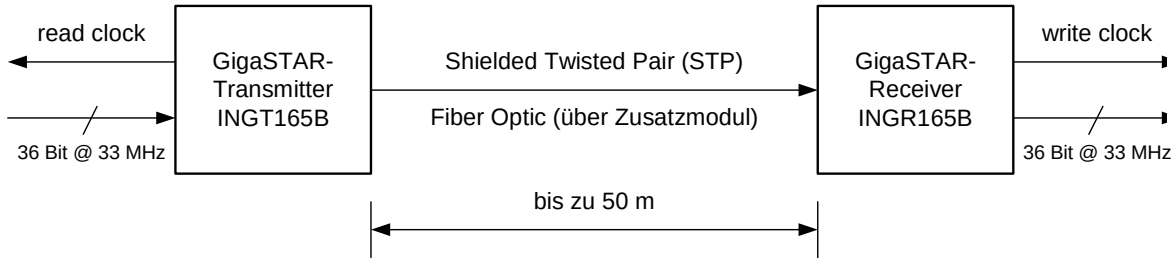


Abb. 8-3 : Aufbau

Literatur

[1] GigaSTAR Transceiver ING165B_DS Data Sheet, 07/2001 – rev. 1.6, http://www.inova-semiconductors.de/

[2] Robert Kraus, Gigabit-Bus – GigaStar bringt 1,2 Gbit / s Nutzdatenrate ins Auto,
F&M Jahrg. 109 (2001) 7-8, S. 25-29, Carl Hansa Verlag, München

[3] Robert Kraus, Ein Bus für alle Fälle, Elektronik Automotive, Heft April 2002, S. 44-49, Franzis Verlag, München, April 2002

8.4 DVI (Digital Visual Interface)

8.4.1 Einleitung

1998 formierte sich ein Konsortium aus Herstellern mit dem Namen **Digital Display Working Group**. Ziel war es, eine auf die Marktbelange abgestimmte Schnittstelle zu definieren. Hinter der DDWG stehen die Firmen *Intel, Compaq, Fujitsu, Hewlett Packard, IBM, NEC* und *Silicon Image*. Basis der Übertragung ist TMDS[1] . Um die begrenzte Bandbreite zu erhöhen, wurde das Verfahren erweitert.

Der neue Schnittstellenstandard DVI wurde am 2. April 1999 in der heute gültigen Version 1.0 veröffentlicht.
DVI hat sehr gute Aussichtschancen, denn das digitale Übertragungsprotokoll ist das TMDS. Im Gegensatz zu P&D und DFP, die nur einen Link besitzen, verdoppelt ein zweiter Link die gesamte Übertragungsbandbreite und somit die maximale Pixelrate. Damit werden Auflösungen über 1280 x 1024 Pixels möglich.
Ein weiterer Vorteil von DVI ist die Tatsache, dass analoge Bildsignale übertragen werden können. Damit ließen sich auch noch Röhrenmonitore anschließen.

Schnittstelle	**Plug&Display**	**DFP**	**DVI**
Info	VESA	DFP	DDWG
Bilddaten	analog/digital	digital	analog/digital
Protokoll	TMDS	TMDS	TMDS
Kanäle	3 Kanäle	3 Kanäle	6 Kanäle
Bandbreite	165 MHz	165 MHz	330 MHz
Anschluss-Pins	30/34 (analog)	20	24/29 (analog)
max. Auflösung	1280 x 1024	1280 x 1024	2048 x 1536
max.Kabellänge (abgeschirmt)	5 m	10 m	10 m
Farbtiefen	12/24 Bit	12/24 Bit	12/24 Bit
Anschlussbreite	40,6 mm	33,4 mm	37,0 mm
Kompatibilität	--	zu Plug&Display über Adapter	zu Plug&Display und DFP über Adapter
zusätzliche Features	USB, IEEE 1394	--	--

Abb. 8-4 : Vergleich einzelner Standards

[1] Transition Minimized Differential Signaling. Übertragungsverfahren für digitale Bilddaten, das mit Parallel/Seriell-Wandlung und Spannungsdifferenzen arbeitet.

8.4.2 TMDS

Dieser Standard wurde von Silicon Image geformt und von der VESA (Video Electronics Standards Association) als Standard anerkannt. Er unterstützt die gesamte Bandbreite der Display-Technologie, also neben aktiven und passiven LCDs auch Plasma-Displays und Projektoren. Als Quelle können entsprechend ausgerüstete Grafikkarten, Notebooks, PDAs oder DVD-Player dienen.

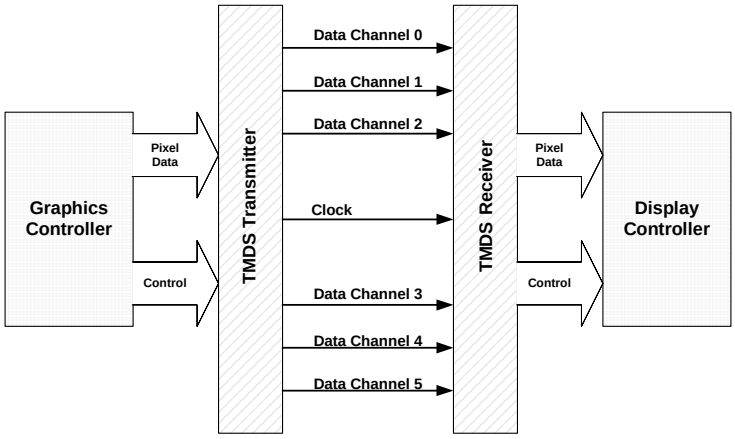


Abb. 8-5 : TMDS [2]

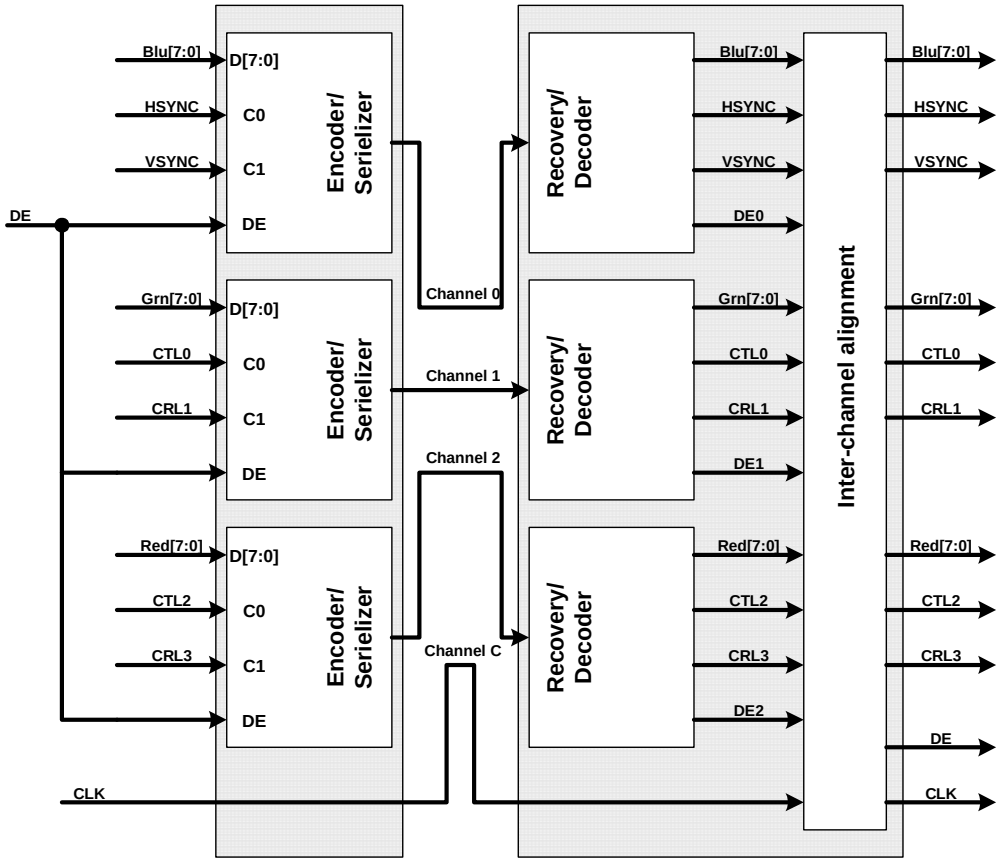


Abb. 8-6 : Single link

Bei DVI kommen zu den drei Kanälen von TMDS drei weitere, wodurch sich die Bandbreite auf 330 MHz verdoppelt. Dieses so genannte Duallink-Verfahren verteilt die Bandbreite auf beide Links gleichermaßen. Für beide Verbindungen kommt dieselbe Taktleitung zum Einsatz. Bei der Datenübertragung kümmert sich die erste Verbindung um die ungeraden Pixels, die zweite um die geraden. Nach dem Start des Systems baut sich zunächst eine Single-Link-Verbindung auf, erst wenn der Monitor seine Tauglichkeit für eine zweite Verbindung bescheinigt hat, wird diese aktiviert.

Die Schnittstelle ist rein digital, bei älteren Varianten müssen die Daten erst analog und dann wieder digital gewandelt werden, was preislich gesehen ein großer Nachteil ist. Gelangen die Bildinformationen dagegen direkt vom Grafikcontroller zum Display, ist die umständliche und vor allem verlustbehaftete DA-AD-Wandlung überflüssig. Die teuren Konverter können entfallen. Zwar müssen ein Sende- und Empfangsbaustein vorhanden sein, doch der Preis für diese ICs liegt weit unter denen für die Wandler.

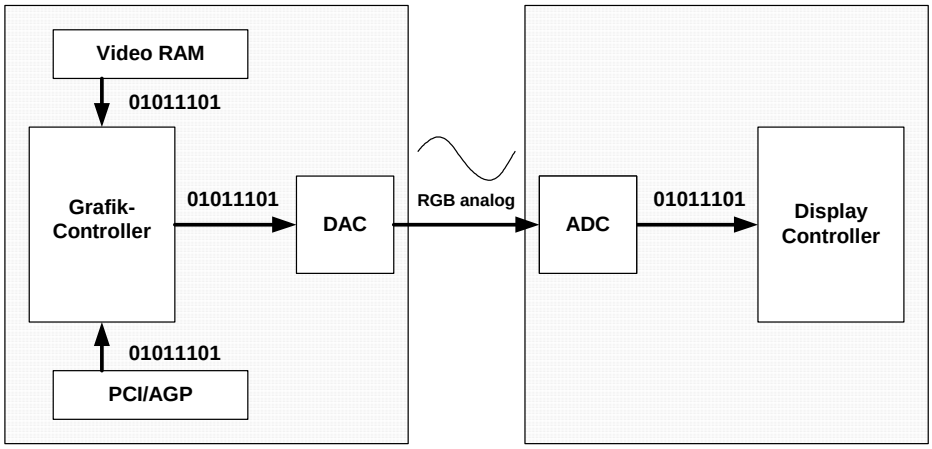


Abb. 8-7 : Display mit analogem Eingang

8.4.3 DVI-Steckverbindungen

Es existieren zwei Varianten des DVI-Steckers. DVI-V besitzt 24 Pins, und ist nur für digitale Verbindungen zuständig. Für die Verbindung zu einem analogen Gerät wäre weiterhin ein VGA-Anschluss (RGB) notwendig. Mittlerweile fast alle Grafikkartenherstellen bieten neben dem herkömmlichen VGA-Anschluss auch einen DVI-Stecker.

Abb. 8-8 : DVI-V(GA) / DVI-I(ntegrated) [1]

Der DVI-I besitzt vier weitere Pins hinzu für die Übertragung von analogen Signalen. So können auch herkömmliche Röhren-Bildschirme angeschlossen werden.

8.4.4 Fazit

Unter den digitalen Schnittstellen bietet die DVI-Variante die besten Voraussetzungen für die Zukunft. Neben der hohen Auflösung und Übertragungsrate (4,95 Gbit/s) bleiben die analogen Geräte nicht auf der Strecke. Fast alle Grafikkartenhersteller bieten neben S-Video (TV) auch eine DVI-Verbindung.

Literatur

[1] Digitale Grafikschnittstellen, Januar 2000, www.tecchannel.de
[2] Digital Visual Interface Specification, Revision 1.0 April 1999, www.ddwg.org

9. Subbus

9.1 K-Line / L-Line / ISO9141 / KWP2000

9.1.1 ISO 9141

Ursprünglich Physical Layer von K/L-Diagnosebus und K-Bus, im wesentlichen ein Level Shifter von 5 V aus dem µC auf 12 V. Viele ASICs und ASSPs haben den Transceiver mit integriert. Es gibt auch etliche Stand-Alone-Transceiver auf dem Markt (Philips, Motorola, Siliconix, …). Wenn die Übertragungsgeschwindigkeit reicht (bis 20 kBit/s), dann ist er genauso für die LIN-Übertragung geeignet.

9.1.2 K-Bus

Der „Haus-Bus“ von BMW, aber auf absteigendem Ast. Wird langfristig durch LIN ersetzt

9.2 I2C (Inter IC Bus)

9.2.1 Beschreibung

Der I^2C-Bus ist von Philips entwickelt worden und wurde in der ersten Zeit speziell für die Steuerung von Video-und Audio-IC’s verwendet. Seitdem wurde aber auch der I^2C-Bus für Module in der Industrie entdeckt, da er nur 2 IO-Pins am µC benötigt. Der I^2C-Bus arbeitet mit den Leitungen SCL (Clock) und SDA (Daten). Beide Leitungen werden über je eine Stromquelle mit I_{i2cmax}=3mA – die in der Regel aus einem Widerstand besteht – auf den 1 Pegel gesetzt, wenn weder Master noch Slave aktiv sind. Daher arbeitet weder der Master, noch der Slave im Push-Pull-Betrieb, sondern setzt den Ausgang auf den 0-Pegel oder wird hochohmig. Da der 1-Pegel daher rezessiv ist, ist ein Multimasterbetrieb einfach zu realisieren, indem im Adressenfeld des Telegramms die niederwertigste Adresse bei gleichzeitigem Senden die 1-Pegel überlagert und so eine einfache Arbitrierung ermöglicht.

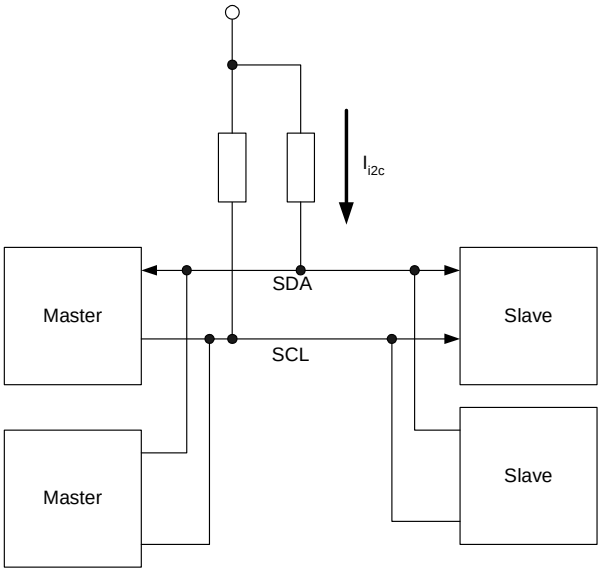


Abb. 9-1 : I2C Aufbau

Gestartet wird eine Übertragung mit einer Startbedingung in dem bei SCL=1, die Leitung SDA auf 0 gesetzt wird. Beendet wird die Übertragung indem bei gesetztem SCL die SDA-Leitung wieder auf 1 gesetzt wird. Die Übertragung eines Bits erfolgt, indem zuerst das Datenbit gesetzt wird, um anschließend nach einer Setupzeit die SCL-Leitung auf 1 und dann wieder zurückzusetzen. Erst nach einer weiteren Setupzeit darf dann die Datenleitung wieder zurückgesetzt werden.

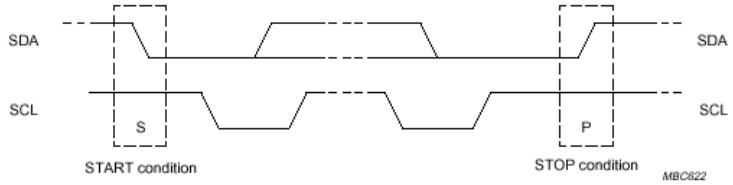


Abb. 9-2 : I2C Übertragungsformat

Ein komplettes Telegramm zum Senden von Daten an einen Slave besteht aus der Startbedingung, der Adresse, dem RW-Bit, einem Acknowlegde-Bit und weiteren Datenbytes mit je einem Acknowledge-Bit im Anschluss. Ein Acknowledge-Bit bestätigt den Empfang von Daten, indem der Empfänger gültige Daten durch das Setzen der SDA-Leitung auf 0 bestätigt. Wenn der Master Daten anfordert, so sendet er eine gültige Adresse mit dem RW-Bit auf 1-Pegel gesetzt und erwartet ein Ack-Bit vom Slave. Anschließend sendet der Slave mit jedem SCL-Takt seinen Daten auf die SDA-Leitung und der Master muss jedes Byte mit einem Ack-Bit bestätigen. Das letzte Ack-Bit kann vom Empfänger nicht bestätigt werden, um anzuzeigen, dass keine weiteren Daten empfangen werden sollen.

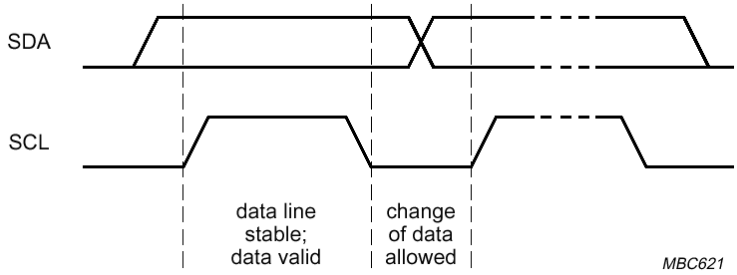


Abb. 9-3 : I2C Übertragungsformat

9.2.2 Positive Eigenschaften

Der I^2C-Bus ist billig als Master in einem Mikrocontroller zu implementieren. Zudem benötigt er nur 2 IO-Pins an einem Mikrocontroller. Die Flankensteilheit lässt sich zumindest bei den positiven Flanken gut kontrollieren.

9.2.3 Negative Eigenschaften

Der I^2C-Bus ist nicht gegen Fehler gesichert ist und ist relativ langsam. Die Implementation eines Slaves per Software erfordert erhöhte Fähigkeiten, was die Start- und Stopbedingung betrifft, da diese Zeitkritisch sind. Allerdings besteht immer noch die Freiheit die Vorgaben von Philips in eigenen Implementationen einzuschränken.

9.3 SPI

9.3.1 Beschreibung

Der SPI-Bus wird hauptsächlich für die Kommunikation zwischen einem Mikrokontroller und peripheren ICs verwendet. Seine Verwendung ist einfach, da eine synchrone Kommunikation verwendet wird und kein Adressenfeld in den Daten ausgewertet werden muss, sowie der Mikrocontroller immer der einzige Master am Bus ist. Der SPI-Bus besteht aus den Signalen Clock, Dateneingang und Datenausgang, sowie einer ChipSelect-Leitung für die Auswahl eines Slave. Allerdings ist das SPI-Protokoll nicht standardisiert, so das die beispielhafte Beschreibung hier nur stellvertretend für einen Ausschnitt von existierenden SPI-Alternativen sein kann. Am Beginn einer Kommunikation wird die ChipSelect-Leitung (CS) auf den Pegel 0 gesetzt, um den Slave in den aktiven Zustand zu versetzen. Zugleich kann der Datenausgang für das erste Bit gesetzt werden, um dann anschließend – nach einer einzuhalten Setupzeit – die Clock-Leitung von 0 auf 1 zu setzen. In diesem Augenblick wird das Bit in das Schieberegister des Slaves übernommen und gleichzeitig kann ein Slave auf dem Datenausgang ein Datenbit ausgeben, da anschließend vom Mikrocontroller übernommen werden kann. Anschließend wiederholt sich der Vorgang mit jeder steigenden Flanke des Clock-Signals. Nach Abschluss der Übertragung wird das ChipSelect-Signal wieder auf 0

gesetzt und die Kommunikation ist beendet. Das Beispiel in verdeutlicht die Beschreibung. Es wird ein Wort mit 4 Bit Breite gesendet und empfangen. Gesendetes Wort: „1010“, empfangenes Wort „1100“

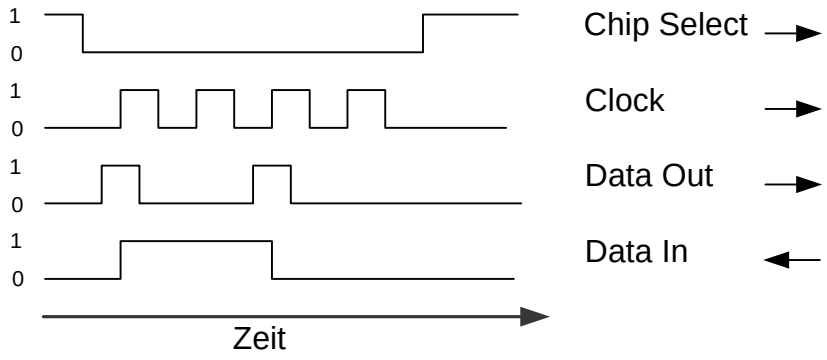


Abb. 9-4 : Übertragungsformat

9.3.2 Positive Eigenschaften

Der SPI-Bus ist bis zu etwa 5Megabit /Sekunde schnell und ist einfach per Software im Master zu implementieren und daher gut geeignet für die Kommunikation in kleinen Systemen. Zum Beispiel ist es möglich mit einem ATmega323-µC von Atmel ohne Probleme mehr als 4 Peripheriechips anzusprechen.

9.3.3 Negative Eigenschaften

Die Flankensteilheit der Signale wird nicht kontrolliert und die Übertragung ist nicht gegen Fehler abgesichert. Daher ist auf die EMV-Eigenschaften der Schaltung zu achten.

9.4 RS-232

9.4.1 Beschreibung

Die RS232-Schnittstelle ist den meisten bekannt und zeichnet sich immer noch durch ihre Beliebtheit in der Elektrotechnik aus. Es ist eine reine Punkt zu Punkt Verbindung. Die RS232-Übertragung (Abb. 9-5 : RS-232 Verbindung) besteht im einfachsten Fall aus 3 Leitungen, der TXD-Leitung zum Senden der Daten vom PC zu einer Baugruppe, der RXD-Leitung zum Empfangen der Daten von einer Baugruppe und der Masse-Leitung. Daneben können noch Handshake-Leitungen benutzt werden, auf die aber beim Betrieb mit Mikrocontrollern häufig verzichtet wird.

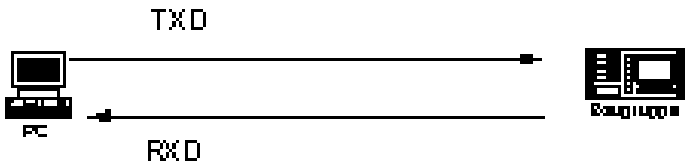


Abb. 9-5 : RS-232 Verbindung

Das RS232-Protokoll ist Asynchron und deshalb wird in einem festen Zeitraster gearbeitet (Abb. 9-6). Im Ruhezustand sendet die RS232-Schnittstelle Stopbits mit dem 1-Pegel. Soll ein Wort (5 Bits-8 Bits) übertragen werden, so wird die Leitung für eine Bitzeit auf 0-Pegel gesetzt (Startbit). Anschließend werden alle Datenbits übertragen. Wird noch ein Paritybit gewünscht, so wird dieses im Anschluss gesendet. Zum Schluss wird noch mindestens ein Stopbit gesendet. Die Anzahl der Stopbit ist entweder eins oder zwei, wobei in der Regel nur ein Stopbit gewählt wird. Die Geschwindigkeit einer RS232-Schnittstelle wird in Baud angeben und stellt nichts anderes dar als die Einheit Bit/Sekunde. Es gibt die folgenden Standartbaudraten: 110, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600 und 115200. Auf der Leitung zum Empfänger werden die Pegel mit Hilfe eines Tranceivers noch codiert, so dass ein 1-Pegel mit einem Spannungsbereich von –3V bis –12V repräsentiert wird und ein 0-Pegel wird von einem Spannungsbereich von 3V bis 12V repräsentiert.

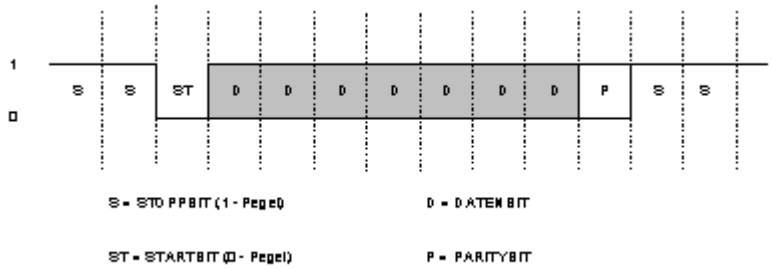


Abb. 9-6 : UART Format

9.4.2 Positive Eigenschaften

RS232 ist eine viel gebrauchte Schnittstelle und wird Hardware- und Softwareseitig gut unterstützt und ist daher einfach in eigenen Projekten zu integrieren.

9.4.3 Negative Eigenschaften

Es ist kein Netzwerkbetrieb möglich und die Übertragungsgeschwindigkeiten niedrig.

9.5 RS-485

9.5.1 Einleitung

Die RS485-Schnittstelle ist besonders in der Automatisierungstechnik und der Messtechnik bekannt. Insbesondere besteht die einfache Möglichkeit kleinere Netzwerke zu bilden. Das RS485-Netzwerk besteht im einfachsten Fall aus einem Master mit seinen Slaves, der auf der differentiellen Leitung eine Adresse und beliebig viele Daten sendet.

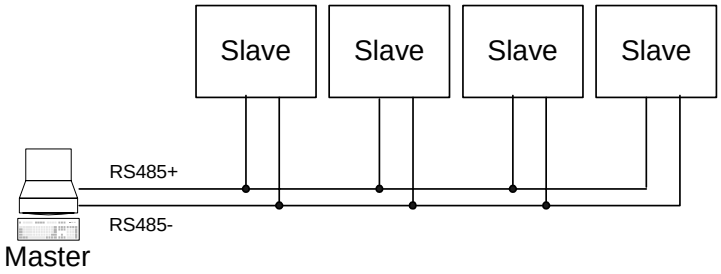


Abb. 9-7 : Einfaches RS-485 Netzwerk

Die Slaves antworten nur auf Anforderung des Masters. Wenn klassische Transceiver – welche die Bedingungen der RS485-Spezifikation einhalten – können bis zu 32 Busteilnehmer ohne Repeater am Bussegment angeschlossen werden. Wenn mehr Busteilnehmer erwünscht sind, lassen sich auch Transceiver verwenden, die einen Innenwiderstand von 48kOhm haben und bis zu 128 Busteilnehmer ermöglichen.

Die RS485-Busleitung ist differentiell ausgelegt und eignet sich bei Verwendung entsprechender Kabel für Übertragungsraten bis zu 10 Megabit pro Sekunde. Die Pegel auf den Leitung reichen von –7V bis 12V. Normalerweise wird für die Übertragung eine ganz normale RS232-Uart mit Rs485-Transceivern verwendet, so dass die Implementation einer RS485-Schnittstelle kein großes Problem darstellt.

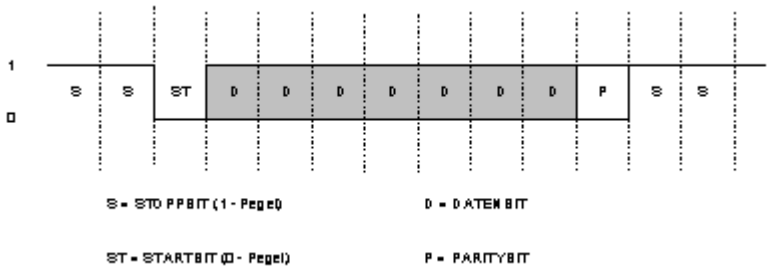


Abb. 9-8 : UART Format

Die Geschwindigkeit einer RS485-Schnittstelle wird in Baud angeben und stellt nichts anderes dar als die Einheit Bit/Sekunde. Es gibt die folgenden Standartbaudraten: 110, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600 und 115200.

9.5.2 Positive Eigenschaften

RS485 bietet Netzwerkbetrieb und hohe Übertragungsgeschwindigkeiten. Die Implementation ist billig und kann einfach an die eigenen Bedürfnisse angepasst werden.

9.5.3 Negative Eigenschaften

Das Protokoll sichert nur die Definition der physikalischen Übertragung, aber nicht zum Beispiel die Länge des Adressenfeldes und die Sicherung der Daten über eine Checksumme. Daher sollte auf zu hohe Übertragungsgeschwindigkeiten verzichtet werden oder eine Checksummenüberprüfung implementiert werden.

10. High-Speed / Real Time

10.1 Byteflight

10.1.1 Einleitung

Durch die steigenden Anforderungen und Anzahl der Funktionen (Sensoren und Aktoren) in der Automobilentwicklung, werden immer mehr elektronische Systeme statt mechanischer Komponenten eingesetzt. Diese können nicht mit einem zentralen Steuergerät realisieren werden. Die Lösung liegt bei einer Vernetzung der verschiedenen Elektronikbaugruppen über einen Hochleistungs-Datenbus zur Reduzierung des Verkabelungsaufwands und Mehrfachnutzung von Sensordaten.

Um diese Anforderungen zu erfüllen, wurde von der *BMW AG* zusammen mit den Firmen *Motorola*, *Siemens AG* und *ELMOS AG* das Hochleistungs-Datenbussystem **byteflight** entwickelt.

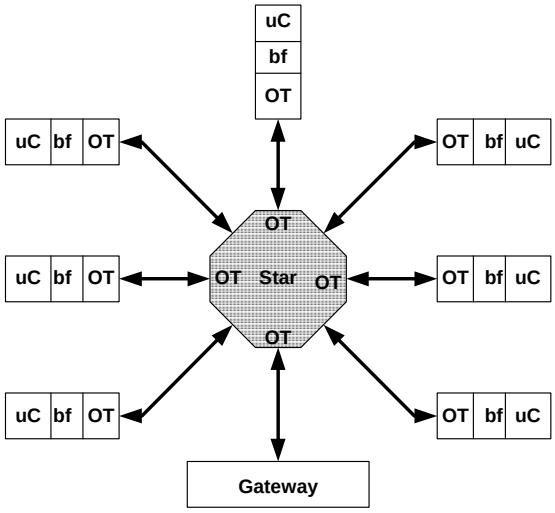
10.1.2 Buszugriff und Struktur

Zeitgesteuerte Datenprotokolle stellen jedem Teilnehmer ein vordefiniertes Raster mit einer Übertragungszeit zur Verfügung (z.B. TTP). Die Anzahl der Nachrichten ist vorgegeben und kann nicht im Betrieb verändert werden. Ereignisgesteuerte Datenprotokolle übertragen nur dann Daten, wenn einer Sendeanforderung vorliegt (z.B. CAN).

Byteflight vereint Vorteile von synchronen und asynchronen Verfahren. Es garantiert deterministische Latenzzeiten für eine bestimmte Anzahl von hochprioren Nachrichten und flexible Nutzung der Übertragungsbandbreite durch niederpriore Telegramme.
Der Buszugriff erfolgt nach **FTDMA** (Flexible Time Division Multiple Access).

FTDMA ist ein rein zeitgesteuertes Verfahren. Alle Teilnehmer starten, durch einen Synchronisations-Impuls getriggert, sog. Slotzähler, die bis zum höchstmöglichen Identifierwert zählen. Wenn die Zähler einen ID-Wert erreichen, für den eine Sendeanforderung vorliegt, wird die Nachricht mit diesem Identifier über den Bus übertragen. Für die Dauer der Übertragung stoppen die Slotzähler auf dem aktuellen Wert. Am Ende der Übertragung zählen die Slots weiter.

Bei Ausfall des Sync-Masters und damit der Synchronisationsimpulse kann ein bestimmter Teilnehmer (Ersatz-SYNC-Master) durch seinen uC als Master konfiguriert werden, der dann die Sync-Pulse sendet. Die Datenübertragung ist durch die Sternkopplertopologie gewährleistet (s. Abbildung 1).



OT:
optical
Tranceiver

bf:
byteflight-
controller

uC:
uController

Abb. 10-1 : Struktur des Bussystems nach [2]

10.1.3 Telegrammaufbau

Byteflight-Nachrichten bestehen aus einer 6Bit-Startsequenz, einem Identifier- und Längenbyte, bis 12 Datenbytes und zwei CRC(Circle Redundancy Check) Bytes (40Bit Overhead). Zur Bitsynchronisation ist jedes Byte von je einem Start- und Stop-Bit begrenzt. Die Bitdauer beträgt 100ns bei einer Datenrate von 10 Mbit/s.

ID: 8bit, 1_{10}... 255_{10}, wobei 1_{10} höchste Priorität.
LEN: Bit 0...3 (LSBs): LEN = number of data bytes (0_{10}...12_{10})
Bit 4…7 (MSBs): 4 additional information/data bits
D0…D11: Data bytes, max. 12 Datenbytes
CRCH/ CRCL: Cyclic Redundancy check sequence

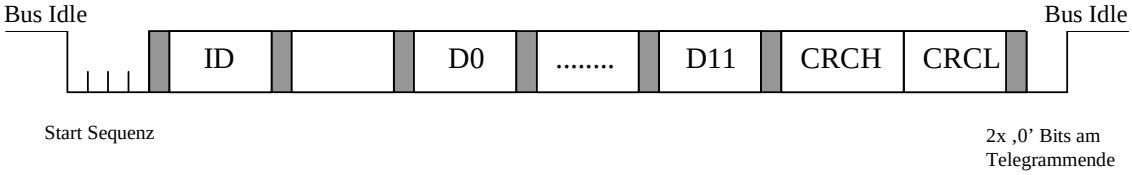


Abb. 10-2 : Telegrammaufbau nach [2]

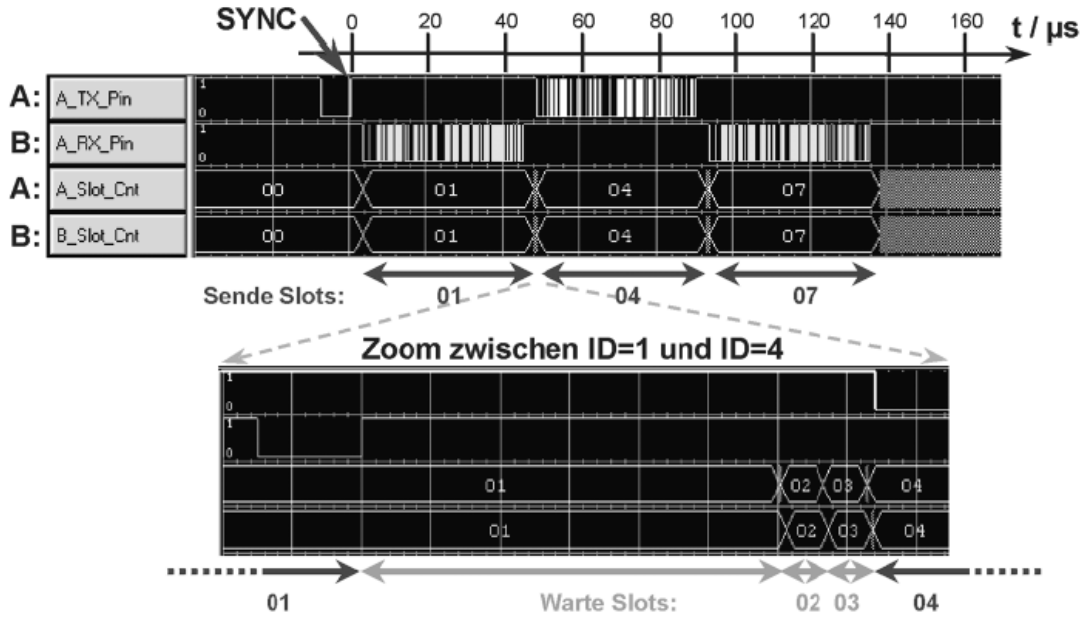


Abb. 10-3 : Datenübertragung nach [2]

Übertragungsbeispiel:
Teilnehmer A sendet den Identifier 4, Teilnehmer B die IDs 1 und 7. Die Sende-Slots 1, 4 und 7 dauern so lange, wie es die Übertragung der Nachrichten erfordert. Für die ID 2 und 3 liegen keine Sendeanforderungen vor, die Slots 2 und 3 sind nicht belegt und erscheinen nur als sehr kurze Warte-Slots.

Das TDMA-Verfahren ist ein rein zeitgesteuertes Buszugriffsverfahren. Es ermöglicht einerseits die Übertragung einer bestimmen Anzahl von hochprioren Nachrichten in jedem Kommunikationszyklus und andererseits die flexible und statische Zuteilung der Bandbreite an restlichen Nachrichten (s. Abbildung 10-4). Hier werden die höchstprioren Nachrichten synchron und zyklisch alle 250us übertragen. Der zweite Teil des Kommunikationszyklus kann dagegen für ereignisgesteuerte Nachrichten verwendet werden.

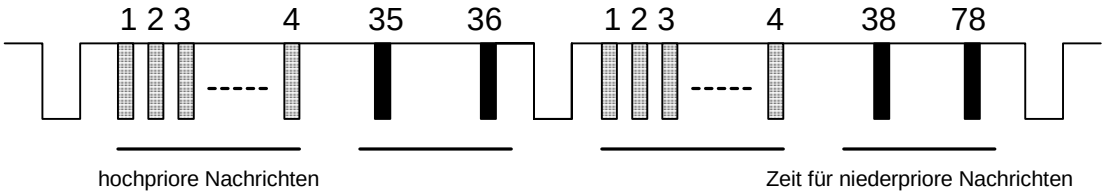


Abb. 10-4 : Synchrone und asynchrone Nutzung der Bandbreite nach [2]

Da die Datenübertragung von sicherheitsrelevanten Signalen generell zyklisch mit einer Updaterate von 250us erfolgt, ist bei Übertragungsfehlern auf Protokollebene kein Wiederholungsmechanismus vorgesehen. Das Protokoll ist so ausgelegt, dass selbst bei starken Störungen spätestens mit dem nächsten Synchronisationsimpuls die Kommunikation wieder definiert weiterläuft.

Das Protokoll bietet eine weitere spezielle Eigenschaft, die besonders für passive Sicherheitssysteme notwendig ist. Der Sync-Master hat die Möglichkeit, die Art des Synchronisationsimpulses zu verändern, um ein Alarm-Zustand an zu zeigen. Dieser Zustand wird sowohl von allen byteflight-Controllern und Treiberbausteinen erkannt und kann dazu verwendet werden, bestimmte Sicherheitsfunktionen frei zu schalten. Die Umschaltung des Synchronisationsimpulses hat auf den Kommunikationszyklus keinerlei Einfluss.

Es werden nur fehlerfrei empfangene Nachrichten, mit gültigem CRC, der Host-CPU zum Auslesen bereit gestellt. Die Sterntopologie ermöglicht die Abschaltung von Teilnehmern, die zum Beispiel das Busprotokoll verletzen oder Übertragungsfehler verursachen. Die optischen Transceiver bieten durch eine Abschaltfunktion Schutz vor Blockade des Datenbusses.

10.1.4 Aussichten

Wegen des deterministischen Verhaltens, der flexiblen Bandbreitennutzung, Systemerweitbarkeit und der hohen Übertragungsrate ist dieses Protokoll ein prädestiniertes Übertragungsverfahren im Automobilbereich.
Der erste Einsatz von byteflight soll in den nächsten zwei Jahren bei BMW in passiven Sicherheitssystemen stattfinden.

Durch die Übertragungsrate von 10Mbit/s und relativ hohen Störsicherheit durch die optische Übertragung, sind Anwendungen in der industriellen Automatisierungstechnik und Luft- und Raumfahrttechnik denkbar.

Literatur

[1] byteflight Spezifikation, http://www.byteflight.com/specification/index.html

[2] ATZ Sonderausgabe,
http://www.byteflight.com/presentations/atz_sonderausgabe.pdf

[3 Introduction to byteflight Technology,
http://www.byteflight.com/presentations/introduction.pdf

11. Wireless

11.1 Bluetooth

Der Bluetooth-Standard bezeichnet eine Datenfunktechnologie, die ursprünglich dazu entwickelt worden ist, Kabelverbindungen zwischen dem PC und Peripheriegeräten zu ersetzen. Mittlerweile ist daraus ein universell einsetzbarer Standard zur Übertragung von Daten und Sprache über kurze Distanzen (bis ca. 10 Meter) geworden. Die Spezifikation liegt zur Zeit in der Version 1.1 vor und kann bei der Bluetooth-SIG (Special Interest Group) heruntergeladen werden (http://www.bluetooth.com).

Das Haupteinsatzgebiet von Bluetooth ist die Vernetzung von Computern und Peripheriegeräten und die spontane Bildung sehr kleiner Datennetze (Personal Area Networks (PAN), bei Bluetooth auch Piconet genannt) an beliebigen Orten. So wird der Austausch von Daten zwischen verschiedensten Geräten möglich, ohne Kabelverbindungen herstellen zu müssen. Dabei ist in einem Bluetooth-Netzwerk (Piconet) entweder eine Punkt-zu-Punkt oder eine Punkt-zu-Mehrpunkt-Verbindung möglich, mit jeweils einem Master und einem oder mehreren Slaves. Es kann auch weitere Master geben, die jedoch inaktiv, aber synchronisiert zum aktuellen Master bleiben, solange dieser aktiv ist. Damit ist ein Bluetooth-Netzwerk bedingt multimasterfähig.

Das System arbeitet im lizenzfreien 2,4GHz-Band. Schon aus dieser Tatsache heraus ergibt sich jedoch, dass ein Bluetooth-Netzwerk leicht durch andere, ebenfalls im 2,4GHz-Band arbeitende Funksysteme gestört werden kann. Damit ist das System absolut nicht für sicherheitskritische Systeme im Kraftfahrzeug (beispielsweise Fahrzeuglenkung, Motorsteuerung oder gar Airbag) geeignet, wohl aber für unkritische Komfortanwendungen, wie z.B. Multimedia-Systeme im Kraftfahrzeug oder ständige Überwachung des Reifendrucks während der Fahrt. Auch für Abrechnungssysteme in Parkhäusern oder an Mautstellen und für Diagnosegeräte mit drahtloser Anbindung ist der Einsatz von Bluetooth im Kraftfahrzeug sehr gut denkbar. Es ist dabei jedoch darauf zu achten, dass sich Bluetooth und andere im Fahrzeug vorhandene Systeme nicht gegenseitig stören.

Der Bluetooth-Standard teilt das zur Verfügung stehende 2,4GHz-Band in 79 Kanäle ein. Bluetooth-Knoten wechseln automatisch bis zu 1600 Mal pro Sekunde den Kanal, so dass andere Sender, die zufällig gerade denselben Kanal benutzen, nur kurz (für sicherheitsrelevante bzw. auf Echtzeitbetrieb angewiesene Systeme jedoch zu lange) stören können. Jedes Datenpaket wird auf einem anderen Kanal übertragen. Werden mehrere Bluetooth-Netzwerke nahe beieinander betrieben (was leicht der Fall sein kann, wenn mehrere mit Bluetooth ausgerüstete Fahrzeuge hintereinander oder nebeneinander stehen oder fahren), nimmt die Übertragungsgeschwindigkeit stark ab, da dann häufiger Wiederholungen von Datenpaketen notwendig sind. Somit kann eine Echtzeitfähigkeit nicht garantiert werden.

In einer Bluetooth-Verbindung stehen mehrere Übertragungskanäle zur Verfügung: ein asynchroner Datenkanal, der entweder im asymmetrischen Betrieb mit 723,2 kBit/s in der einen und 57,6 kBit/s in der anderen Richtung oder im symmetrischen Betrieb mit 433,9 kBit/s in beiden Richtungen arbeitet. Außerdem stehen drei synchrone Kanäle mit je 64 kBit/s zur Übertragung von Sprache zur Verfügung.

Jedes Datenpaket besteht aus einem Header und einem Nutzdatenteil. Der Header besteht dabei aus dem 72 Bit breiten „Access Code“ sowie einem 54 Bit breiten Adressfeld, welches sich aus einer 48 Bit breiten „Company ID“, einem beliebigen (vom Hersteller des Gerätes vergebenen) 16-Bit-Wert sowie Fehlerkorrekturbits zusammensetzt. Die Company ID wird dabei durch die Bluetooth Special Interest Group vergeben. Im Nutzdatenteil können pro Paket bis zu 2745 Bits (ungefähr 343 Bytes) übertragen werden. Zur Übertragung des Headers wird ein fehlerkorrigierender Code (FEC, Forward Error Correction) und für die Nutzdaten ein fehlererkennender Code (mit CRC-Prüfsummen) benutzt. Die Daten können verschlüsselt (mit 64 bis 128 Bits Schlüssellänge) oder unverschlüsselt übertragen werden.

Die Arbitrierung erfolgt nach dem TDD (Time Division Duplex)-Schema. Dabei wird jeder Kanal in 625µs lange Zeitscheiben aufgeteilt. Die Synchronisierung erfolgt durch einen vom Master vorgegebenen Zähler. Enthält der Zähler einen geraden Wert, darf der Master eine Übertragung starten. Enthält hingegen der Zähler einen ungeraden Wert, darf ein Slave eine Übertragung starten. Eine Übertragung darf insgesamt maximal 5 Zeitscheiben beanspruchen.

Da Bluetooth unter anderem auch für batteriebetriebene Mobilgeräte entwickelt wurde, wurde Wert auf minimalen Energieverbrauch der Interface-Module gelegt. Der Energieverbrauch solcher Module liegt bei 30 bis 100mA und ist somit für Anwendungen im Kraftfahrzeug als eher niedrig einzustufen.

Von mehreren Herstellern, unter anderem Ericsson, Nokia, Motorola, Intel und IBM, sind Chips und Multichip-Module für Bluetooth verfügbar. Dabei machen die Multichip-Module die Integration von Bluetooth in vorhandene oder neue Entwicklungen besonders einfach, da diese bereits alle notwendigen Komponenten enthalten, um eine Bluetooth-Schnittstelle zu realisieren.

Literatur

[1] http://www.bluetooth.com

12. Systemarchitektur

Diskussion.

Anhang

Tabellarische Auflistung der einzelnen Bussysteme

General Purpose

A-Bus

Eigenschaft / Bussystem	A-BUS „Automotive Bit-Serial Interface System“
Applikation: Automobil?	Gedacht als universeller Bus, Alternative zu CAN
Applikation: Home?	keine (nicht bekannt)
Applikation: Industrie?	Keine (nicht bekannt)
Standard?	Volkswagen-intern
Website für Standard	Unbekannt
Wer steht dahinter? (Organisation)	Volkswagen AG (proprietär)
Medium (phys. Layer)	Eindraht, Bordnetz (nicht festgelegt, aber typ. Realisierung)
Encoding	NRZ
Synchron, Asynchron?	Asynchron
Media Access, Arbitration, multi Master fähig?	
Priorisierung von Transfers möglich?	Nein
Echtzeitfähig: Zeit für MS-Datentransfer (Read / Write)	Nicht echtzeitfähig, abhängig davon, wie viele andere Sender sich in Reichweite befinden (je mehr, desto langsamer, da dann eventuell mehrere Wiederholungen nötig)
Overhead pro Datenpaket (Bytes)	15 Bit Overhead bei 16 Bit Nutzdaten
Datenblocklängen (von ... bis)	2 Byte, fest
Genauigkeit clock Übereinstimmung	
Clock synchronisation	
Error detection / correction	Start Bit Error, TX Error, RX Error, Short Circuit werden erkannt
Sicherheit / Redundanz	8-fach oversampling
Bitrate (von...bis)	Nicht spezifiziert, 500kBit maximal
Buslänge (von...bis)	Nicht spezifiziert - typ. 30m
Anzahl Nodes Identifier	2048
Anzahl Nodes Physikalisch	Ca. 30
Hardware verfügbar?	Früher ja, aktuell nicht mehr
EMV-Aspekte	
Wake-Up?	Ja, unter Verlust erste Message
Lizenzgebühr	Unklar
Bewertung: Kosten für Master / Slave	<2$
Bewertung: Zukunftsaussichten (Anwendungsgebiet)	

CAN / TTCAN

Eigenschaft / Bussystem	CAN	TTCAN
Applikation: Automobil?	Antriebsstrang, Karosserie	←
Applikation: Home?	-	-
Applikation: Industrie?	Maschinensteuerung, Fabrikvernetzung	möglich
Standard?	ISO CAN2.0	Ja
Website für Standard	http://www.can-cia.de	←
Wer steht dahinter? (Organisation)	Can in Automation (CiA) Ursprung: Bosch	←
Medium (phys. Layer)	2-Draht, twisted pair	←
Encoding	NRZ (5 Bit)	←
Synchron, Asynchron?	Synchron	←
Media Access, Arbitration, multi Master fähig?	CSMA/CD; Beliebiger Zugriff mit Priorisierung durch bitweiser Ver-Undung der Identifier	Aufteilung in Zeitslots, zusätzlich herkömmliche Arbitration
Priorisierung von Transfers möglich?	Ja	Ja, je Zeitslot
Echtzeitfähig: Zeit für MS-Datentransfer (Read / Write)	Delay bei max. Priorität: 130 Bitzeiten. Für niedrigere Priorität nicht deterministisch.	Entsprechend der Abstände der Zeitslots pro Teilnehmer
Overhead pro Datenpaket (Bit)	31 + 11 (2.0A) /29 (2.0B) Adresse, 15 CRC	←
Datenblocklängen (von ... bis)	0 ... 8 Byte	←
Genauigkeit clock Übereinstimmung	0,5%, Quarz nötig	←
Clock synchronisation	DPLL	← und auf Ebene der Basiszyklen
Error detection / correction	detection, 15 bit CRC: 5 single bit / burst 14 bit	←
Sicherheit / Redundanz	Abgesicherte Übertragung, autom. retry	← und redundanter Zeitmaster
Bitrate (von...bis)	10k ... 1Mbit, innerhalb eines Systems fest	←
Buslänge (von...bis)	40m (1Mbit) ... 1000m	←
Anzahl Nodes Identifier	11 bit (2.0A) / 29 bit (2.0B)	←
Anzahl Nodes Physikalisch	bis zu einige 10	←
Hardware verfügbar?	CAN-Interfaces und diverse Controller, sowie Microcontroller	Erste Implementationen in HW, weitere angekündigt
EMV-Aspekte	gering, das twisted pair	←
Wake-Up?	Möglich, spez. ID	←
Lizenzgebühr	Ja	←
Bewertung: Kosten für Master / Slave	0,50€ Tranceiver, 0,5-4 € Controller	←
Bewertung: Zukunftsaussichten (Anwendungsgebiet)	Gut, hat sich in vielen Bereichen etabliert	Gut, da immernoch einfaches und durchgängiges Konzept

VAN / single-Wire CAN / low-speed CAN

Eigenschaft / Bussystem	Low-Speed CAN	Single Wire CAN	VAN
Applikation: Automobil?	Antriebsstrang, Karosserie	Karosserie	←
Applikation: Home?	-	-	-
Applikation: Industrie?	-	-	-
Standard?	Ja	Ja	Ja
Website für Standard	http://www.can-cia.de	←	http://www.van-mux.org
Wer steht dahinter? (Organisation)	Can in Automation (CiA)	←	Ursprung: PSA
Medium (phys. Layer)	2-Draht, twisted pair, keine Abschlußwid.	1-Draht	2-Draht, twisted pair
Encoding			“Enhanced Manchester”
Synchron, Asynchron?			
Media Access, Arbitration, multi Master fähig?			CSMA/CD; wie CAN
Priorisierung von Transfers möglich?			Ja
Echtzeitfähig: Zeit für MS-Datentransfer (Read / Write)			In-frame-reply-Mechanismus
Overhead pro Datenpaket (Bit)			64 incl. CRC, plus 2 pro Byte
Datenblocklängen (von ... bis)			0 .. 28 Byte
Genauigkeit clock Übereinstimmung	1,5% durch geringf. Anpass. Keramikresonator		3%, Slaves 20% durch Längenmeßfeld
Clock synchronisation			DPLL, Frequenzmessung
Error detection / correction			←
Sicherheit / Redundanz	Abgesicherte Übertragung, autom. Retry + Fehlertoleranz gegen Kurzschluß an VB/GND oder zw. CAN+/CAN-	Abklemmen des Empfängers bei Masseverlust	wie CAN
Bitrate (von...bis)	10K ... 125 KBit	33Kbit, (80)	60k ... 1Mbit
Buslänge (von...bis)	einige m	einige m	
Anzahl Nodes Identifier			12 bit
Anzahl Nodes Physikalisch	typ. unter 40	typ. unter 40	←
Hardware verfügbar?	div. Treiber	div. Treiber	Als Peripheriechip, ca. 3-4KGate komplex
EMV-Aspekte	gering, das twisted pair	Kontrolle der Flankensteilheit	wie CAN
Wake-Up?	Ja	Ja	Möglich
Lizenzgebühr	Ja	←	?
Bewertung: Kosten für Master / Slave	0,50€ Tranceiver, 0,5-4 € Controller	←	ähnlich CAN
Bewertung: Zukunftsaussichten (Anwendungsgebiet)	Gut, hat sich in vielen Bereichen etabliert	geringe Baudrate vs. Einsparung eines Drahtes	Steht im Schatten von CAN, da sehr ähnlich

SAE J1567

Eigenschaft / Bussystem	SAE J1567 C^2D
Applikation: Automobil?	Auto in-vehicle
Applikation: Home?	
Applikation: Industrie?	
Standard?	
Website für Standard	-
Wer steht dahinter? (Organisation)	Chrysler
Medium (phys. Layer)	2-draht, twisted pair
Encoding	NRZ (10 Bit)
Synchron, Asynchron?	asynchron
Media Access, Arbitration, multi Master fähig?	Ja, bitweise über Identifier
Priorisierung von Transfers möglich?	Ja
Echtzeitfähig: Zeit für MS-Datentransfer (Read / Write)	Delay bei max. Priorität: 82 Bitzeiten
Overhead pro Datenpaket (Bit)	34 Bit, 68 % bei 16 Datenbits
Datenblocklängen (von ... bis)	1 – 6 Byte
Genauigkeit clock Übereinstimmung	
Clock synchronisation	
Error detection / correction	checksum
Sicherheit / Redundanz	
Bitrate (von...bis)	7.812 Kbps
Buslänge (von...bis)	not specified, typical > 30 m
Anzahl Nodes Identifier	not specified
Anzahl Nodes Physikalisch	
Hardware verfügbar?	Harris Corporation
EMV-Aspekte	Tiefpassfilter, um EMI herauszufiltern
Wake-Up?	möglich
Lizenzgebühr	
Bewertung: Kosten für Master / Slave	
Bewertung: Zukunftsaussichten (Anwendungsgebiet)	

USB

Eigenschaft / Bussystem	USB 1.1	USB 2.0
Applikation: Automobil?	MP3	Multimedia
Applikation: Home?	Kommunikation mit Peripheriegeräten (Drucker, Maus, Tastatur)	Kommunikation mit externen Datenspeichern (externe CD-Brenner und Festplatten)
Applikation: Industrie?	-	-
Standard?	USB 1.1	USB 2.0
Website für Standard	http://www.usb.org	http://www.usb.org
Wer steht dahinter? (Organisation)	Compaq, Intel, Microsoft, NEC	Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, Philips
Medium (phys. Layer)	4-Draht: 2x Versorgung, 2x Daten (differenziell)	4-Draht: 2x Versorgung, 2x Daten (differenziell)
Encoding	NRZI	NRZI
Synchron, Asynchron?	Isynchron, Asynchron	Isynchron, Asynchron
Media Access, Arbitration, multi Master fähig?	Host Zentriert	Host Zentriert
Priorisierung von Transfers möglich?	Ja	Ja
Echtzeitfähig: Zeit für MS-Datentransfer (Read / Write)	Ja	Ja
Overhead pro Datenpaket (Bit)	9…45 Bytes je nach Übertragungsart	9…173 Bytes je nach Übertragungsart
Datenblocklängen (von ... bis)	0 – 1023 Bytes	0 – 3072 Bytes
Genauigkeit clock Übereinstimmung	Low Speed +/- 15000ppm Full Speed +/- 2500 ppm	Low Speed +/- 15000ppm Full Speed +/- 2500 ppm High Speed +/- 500ppm
Clock synchronisation	PLL	PLL
Error detection / correction	CRC in Control + Data Fields	CRC in Control + Data Fields
Sicherheit / Redundanz	Timeout, ACK, Retry 3x (Nur Asynchron)	Timeout, ACK, Retry 3x (Nur Asynchron)
Bitrate (von...bis)	Low Speed: 10-100 kbit/s Full Speed: 500k – 10Mbit/s	Low Speed: 10-100 kbit/s Full Speed: 500k – 10Mbit/s High Speed: 25 – 400Mbit/s
Buslänge (von...bis)	Bis 5m (zwischen 2 Geräten) Bis 30m Gesamtdistanz	Bis 5m (zwischen 2 Geräten) Bis 30m Gesamtdistanz
Anzahl Nodes Identifier	127	127
Anzahl Nodes Physikalisch	127	127
Hardware verfügbar?	USB-Interfaces, Hubs, Endgeräte	USB-Interfaces, Hubs, Endgeräte
EMV-Aspekte	Nein	Nein
Wake-Up?	Suspend Mode	Suspend Mode
Lizenzgebühr	Keine	Keine
Bewertung: Kosten für Master / Slave	Für mehrere USB Geräte ist ein HUB notwendig!	Für mehrere USB Geräte ist ein HUB notwendig!
Bewertung: Zukunftsaussichten (Anwendungsgebiet)	Weite Verbreitung, USB 1.1, könnte durch Bluetooth abgelöst werden	USB 2.0 oder Firewire wird sich durchsetzen.

Multimedia

IEEE1394

Eigenschaft / Bussystem	IEEE 1394
Applikation: Automobil?	Navigationssystem Rückfahrkamera Video-Infotainment
Applikation: Home?	Home A/V Network, Konsumerelektronik
Applikation: Industrie?	Industrieautomatisierung, Medizintechnik
Standard?	IEEE 1394 – 1995
Website für Standard	http://www.ieee.org
Wer steht dahinter? (Organisation)	IEEE, Apple, Sony
Medium (phys. Layer)	4-Draht: 2x Twisted Pair für Daten 6-Draht: wie oben, 2x Versorgungsspannung
Encoding	NRZ
Synchron, Asynchron?	Isynchron, Asynchron
Media Access, Arbitration, multi Master fähig?	Multimaster, Point-to-point
Priorisierung von Transfers möglich?	
Echtzeitfähig: Zeit für MS-Datentransfer (Read / Write)	
Overhead pro Datenpaket (Bit)	20% Overhead, 80% Nutzdaten
Datenblocklängen (von ... bis)	
Genauigkeit clock Übereinstimmung	maximal 1/1000% Taktabweichung
Clock synchronisation	Nodes wirken als Repeater
Error detection / correction	CRC
Sicherheit / Redundanz	Timeout, ACK, Retransmission (Nur Asynchron)
Bitrate (von...bis)	100, 200 und 400Mbit/s
Buslänge (von...bis)	4,5 zwischen Zwei Nodes maximale Länge 72m (mit Nodes als Repeater)
Anzahl Nodes Identifier	64
Anzahl Nodes Physikalisch	64
Hardware verfügbar?	Vorallem im Bereich externe Speicher und Video
EMV-Aspekte	
Wake-Up?	
Lizenzgebühr	0,25$ pro System
Bewertung: Kosten für Master / Slave	< 5$
Bewertung: Zukunftsaussichten (Anwendungsgebiet)	Neuer Standard IEEE1394 ist in der Entwicklung und verspricht Datenraten bis zu 3.2 Gbit/s

GIGASTAR / MOST

Eigenschaft / Bussystem	MOST (Media Oriented Systems Transfer)	GigaSTAR
Applikation: Automobil?	Multimedia im Automobil	Multimedia & General Purpose
Applikation: Home?	Multimedia in Haus	
Applikation: Industrie?		Datenübertragung in Maschienen
Standard?	MOST Cooperation	
Website für Standard	http://www.mostnet.de/	http://www.inova-semiconductors.de/
Wer steht dahinter? (Organisation)	MOST Cooperation gegründet 1998 von BMW, Daimler-Chrysler, Harman/Becker, OASIS Silicon Systems	Inova Semiconductors
Medium (phys. Layer)	optisch	Kabel (STP) oder Fiberoptik
Encoding	biphase mark coding	proprietär (DC-balanced)
Synchron, Asynchron?	Fenster im Daten-Frame für synchrone & asynchrone Kommunikation	synchron mit Taktrückgewinnung
Media Access und Arbitration, Multi Master fähig?	Single-/ Multi-Master	peer-to-peer
Priorisierung von Transfers möglich?	Ja	
Echtzeitfähig: Verzögerungszeit für MS-Datentransfer (Read / Write)		Latenz anhängig von Kabellänge
Overhead pro Datenpaket (Bit)	32 Bits pro 512 Bit-Frame	10%
Datenblocklängen (von ... bis)	60 Byte (bei 512 Bit-Frame)	transparent, da parallel-seriell-parallel
Genauigkeit der Taktfrequenz	Master-Takt wird von Slaves rekonstruiert	
Clock-Synchronisation	PLL	
Error detection / correction	Frame Parity Bit, 16-Bit-CRC	Parity-Bit intern
Sicherheit / Redundanz	Bitfehlerrate < 10^{-10}	-
Bitrate (von...bis)	bis 25 Mbit / s	1,32 G bits / s
Buslänge (von...bis)		bis zu 50 m
Anzahl Nodes Identifier	physikalische Adresse, Logische Adresse (2 Bytes), Gruppenadresse (1 Byte)	Sender – Empfänger
Zahl der Netzknoten (physikalisch)	2 bis 64	2
Hardware verfügbar?	Transceiver (elektrisch & optisch), Controller (http://www.oasis.com/)	Transmitter und Receiver (http://www.inova-semiconductors.de/)
EMV-Aspekte (Ausstrahlung, Störempfindlichkeit)	optische Datenübertragung	optisch / STP
Wake-Up?	möglich	
Lizenzgebühr	Ja	
Bewertung: Kosten für Master / Slave		
Bewertung: Zukunftsaussichten (Anwendungsgebiet)	Primär Automotive ggf. in Kombination mit Geräten aus Consumer / Home / Office	nur peer-to-peer, 36 Bit parallel, seriell übertragen

D2B

Eigenschaft / Bussystem	D2B
Applikation: Automobil?	Multimedia im Automobil
Applikation: Home?	Multimedia in Haus
Applikation: Industrie?	
Standard?	Philips
Website für Standard	http://www.philips.com/
Wer steht dahinter? (Organisation)	Philips
Medium (phys. Layer)	twisted pair
Encoding	pwm bit encoding
Synchron, Asynchron?	
Media Access und Arbitration, Multi Master fähig?	
Priorisierung von Transfers möglich?	Ja
Echtzeitfähig: Verzögerungszeit für MS-Datentransfer (Read / Write)	
Overhead pro Datenpaket (Bit)	34 / (34 + 256) = 11,7 %
Datenblocklängen (von ... bis)	1 ... 32 Data Bytes pro Frame
Genauigkeit der Taktfrequenz	
Clock-Synchronisation	
Error detection / correction	Parity Bits
Sicherheit / Redundanz	
Bitrate (von...bis)	bis 1 Mbit / s
Buslänge (von...bis)	
Anzahl Nodes Identifier	
Zahl der Netzknoten (physikalisch)	
Hardware verfügbar?	
EMV-Aspekte (Ausstrahlung, Störempfindlichkeit)	twisted pair
Wake-Up?	Reduzierung der Leistungsaufnahme für Bus-Controller möglich
Lizenzgebühr	
Bewertung: Kosten für Master / Slave	
Bewertung: Zukunftsaussichten (Anwendungsgebiet)	Baudrate ist für zukünftige Multimediaanwendungen zu gering

Subbus

LIN

Eigenschaft / Bussystem	Local Interconnect Network
Applikation: Automobil?	Comfort electronics (Seat, Mirrors, Instrumentation)
Applikation: Home?	
Applikation: Industrie?	
Standard?	LIN Specification Rev. 1.2
Website für Standard	http://www.lin-subbus.org/
Wer steht dahinter? (Organisation)	LIN Sub-Bus-Konsortium
Medium (phys. Layer)	Single wire 12 (14) V
Encoding	Single Master, Multiple Slave
Synchron, Asynchron?	NRZ 8N1 (UART)
Media Access, Arbitration, multi Master fähig?	Asynchron
Priorisierung von Transfers möglich?	?
Echtzeitfähig: Zeit für MS-Datentransfer (Read / Write)	Nein
Overhead pro Datenpaket (Bit)	14 Bit Sync Break + 8 Bit Header + 6 Bit Identifier
Datenblocklängen (von ... bis)	2 … 8 Byte
Genauigkeit clock Übereinstimmung	?
Clock synchronisation	Selbstsynchronisation des Slaves ohne Quarz
Error detection / correction	8-Bit Checksumme
Sicherheit / Redundanz	
Bitrate (von...bis)	1.4 kb/s, 9.6 kb/s, und 19.2kb/s
Buslänge (von...bis)	
Anzahl Nodes Identifier	2 … 10 Knoten
Anzahl Nodes Physikalisch	≤ 12 Knoten
Hardware verfügbar?	LIN Transceiver und Controller
EMV-Aspekte	
Wake-Up?	unterstützt
Lizenzgebühr	Nein
Bewertung: Kosten für Master / Slave	50Cent Transceiver, 0.5 – 2 € Controller
Bewertung: Zukunftsaussichten (Anwendungsgebiet)	Wird voraussichtlich **der** Low End Class A Automotive Bus

RS232 / RS485

Eigenschaft / Bussystem	RS232	RS485
Applikation: Automobil?	-	-
Applikation: Home?	PC-Maus,Modem	-
Applikation: Industrie?	Parametrierungen und Programmierung von Steuerungen	Parametrierungen und Programmierung von Steuerungen
Standard?	V24	**EIA-RS485**
Website für Standard		
Wer steht dahinter? (Organisation)	CCITT	ISO/ITU
Medium (phys. Layer)	minimum 2-Draht-Kommunikation	minimum 2-Draht-Kommunikation
Encoding	NRZ	NRZ
Synchron, Asynchron?	Asynchron	Asynchron
Media Access, Arbitration, multi Master fähig?	Nein	Nein
Priorisierung von Transfers möglich?	Nein	Nein
Echtzeitfähig: Zeit für MS-Datentransfer (Read / Write)	Maximaler Zeitbedarf sind 11 Bitzeiten für die Übertragung eines Bytes.	Maximaler Zeitbedarf sind 11 Bitzeiten für die Übertragung eines Bytes plus die Zeit für die Senderichtungsumschaltung.
Overhead pro Datenpaket (Bit)	max. 4 Bit pro Byte	max. 4 Bit pro Byte
Datenblocklängen (von ... bis)	0..beliebig	0..beliebig
Genauigkeit clock Übereinstimmung	Abhängig von UART (ca. 1%-3%)	Abhängig von UART (ca. 1%-3%)
Clock synchronisation	Erfolgt durch das Startbit	Erfolgt durch das Startbit
Error detection / correction	max. 1 Paritybit	max. 1 Paritybit
Sicherheit / Redundanz	-	-
Bitrate (von...bis)	110 bit/s .. ca. 230 kbit/s	110 bit/s .. 10 Megabit/s
Buslänge (von...bis)		
Anzahl Nodes Identifier	-	Implementationsabhängig
Anzahl Nodes Physikalisch	2	32
Hardware verfügbar?	fast alle µC, RS232-Controller 16550	fast alle µC, RS232-Controller 16550, Tranceiver von Texas, Linear Technology usw.
EMV-Aspekte	Flanken werden in der Regel nicht kontrolliert und können eine Störquelle darstellen	
Wake-Up?	Möglich	Möglich
Lizenzgebühr	Nein	Nein
Bewertung: Kosten für Master / Slave	ca. 2$ / 2-IO-Pins des µC	ca 2$ / 3 IO-Pins des µC
Bewertung: Zukunftsaussichten (Anwendungsgebiet)	Wird vorraussichtlich noch viele Jahre eingesetzt werden.	Wird als günstige Alternative in der Industrie zu CAN eingesetzt

SPI / I2C

Eigenschaft / Bussystem	SPI	I^2C
Applikation: Automobil?	Steuerung von IO-Chips auf Modulen (Eeprom, IO-Expander, Motioncontroller TMC428)	z.B. Steuerung von Receiver-Modulen und RDS-Decoder, aber auch Steuerung von Schrittmotorsteuerungen,
Applikation: Home?		
Applikation: Industrie?		
Standard?	-	Philips „THE I^2C-Bus SPECIFICATION Version 2.1“
Website für Standard		www.philips-semiconductors.com
Wer steht dahinter? (Organisation)		Philips
Medium (phys. Layer)	4 Draht-Kommunikation	2-Draht-Kommunikation
Encoding	NRZ	NRZ
Synchron, Asynchron?	Synchron	Synchron
Media Access, Arbitration, multi Master fähig?	Nein	Ja
Priorisierung von Transfers möglich?	Nein	Ja
Echtzeitfähig: Zeit für MS-Datentransfer (Read / Write)	1 SPI-CLOCK für jedes zu übertragende Bit + Zeit für Start und Stop	1 I^2C-CLOCK für jedes zu übertragende Bit + 1 Ack-Bit für jedes Byte+Zeit für Start und Stop bei Single-Masterbetrieb.Im Multimasterbetrieb ist die Arbitrierung des Busses ausschlaggebend.
Overhead pro Datenpaket (Bit)	ca. 2 Bitzeiten für Start und Stop.	bis zu 2 Adressbytes und 1 Ack.-Bit für jedes Byte
Datenblocklängen (von ... bis)	0 ... beliebig	0..beliebig
Genauigkeit clock Übereinstimmung	nicht erforderlich	nicht erforderlich
Clock synchronisation	-	-
Error detection / correction	-	-
Sicherheit / Redundanz	-	-
Bitrate (von...bis)	1 bit/s..ca. 5 Megabit/s	1 bit/s .. 3.4 Megabit/s
Buslänge (von...bis)	Ist abhängig von der Geschwindigkeit und der notwendigen Flankensteilheit.	Die Kapazität der Leitungen gegenüber GND sollte 400pF nicht übersteigen. Wenn HighSpeed-I^2C gefordert ist, dann entsprechend weniger.
Anzahl Nodes Identifier	Ist weitgehend abhängig von der zur Verfügung stehenden Anzahl der Chip-Select-leitungen	Ist abhängig vom Adressraum (max. 1024 Slaves)
Anzahl Nodes Physikalisch		Ist begrenzt von der Leitungskapazität
Hardware verfügbar?	µC, EEprom, IO-Expander	Siehe TMC453, Philips, Atmel µC
EMV-Aspekte	Flanken werden in der Regel nicht kontrolliert und können eine Störquelle darstellen	Durch Abstimmen von Geschwindigkeit und den Stromquellen, lassen sich die Flanken gut kontrollieren. Daher sind wenig EMV-Probleme zu erwarten.
Wake-Up?	Möglich	Möglich
Lizenzgebühr	Nein	Ja, für die Chiphersteller
Bewertung: Kosten für Master / Slave	3 IO-Pins des Microcontrollers für den Bus und für jeden Slave eine CS-Leitung	2 IO-Pins des Microcontrollers
Bewertung: Zukunftsaussichten (Anwendungsgebiet)	Dieses Protokoll ist so einfach, das ein Ersatz nicht zu sehen ist.	Wird weiterhin in der Unterhaltungselektronik massiv eingesetzt.

High-Speed / Real-Time

TTP

Eigenschaft / Bussystem	TTP/C	TTP/A
Applikation: Automobil?	X-by-Wire, Automobil, Flugzeug und Eisenbahn	Motor, Body
Applikation: Home?	-	
Applikation: Industrie?	-	Maschinensteuerung
Standard?	TTP/C	TTP/A
Website für Standard	http://www.ttpforum.org	http://www.ttpforum. org
Wer steht dahinter? (Organisation)	TTTech, TU-Wien, VW, Audi, OKI, NEC, u.a.	TTTech, TU-Wien, VW, Audi, OKI, NEC, u.a.
Medium (phys. Layer)	2-draht, twisted pair copper and fibre (aber 2 redundante Kanäle, um Sicherheit zu erhöhen) auch Star-Connection für noch mehr Sicherheit	1-draht
Encoding	MFM(Modified frequency modulation), NRZ	NRZ
Synchron, Asynchron?	synchron und asynchron	asynchron
Media Access, Arbitration, multi Master fähig?	TDMA (Time Division Multiple Access), Arbitration nur zum Starten des Netzwerks und hinzufügen von Knoten	ein Master, aber zweiter Shadow Master möglich, der bei Ausfall des Masters eingreift Polling
Priorisierung von Transfers möglich?	Ja, im asynchronen Betrieb	Ja
Echtzeitfähig: Zeit für MS-Datentransfer (Read / Write)	hard real time	soft real time
Overhead pro Datenpaket (Bit)	4 Bit Header 2 bzw. 3 Byte CRC	Startbit, Stopbit, Paritybit + 2 Bit IBG (Inter Byte Gap) – Datenbyte: Overhead : 5 bit
Datenblocklängen (von ... bis)	1 bis 16 Byte, nächste Generation: 1 – 240 Byte	1 Byte
Genauigkeit clock Übereinstimmung	mit aktuellen Halbleitern: besser 500 ns	Abweichung kleiner 100 ppm im Master, Slaves benötigen nur Oszillator mit geringer Präzision
Clock synchronisation	Clock Synchronisation findet durch den Membership-Service statt	alle Slave Knoten werden vom Master synchronisiert
Error detection / correction	detection, 16 Bit CRC Sender und Empfänger	detection, parity bit
Sicherheit / Redundanz	2 Kanäle, um Redundanz zu gewährleisten, Watchdog, Bus Guardian (Fail Silence), Fault tolerant, Membership Service	
Bitrate (von...bis)	bis 2 Mbit/s, next generation: asynchron: 5Mbit/s synchron 25 Mbit/s	20 kbit/s
Buslänge (von...bis)	Buslängen >20m sind kein Problem	
Anzahl Nodes Identifier	64	256, inkl. master
Anzahl Nodes Physikalisch	typisch: 4 bis 40	
Hardware verfügbar?	TTP/C-C1Communication	Standardmicrocontroller,

	Controller AS8201	Slave Nodes : Low Cost Controller Master Node: z.B. Motorola 68376
EMV-Aspekte		
Wake-Up?	Nein, da alle Knoten ständig wach sein müssen	möglich
Lizenzgebühr	Nein	Nein
Bewertung: Kosten für Master / Slave	?	TTP/A: 2$
Bewertung: Zukunftsaussichten (Anwendungsgebiet)	Für X-by-Wire recht gute Aussichten, steht jedoch in Konkurrenz zu Flexray	doppelt so teuer wie LIN, aber nur halb so teuer wie CAN, mit Sicherheit liegt es auch zwischen LIN und CAN

FlexRay

Eigenschaft / Bussystem	*13. FlexRay*
Applikation: Automobil?	X-by-Wire, Automobile, Flugzeug
Applikation: Home?	-
Applikation: Industrie?	-
Standard	TDMA (Flexible Time Devision Multiple Access Verfahren)
Website für Standard	www.flexraygroup.com
Wer steht dahinter? (Organisation)	BMW, DaimlerChrysler, Motorola,Phillips, BOSCH
Medium (phys. Layer)	2-draht (aber 2 Busse),
Encoding	MFM
Synchron, Asynchron	Synchron(wichtige Nachrichten)/asynchron(weniger wichtige Nachrichten)
Media Access, Arbitration, multi Master fähig?	TDMA, FTDMA
Priorisierung von Transfers möglich	Ja
Echtzeitfähig: Zeit für MS-Datentransfer (Read / Write)	Ja
Overhead pro Datenpaket (Bit)	40 Bit Header, 24 Bit CRC,
Datenblocklängen (von ... bis)	1 bis 246 Bytes
Genauigkeit clock Übereinstimmung	<=50ns
Clock synchronisation	Vom Master in 100ns Abstand
Error detection / correction	Detektion via CRC
Sicherheit / Redundanz	
Bitrate (von...bis)	10MBit
Buslänge (von...bis)	-
Anzahl Nodes Identifier	64
Anzahl Nodes Physikalisch	
Hardware verfügbar?	Bisher nicht, erste Muster 2003
EMV-Aspekte	
Wake-Up?	Wecken über den Datenbus
Lizenzgebühr	-
Bewertung: Kosten für Master / Slave	-
Bewertung: Zukunftsaussichten (Anwendungsgebiet)	

Byteflight

Eigenschaft / Bussystem	**Byteflight**
Applikation: Automobil?	Sicherheitssysteme in Automotive/Aerospace
Applikation: Home?	-
Applikation: Industrie?	-
Standard	FTDMA (Flexible Time Devision Multiple Access Verfahren)
Website für Standard	www.byteflight.com
Wer steht dahinter? (Organisation)	BMW, Motorola, SiemensAG und ELMOS AG
Medium (phys. Layer)	Sterntopologie mit bidirektionaler Kommunikation via POF (plastic optical fiber) , optischer Transiver
Encoding	NRZ mit start/stop bits
Synchron, Asynchron	Synchron(wichtige Nachrichten)/asynchron(weniger wichtige Nachrichten)
Media Access, Arbitration, multi Master fähig?	Multimasterfähig, jeder Teilnehmer kann als Master konfiguriert werden.
Priorisierung von Transfers möglich	Ja für Alarmsignale
Echtzeitfähig: Zeit für MS-Datentransfer (Read / Write)	100ns bei einer Bitrate von 10Mbit/s
Overhead pro Datenpaket (Bit)	40Bit
Datenblocklängen (von ... bis)	Bis 12 Databytes
Genauigkeit clock Übereinstimmung	-
Clock synchronisation	Vom Master in 100ns Abstand
Error detection / correction	Detektion via CRC
Sicherheit / Redundanz	Sicherheit durch ***Sternenkoppler***, bei Ausfall des Masters kann ein best. Teilnehmer durch seinen uC konf. werden
Bitrate (von...bis)	10MBit
Buslänge (von...bis)	-
Anzahl Nodes Identifier	-
Anzahl Nodes Physikalisch	Abhängig von Phy. Layer, Existierende Sternkoppler unterstützen 22 Nodes
Hardware verfügbar?	**MC68HC912BD32** von Motorola; **E100.38** von ELMOS
EMV-Aspekte	Optische Verdrahtung, dadurch wenig Einfluss durch elektromagnetische Störungen.
Wake-Up?	Optisches Wecken über den Datenbus
Lizenzgebühr	-
Bewertung: Kosten für Master / Slave	-
Bewertung: Zukunftsaussichten (Anwendungsgebiet)	DaimlerChrysler und BMW erweitern den Bus für X-by-wire Anwendungen (Bremssystem, Lenkung), Anwendungen im Bereich Aerospace denkbar

Wireless

Bluetooth

Eigenschaft / Bussystem	Bluetooth
Applikation: Automobil?	Nicht sicherheitskritische Systeme, z. B. Multimedia, kabellose Diagnose in der Werkstatt, Schlüsselfernbedienung, Abrechnungssysteme (Parkhaus, Maut)
Applikation: Home?	Vernetzung von PC und Peripheriegeräten sowie Multimedia
Applikation: Industrie?	Büro: Vernetzung von PCs sowie Palms / Organizers, Kleinstnetzwerke („Personal Area Network“ (PAN))
Standard	Bluetooth 1.1
Website für Standard	http://www.bluetooth.com
Wer steht dahinter? (Organisation)	Bluetooth SIG (Special Interest Group)
Medium (phys. Layer)	Funk (2,4 GHz lizenzfreie Frequenz), automatischer Kanalwechsel bis zu 1600 Mal pro Sekunde (79 Kanäle)
Encoding	FEC (Forward Error Correction Code) Data Whitening (Vermeidung von Gleichspannungsanteilen)
Synchron, Asynchron	1 asynchroner Datenkanal und bis zu 3 synchrone Sprachkanäle
Media Access, Arbitration, multi Master fähig?	TDD (Time division duplex (abwechselndes Senden und Empfangen bei jedem Knoten)) Multi-Master-Fähigkeit: bedingt
Priorisierung von Transfers möglich?	Nein
Echtzeitfähig: Zeit für MS-Datentransfer (Read / Write)	Nicht echtzeitfähig, abhängig davon, wie viele andere Sender sich in Reichweite befinden (je mehr, desto langsamer, da dann eventuell mehrere Wiederholungen nötig)
Overhead pro Datenpaket (Bytes)	72 Bit Acces Code, 54 Bit Adreßfeld
Datenblocklängen (von ... bis)	Max. 2745 Bits (~343 Bytes)
Genauigkeit clock Übereinstimmung	-
Clock synchronisation	Vom Master vorgegebener Zähler
Error detection / correction	Fehlererkennung und –korrektur (Fehlererkennende Kodierung und CRC)
Sicherheit / Redundanz	Verschlüsselte Übertragung (Schlüssellänge 64-128 Bits) Automatische Fehlerkorrektur, Fehlererkennung und Wiederholung, ARQ (Automatic Repeat Request)
Bitrate (von...bis)	Asynchron (Daten): asymmetrisch 723,2 kBit/s + 57.6 kBit/s oder symmetrisch 433.9 kBit/s beide Richtungen Synchron (Sprache): 64 kBit/s beide Richtungen
Buslänge (von...bis)	Reichweite ca. 10m
Anzahl Nodes Identifier	48-Bit-Adressierung (32 Bit Company ID + 16 Bit beliebig)
Anzahl Nodes Physikalisch	Einige 10
Hardware verfügbar?	Chipsätze und Multichipmodule verfügbar
EMV-Aspekte	Funk 2,4 GHz; Code minimiert Gleichspannungsanteile
Wake-Up?	-
Lizenzgebühr	Ja (?), 7000$ bis 40000$ im Jahr (abhängig von Firmengröße) für Mitgliedschaft in Bluetooth-SIG zur Vergabe einer Company ID
Bewertung: Kosten für Master / Slave	<5$
Bewertung: Zukunftsaussichten (Anwendungsgebiet)	Bisher nur bei Vernetzung PC mit Peripherie (oder z. B. Notebook mit Mobiltelefon) verwendet, noch nicht weit verbreitet (Verbreitung steigend). Hauptsächlich als „Kabelersatz“ zur Verbindung nahe beieinander stehender Geräte entwickelt.