

# Echtzeitsysteme

## Rekapitulation

Peter Wägemann

Lehrstuhl für Verteilte Systeme und Betriebssysteme

Friedrich-Alexander-Universität Erlangen-Nürnberg

<https://sys.cs.fau.de/lehre/ss22/ezs/>

19. Juli 2022



# Verlässliche Echtzeitsysteme – Übersicht



Wir haben uns bisher nur um **Rechtzeitigkeit** gekümmert!

**Fehlerfall** übersteigt die Kosten des Normalfalls um Größenordnungen

- Fragestellung: Wie werden verlässliche Echtzeitsysteme entwickelt?
  - Wie wird die Korrektheit von Software sichergestellt?
  - Welche Laufzeitfehler sind insbesondere von Belang?
  - Welche Fehlertoleranzmechanismen werden implementiert?

Ziel: Zuverlässiger Betrieb, minimierte Ausfallwahrscheinlichkeit



# Verlässliche Echtzeitsysteme – Termin

## ■ Vorlesung

- Dozent: Peter Wägemann
- ECTS: 5 oder 7,5
- Raum: 0.031-113 (Aquarium)
- Termin: Montag, 16:15-17:45 Uhr

## ■ Übungen

- Übungsleiter: Simon Schuster (et al.)
- Basis + Erweiterte Übungen, Rechnerübung



Raum- und Zeitangaben sind vorläufig!



# Gliederung

1 Verlässliche Echtzeitsysteme

2 Wiederholung



## II – Einleitung

- **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**



## III-1 – Zusammenspiel

**Zusammenspiel** Kontrolliertes Objekt ↔ Kontrollierendes Rechensystem

- die **Objektdynamik** definiert den zeitlichen Rahmen durch Termine
- die Echtzeitanwendung muss diese Termine einhalten
- ihr Anteil am kontrollierenden Rechensystem ist eher gering

**Programmunterbrechung** in synchroner oder asynchroner Ausprägung

- beeinflussen den Ablauf der Echtzeitanwendung
- Zustandssicherung, Verwaltungsgemeinkosten des schlimmsten Falls

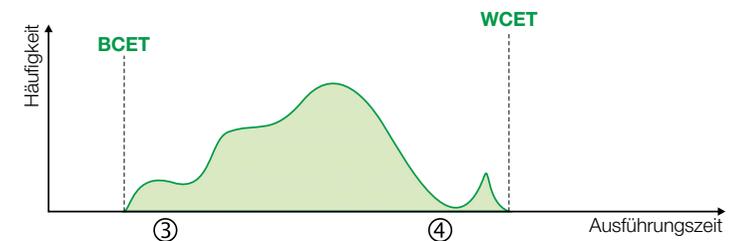


## III-2 – Echtzeitanwendung

- **Einplanungseinheit** → Prozedur, Faden und/oder Fadengruppe
  - Aufgaben (*Tasks*) und Arbeitsaufträgen (*Jobs*)
  - Verwaltungsgemeinkosten ein- und mehrfädiger Aufgaben
  - Einplanung als zweiphasiger Prozess
- **Ablaufsteuerung** → Strategie & Mechanismus
  - Einplanung ist die Strategie, Einlastung ist der Mechanismus
  - entkoppelt vs. gekoppelt, Zeitsteuerung vs. Ereignissteuerung
- **Zeitparameter** sind Punkte und Intervalle auf der Echtzeitachse
  - Auslösezeit, (absoluter) Termin
  - Antwortzeit, relativer Termin, Schlupfzeit, Ausführungszeit
- **Planbarkeit** sichert Rechtzeitigkeit der Echtzeitanwendung
  - gültige und zulässige Ablaufpläne
  - optimale Einplanungsalgorithmen
  - konstruktive vs. analytische Überprüfung der Planbarkeit



## III-3 – WCET



**WCET-Bestimmung** gliedert sich grob in zwei Teilprobleme

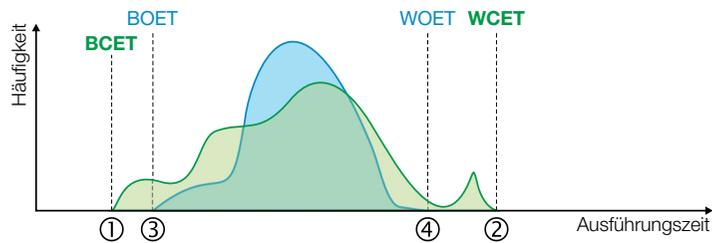
- **Programmiersprachenebene** (makroskopisch) → finde die längsten Pfade durch ein Programm
- **Maschinenprogrammzebene** (mikroskopisch) → bestimme die WCET der Elementaroperationen



Tatsächliche Ausführungszeit: **BCET / WCET**



### III-3 – WCET

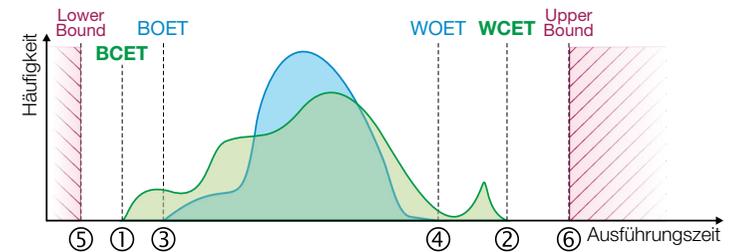


**Dynamische Analyse** → Beobachtung der Ausführungszeit

- Messung bezieht beide Ebenen mit ein
- Vollständige Messung im Allgemeinen **nicht möglich** → **Unterapproximation**

**Gemessene Ausführungszeit: BOET / WOET**

### III-3 – WCET

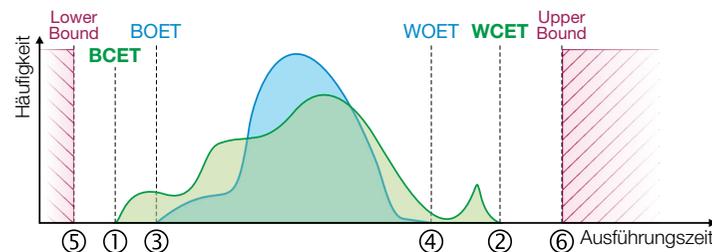


**Statische Analyse** → schätzt die Ausführungszeit

- Pfadanalyse (Programmiersprachenebene)
- Lösungswege: Abstraktion (Timing Schema vs. IPET)
- Gibt pessimistische Schranken an → **Überapproximation**

**Geschätzte Ausführungszeitgrenzen: Lower- / Upper Bound**

### III-3 – WCET



**Hardware-Analyse** → Eingaben für die WCET-Berechnung

- Hauptaufgaben: **Cache-** und **Pipeline-Analyse**
- must-Approximation und may-Approximation

**Werkzeugunterstützung** kombiniert Ebenen und macht die WCET-Analyse handhabbar

### IV-1 – Periodische EZS-Ausführung

**Periodische Aufgaben** haben in Echtzeitsystemen eine weite Verbreitung

- Periode, Phase, Hyperperiode, digitale Kontrollschleife
- Restriktionen periodischer Aufgaben und ihre Einschränkungen

**Zeitgesteuerte Ausführung** periodischer Aufgaben

- naive „Busy Loop“-Implementierung und Ablauftabellen
- Laufzeitkontrolle im Abfrage- und Unterbrecherbetrieb
- Tickless Systeme
- stapelbasierte Ablaufplanung

**Ereignisgesteuerte Ausführung** periodischer Aufgaben

- Ereignis- bzw. prioritätsorientierte Einplanung
- Feste und dynamische Prioritäten auf Task- bzw. Job-Ebene
- Auslösung vs. Auswahl, Ablaufliste vs. Ablauftabelle
- *Multi-Level-Queue-Scheduler*

## IV-2 – Periodische EZS

Ereignisgesteuerte Ablaufplanung

**Ablaufplanung** gebräuchliche, ereignisgesteuerte Verfahren

- **statische Prioritäten**  $\leadsto$  RM, DM
  - **Prioritätsabbildung** im Falle nicht ausreichender Systemprioritäten
- **dynamische Prioritäten**  $\leadsto$  EDF
- **Priorität/Dringlichkeit**  $\neq$  **Wichtigkeit/Kritikalität**

**Optimalität und Nichtoptimalität** von RM, DM und EDF

- Hängt von den Eigenschaften der betrachteten Aufgaben ab
- **Nichtoptimalität** von statischen Prioritäten und Ereignissteuerung

**Planbarkeitsanalyse** ereignisgesteuerter Ablaufplanungsverfahren

- **maximalen, kumulativen CPU-Auslastung** und **Antwortzeitanalyse**
- **relative Planbarkeit** im Falle nicht ausreichender Systemprioritäten



## IV-3 – Periodische EZS

- **Entwicklungsprozesse** führen verschiedenste Akteure zusammen
  - **Firmen/Arbeitsgruppen** sind u.U. über den ganzen Globus verstreut
  - $\rightarrow$  Eine **zeitliche Spezifikation** der Abläufe ist wünschenswert
    - Sie ermöglicht die Entwicklung **top-down** zu strukturieren
    - Wird durch eine manuelle, statische Ablaufplanung unterstützt
- **Struktur zyklischer Ablaufpläne**  $\leadsto$  gute Anordnung, Determinismus
  - **Rahmen, Rahmenlänge, Scheiben; major/minor cycle**
- **Algorithmische Einplanung** ordnet gerichtete, azyklische Graphen
  - $\rightarrow$  Entlastung bei der Lösung eines komplexen Problems
  - **List-Scheduling- und Branch&Bound-Algorithmen**
- **Moduswechsel** durch aperiodischen oder sporadischen Auftrag
  - **Tabellenwechsel, Betriebsmittelfreigabe/-anforderung, Nachladen**



## V-1 – Nicht-periodische EZS

Basistechniken

**Nicht-periodische Aufgaben** werden ereignisgesteuert ausgelöst

- **Harte o. feste/weiche Termine** (sporadische/aperiodische Aufgaben)
- **Mischbetrieb** ist eine Herausforderung

**Unterbrecherbetrieb** bevorzugt nicht-periodische Aufgaben

- **Sehr gut Antwortzeiten**, anfällig für **Überlast**
- **Gefährdet statische Garantien**  $\leadsto$  kontrollierter Unterbrecherbetrieb

**Hintergrundbetrieb** stellt nicht-periodische Aufgaben hinten an

- **Antwortzeiten** hängen von der Last periodischer Aufgaben ab

**Abfragende Zusteller** konvertieren sie in periodische Aufgaben

- **Schlechte Antwortzeiten**, Ausführungsbudget, Auffüllperiode

**Slack-Stealing** ist ein guter Kompromiss

- **Einfache Umsetzung** in gut strukturierten, zeitgesteuerten Systemen
- **Nicht praktikabel** in ereignisgesteuerten Systemen



## V-2 – Nicht-periodische EZS

Zustellerkonzepte

- **Bandbreite bewahrende Zusteller**  $\leadsto$  **Verbrauchs-/Auffüllregeln**
  - **Aufschiebbar**: ohne/mit Hintergrundzusteller, **Doppeltreffer**
  - **Sporadisch**: **SpSL Sporadic Server**, **Komplexität**
- **POSIX Sporadic Server**: Umsetzung des SpSL Sporadic Server
  - **Bedeutung** innerhalb des POSIX-Standard
  - **Ausweitung des Budgets**, verfrühte Auffüllung
  - **Unzureichende zeitliche Isolation**
- **Übernahmeprüfungen** für dynamische und statische Prioritäten
  - **Dichte-basierter Akzeptanztest** für die EDF-Ablaufplanung
  - **Schlupf-basierter Akzeptanztest** für sporadische Zusteller



## VI – Rangfolge

**Rangfolge**  $\leadsto$  gerichtete Abhängigkeiten

- resultieren oft aus Datenabhängigkeiten
- gerichtete Abhängigkeiten in nebenläufigen Ausführungsumgebungen erfordern Koordinierung

**Umsetzung gerichteter Abhängigkeiten**  $\leadsto$  Koordinierung

- wohlgeordneter Ablauf von Produzent und Konsument
- Übergang zwischen zeitlichen Domänen
- Implementierung gerichteter Abhängigkeiten
  - **implizit**  $\leadsto$  statische Ablaufabellen, Phasenverschiebung
  - **explizit**  $\leadsto$  Aktivierung, Zeitsignale, Nachrichten

**Ablaufplanung** nutzt die Einschränkung des Ablaufverhaltens

- **Nachfolger**  $\leadsto$  modifizierte Auslösezeiten
- **Vorgänger**  $\leadsto$  modifizierte Termine



## VII – Zugriffskontrolle

**Konkurrenz und Koordination** nebenläufiger Aktivitäten

- Nebenläufigkeit, Kausalität, Kausalordnung
- Konfliktsituationen  $\leadsto$  **synchronisieren ohne Prioritätsumkehr**

**Verdrängungssteuerung**  $\mapsto$  verdrängungsfreie kritische Abschnitte

- benötigt kein *à priori* Wissen; Verklemmungsvorbeugung
- pragmatisch/effektiv, beeinträchtigt unabhängige Jobs

**Prioritätsvererbung**  $\mapsto$  Priorität zeitweise erhöhen

- benötigt kein *à priori* Wissen
- direkte Blockierung, Blockierung durch Vererbung; transitiv

**Prioritätsobergrenzen**  $\mapsto$  Priorität zeitweise deckeln

- benötigt *à priori* Wissen; Verklemmungsvorbeugung
- Grundmodell vs. (einfachere) stapelorientierte Variante

**Ablaufplanung**  $\mapsto$  berücksichtigt Blockierungszeit

- Verzicht auf den Prozessor ermöglicht eine mehrfache Blockierung



## VIII – Mehrkernechtzeitsysteme

### ■ Mehrkernechtzeitsysteme sind die Zukunft

- Leistungssteigerung durch Parallelisierung

### ■ Ablaufplanung ist eine Herausforderung

- Wissen aus Einkernsystemen im Allgemeinen nicht übertragbar
- Zeitliche Anomalien  $\leadsto$  Kritischer Zeitpunkt, Prioritätsordnung
- Prioritätsproblem und Allokationsproblem

### ■ Partitionierte Ablaufplanung

- Verteilen der Aufgaben auf Kerne zum Entwurfszeitpunkt
- Transformation in mehrere Einkernsysteme
- $\rightarrow$  Bekannte Techniken und Algorithmen sind wieder anwendbar
- $\rightarrow$  Garantiert planbare Auslastung sehr schlecht

### ■ Globale Ablaufplanung

- Findet zur Laufzeit statt und erfordert Migration
- Verfahren mit dynamischen Prioritäten auf Auftragsebene erlauben vollständige Auslastung
- $\rightarrow$  In der Praxis mit hohen Kosten und Unwägbarkeiten behaftet



## VIII – Mehrkernechtzeitsysteme (Forts.)

### ■ Hybride Ablaufplanung

- Verbinden die Vor- und Nachteile der anderen Verfahren
- Teilpartitionierte und gruppierende Ablaufplanung

### ■ WCET-Analyse

- Komplexität nimmt stark zu
- Zusätzliche Kosten durch Migration und Synchronisation

