

Echtzeitsysteme

Einleitung

Peter Ulbrich

Lehrstuhl für Verteilte Systeme und Betriebssysteme
Friedrich-Alexander-Universität Erlangen-Nürnberg
<https://www4.cs.fau.de>

21. Oktober 2016



Das erste Echtzeitrechnungssystem

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



(Quelle: Alex Handy from Oakland, Nmbia)

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



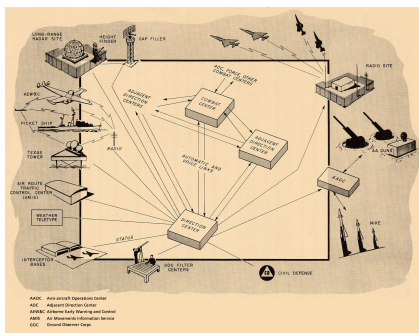
Spätere Nutzung in **SAGE** durch die U.S. Air Force



SAGE – Semi-AutomatGround Environment

Erstes verteiltes Echtzeitrechnungssystem als Schöpfung des „Kalten Krieges“

- Automatisiertes Kontroll- und Abwehrsystem gegen Bomber



- 27 Installationen
 - verteilt über die USA
 - *Nonstop*-Betrieb
 - 25 Jahre
- Kopplung durch Datenfernleitungen
 - Telefonleitungen
 - Internet-„Mutter“
- pro Installation...
 - 100 Konsolen
 - 500 KLOC Ass.



Entwicklung eines leistungsfähigeren Nachfolgers: Whirlwind II



AN/FSQ-7 Echtzeitrechnungssystem

- 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 m², 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) ~ ca. 55 Milliarden \$ (2000)
- **Nachfolger:** u.a. AWACS



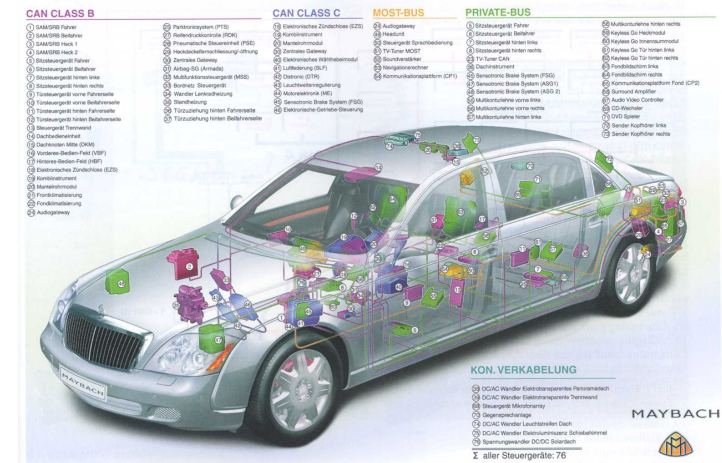
Moderne Echtzeitsysteme

Wo immer Rechensysteme mit ihrer physikalischen Umwelt interagieren ...



Spezialzwecksysteme (Forts.)

Verteiltes System auf Rädern



(Quelle: DaimlerChrysler [1])



Gliederung

- 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



DIN 44300

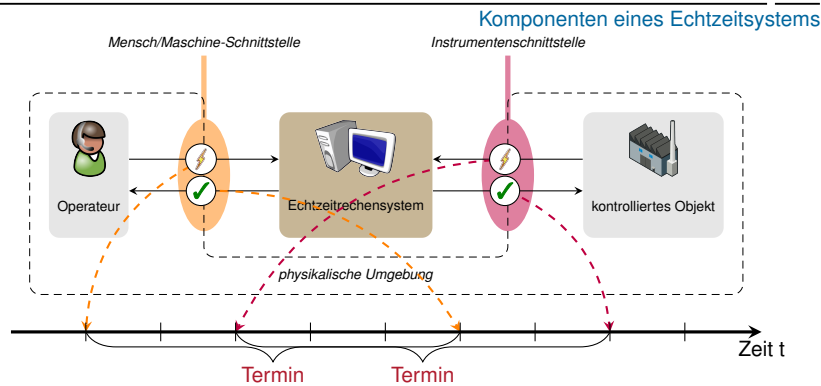
Ereignis- oder zeitgesteuerte Programmverarbeitung

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



- 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



Verarbeitung von Programmen in Echtzeit

Realzeitverarbeitung (engl. *real-time processing*)



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 impliziert nicht unbedingt Rechtzeitigkeit

Zuverlässige Reaktion des Rechensystems auf Umgebungsereignisse



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?
- \rightarrow Temporale Eigenschaften des kontrollierten (physikalischen) Objekts müssen im Rechensystem 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)
- \rightarrow Terminverletzung ist tolerierbar



Fest (engl. *firm*) auch „stark“

- **Ergebnis wird** durch eine Terminüberschreitung **wertlos** und wird verworfen (z.B. Abgabetermin einer Übungsaufgabe)
- \rightarrow Terminverletzung ist tolerierbar, führt zum Arbeitsabbruch



Hart (engl. *hard*) auch „strikt“

- **Terminüberschreitung** kann zum **Systemversagen** führen und eine „Katastrophe“ hervorrufen (z.B. Airbag)
- \rightarrow Terminverletzung ist keinesfalls tolerierbar



Arten von Echtzeitsystemen

Fest ↔ Hart

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

Fest → plangemäß weiterarbeiten

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

hart → sicheren Zustand finden

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

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



Arten von Echtzeitsystemen (Forts.)

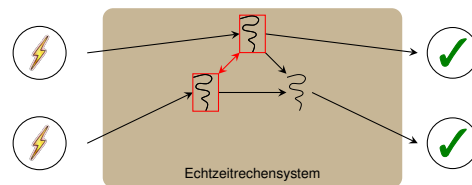
Radikale Unterschiede im Systementwurf zeichnen sich ab...

- **Hard real-time computer system**
(dt. Hartes Echtzeitrechnungssystem)
 - Rechen-system mit mind. einem hartem Termin
 - Garantiert unter allen (spezifizierten) Last- und Fehlerbedingungen
 - Laufzeitverhalten ist ausnahmslos **deterministisch**
 - Typisch für **sicherheitskritische Echtzeitrechnungssysteme**
 - engl. *safety-critical real-time computer system*
 - Beispiel: Fluglageregelung, Airbar, ...
- **Soft real-time computer system**
(dt. Weiches Echtzeitrechnungssystem)
 - Rechen-system, dass 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 → Konkurrenz
 - Verwaltung gemeinsamer Betriebsmittel, allen voran die CPU.
 - Abhängigkeiten zwischen verschiedenen Ereignisbehandlungen



Vorhersagbarkeit des Laufzeitverhaltens

Echtzeitsysteme sind (schwach, stark oder strikt) deterministisch

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 des Laufzeitverhaltens (Forts.)

Echtzeitsysteme sind (schwach, stark oder strikt) deterministisch

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 a priori möglich
- Von Umgebung und Eingaben entkoppeltes Laufzeitverhalten
→ Aktivitäten folgen einem strikt vorgegebenem Stundenplan

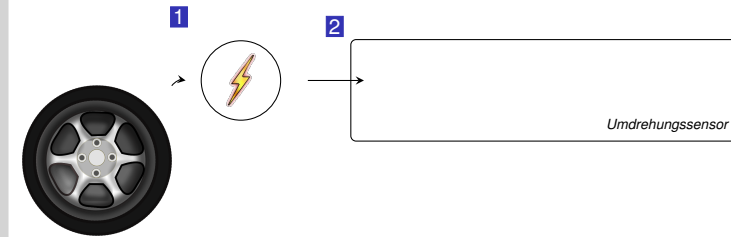
Echtzeitsysteme müssen stets ein deterministisches oder besser vorhersagbares Laufzeitverhalten gewährleisten!

- 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

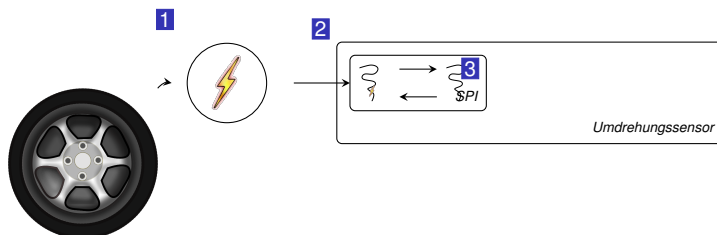


- ABS überwacht kontinuierlich Umdrehungszahl des Rads
→ Messfühler erzeugt Signale (Ereignisse)
- Intelligenter Sensor (engl. *smart sensor*) führt Vorverarbeitung der Daten durch (erkennt z.B. Stillstand)



Beispiel: Ein (fiktives) Anti-Blockier-System

Funktion eines verteilten Echtzeitrechnungssystems

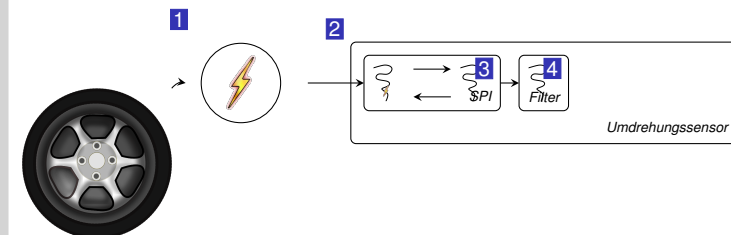


- Meßfühlerdaten werden über den SPI-Bus entgegengenommen
 - Buskommunikation erfordert eine ISR und einen Faden
 - Wann wird die ISR angesprungen? Sind Unterbrechungen gesperrt?
 - Wann wird der Faden eingeplant? Muss er auf Betriebsmittel warten?



Beispiel: Ein (fiktives) Anti-Blockier-System

Funktion eines verteilten Echtzeitrechnungssystems

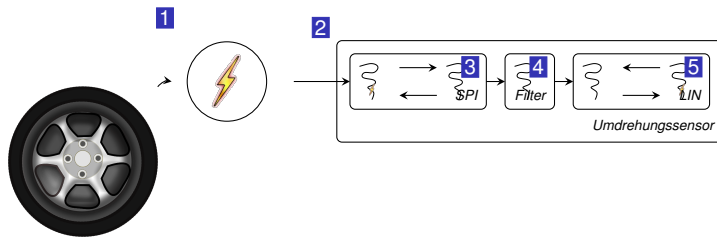


- Filter übernimmt die Signalvorverarbeitung
 - Angleichung diverser Abtastraten durch gesonderten Faden
 - der Filter verarbeitet immer mehrere Messwerte auf einmal
 - Wann wird der Faden eingeplant? Muss er auf Betriebsmittel warten?



Beispiel: Ein (fiktives) Anti-Blockier-System

Funktion eines verteilten Echtzeitrechnungssystems

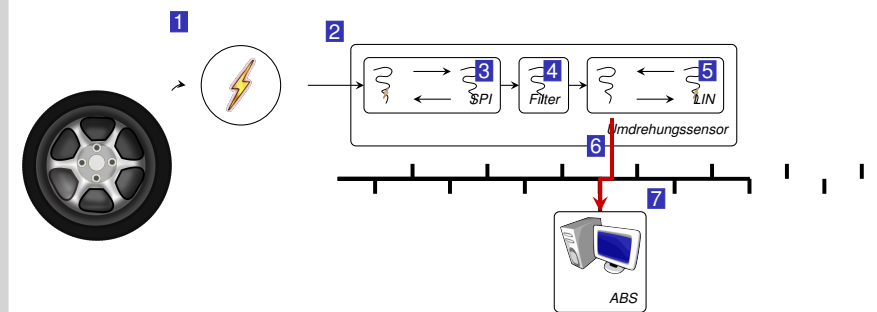


- Konsolidierte Messwerte werden an ABS-Steuergerät gesendet
 - Komplexer Gerätetreiber notwendig
 - Wann wird die ISR angesprungen? Sind Unterbrechungen gesperrt?
 - Wann wird der Faden eingeplant? Muss er auf Betriebsmittel warten?
 - Können alle Daten „auf einmal“ übertragen werden?



Beispiel: Ein (fiktives) Anti-Blockier-System

Funktion eines verteilten Echtzeitrechnungssystems

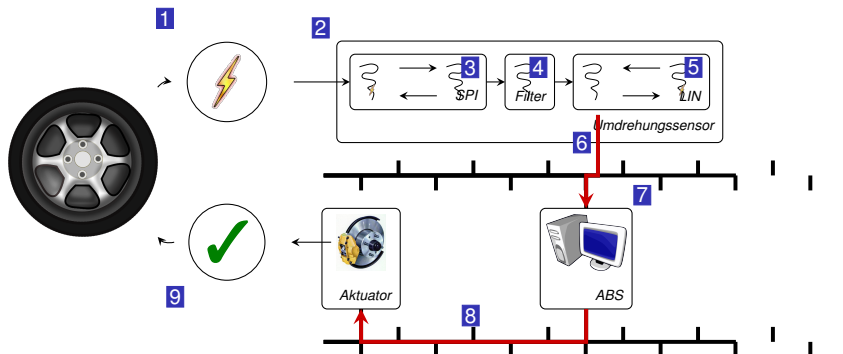


- Sensor und ABS-Steuergerät sind per LIN-Bus verbunden
 - Datenübertragung benötigt Zeit ...
 - Wie lange muss ich warten, bis ich auf das Medium zugreifen kann?
- ⚠ Vorgänge im ABS-Steuergerät sind noch deutlich komplexer



Beispiel: Ein (fiktives) Anti-Blockier-System

Funktion eines verteilten Echtzeitrechnungssystems



- 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 auf das Medium zugreifen kann?

schließlich wird die Bremskraft geeignet beeinflusst



Beispiel: Ein (fiktives) Anti-Blockier-System (Forts.)

Wie lange dauert das ganze nun?

- ⚠ 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** ~ „Fortschritt“ im ABS
 - Berechnungen benötigen Zeit ~ **maximale Ausführungszeit**
 - Geschwindigkeit der Datenübertragung ist beschränkt
 - Inaktive Zeitintervalle** ~ „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!



Charakterisierung von Echtzeitanwendungen [4, S. 25]

- ☞ 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



Gliederung

- 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

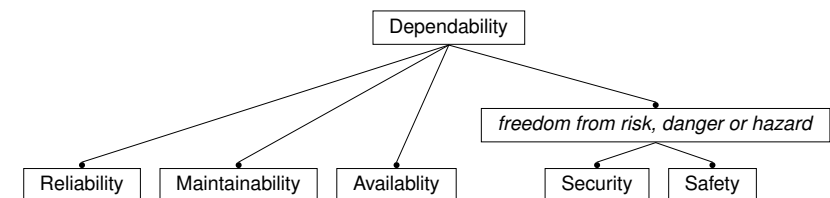
	vorranggesteuerte Systeme	taktgesteuerte Systeme	Analyse
periodische Echtzeitsysteme			
nicht-periodische Echtzeitsysteme			
Rangfolge			
Zugriffskontrolle			
Aktuelle Forschungsthemen (Mehrkernrechensysteme)			
Aktuelle Forschungsthemen II / Industrievortrag (optional)			
Zusammenfassung und Ausblick			



Abgrenzung

Verlässlichkeit (engl. *dependability*)

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]



Abgrenzung

Zusammenspiel von Rechtzeitigkeit und Verlässlichkeit

- ⚠ 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 ;-)



Gliederung

- 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



Resümee

- **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**



Literaturverzeichnis

- [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

