

# Betriebssysteme (BS)

## VL 1 – Einführung

**Daniel Lohmann**

Lehrstuhl für Informatik 4  
Verteilte Systeme und Betriebssysteme

Friedrich-Alexander-Universität  
Erlangen Nürnberg

WS 15 – 13. Oktober 2015

[https://www4.cs.fau.de/Lehre/WS15/V\\_BS](https://www4.cs.fau.de/Lehre/WS15/V_BS)



## Lernziele

- **Vertiefen** des Wissens über die interne Funktionsweise von Betriebssystemen
  - Ausgangspunkt: Systemprogrammierung
  - Schwerpunkt: Nebenläufigkeit und Synchronisation
- **Entwickeln** eines Betriebssystems *von der Pike auf*
  - OOSTuBS / MPStuBS Lehrbetriebssysteme
  - **Praktische** Erfahrungen im Betriebssystembau machen
- **Verstehen** der technologischen Hardware-Grundlagen
  - PC-Technologie verstehen und einschätzen können
  - Schwerpunkt: Intel x86 / IA-32



## Vorwort

*Die Lehrveranstaltung ist grundsätzlich für alle Studiengänge offen. Sie verlangt allerdings gewisse Vorkenntnisse. Diese müssen nicht durch Teilnahme an den Lehrveranstaltungen von 14 erworben worden sein.*



## Voraussetzungen

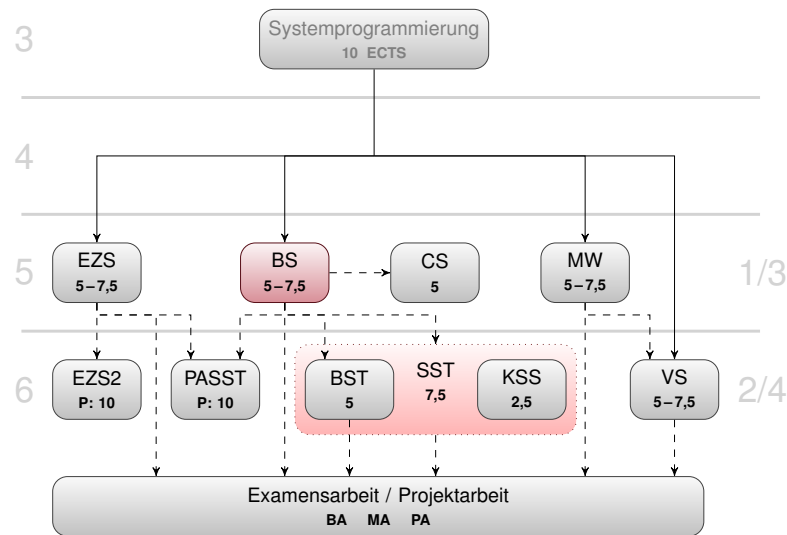
- Rechnerarchitektur, **Systemprogrammierung**
- C / **C++**, Assembler (x86)
- Ein gewisses Maß an **Durchhaltevermögen**
- Freude an systemnaher und **hardwarenaher Programmierung**

Wir arbeiten auf der "nackten Maschine" (*bare metal*)!

Die meisten sind überrascht, wie viel Spaß das macht :-)



## Einpassung in den Musterstudienplan (Bachelor/Master)



## Verwendbarkeit, Scheinerwerb und Modulnote

- **Wahlpflichtmodul** (Bachelor/Master) der Vertiefungsrichtung **Verteilte Systeme und Betriebssysteme**
  - eigenständig (nur BS) VL + Ü oder VL + EÜ
  - mit weiteren Veranstaltungen VL oder VL + Ü oder VL + EÜ
- Studien- und Prüfungsleistungen
  - Bachelor Prüfungsleistung
  - Master Prüfungsleistung
    - erworben durch
      - erfolgreiche Teilnahme an den Übungen
      - erfolgreiche Bearbeitung aller Übungsaufgaben
      - 30 min. mündliche Prüfung
- Berechnung der Modulnote
  - Note der mündlichen Prüfung + "Übungsbonus" in Zweifelsfällen

## Veranstaltungsbestandteile

(5 – 7,5 ECTS Punkte)

### VL – Vorlesung 2,5

Vorstellung und detaillierte Behandlung des Lehrstoffs

### Ü – Übung 2,5

- Übung **OOSTuBS**
- 6 – 7 Übungsaufgaben
- Abnahme alle 14 Tage

+

oder

+

### EÜ – Erweiterte Übung 5

- Übung **MPStuBS**
- erweiterte Aufgaben
- Rechnerübung "Pflicht"

### RÜ – Rechnerübung 0

- **Betreutes** Arbeiten am Rechner
- Hilfe zu OOSTuBS und **MPStuBS**

## Übungsbetrieb

### Übung

Raum 0.031 oder 00.152, Abgaben in 0.01 (Huber-CIP)

- Zwei Termine zur Auswahl
  - Do, 12:15 – 13:45 (00.151) oder Fr, 12:15 – 13:45 (0.031)
- Übungsaufgaben sind in 2er-Gruppen zu bearbeiten
- Anmeldung über **WAFFEL** (URL siehe Webseite)
  - Freischaltung erfolgt nach der Vorlesung, heute im Tagesverlauf

### Rechnerübung

Raum 0.01 (Huber-CIP)

- Zwei Termine zur Auswahl
  - Do, 14:00 – 15:30 (0.01) oder Fr, 14:00 – 15:30 (0.01)
- Betreuer können auch jederzeit direkt angesprochen werden

## Terminübersicht Wintersemester 2015

KW	Di 10-12	Do 12-14	Do 14-16	Fr 12-14	Fr 14-16	Raum
12.10.	VL <sub>1</sub>					0.031
19.10.	VL <sub>2</sub>	Ü <sub>1</sub>		Ü <sub>1</sub>		00.151
26.10.	VL <sub>3</sub>	Ü <sub>2</sub>	RÜ	Ü <sub>2</sub>	RÜ	0.031
02.11.	VL <sub>4</sub>	A <sub>1</sub>	RÜ	A <sub>1</sub>	RÜ	0.01
09.11.	VL <sub>5</sub>	Ü <sub>3</sub>	RÜ	Ü <sub>3</sub>	RÜ	0.01
16.11.	VL <sub>6</sub>	A <sub>2</sub>	RÜ	A <sub>2</sub>	RÜ	
23.11.	VL <sub>7</sub>	Ü <sub>4</sub>	RÜ	Ü <sub>4</sub>	RÜ	
30.11.		A <sub>3</sub>	RÜ	A <sub>3</sub>	RÜ	
07.12.	VL <sub>8</sub>	Ü <sub>5</sub>	RÜ	Ü <sub>5</sub>	RÜ	
14.12.	VL <sub>9</sub>	A <sub>4</sub>	RÜ	A <sub>4</sub>	RÜ	
21.12.						
11.01.	VL <sub>10</sub>	Ü <sub>6</sub>	RÜ	Ü <sub>6</sub>	RÜ	
18.01.	VL <sub>11</sub>	A <sub>5</sub>	RÜ	A <sub>5</sub>	RÜ	
25.01.	VL <sub>12</sub>	A <sub>6</sub>	RÜ	A <sub>6</sub>	RÜ	
01.02.	VL <sub>13</sub>	Ü <sub>7</sub>	RÜ	Ü <sub>7</sub>	RÜ	



## Beteiligte Personen

### Dozenten Vorlesung



Daniel Lohmann

### Tafel- und Rechnerübung



Christian Dietrich



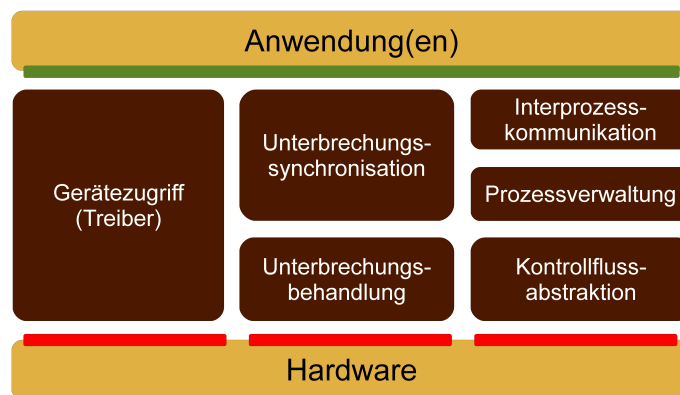
Valentin Rothberg



Sebastian Maier

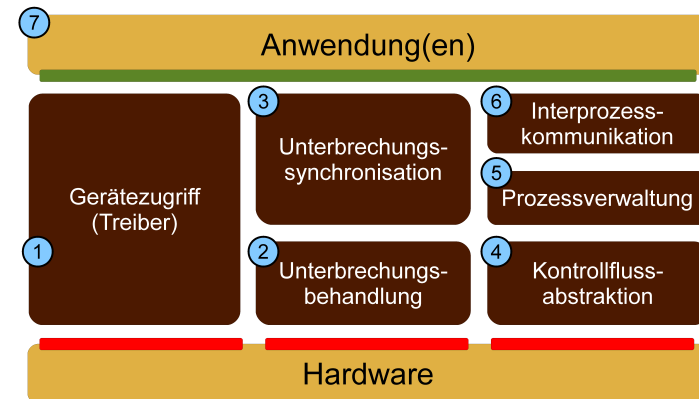


## Aufbau eines Betriebssystems



## Themenübersicht Übung

Am Beispiel von: OOSTuBS, MPStuBS

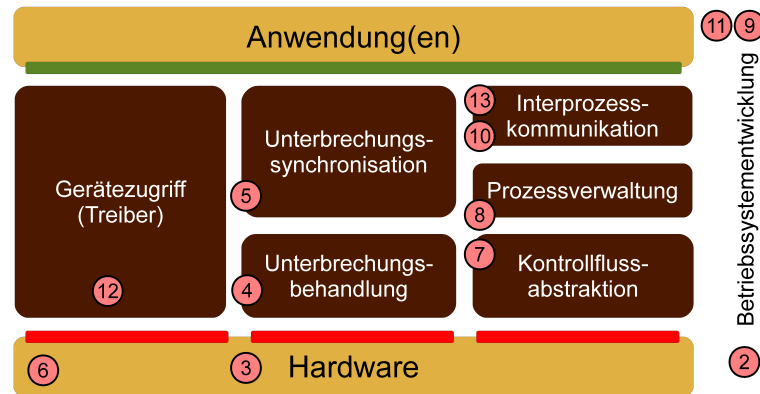


Betriebssystementwicklung



## Themenübersicht Vorlesung

Am Beispiel von: x86, MC68k, TriCore; Windows, Linux



## VL<sub>2</sub> Betriebssystementwicklung (oft ein harter Kampf!)

### ■ Erste Schritte

Wie bringt man sein System auf die Zielhardware?

- Übersetzen und Linken für "nackte Hardware"
- Bootvorgang

### ■ Testen und Fehlersuche

Was tun, wenn das System nicht reagiert?

- "printf"-Debugging
- Simulatoren
- Debugger
- Remote debugging
- Hardwareunterstützung



## Verzahnung von Vorlesung und Übungsaufgaben

### KW

12.10.	VL <sub>1</sub> Einführung	
19.10.	VL <sub>2</sub> BS-Entwicklung	→ Ü <sub>1</sub> Ein-/Ausgabe
26.10.	VL <sub>3</sub> IRQs (Hardware)	→ Ü <sub>2</sub> IRQ-Behandlung
02.11.	VL <sub>4</sub> IRQs (Software)	
09.11.	VL <sub>5</sub> IRQs (Synchronisation)	
16.11.	VL <sub>6</sub> Intel IA-32	→ Ü <sub>3</sub> IRQ-Synchronisation
23.11.	VL <sub>7</sub> Koroutinen und Fäden	
30.11.		→ Ü <sub>4</sub> Fadenumschaltung
07.12.	VL <sub>8</sub> Scheduling	
14.12.	VL <sub>9</sub> Architekturen	→ Ü <sub>5</sub> Zeitscheiben-Scheduling
21.12.		
11.01.	VL <sub>10</sub> Fadensynchronisation	→ Ü <sub>6</sub> Fadensynchronisation
18.01.	VL <sub>11</sub> Gerätetreiber	
25.01.	VL <sub>12</sub> IPC	
01.02.	VL <sub>13</sub> Ausblick	→ Ü <sub>7</sub> "Eine Anwendung" (opt.)



## VL<sub>3–4</sub> Programmunterbrechungen

### ■ im Prinzip

- Unterbrechungen, *Traps* und Ausnahmen
- Vektortabellen
- geschachtelte Unterbrechungen
- *spurious interrupts*

### ■ beim PC

- CPU und APIC
- Unterbrechungen in Multiprozessorsystemen

### ■ Behandlung im Betriebssystem

- Kopplungsfunktion
- Zustandssicherung



## VL<sub>5</sub> Unterbrechungssynchronisation

- Zusammenspiel zwischen Unterbrechungsbehandlung und “normalem” Kontrollfluss
  - Ursache und Problem
  - Kontrollflussebenenmodell
- Hardware-Mechanismen zur “harten Synchronisation”
  - cli und sti
  - Unterbrechungsebenen
- Software-Mechanismen zur “weichen Synchronisation”
  - Pro-/Epilogmodell und Varianten
  - Unterbrechungstransparente Algorithmen



## VL<sub>7</sub> Programmfäden

- Realisierung von Programmfäden
  - beim MC68k, Infineon TriCore, Intel x86
  - Fortsetzungen und Koroutinen als Basis
  - Implementierung des Kontextwechsels
- Fadenmodelle
  - leicht vs. schwer vs. federgewichtig vs. ...
  - Umsetzung in einer Systemfamilie



## VL<sub>6</sub> Das Intel IA-32 Programmiermodell

- Die Entwicklung der x86 CPU-Familie
  - vom 8086 bis zum Core i7
- Relikte und Eigenarten (*quirks*)
  - *Real Mode*
  - *A20 Gate*
- Neuerungen des *Protected Mode*
  - Ringe und Schutzmodell
  - *Task*-Modell
- Hardwarevirtualisierung



## VL<sub>8</sub> Ablaufplanung

- Kurze Wiederholung und Vertiefung
  - Grundprinzipien
  - Klassifikation
  - neue Strategien
- Beispiele aus der Praxis
  - Windows
  - Linux
  - Scheduling auf Multiprozessor-Systemen
- Herausforderungen beim Betriebssystembau
  - Zusammenspiel Ablaufplanung  $\Rightarrow$  Unterbrechungssynchronisation



- Wie organisiert man ein Betriebssystem: Architekturmodelle
  - Bibliotheken
  - Monolithen
  - Mikrokerne
  - Exokerne
  - Hypervisor
- Geschichte: Revolutionen, Religionen ... und die Realität
  - Bewertungskriterien
  - Erfolgs- und Misserfolgsgeschichten
- Beispiele aus der Praxis
  - OS360, Unix, Linux, L4, Windows
  - exoKernel, xen, vmware
  - ...



- Grundsätzliches
  - Voraussetzungen
  - aktives und passives Warten
- Synchronisationsprimitiven
  - *Mutex*, *Semaphore* und *Condition*
  - aus der Sicht des BS-Entwicklers
- spezielle Probleme
  - Wechselwirkung Synchronisation  $\Rightarrow$  Ablaufplanung
  - Fortschrittsgarantie und Verklemmung
- Beispiele aus der Praxis
  - Synchronisationsprimitiven in Windows



- Treiber und ihre Bedeutung
  - Vielfalt von Geräten
  - Probleme
- Komponentenmodell für Treiber
  - Struktur eines E/A-Systems
  - Treiberklassen und -schnittstellen
- Beispiele aus der Praxis
  - Windows
  - Linux



- Grundsätzliches
  - Wechselwirkung  $\Rightarrow$  Synchronisation
  - implizite und explizite Synchronisation
- Abstraktionen jenseits von *Semaphore*
  - gemeinsamer und verteilter Speicher
  - Fern- und Nahaufrufe
- Dualität nachrichtenbasierter und prozeduraler Systeme
  - konkrete Beispiele
  - Mikrokern  $\Leftrightarrow$  Monolith



### Quo Vadis Betriebssysteme?

- Zusammenfassung
  - Zusammenfassung des Lernstoffes
  - Diskussion der Evaluationsergebnisse
  - Tipps und Hinweise für die Prüfung
- Ausblick: Neue Herausforderungen
  - Multi- und Manycore Systeme
  - Heterogene Hardware
- Ausblick: Systeme aus der Forschung
  - Corey
  - Barrelfish/Multikernel
  - Factored OS
  - TxOS
  - OctoPOS/Invasive Computing
  - ...



# Viel Spaß!

