Echtzeitsysteme

Einleitung

Lehrstuhl Informatik 4

17. Oktober 2013

Gliederung

Whirlwind I

Das erste Echtzeitrechensystem

Zweck: Flugsimulator
(Ausbildung von Bomberbesatzungen)

Auftraggeber: U.S. Navy

Auftragnehmer: MIT

Laufzeit: 1945 – 1952

(Quelle: Alex Handy from Oakland, Nmibia)

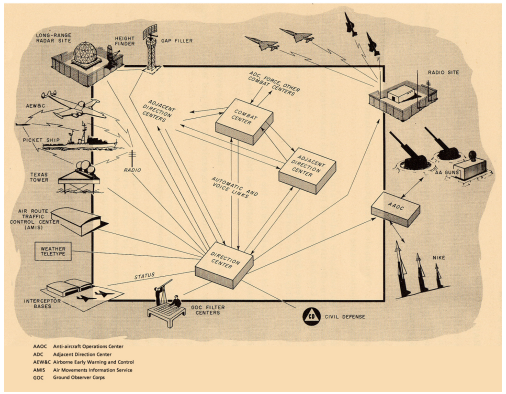
Bauweise: Digitalrechner, bit-parallele Operationen, 5000 Röhren, 11000 Halbleiterdioden, magnetischer Kernspeicher, Röhrenmonitore, Lichtgriffel

später: Nutzung durch die U.S. Air Force im SAGE

SAGE – **S**emi-**A**utomatic **G**round **E**nvironment

Erstes verteiltes Echtzeitrechensystem als Schöpfung des „Kalten Krieges“

Automatisiertes Kontroll- und Abwehrsystem für feindliche 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

SAGE und AN/FSQ-7 alias „Whirlwind II“

Daten AN/FSQ-7 alias „Whirlwind II“:

SAGE Bedienstation

(Quelle: Steve Jurvetson from Menlo Park, USA)

Auftraggeber: U.S. Air Force

Auftragnehmer: MIT, später IBM

Bauweise: 55000 Röhren, 2000 m^2, 275 Tonnen, 3 MW, 75 KIPS

Betriebsdaten SAGE:

Installation: 22 - 23 Stationen im Zeitraum 1959 - 1963

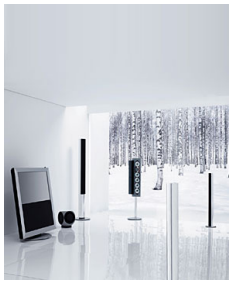
Betrieb: bis 1983 (Whirlwind I bis 1979)

Nachfolgesysteme: z.B. AWACS

Kosten: 8 – 12 Milliarden $ (1964), entspricht ca. 55 Milliarden $ (2000)

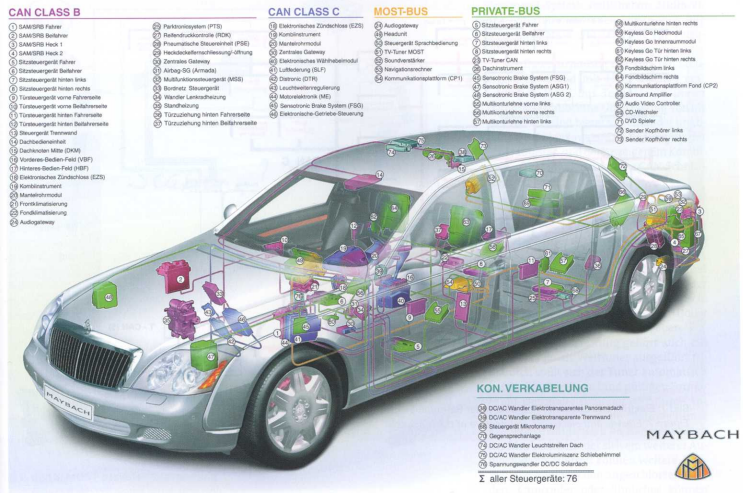
Moderne Echtzeitsysteme

Wo immer Rechensysteme mit ihrer physikalischen Umwelt interagieren ...



Spezialzwecksysteme (Forts.)

Verteiltes System auf Rädern



(Quelle: DaimlerChrysler [1])

Gliederung

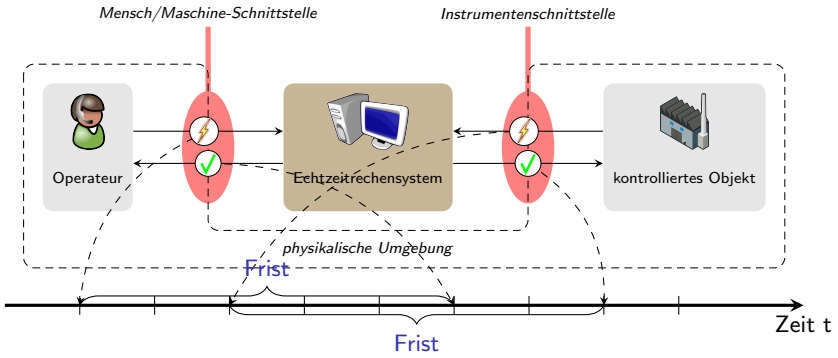
DIN 44300

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.*

Komponenten eines Echtzeitsystems

Echtzeitrechensystem und seine Umgebung



- das Echtzeitrechensystem berechnet als Reaktion auf Stimuli bzw. Ereignisse (engl. *event*) der Umgebung Ergebnisse
- der 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

- korrektes Systemverhalten hängt nicht nur von den logischen Ergebnissen der Berechnungen ab
- zusätzlicher Aspekt ist der physikalische Zeitpunkt der Erzeugung und Verwendung der Berechnungsergebnisse
- den Rahmen stecken der Eintrittspunkt des Ereignisses und die entsprechende Frist 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 Bildmaterial)
Sekunden, Minuten, Stunden	z.B. Prozessanlagen (Erhitzen von Wasser)

Geschwindigkeit impliziert nicht unbedingt Rechtzeitigkeit

Zuverlässige Reaktion des Rechensystems auf Umgebungsereignisse

Geschwindigkeit liefert keine Garantie, um rechtzeitig Ergebnisse von Berechnungen abliefern und Reaktionen darauf auslösen zu können

- 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 im Rechensystem verwendete Zeitskala muss nicht mit der durch die Umgebung vorgegebenen identisch sein
- die zeitlichen Gegebenheiten des kontrollierten Objekts müssen im Rechensystem geeignet abgebildet werden

Konsequenzen überschrittener Termine

Verbindlichkeit von Terminvorgaben

weich (engl. *soft*) auch „schwach“
- das Ergebnis verliert mit zunehmender Terminüberschreitung an Wert (z.B. Bildrate bei Multimediasystemen)
- Terminverletzung ist tolerierbar

fest (engl. *firm*) auch „stark“
- das 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“
- eine 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 ausgeschlossen[1]
- die Terminverletzung wird vom Betriebssystem erkannt

fest $\rightsquigarrow$ plangemäß weiterarbeiten
- das Betriebssystem bricht den Arbeitsauftrag ab
- der nächste Arbeitsauftrag wird gestartet
- ist transparent für die Anwendung

hart $\rightsquigarrow$ sicheren Zustand finden
- das Betriebssystem löst eine Ausnahmesituation aus
- die Ausnahme ist intransparent für die Anwendung
- die Anwendung behandelt diese Ausnahme

[1] Auch wenn Ablaufplan und Betriebssystem auf dem Blatt Papier Determinismus zeigen, kann das im Feld eingesetzte technische System von Störeinflüssen betroffen sein, die ggf. die Verletzung auch eines harten Termins nach sich ziehen.

Arten von Echtzeitsystemen (Forts.)

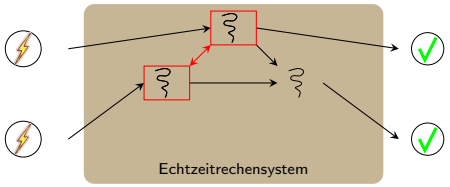
Radikale Unterschiede im Systementwurf zeichnen sich ab...

hard real-time computer system
- ein Rechensystem, das mind. einen strikten Termin erreichen muss
  - garantiert unter allen (spezifizierten) Last- und Fehlerbedingungen
  - das Laufzeitverhalten ist ausnahmslos deterministisch
- typisch für ein sicherheitskritisches Echtzeitrechensystem
  - engl. *safety-critical real-time computer system*

soft real-time computer system
- ein Rechensystem, das keinen strikten Termin erreichen muss
- es ist erlaubt, gelegentlich Termine zu verpassen

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 $\rightsquigarrow$ Konkurrenz
  - Verwaltung gemeinsamer Betriebsmittel, allen voran die CPU.
- Abhängigkeiten zwischen verschiedenen Ereignisbehandlungen

Vorhersagbarkeit des Laufzeitverhaltens

Echtzeitsysteme sind (schwach, stark oder strikt) deterministisch

Determiniertheit

Bei ein und derselben Eingabe sind verschiedene Abläufe zulässig, alle Abläufe liefern jedoch stets das gleiche Resultat.

- Transparenz von Programmunterbrechungen
  - *Interrupts* verursachen vom „normalen Ablauf“ verschiedene ausnahmebedingte Abläufe

☞ unzureichend, falls zeitliche Aspekte bedeutend sind

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.

- von Umgebung und Eingaben entkoppeltes Laufzeitverhalten
  - Aktivitäten folgen einem strikt vorgegebenem Stundenplan

☞ vorteilhaft für zeitkritische Systeme
- exakte Angaben zum zeitlichen Ablauf sind bereits à priori möglich

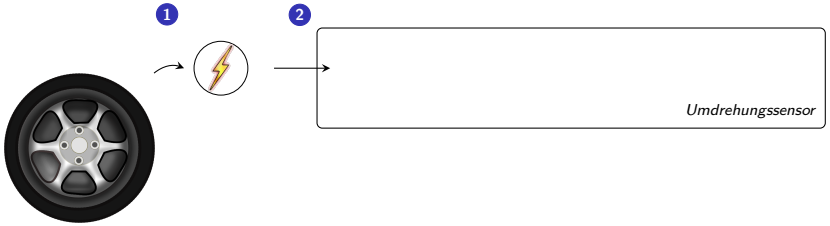
☞ Das Echtzeitsystem muss stets ein deterministisches oder besser vorhersagbares Laufzeitverhalten gewährleisten.
- insbesondere beim Zugriff auf gemeinsame Betriebsmittel

  CPU $\mapsto$ Umschaltung zwischen verschiedenen Aktivitäten

  Kommunikationsmedium $\mapsto$ Versand von Nachrichten

Beispiel: ein (fiktives) AB-System

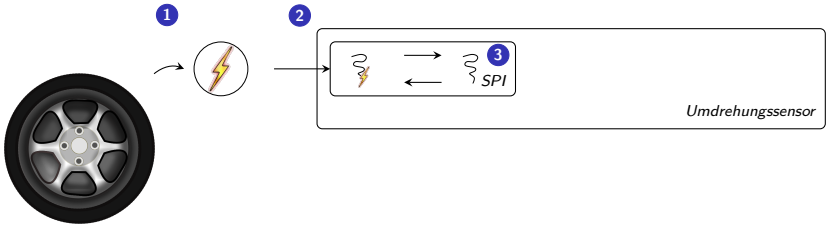
Ein Gemeinschaftserzeugnis eines verteilten Echtzeitrechensystems



- das ABS überwacht kontinuierlich Umdrehungszahl des Rads
  - so kann z. B. ein Blockieren des Rades erkannt werden
- in einem intelligenten Sensor (engl. *smart sensor*) findet zunächst eine Vorverarbeitung der erfassten Daten statt

Beispiel: ein (fiktives) AB-System

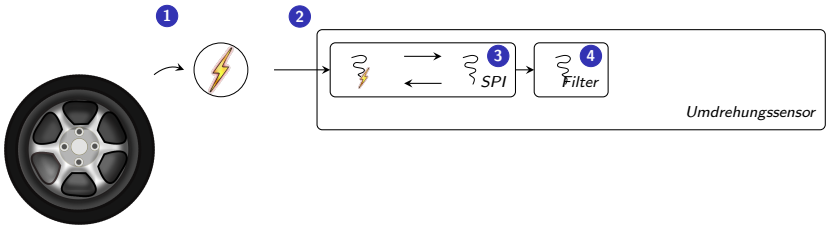
Ein Gemeinschaftserzeugnis eines verteilten Echtzeitrechensystems



- die Daten selbst werden über den SPI-Bus entgegengenommen
  - die Buskommunikation erfordert einen 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) AB-System

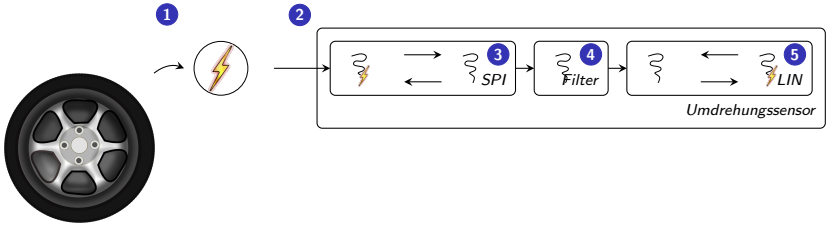
Ein Gemeinschaftserzeugnis eines verteilten Echtzeitrechensystems



- anschließend übernimmt ein Filter die Vorverarbeitung
  - Angleichung diverser Ausführungsraten durch den gesonderten Faden
    - der Filter verarbeitet immer mehrere Messwerte auf einmal
    - ⇝ Wann wird der Faden eingeplant? Muss er auf Betriebsmittel warten?

Beispiel: ein (fiktives) AB-System

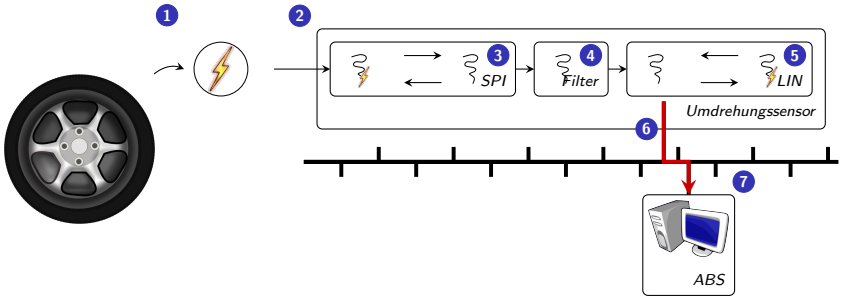
Ein Gemeinschaftserzeugnis eines verteilten Echtzeitrechensystems



- die Messwerte werden dann an das ABS-Steuergerät gesendet
  - auch hier ist ein 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“ übetragen werden?

Beispiel: ein (fiktives) AB-System

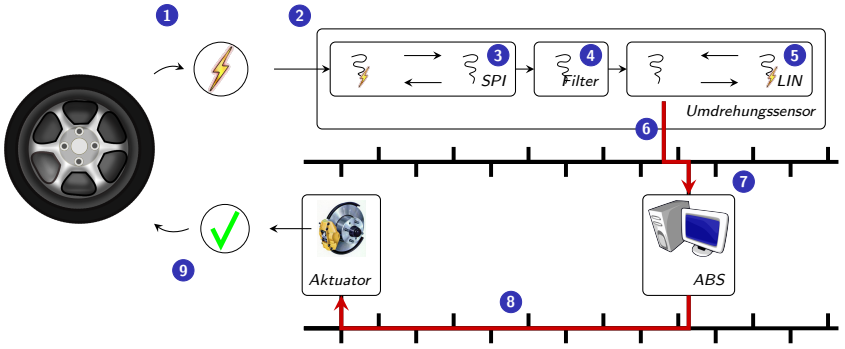
Ein Gemeinschaftserzeugnis eines verteilten Echtzeitrechensystems



- Sensor und ABS-Steuergerät sind über einen LIN-Bus verbunden
  - auch die Datenübertragung benötigt Zeit ...
    - ⇝ Wieviele Bytes schafft der Bus in einer bestimmten Zeit?
    - ⇝ Wie lange muss ich warten, bis ich auf das Medium zugreifen kann?
- die Vorgänge auf dem ABS-Steuergerät sind noch deutlich komplexer

Beispiel: ein (fiktives) AB-System

Ein Gemeinschaftserzeugnis eines verteilten Echtzeitrechensystems



- schließlich wird der berechnete Stellwert dem Aktor zugestellt
  - dieser ist z. B. über einen CAN-Bus an das Steuergerät angebunden
    - ⇝ Wieviele Bytes schafft der Bus in einer bestimmten Zeit?
    - ⇝ Wie lange muss ich warten, bis auf das Medium zufgreifen kann?

☞ schließlich wird die Bremse ggf. gelöst

Beispiel: ein (fiktives) AB-System (Forts.)

Wie lange dauert das ganze nun?

Für eine korrekte Funktion des AB-Systems muss die Reaktion auf eine Blockierung des Rades in einer bestimmten Zeitspanne gewährleistet sein.

Zu dieser Zeitspanne tragen zwei Komponenten bei:

„aktive“ Zeitintervalle $\rightsquigarrow$ „Fortschritt“ im ABS

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

„inaktive“ Zeitintervalle $\rightsquigarrow$ „Wartezeit“ für das ABS

- Fortschritt erfordert die Zuteilung von Betriebsmitteln
  - z. B. der CPU für eine Berechnung oder des Kommunikationsmediums für die Übertragung der Daten

☞ 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 direkt eine Aussage zu dieser Frage!

Spezialzweckbetrieb

Verhalten von Echtzeitanwendungen [4, S. 25]

☞ deterministische Abarbeitung von Ereignisbehandlungen

rein zyklisch $\rightsquigarrow$ nur periodische Ereignisbehandlungen, Abfrage-Betrieb

- nahezu konstanter Betriebsmittelbedarf von Periode zu Periode

meist zyklisch $\rightsquigarrow$ überwiegend periodische Ereignisbehandlungen

- das System muss auf externe Ereignisse reagieren können

asynchron/vorhersagbar $\rightsquigarrow$ kaum periodische Ereignisbehandlungen

- aufeinanderfolgende Aktivierungen können zeitlich stark variieren
- Zeitdifferenzen haben eine obere Grenze oder bekannte Statistik

asynchron/nicht vorhersagbar $\rightsquigarrow$ aperiodische Ereignisbehandlungen

- Anwendungen reagieren auf asynchrone Ereignisse
- hohe, nicht deterministische Laufzeitkomplexität einzelner Ereignisbehandlungen

Gliederung

Aufbau der Vorlesung

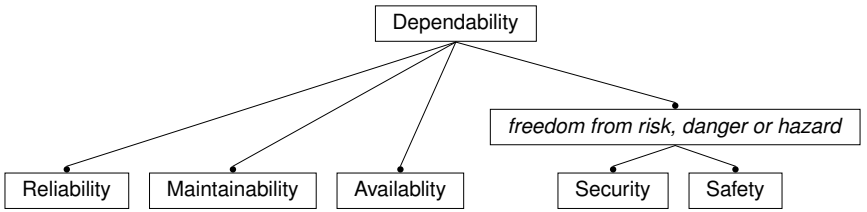
Die Vorlesung orientiert sich vor allem ...

- an der Ausprägung des Spezialzweckbetriebs ...
- und den Eigenschaften der Ereignisse und ihrer Behandlungen,
- blickt aber auch über den Tellerrand.

	vorranggesteuerte Systeme	taktgesteuerte Systeme	Analyse
Einleitung			
Grundlagen			
periodische Echtzeitsysteme			
nicht-periodische Echtzeitsysteme			
Rangfolge			
Zugriffskontrolle			
verteilte Echtzeitsysteme			
Exkurs			
Zusammenfassung und Ausblick			

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]*

Zuverlässigkeit (engl. *reliability*)

Mittlere Betriebsdauer

$R(t)$ die Wahrscheinlichkeit, dass ein System seinen Dienst bis zum Zeitpunkt t leisten wird, sofern es bei $t = t_0$ betriebsbereit war

- Annahme: eine konstante Fehlerrate von λ Fehler/Stunde
- Zuverlässigkeit zum Zeitpunkt t: $R(t) = exp(-\lambda(t - t_0))$
  - mit $t - t_0$ gegeben in Stunden
- Inverse der Fehlerrate $1/\lambda$ ist die *mean time to failure* (MTTF)

ultra-hohe Zuverlässigkeit wenn $\lambda \leq 10^{-9}$ Fehler/Stunde gefordert ist

- Beispiel: elektronisch gesteuerte Bremsanlage im Automobil
  - das Kfz sei durchschnittlich eine Stunde täglich in Betrieb
  - dann darf jährlich nur ein Fehler pro eine Million Kfz auftreten
- andere Beispiele sind Eisenbahnsignalanlagen, Kernkraftwerk- und Flugüberwachungssysteme

Wartbarkeit (engl. *maintainability*)

Mittlere Reparaturdauer

$M(d)$ die Wahrscheinlichkeit, dass das System innerhalb Zeitspanne d nach einem reparierbaren (gutartigen) Fehler wieder hergestellt ist

- Ansatz: konstante Reparaturrate von μ Reparaturen/Stunde
- die Inverse $1/\mu$ ist dann die *mean time to repair* (MTTR)

Fundamentaler Konflikt zwischen Zuverlässigkeit und Wartbarkeit:

- ein wartbares System erfordert einen modularen Aufbau
  - kleinste ersetzbare Einheit (engl. *smallest replaceable unit*, SDU)
  - über Steckverbindungen lose gekoppelt mit anderen SDUs
  - dadurch ist jedoch eine höhere (physikalische) Fehlerrate gegeben
  - darüberhinaus verbuchen sich höhere Herstellungskosten
- ein zuverlässiges System ist aus einem Guss gefertigt...

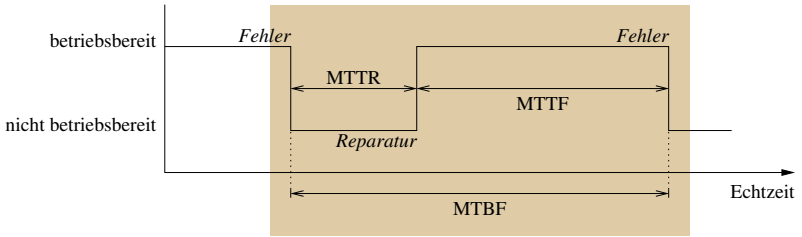
> *Beim Entwurf von Produkten für den Massenmarkt geht die Zuverlässigkeit meist auf Kosten von Wartbarkeit.*

Verfügbarkeit (engl. *availability*)

MTTF und MTTR im Zusammenhang

Maß zur Bereitstellung einer Funktion vor dem Hintergrund eines abwechselnd korrekt und fehlerhaft arbeitenden Systems

- Zeitanteil der Betriebsbereitschaft: $A = MTTF/(MTTF + MTTR)$
- $MTTF + MTTR$ auch kurz: *mean time between failures* (MTBF)



☞ hohe Verfügbarkeit bedeutet kurze MTTR und/oder lange MTTF

Sicherheit $\mapsto$ *Security* und *Safety*

Robustheit des Echtzeitrechensystems stärken

security Schutz von Informationen und Informationsverarbeitung vor „intelligenten“ Angreifern
- allgemein in Bezug auf Datenbasen des Echtzeitsystems
  - Vertraulichkeit (engl. *confidentiality*)
  - Datenschutz (engl. *privacy*)
  - Glaubwürdigkeit (engl. *authenticity*)
- speziell z.B. Diebstahlsicherung: Zündungssperre im Kfz
  - Kryptographie (engl. *cryptography*)

safety Schutz von Menschen und Sachwerten vor dem Versagen technischer Systeme
- Zuverlässigkeit trotz bösartigen (engl. *malign*) Fehlerfall
  - Kosten liegen um Größenordnungen über den Normalbetrieb
- Abgrenzung von unkrit., gutartigen (engl. *benign*) Fehlern
- oft ist Zertifizierung (engl. *certification*) erforderlich

Verlässlichkeit $\leftrightarrow$ Komplexität

Automobil eine Bestandsaufnahme vom Jahr 2005 ...
- etwa 90 % der Innovationen im Auto bringt die Elektronik ein
  - gut 80 % davon sind Software
- etwa ein Drittel aller Pannen liegen an fehlerhafte Elektronik
  - gut 80 % davon sind Softwarefehler

> *Everything should be made as simple as possible, but no simpler. (Albert Einstein)*

> *You know you have achieved perfection in design, not when you have nothing more to add, but when you have nothing more to take away. (Antoine de Saint Exupery)*

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.
  - Entsprechende Maßnahmen zum Erreichen von Fehlertoleranz werden also nicht durch harte Termine impliziert.

☞ Harte Echtzeitsysteme sind häufig auch äußerst verlässlich. ;-)

Gliederung

Resümee

Echtzeitbetrieb eines Rechensystems in seiner Umgebung

- Komponenten eines Echtzeitsystems
  - Operateur, Echtzeitrechensystem, kontrolliertes Objekt
- Determiniertheit, Determinismus, Vorhersagbarkeit
- Verhalten von Echtzeitanwendungen
  - rein/meist zyklisch
  - asynchron und irgendwie/nicht vorhersagbar
- schwache, starke oder strikte Echtzeitbedingungen

Abgrenzung Fokus dieser Vorlesung liegt auf der Rechtzeitigkeit

- Rechtzeitigkeit und Verlässlichkeit bedingen sich oft gegenseitig
- Maßnahmen zu ihrem Erreichen sind grundverschieden:
  - Verlässlichkeit $\leadsto$ Robustheit durch Fehlertoleranz
  - Rechtzeitigkeit $\leadsto$ deterministisches Ablaufverhalten

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.
Prentice-Hall, Inc., 2000. –
ISBN 0–13–099651–3