

Echtzeitsysteme

Einleitung

Peter Ulbrich

Lehrstuhl für Verteilte Systeme und Betriebssysteme

Friedrich-Alexander-Universität Erlangen-Nürnberg

https://www4.cs.fau.de/Lehre/WS18/V_EZS/

18. Oktober 2018



Das erste Echtzeitrechensystem

■ Whirlwind I

- **Zweck:** Flugsimulator
(Ausbildung von Bomberbesatzungen)
- **Auftraggeber:** U.S. Navy
- **Auftragnehmer:** MIT
- **Laufzeit:** 1945 – 1952



(Quelle: Alex Handy from Oakland, Nmibia)

■ Technische Daten

- Digitalrechner, bit-parallele Operationen
- 5000 Röhren, 11000 Halbleiterdioden
- magnetischer Kernspeicher
- Röhrenmonitore mit Lichtgriffel



Spätere Nutzung in **SAGE** durch die U.S. Air Force



- Der Nachfolger AN/FSQ-7 alias „Whirlwind II“:



(Quelle: Steve Jurvetson from Menlo Park, USA)

← SAGE Bedienstation

- Technische Daten

- Auftraggeber: U.S. Air Force
- Auftragnehmer: MIT, später IBM
- Bauweise: 55000 Röhren, 2000 m^2 , 275 t, 3 MW, 75 KIPS

- Betriebsdaten von SAGE:

- Installation: 22 - 23 Stationen im Zeitraum 1959 - 1963
- Betrieb: bis 1983 (Whirlwind I bis 1979)
- Kosten: 8–12 Milliarden \$ (1964) $\leadsto$ ca. 55 Milliarden \$ (2000)
- Nachfolger: u.a. AWACS



Moderne Echtzeitsysteme

Wo immer Rechensysteme mit ihrer physikalischen Umwelt interagieren ...



CAN CLASS B

- ① SAMSBF Fahrer
- ② SAMSBF Beifahrer
- ③ SAMSBF Heck 1
- ④ SAMSBF Heck 2
- ⑤ Sitzsteuergerät Fahrer
- ⑥ Sitzsteuergerät Beifahrer
- ⑦ Sitzsteuergerät hinten links
- ⑧ Sitzsteuergerät hinten rechts
- ⑨ Türsteuergerät vorne Fahrerseite
- ⑩ Türsteuergerät vorne Beifahrerseite
- ⑪ Türsteuergerät hinten Fahrerseite
- ⑫ Türsteuergerät hinten Beifahrerseite
- ⑬ Steuergerät Tennenwand
- ⑭ Dachkondemittel
- ⑮ Dachkondemittel (DSD)
- ⑯ Vorderes Seitenfeld (VSF)
- ⑰ Hinteres Seitenfeld (HSF)
- ⑱ Elektronisches Zündschloß (EZS)
- ⑲ Kombiinstrument
- ⑳ Motorisiermodul
- ㉑ Frontklimatisierung
- ㉒ Fondklimatisierung
- ㉓ Audiogateway

- ⑳ Parktronic (PTC)
- ㉑ Rollendruckkontrolle (RDK)
- ㉒ Pneumatische Steuereinheit (PSE)
- ㉓ Heckschleifkraftmessung/-führung
- ㉔ Zentrales Gateway
- ㉕ Airbag GG (Amada)
- ㉖ Multifunktionssteuergerät (MSF)
- ㉗ Servofix (Steuergerät)
- ㉘ Standheizung
- ㉙ Türzusicherung hinten Fahrerseite
- ㉚ Türzusicherung hinten Beifahrerseite

CAN CLASS C

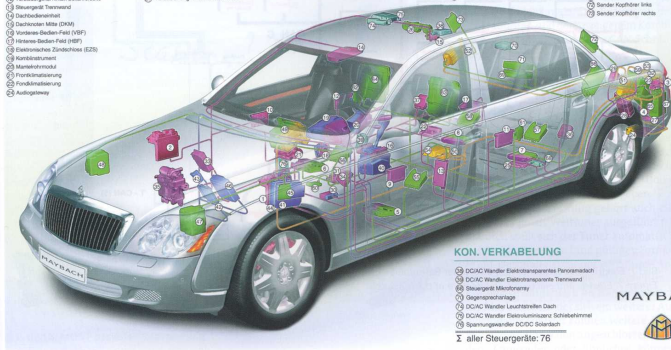
- ① Elektronisches Zündschloß (EZS)
- ② Kombiinstrument
- ③ Motorisiermodul
- ④ Zentrales Gateway
- ⑤ Elektronisches Wählhebelmodul
- ⑥ Lüftung (SLF)
- ⑦ Dreiecke (DTR)
- ⑧ Lichtwellenregulierung
- ⑨ Motorisiermodul (ME)
- ⑩ Sensorische Brake System (FSG)
- ⑪ Elektronische Getriebe-Steuerung

MOST-BUS

- ① Audiogateway
- ② Headunit
- ③ Steuerung Sprachbedienung
- ④ TV-Tuner MOST
- ⑤ Soundverstärker
- ⑥ Navigationsrechner
- ⑦ Kommunikationsplattform (CPT)

PRIVATE-BUS

- ① Sitzsteuergerät: Fahrer
- ② Sitzsteuergerät: Beifahrer
- ③ Sitzsteuergerät hinten links
- ④ Sitzsteuergerät hinten rechts
- ⑤ TV-Tuner CAN
- ⑥ Dachraumtast
- ⑦ Sensorische Brake System (FSG)
- ⑧ Sensorische Brake System (ASG 1)
- ⑨ Sensorische Brake System (ASG 2)
- ⑩ Multikonturlehne vorne links
- ⑪ Multikonturlehne vorne rechts
- ⑫ Multikonturlehne hinten links
- ⑬ Multikonturlehne hinten rechts
- ⑭ Keyless Go Heckmodul
- ⑮ Keyless Go Transurmodul
- ⑯ Keyless Go Tür hinten links
- ⑰ Keyless Go Tür hinten rechts
- ⑱ Fondklimatisierung links
- ⑲ Fondklimatisierung rechts
- ⑳ Kommunikationsplattform Fond (CPT)
- ㉑ Surround Amplifier
- ㉒ Audio Video Controller
- ㉓ CD-Wechsler
- ㉔ DVD-Spieler
- ㉕ Sender Kopfhörer links
- ㉖ Sender Kopfhörer rechts



(Quelle: DaimlerChrysler [1])



- 1 Historischer Bezug
 - Das erste Echtzeitrechensystem
 - SAGE – Der Nachfolger
 - Heutige Echtzeitsysteme
- 2 Echtzeitbetrieb
 - Definition
 - Realzeitbetrieb
 - Termine
 - Deterministische Ausführung
- 3 Aufbau und Abgrenzung
 - Struktur dieser Vorlesung
 - Abgrenzung
- 4 Zusammenfassung



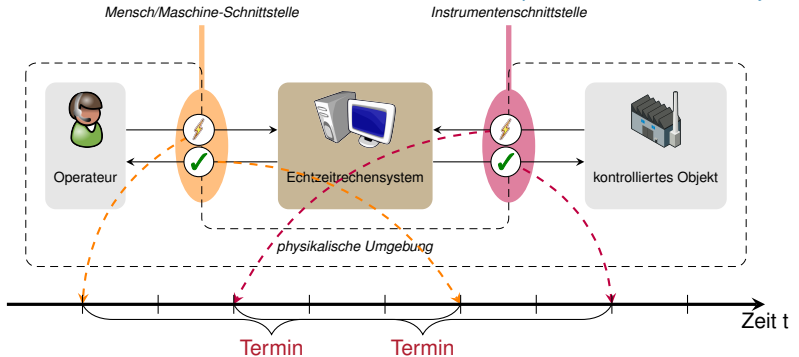
*Echtzeitbetrieb ist ein Betrieb eines Rechensystems, bei dem Programme zur Verarbeitung anfallender Daten ständig betriebsbereit sind derart, dass die **Verarbeitungsergebnisse innerhalb einer vorgegebenen Zeitspanne verfügbar sind.***

*Die Daten können je nach Anwendungsfall nach einer zeitlich **zufälligen Verteilung** oder zu **vorbestimmten Zeitpunkten** anfallen.*



Kopplung mit der (realen) Umwelt

Komponenten eines Echtzeitsystems



- Echtzeitsystem interagiert mit der **physikalischen Umwelt**
- Berechnet als Reaktion auf **Ereignisse** ⚡ (engl. *event*, Stimuli) der Umgebung **Ergebnisse** ✓ (engl. *result*)
- Zeitpunkt, zu dem ein Ergebnis vorliegen muss, wird als **Termin** oder **Frist** (engl. *deadline*) bezeichnet





Echtzeitbetrieb bedeutet **Rechtzeitigkeit**

- Funktionale Korrektheit reicht für korrektes Systemverhalten nicht aus
- **Rechtzeitige** Bereitstellung der Ergebnisse ist **entscheidend**



Den Rahmen stecken der **Eintrittspunkt** des Ereignisses und der entsprechende **Termin** ab



Termine hängen dabei von der Anwendung ab

wenige Mikrosekunden z.B. Drehzahl- und Stromregelung bei der Ansteuerung von Elektromotoren

einige Millisekunden z.B. Multimedia-Anwendungen (Übertragung von Ton- und Video)

Sekunden, Minuten, Stunden z.B. Prozessanlagen (Erhitzen von Wasser)





Geschwindigkeit ist keine Garantie für die rechtzeitige Bereitstellung von Ergebnissen

- **Asynchrone Programmunterbrechungen** (engl. *interrupts*) können **unvorhersagbare Laufzeitvarianzen** verursachen
- Schnelle Programmausführung ist bestenfalls hinreichend für die rechtzeitige Bearbeitung einer Aufgabe



Zeit ist keine intrinsische Eigenschaft des Rechensystems

- Die Zeitskala des Rechensystems muss nicht mit der durch die Umgebung vorgegebenen (Realzeit) übereinstimmen $\leadsto$ Zeitgeber?
- Temporale Eigenschaften des kontrollierten (physikalischen) Objekts müssen im Rechner system geeignet abgebildet werden





Konsequenzen überschrittener Termine

Verbindlichkeit von Terminvorgaben

- **Weich** (engl. *soft*) auch „schwach“
 - **Ergebnis verliert** mit zunehmender Terminüberschreitung **an Wert** (z.B. Bildrate bei Multimediasystemen)
→ Terminverletzung ist tolerierbar
- **Fest** (engl. *firm*) auch „stark“
 - **Ergebnis wird** durch eine Terminüberschreitung **wertlos** und wird verworfen (z.B. Abgabetermin einer Übungsaufgabe)
→ Terminverletzung ist tolerierbar, führt zum Arbeitsabbruch
- **Hart** (engl. *hard*) auch „strikt“
 - **Terminüberschreitung** kann zum **Systemversagen** führen und eine „Katastrophe“ hervorrufen (z.B. Airbag)
→ Terminverletzung ist keinesfalls tolerierbar





Arten von Echtzeitsystemen

Fest $\longleftrightarrow$ Hart

- **Fest/Hart** $\mapsto$ Terminverletzung ist nicht ausgeschlossen¹
 - Terminverletzung wird vom Betriebssystem erkannt
 - $\rightarrow$ Weiteres Vorgehen hängt von der Art des Termins ab

Fest $\leadsto$ plangemäß weiterarbeiten

- Betriebssystem bricht den Arbeitsauftrag ab
- Nächster Arbeitsauftrag wird (planmäßig) gestartet
- $\rightarrow$ Transparent für die Anwendung

hart $\leadsto$ sicheren Zustand finden

- Betriebssystem löst eine **Ausnahmesituation** aus
- Ausnahme ist **intransparent für die Anwendung**
- $\rightarrow$ **Anwendung** behandelt diese Ausnahme

¹ Auch wenn Ablaufplan und Betriebssystem auf dem Blatt Papier Determinismus zeigen, kann das im Feld eingesetzte technische System von unbekannten/unvermeidbaren Störeinflüssen betroffen sein!



■ Hard real-time computer system

(dt. Hartes Echtzeitrechensystem)

- Rechensystem mit mind. einem hartem Termin
- Garantiert unter allen (spezifizierten) Last- und Fehlerbedingungen
- Laufzeitverhalten ist ausnahmslos **deterministisch**
- Typisch für **sicherheitskritische Echtzeitrechensysteme**
 - engl. *safety-critical real-time computer system*
 - Beispiel: Fluglageregelung, Airbag, ...

■ Soft real-time computer system

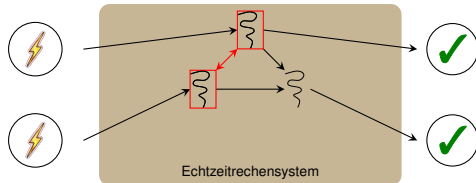
(dt. Weiches Echtzeitrechensystem)

- Rechensystem welches keinen harten Termin erreichen muss
- Termine können gelegentlich verpasst werden



Herausforderung: Gewährleisten von Rechtzeitigkeit

Ereignisbehandlungen müssen termingerecht abgearbeitet werden



- Ereignisse aktivieren **Ereignisbehandlungen**
 - Wie viel Zeit benötigt die Ereignisbehandlung **maximal**?
 - Lösung des trivialen Falls ist (scheinbar) einfach, wenn man die **maximale Ausführungszeit** der Ereignisbehandlung kennt.
- Reale Echtzeitsysteme sind **komplex**
 - Mehrere Ereignisbehandlungen $\leadsto$ Konkurrenz
 - Verwaltung gemeinsamer Betriebsmittel, allen voran die CPU.
 - Abhängigkeiten zwischen verschiedenen Ereignisbehandlungen



Determiniertheit

Bei identischen Eingaben sind verschiedene Abläufe zulässig, sie liefern jedoch stets das gleiche Resultat.



Im allgemeinen **unzureichend** für den Entwurf von Echtzeitsystemen



Transparenz von Programmunterbrechungen

- **Interrupts** verursachen vom normalen Ablauf abweichende **ausnahmebedingte Abläufe**

Determinismus

Identische Eingaben führen zu identischen Abläufen. Zu jedem Zeitpunkt ist bestimmt, wie weitergefahren wird.



Notwendig, falls Termine einzuhalten sind

- Nur so lässt sich das Laufzeitverhalten verlässlich abschätzen



Vorhersagbarkeit

Der Ablauf lässt sich zu jedem Zeitpunkt exakt angeben und hängt nicht von den aktuellen Eingaben oder vom aktuellen Zustand ab.



Vorteilhaft für zeitkritische Systeme

- Exakte Angaben zum zeitlichen Ablauf sind bereits à priori möglich
- Von Umgebung und Eingaben entkoppeltes Laufzeitverhalten
 - Aktivitäten folgen einem strikt vorgegebenem Stundenplan

Echtzeitsysteme müssen stets ein **deterministisches** oder besser **vorhersagbares** Laufzeitverhalten gewährleisten!

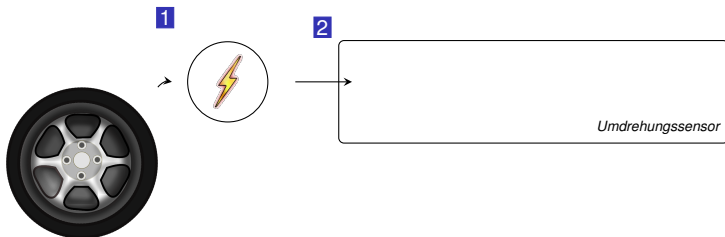
- Insbesondere beim **Zugriff auf gemeinsame Betriebsmittel**
 - CPU** → Umschaltung zwischen verschiedenen Aktivitäten
 - Kommunikationsmedium** → Versand von Nachrichten





Beispiel: Ein (fiktives) Anti-Blockier-System

Funktion eines verteilten Echtzeitrechnungssystems



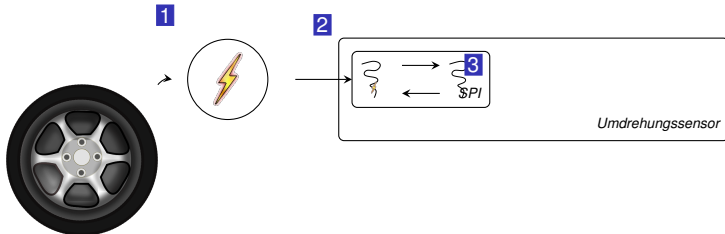
- ABS überwacht kontinuierlich Umdrehungszahl des Rads
→ Messfühler erzeugt Signale (Ereignisse)
- **Intelligenter Sensor** (engl. *smart sensor*) führt Vorverarbeitung der Daten durch (erkennt z.B. Stillstand)





Beispiel: Ein (fiktives) Anti-Blockier-System

Funktion eines verteilten Echtzeitrechnungssystems



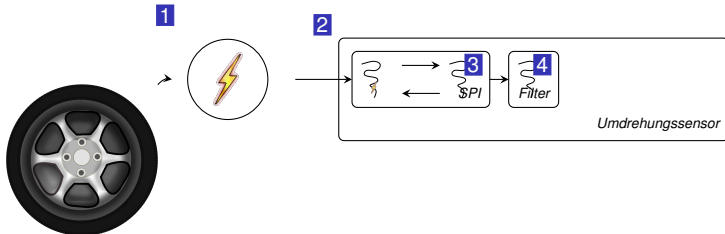
- Meßfühlerdaten werden über den SPI-Bus entgegengenommen
 - Buskommunikation erfordert eine ISR und einen Faden
 - Wann wird die ISR angesprungen? Sind Unterbrechungen gesperrt?
 - Wann wird der Faden eingeplant? Muss er auf Betriebsmittel warten?





Beispiel: Ein (fiktives) Anti-Blockier-System

Funktion eines verteilten Echtzeitrechnungssystems



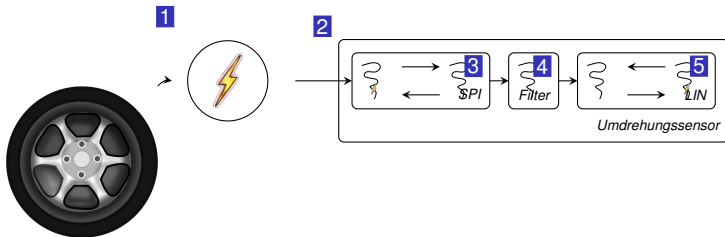
- Filter übernimmt die Signalvorverarbeitung
 - Angleichung diverser Abtastraten durch gesonderten Faden
 - der Filter verarbeitet immer mehrere Messwerte auf einmal
 - Wann wird der Faden eingeplant? Muss er auf Betriebsmittel warten?





Beispiel: Ein (fiktives) Anti-Blockier-System

Funktion eines verteilten Echtzeitrechnungssystems



■ Konsolidierte Messwerte werden an ABS-Steuergerät gesendet

■ Komplexer Gerätetreiber notwendig

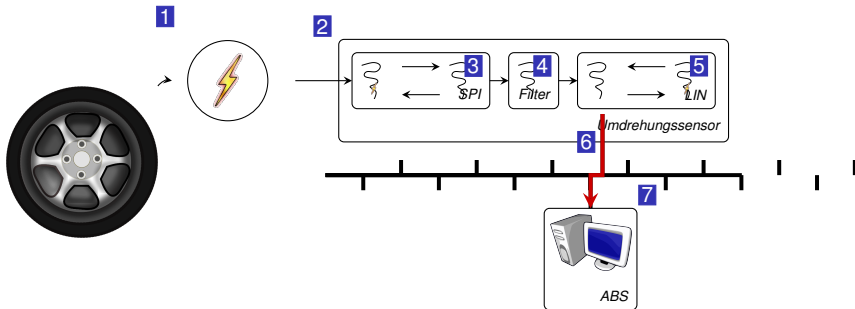
- Wann wird die ISR angesprochen? Sind Unterbrechungen gesperrt?
- Wann wird der Faden eingeplant? Muss er auf Betriebsmittel warten?
- Können alle Daten „auf einmal“ übertragen werden?





Beispiel: Ein (fiktives) Anti-Blockier-System

Funktion eines verteilten Echtzeitsystems



- Sensor und ABS-Steuergerät sind per LIN-Bus verbunden
 - Datenübertragung benötigt Zeit ...
 - Wie lange muss ich warten, bis ich auf das Medium zugreifen kann?



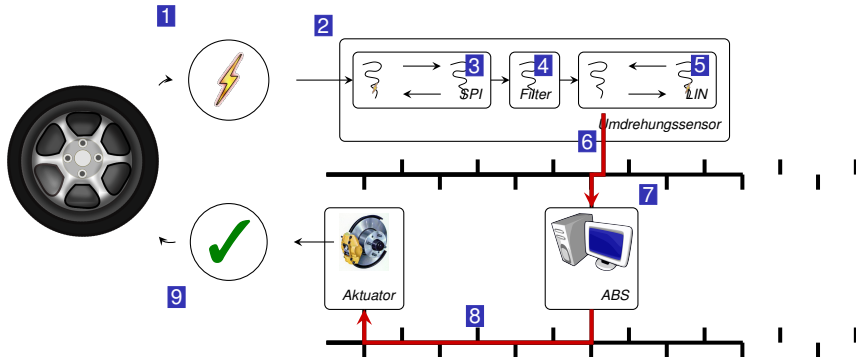
Vorgänge im ABS-Steuergerät sind noch deutlich komplexer





Beispiel: Ein (fiktives) Anti-Blockier-System

Funktion eines verteilten Echtzeitsystems



■ Stellwert wird dem Aktor zugestellt

- CAN-Bus verbindet ABS-Steuergerät und Aktor
 - Wieviele Bytes schafft der Bus in einer bestimmten Zeit?
 - Wie lange muss ich warten, bis ich auf das Medium zugreifen kann?

schließlich wird die Bremskraft geeignet beeinflusst





Die korrekte Funktion des ABS erfordert eine Reaktion auf eine Blockierung des Rades **innerhalb einer bestimmten Zeitspanne**

- Zu dieser Zeitspanne tragen zwei Komponenten bei:

Aktive Zeitintervalle $\leadsto$ „**Fortschritt**“ im ABS

- Berechnungen benötigen Zeit $\leadsto$ **maximale Ausführungszeit**
- Geschwindigkeit der Datenübertragung ist beschränkt

Inaktive Zeitintervalle $\leadsto$ „**Wartezeit**“ für das ABS

- Fortschritt erfordert die Zuteilung von Betriebsmitteln
- z. B. CPU oder Kommunikationsmedium



Die Frage ist, wie lange man auf die Zuteilung warten muss!

- **Determiniertheit** alleine reicht für die Beantwortung nicht aus!
- **Determinismus** erfordert die vollständige Kenntnis der Umgebung!
- **Vorhersagbarkeit** liefert die gewünschte Aussage zu dieser Frage!



Deterministische Abarbeitung von Ereignisbehandlungen?

- **Rein zyklisch** $\leadsto$ periodische Ereignisbehandlungen, Abfrage-Betrieb
 - (Nahezu) konstanter Betriebsmittelbedarf von Periode zu Periode
- **Meist zyklisch** $\leadsto$ überwiegend periodische Ereignisbehandlungen
 - System muss auf externe Ereignisse reagieren können
 - Betriebsmittelbedarf schwankt bedingt von Periode zu Periode
- **Asynchron/vorhersagbar** $\leadsto$ kaum periodische Ereignisbehandlungen
 - Aufeinanderfolgende Aktivierungen können zeitlich stark variieren
 - Zeitdifferenzen haben eine obere Grenze oder bekannte Statistik
 - Stark schwankender Betriebsmittelbedarf
- **Asynchron/nicht vorhersagbar** $\leadsto$ aperiodische Ereignisbehandlungen
 - Ausschließlich externe Ereignisse
 - Hohe, nicht deterministische Laufzeitkomplexität einzelner Ereignisbehandlungen



- 1 Historischer Bezug
 - Das erste Echtzeitrechensystem
 - SAGE – Der Nachfolger
 - Heutige Echtzeitsysteme
- 2 Echtzeitbetrieb
 - Definition
 - Realzeitbetrieb
 - Termine
 - Deterministische Ausführung
- 3 Aufbau und Abgrenzung
 - Struktur dieser Vorlesung
 - Abgrenzung
- 4 Zusammenfassung



Aufbau der Vorlesung

- Die Vorlesung orientiert sich vor allem ...
 - an der Ausprägung des Spezialzweckbetriebs
 - und den Eigenschaften der Ereignisse und ihrer Behandlungen,
 - blickt aber auch über den Tellerrand.

Einleitung

Grundlagen

vorranggesteuerte
Systeme

taktgesteuerte
Systeme

Analyse

periodische Echtzeitsysteme

nicht-periodische Echtzeitsysteme

Rangfolge

Zugriffskontrolle

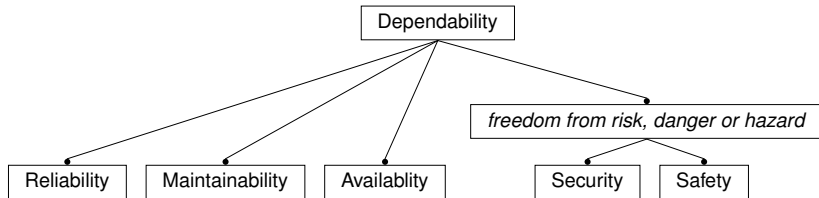
Aktuelle Forschungsthemen (Mehrkernrechnensysteme)

Aktuelle Forschungsthemen II / Industrievortrag (optional)

Zusammenfassung und Ausblick



Echtzeitsysteme sind häufig **sicherheitskritische Systeme** und erfordern ein hohes Maß an **Verlässlichkeit**. Verlässlichkeit selbst hat viele Gesichter ...



The trustworthiness of a computing system which allows reliance to be justifiably placed on the service it delivers. [3]



Verlässlichkeit **erfordert** Rechtzeitigkeit!

- Verpasste Termine stellen Fehler dar
- Diese Fehler müssen ggf. erkannt oder maskiert werden

■ **Andererseits:** Rechtzeitigkeit **erfordert** Verlässlichkeit!

- Fehler können zum Verpassen eines Termins führen
- Maskieren solcher Fehler hilft, die Rechtzeitigkeit zu gewährleisten

■ Betrachtung der Rechtzeitigkeit unter Annahme des *fehlerfreien Falls*

- Verletzte Termine werden auf einer höheren Ebene behandelt
- Toleranz gegenüber Fehlern dient der Verlässlichkeit



Das ist Thema der **Verlässlichen Echtzeitsystem** im SS

;-)



- 1 Historischer Bezug
 - Das erste Echtzeitrechensystem
 - SAGE – Der Nachfolger
 - Heutige Echtzeitsysteme
- 2 Echtzeitbetrieb
 - Definition
 - Realzeitbetrieb
 - Termine
 - Deterministische Ausführung
- 3 Aufbau und Abgrenzung
 - Struktur dieser Vorlesung
 - Abgrenzung
- 4 Zusammenfassung



- **Echtzeitbetrieb** eines Rechensystems in seiner Umgebung
 - Ereignis, Ereignisbehandlung, Ergebnis, Termin
- Komponenten eines Echtzeitsystems
 - Operateur, Echtzeitrechensystem, kontrolliertes Objekt
- **Weiche**, **feste** und **harte** Echtzeitbedingungen
- Determiniertheit, Determinismus, Vorhersagbarkeit
- Verhalten von Echtzeitanwendungen
 - Rein/meist zyklisch
 - Asynchron und irgendwie/nicht vorhersagbar
- **Abgrenzung**: Fokus dieser Vorlesung liegt auf der **Rechtzeitigkeit**



- [1] DaimlerChrysler AG:
Der neue Maybach.
In: *ATZ/MTZ Sonderheft* (2002), Sept., S. 125
- [2] Deutsches Institut für Normung:
DIN 44300: Informationsverarbeitung — Begriffe.
Berlin, Köln : Beuth-Verlag, 1985
- [3] IFIP:
Working Group 10.4 on Dependable Computing and Fault Tolerance.
<http://www.dependability.org/wg10.4>, 2003
- [4] Liu, J. W. S.:
Real-Time Systems.
Englewood Cliffs, NJ, USA : Prentice Hall PTR, 2000. –
ISBN 0-13-099651-3

