

Echtzeitkommunikation und Feldbusse

Michaela Urvat

Prozesssteuerung und Robotik – Prof. Polze



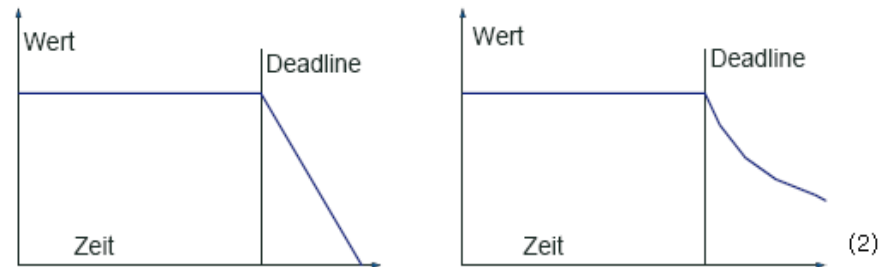
Echtzeitkommunikation - Einführung

2

„Echtzeitsysteme sind Computersysteme bei denen der Nutzen eines Resultates nicht nur vom Resultat abhängt, sondern auch vom Zeitpunkt der Auslieferung des Resultates.“ (2)

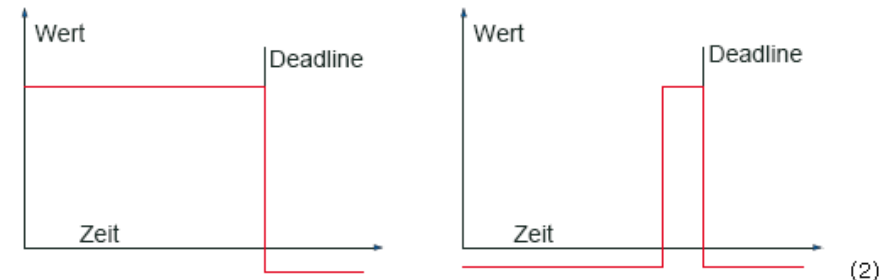
Weiche Echtzeit:

- gelegentliche Überschreitung der Deadline ist nicht fatal



harte Echtzeit:

- eine Überschreitung der Antwortzeit wird als Fehler gewertet

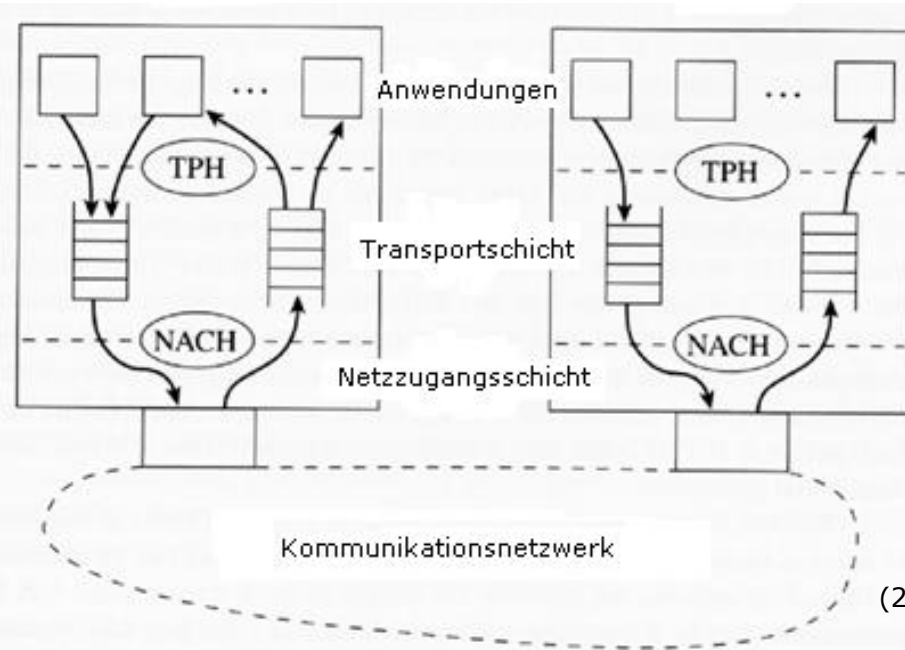


Ziel der Echtzeitkommunikation:

Vorhersagbarkeit von rechtzeitigen Ergebnissen maximieren und den Nachrichtenverlust zu minimieren

Modell für Echtzeitkommunikation

3 Zur Vereinfachung wird alles über der Transportschicht als Anwendung betrachtet.



TPH

(*Transport Protocol Handler*)

- Interface zu lokalen Anwendungen
- bietet einen Nachrichtentransport-Service

NACH

(*Network Access Control Handler*)

- Interface zum Netzwerk
- bietet TPH Netzwerkzugriff & Nachrichtenübertragungsservice

(2)

Echtzeitkommunikation:

- Queues für zu sendende und für empfangene Nachrichten
- Nachrichten werden zur Übertragung in Pakete einer definierten Größe zerlegt
- Übertragung eines Paketes ist nicht unterbrechbar

4 • **Periodische Nachrichten:**

- Übertragung einer periodischen Nachricht ist periodische Task
- Charakterisiert durch
 - Periode (p)
 - Dauer der Übertragung (e)
 - Deadline auf Empfängerseite (D)

• **Aperiodische Nachrichten:**

- Kein Wissen über Deadlines und Interarrival-Zeiten
- Ziel: Baldmögliche Auslieferung

• **Sporadische Nachrichten:**

- aperiodische Nachrichten mit Wissen über Interarrival-Zeiten und Deadlines
- charakterisiert durch:
 - durchschnittliche Interarrival-Zeit (p)
 - Länge des Zeitintervalls für Durchschnitt (l)
 - minimale Interarrival-Zeit (p)

5

- **Raten** (Nachrichtenverlust, verpasste Deadlines und beschädigte Nachrichten)
 - Anteil am Gesamtnachrichtenaufkommen
 - Anforderung: Einhaltung bestimmter Mindestwerte
- **Durchsatz**
 - Anzahl von Nachrichten, die pro Zeiteinheit transportiert werden können
 - Mindestanforderungen an den Durchsatz sind Basis von Scheduling und Flußsteuerungsalgorithmen
- **Delay** oder **End-zu-End-Zeiten**
 - Verzögerung einer Nachricht von sendender Task zu empfangender Task
 - harte Echtzeit essentiell, bei weicher Echtzeit irrelevant
- **Delay-Jitter**
 - Schwankungen der End-zu-End-Übertragungsdauer
 - Abhilfe gegen Jitter: große Puffer → erhöhen aber Delay

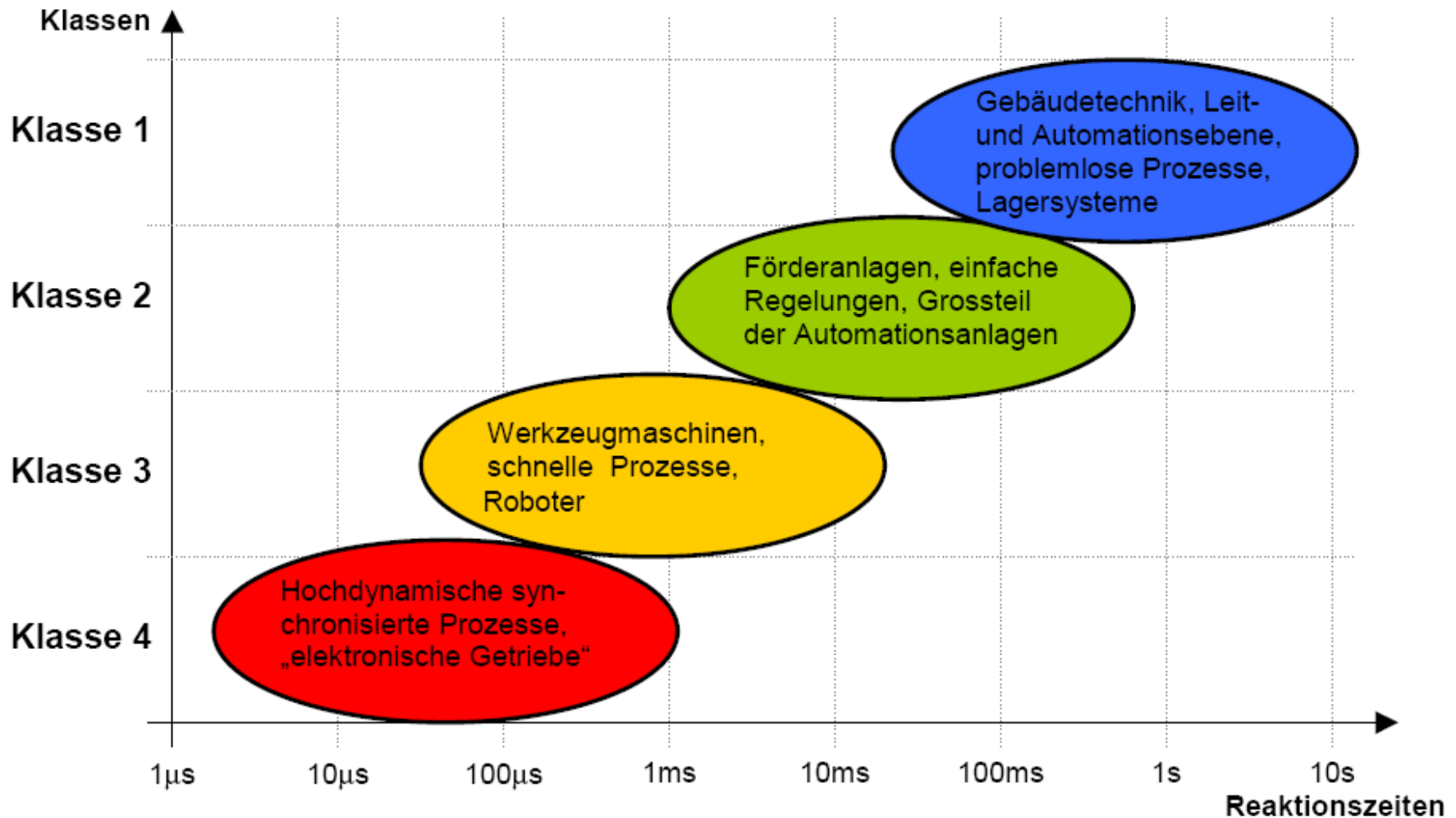
Netzwerktopologien & Routing

6

- Echtzeitkommunikationssysteme können in beliebigen Netzwerk-Topologien aufgebaut sein, wie z.B. Stern, Ring, Bus, eng vermaschtes Netz, Baum und auch in Mischformen
- Strukturen mit nicht direkter Verbindung zwischen Nodes erfordern Routing:
 - Packet Switching
 - Pakete einer Nachricht werden individuell zum Empfänger transportiert
→ einzelne Pakete nehmen unter Umständen andere Wege
 - Circuit Switching
 - Verbindung zwischen Sender und Empfänger wird geschaltet
→ ganzer Nachrichtenstrom nimmt diesen Weg
 - Wormhole Routing
 - Pakete werden zerlegt und suchen sich stückweise den Weg durch das Netz (nur Anfangsteil hat Zieladresse)
 - Vorteil: Erfordert weniger Pufferplatz

Echtzeitklassen

7



8

- Nutzung bereits vorhandener Technologien, wie z.B. Ethernet, ist billig

Protokoll muss erweitert werden, damit man die Vorhersagbarkeit der Kommunikation erhöhen kann.

Beispiele:

- **Ethernet POWERLINK** (Protokollerweiterung zum Ethernet-Standard)
- **ProfiNet** (offene Industrial Ethernet Standard)
- **SERCOS III** (SERCOS Interface Reihe auf der Basis von Standard-Ethernet)
- **Rtnet** (Echtzeitvariante des Ethernets von Token Bus, für Real-Time Linux)
- **EtherCAT** (ethernetbasierter Feldbus)
- **SafetyNET p** (ethernetbasierter Feldbus)

➔ Systeme erreichen alle Buszyklen von rund 100 μ s

Ethernet POWERLINK

9

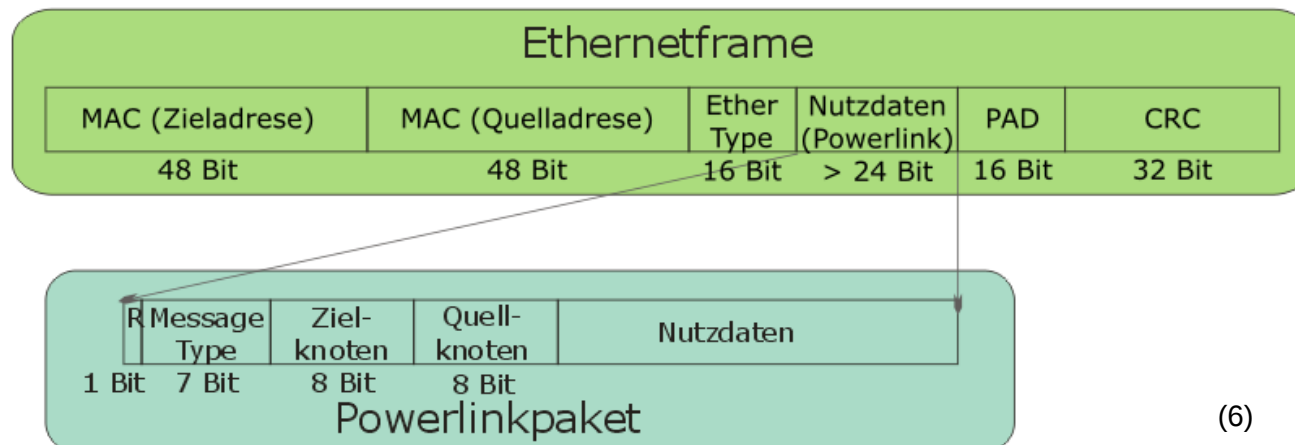
Protokollerweiterung zum Ethernet-Standard um Echtzeitdaten im Mikrosekundenbereich zu übertragen.

- Standardkonformität bei der Entwicklung
 - gemischter Polling- und Zeitscheibenmechanismus zur deterministischen Übertragung von Daten:
 - garantierte Übertragung von zeitkritischen Daten in sehr kurzen isochronen Zyklen mit konfigurierbarem Zeitverhalten
 - zeitliche Synchronisation aller Netzwerkknoten mit sehr hoher Präzision
 - Übertragung des weniger zeitkritischen Datenaufkommens im reservierten asynchronen Kanal
- ➔ aktuelle Implementierungen erreichen Zykluszeiten von unter 200µs und eine zeitliche Präzision (Jitter) von weniger als 1µs

- auf den Schichten 3 und 7 im OSI-Schichtenmodell angesiedelt
- ist grundsätzlich von der verwendeten Physik (Layer 1) unabhängig
- Repeating Hubs anstelle von Switches empfohlen

Datenformat:

- jedes Paket besteht aus einem Header und den eigentlichen Nutzdaten
- das Paket ist im normalen Ethernetframe eingebunden



(6)

Ethernet POWERLINK – Datenübertragung II

11

Message Type	ID	Bezeichnung der Nachricht	Verwendung	Ethernet Transfertyp
SoC	0x01	Start of Cycle	Definiert den Start eines neuen Zyklus	Multicast
PReq	0x03	PollRequest	Erfrage zyklische Daten des CN	Unicast
PRes	0x04	PollResponse	Sende aktuelle zyklische Daten des CN	Multicast
SoA	0x05	Start of Asynchronous	Signalisiere den Start der asynchronen Phase	Multicast
ASend	0x06	Asynchronous Send	Senden von asynchronen Daten	Multicast

(6)

- Datenübertragung durch *Managing Node (MN)* gesteuert
- *Controlled Nodes (CN, die Netzwerkteilnehmer)* dürfen nur dann senden, wenn sie dezidiert dazu aufgefordert wurden

Kommunikationsablauf:



(6)

- alle Kommunikations- und Anwenderobjekte werden in einem *Objektverzeichnis* (OV) zusammengefasst = Bindeglied zw. Anwendung & Kommunikationseinheit
 - jeder Eintrag im OV steht für ein Objekt (16 Bit Index)
 - pro Index können wiederum bis zu 256 Subindizes enthalten sein
 - 65536×254 Nutzeinträge pro Gerät
 - Zuordnung von Kommunikations- und Geräteprofilobjekten zu einem jeweiligen Index genau im *Profil* definiert → OV ist eindeutige Schnittstelle
- elektronische Datenblätter sind nötig (XDD-Dateien= XML Device Description)
 - wichtigsten Parameter der Objekte des OV eines Gerätes
 - weitere Parameter wie z.B. die unterstützten Kommunikationsdienste
 - Konfigurationstools können Datenblatt-Dateien einlesen und mit dem jeweiligen Gerät kommunizieren und es ggf. parametrisieren
- sicherheitskritische Anwendungen: Powerlink Safety Erweiterung
 - sicherheitskritische Daten werden in zwei Subframes gleichen Inhalts aufgeteilt und mit unterschiedlich berechneten Prüfsummen gesichert
 - Sicherheit stellt dabei eine eigene Sicherheitssteuerung her

Token-Ring - Timed-Token-Protokoll

13

Jedem Knoten wird Zeit für den Zugang zum Netz garantiert Übertragung von zyklischen Nachrichten (für harte Echtzeit)

- garantiert bestimmte Menge von synchroner Datenverkehr alle T Zeiteinheiten
- Bandbreite ist bekannt und garantiert

→ synchrone Nachrichtenübertragung = Echtzeit

Übertragung von SRT-oder NRT-Nachrichten darf den Zyklus nicht verlängern

- verwendet ungenutzte Bandbreite
- asynchrone Nachrichten = Nicht-Echtzeit

→ Ausschlaggebend ist target token-rotation time (TTRT)

- möglichst : Umlaufzeit $< \text{TTRT}$
- Aber nur Umlaufzeit $\leq 2 \text{ TTRT}$ garantiert
- Node kann bei jeder Tokenrunde einen festgelegten Anteil übermitteln

Timed-Token-Protokoll- Zyklus-Schema

14

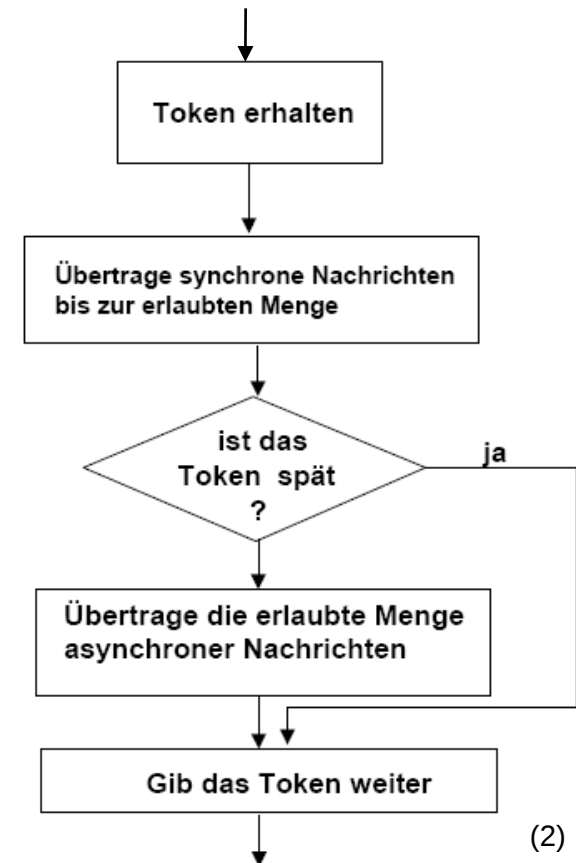
1. Zyklus:

Broadcast um TTRT zu bestimmen
→ kleinste angefragte Wert ist TTRT

- Node kann pro Umlauf nur so viele asynchrone Daten verschicken, wie kein Überlauf entsteht
 - Tokenverlust / Tokenfehler
 - Token in zwei aufeinanderfolgenden Umläufen zu spät ist → Anzeichen für Fehler
 - TTRT wird neu ausgehandelt
- Claim Token Protocol

Das Token ist:

- **früh**, wenn: $\text{Zyklus} \leq \text{TTRT}$
- **spät**, wenn: $\text{Zyklus} > \text{TTRT}$

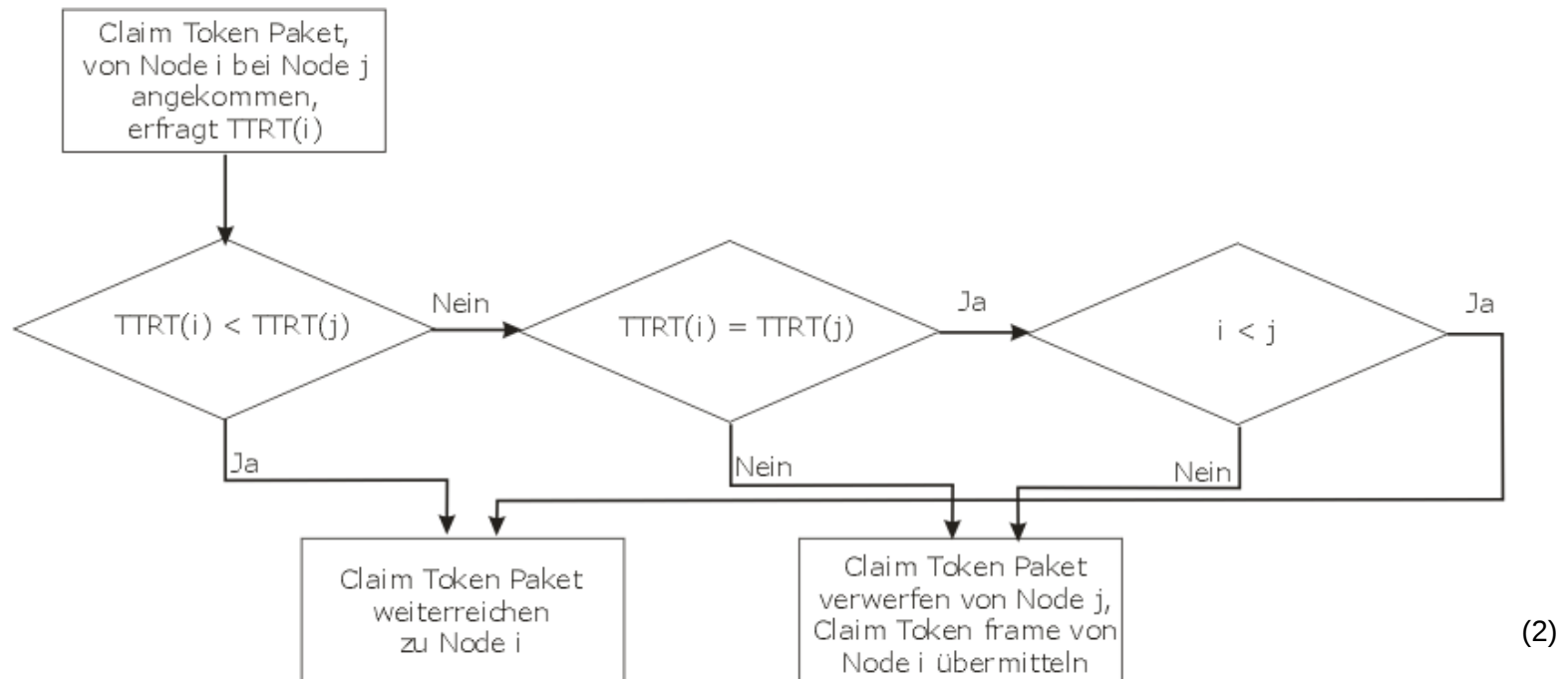


(2)

Ablauf TTP nach 2. Zyklus

Timed-Token-Protokoll – Claim Prozess

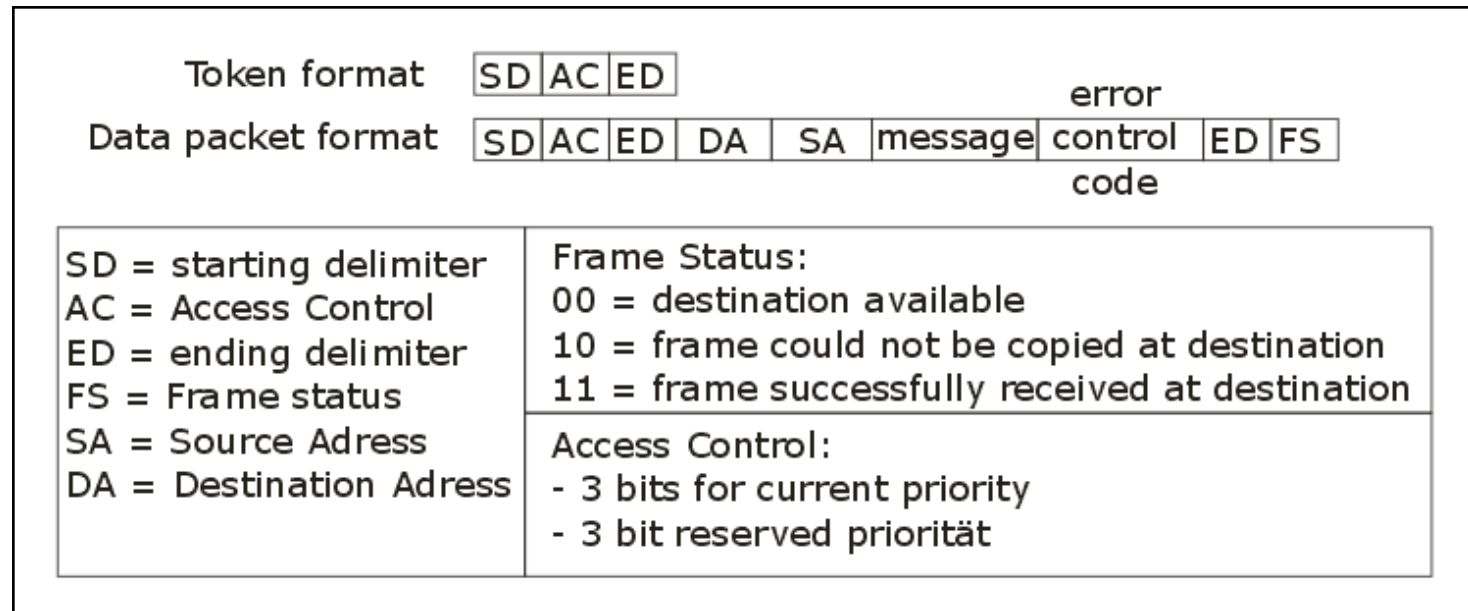
15



- zur Bestimmung der Target Token Rotation Time (TTRT)
- bei Erstinbetriebnahme des Rings oder das Einfügen von neuen Stationen
- MACs der angeschlossenen Stationen handeln die TTRT untereinander aus
- nach Initialisierungsprozess verfügt jede einzelne Station über operative TTRT → wird in Token Rotation Time (TRT) kopiert
- Zähler der regelmäßig dekrementiert wird

Timed-Token-Protokoll – Token & Datenpaket

16



(1)

- Node i überprüft Priorität (reservation bits) des Token
 - Priorität $\geq$ Priorität von Node i $\rightarrow$ nichts unternehmen
 - Priorität $<$ Priorität von Node i $\rightarrow$ Node i schreibt eigene Priorität in reservation bits
 - Übertragung komplett $\rightarrow$ Sender gibt Token mit neuer Priorität weiter

Feldbus

- entwickelt um 1980
- industrielles Kommunikationssystem für digitale Übertragungstechnik
- dienen der Ansteuerung von Maschinen
- viele Feldgeräte werden mit **einem** Steuerungsgerät verbunden
 - serielle Verdrahtung
 - ca. 40% der Kosten können eingespart werden
- der Zeit existieren ca. 50 unterschiedliche Feldbusse

Kriterien für Echtzeit:

- Übertragungsrate
- Zugriffsverfahren
- Zugriffsgarantie
- Störsicherheit
- Max. zulässige Teilnehmerzahl

Vor- und Nachteile von Feldbussen gegenüber paralleler Verdrahtung

18

+ Vorteile	- Nachteile
+ weniger Kabel, Rangierverteiler etc.	- komplexeres System
+ weniger Planungszeit bei Installation	- höherer Preis der Komponenten
+ verkürzte Ausfall- und Wartungszeiten	- aufwändige Messgeräte
+ bessere Zuverlässigkeit und Verfügbarkeit	- etwas längere Reaktionszeit
+ erhöhter Schutz vor Störungen	- Sensor-/Aktor-Hersteller sind gezwungen, mehrere Feldbusse zu unterstützen → zusätzliche Kosten
+ Komponenten verschiedener Hersteller sind hinsichtlich der Basiskommunikation leichter austauschbar	- zentrales Anbindungsprinzip kann bei einer Busstörung das Leitsystem von allen Sensoren und Aktoren abgeschnitten sein
+ Erweiterungen oder Änderungen sind einfach durchzuführen	

Feldbusse - Beispiele

CAN

- Motorsteuerung, Airbag, ABS, Fensterheber, Klima, ...

Time-Triggered Protokoll

- feldbusbasierendes Übertragungsprotokoll für fehlertolerante zeitgesteuerte Kommunikation

PROFIBUS

- Roboter, Maschinenbau, Anlagenbau → Automatisierungstechnik

Interbus-S

- zentrales Master-Slave-Zugriffsverfahren

ARCNET

- deterministischer, echtzeitfähiger Feldbus
- Automotive, Industrieautomatisierung und Medizintechnik

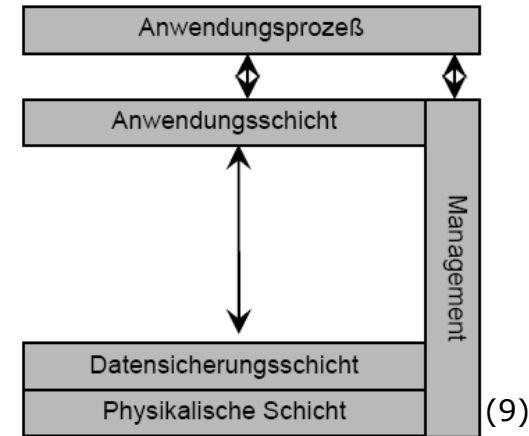
LON

- Beleuchtung, Heizung, Jalousiesteuerung, ...

CAN - Controller Area Network

20

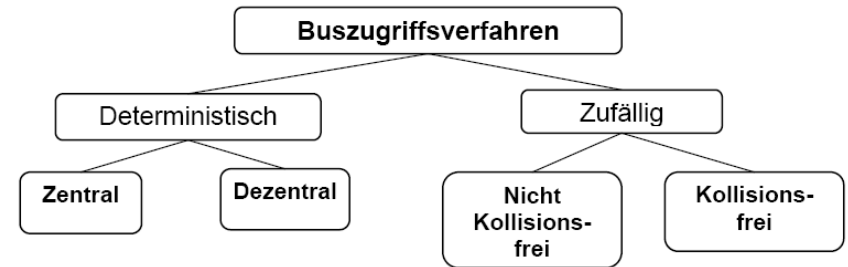
- OSI-Schichten 1, 2 und 7
- fortlaufenden Synchronisierung der Busteilnehmer mit
- Kollisionen bei Buszugriff durch die (Bit-)Arbitrierung aufgelöst
- zyklische Redundanzprüfung (CRC)
- Bitstopfen (*bit stuffing*) verwendet
- Producer-Consumer Prinzip
 - sendewillige Teilnehmer (Producer) sendet Nachricht
 - alle anderen Teilnehmer (Consumer) können Nachricht aufnehmen und verarbeiten.
- Netzwerk-Management für Initialisierung, Überwachung und Konfiguration von Knoten in standardisierter Form & schichtenbezogene Funktionen
- Bus entweder Kupferleitungen oder Glasfaser
- max. Datenübertragungsrate :
 - Highspeed-Bus = 1 Mbit/s
 - Lowspeed-Bus = 125 kbit/s



CAN - Eigenschaften

21

- Zugriffsteuerung erfolgt dezentral
- Kollisionsfreies Kommunikationsprotokoll: CSMA / CA
- nachrichtenorientiertes Verfahren
- Multi-Master-Bus, der nach dem Producer-Consumer Prinzip arbeitet
 - sendewillige Teilnehmer (Producer) sendet Nachricht über Bus
 - alle anderen Teilnehmer (Consumer) können Nachricht aufnehmen und verarbeiten.
- Versand von Nachrichten verwendet keine Zieladressen sondern eindeutigen Nachrichtenidentifizier zur Regelung des Busverkehrs
 - jeder Teilnehmer kann jederzeit Nachrichten versenden
 - damit Nachrichten sich nicht gegenseitig stören bzw. zerstören, Bit-Codierung und der Umgang mit Fehlern realisiert



(9)

CAN – Codierung

22

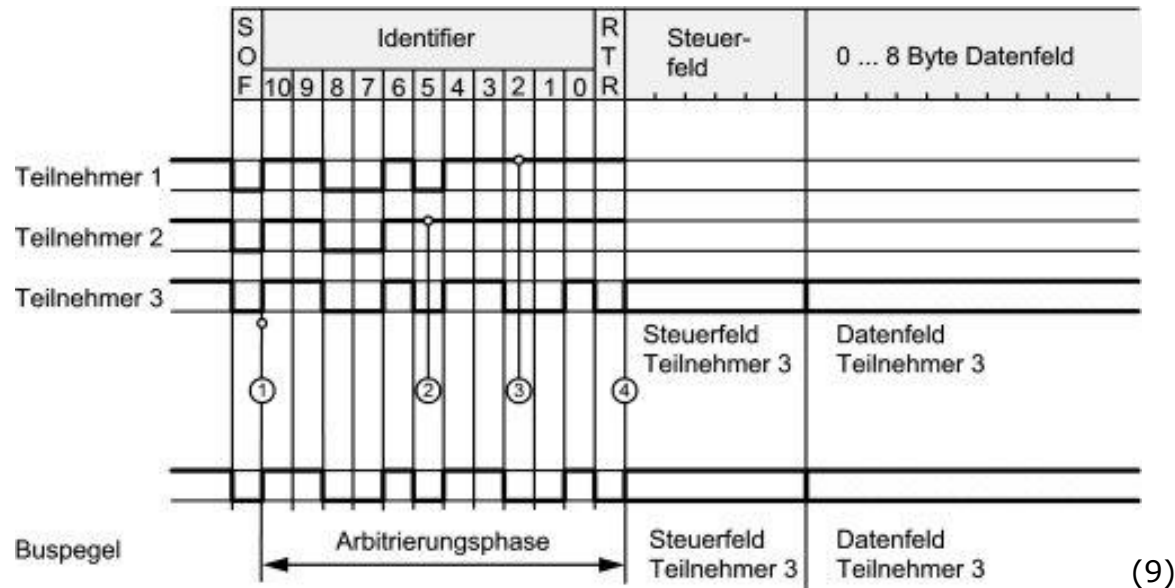
- es existieren zwei grundsätzliche Buspegel, ein rezessiver Pegel (1, rezessive Bit) und ein dominanter Pegel (0, dominante Bit)
 - rezessive Bit = nachgebender Pegel
 - niemand belegt Busleitung oder alle senden ein rezessives Bit
 - dominantes Bit = dominanter Pegel
 - ein Bit, ist je nach Zustand, entweder **dominant** oder **rezessiv** auf den Busleitungen → ein dominantes überschreibt dabei ein rezessives Bit
- über diese beiden Zustände werden Nachrichten codiert und versendet
- ➔ Grundlage der Arbitrierung um Zugriffskonflikt zu verhindern:
Teilnehmer, der die Nachricht mit der höchsten Priorität versenden will erhält Zugriff auf den Bus

23

Teilnehmer, der die Nachricht mit der höchsten Priorität versenden will erhält den Vorzug auf den Zugriff der Busleitung → Arbitrierung:

- Teilnehmer senden gleichzeitig ihren Nachrichtenidentifizier (11-stellige Binärzahl) und hören dabei den Pegel, der zeitgleich auf der Busleitung liegt, ab.
- Nachrichtenidentifizier wird während der Arbitrierung bitweise von links nach rechts übertragen
- Regel für die Teilnehmer gilt: Sendet man ein rezessives Bit (1) und empfängt zeitgleich einen dominanten Buspegel (0), muss das Senden sofort eingestellt werden und der Teilnehmer fungiert nur noch als Empfänger → Nachrichten mit der kleinsten Binärzahl erhalten den Vorzug zum Senden
- Nachrichten mit hoher Priorität erhalten niedrige Binärzahlen bzw. Identifizier
Nachrichten mit niedere Priorität erhalten hohe Binärzahlen bzw. Identifizier

CAN – Arbitrierung - Beispiel



1. Gleichzeitiger Start des Sendens der Identifier
2. Teilnehmer 2 sendet das Bit 5 des Identifiers als rezessives Bit → empfängt aber dominanten Pegel (Teilnehmer 2 beendet Sendevorgang)
3. Teilnehmer 1 sendet das Bit 2 des Identifiers als rezessives Bit → empfängt aber dominanten Pegel (Teilnehmer 1 beendet Sendevorgang)
4. Teilnehmer 3 hat niedrigste Binärzahl & erhält das Zugriffsrecht auf die Busleitung zum Versenden einer Nachricht

CAN – Fehlererkennung & -vermeidung

Fehlererkennung und Elimination für fünf **Fehlertypen**:

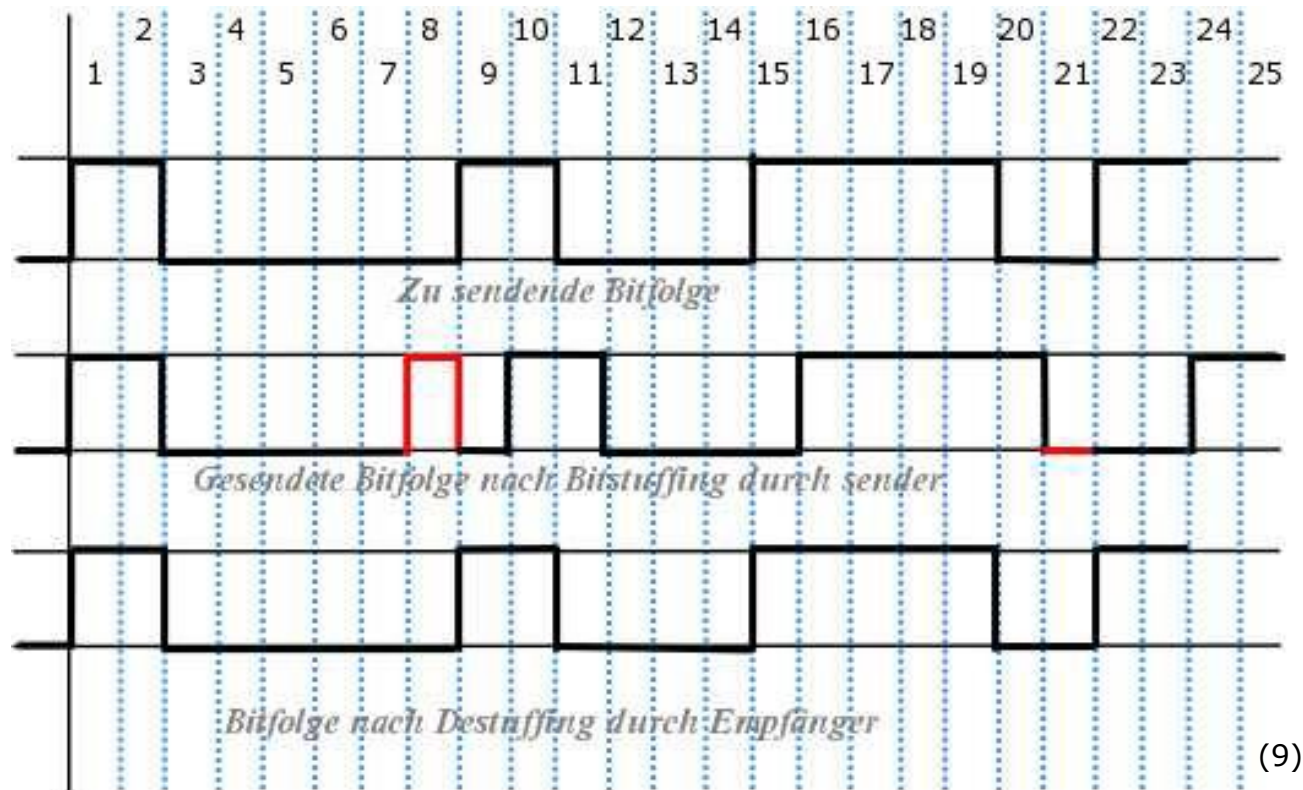
- Bit-Fehler
- Bit-Stuffing-Fehler
- CRC-Fehler (Cyclic-Reduncy-Check)
- Format-Fehler
- Acknowledgement-Fehler

Bitstuffing:

- präventiv den großen Nachteil der NRZ-Codierung verhindern soll:
bei langen gleichen Buspegeln → Synchronisationsverlust
- Maßnahme: nach fünf gleichen Bits wird ein komplementäres Bit in die Übertragung eingebaut → empfangende Netzknoten erkennt dies und filtert die Komplementärbits wieder heraus

CAN – Bitstopfen - Beispiel

26



- zur Fehlervermeidung durch zu lang anliegende Signale
- fünf mal das gleiche Signal
 - ein komplementäres Bit wird eingefügt

CAN – Anwendungsschicht (CAL)

27

- anwendungsunabhängige, objektorientierte Umgebung für die Implementierung von CAN-basierten verteilten Systemen
 - Anwendungsschicht = Schnittstelle zwischen der Datenübertragungsschicht und den Anwendungen
- zusätzliche Vereinbarungen über die Wirkung der ausgetauschten Nachrichten
- Notwendigkeit: Viele CAN-Anwendungen benötigen Dienste, die über der Datenübertragungsschicht liegen
- beinhaltet nur generelle Kommunikationsprozeduren
 - verteilte Anwendungen werden über standardisierte Kommunikationsobjekte beschrieben und entsprechende Protokolle für den Datenaustausch definiert
 - Kommunikationsobjekten werden CAN-Identifizier dynamisch zugeordnet

Feldbus – Time-Triggered Protocol I

28

- feldbusbasierendes Übertragungsprotokoll
- fehlertolerante zeitgesteuerte Kommunikation
- Zeitgesteuert = gemeinsame Uhrzeit als Steuersignale
- Uhren ständig synchronisieren durch Mittelwertbildung der Differenzen zwischen realer Ankunft der Nachricht und erwarteter Ankunft der Nachricht.
- Uhrensynchronisationsalgorithmus arbeitet dezentral und ist fehlertolerant gegenüber Einzelfehlern
- Ankunftszeiten werden a priori in der Message Descriptor List festgelegt
- Kommunikationsaufteilung = TDMA (Time Division Multiple Access)
 - jeder Teilnehmer hat die Möglichkeit für eine bestimmte Zeit seine Nachricht zu senden
 - TDMA-Runde zyklisch
 - jeder Teilnehmer hat einen lokalen Buswächter, der unabhängig vom Teilnehmer das Senden auf die Kanäle nur während der in der MEDL festgelegten Sendezeit zulässt

Time-Triggered Protocol II

29

- auch für sicherheitskritische Anwendungen geeignet zu sein: Membership-Service → jeder Knoten führt Liste der fehlerfrei angenommenen Knoten
- Membership-Liste stimmt nicht mit denen der anderen Knoten überein → dieser Knoten nimmt einen Fehler bei sich an und schaltet sich ab
- bei jeder Übertragung wird die Membership-Liste aktualisiert.
- Knoten sendet einen fehlerhaften Frame → wird aus der Membership ausgeschlossen.
- Cliques-Problem: zwei oder mehrere Gruppen, die sich untereinander als fehlerfrei ansehen und nur miteinander kommunizieren
→ Vermeidung: jeder Knoten prüft ob er Mitglied der größten 'Clique' ist und schaltet bei negativem Ergebnis in den passiven Modus.

Einsatzgebiete:

- Airbus A380: zur Steuerung des von Nord-Micro hergestellten Kabinendrucksystems eingesetzt
- Alcatel benutzt das Protokoll seit Juni 2002 als Feldbusprotokoll in der Bahnhofssignalsteueranlage ELEKTRA 2.

Quellen

30

- (1) Real-Time Systems,**
C.M. Krishna, K.G. Shin, McGraw-Hill, 1997
- (2) Echtzeitkommunikation,**
Dr.-Ing. Matthias Werner & Dipl. Inf. Jan Richling (HU Berlin)
<http://www2.informatik.hu-berlin.de/~mwerner/lec/emes/13-rtcomm.pdf>
- (3) Feldbusse,**
Dr.-Ing. Matthias Werner & Dipl. Inf. Jan Richling (HU Berlin)
<http://www2.informatik.hu-berlin.de/~mwerner/lec/emes/15-bus.pdf>
- (4) Telekommunikationstechnik,**
Prof. Dr.-Ing. Sven Zeisberg (HTW Dresden)
[http://www.htw-dresden.de/~s57820/Dokumente/Praktikum/T/T3/\(6\)%20Feldbusse.pdf](http://www.htw-dresden.de/~s57820/Dokumente/Praktikum/T/T3/(6)%20Feldbusse.pdf)
- (5) Feldbusse,**
<http://www.feldbusse.de/>
- (6) Wikipedia:** Ethernet Powerlink
http://de.wikipedia.org/wiki/Ethernet_Powerlink
- (7) Industrial Ethernet**
Prof. Hans Scheitlin (Zürcher Hochschule Winterthur)
https://home.zhaw.ch/~sln/Industrial_Ethernet/Echtzeit-Ethernet.pdf
- (8) Protokolle**
<http://www.informatik.uni-ulm.de/rs/projekte/core/rts1/networks%20and%20protocols.3.pdf>
- (9) Controller Area Network,**
Truong Kim Cuong, Ramin Hedayati, Sandro Butzki, Till Jaspersen