

Betriebssysteme (BS)

VL 1 – Einführung

Volkmar Sieh / Daniel Lohmann

Lehrstuhl für Informatik 4
Verteilte Systeme und Betriebssysteme

Friedrich-Alexander-Universität
Erlangen Nürnberg

WS 18 – 18. Oktober 2018

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



Die Lehrveranstaltung ist grundsätzlich für alle Studiengänge offen. Sie verlangt allerdings gewisse Vorkenntnisse. Diese müssen nicht durch Teilnahme an den Lehrveranstaltungen von I4 erworben worden sein.



- **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



Voraussetzungen

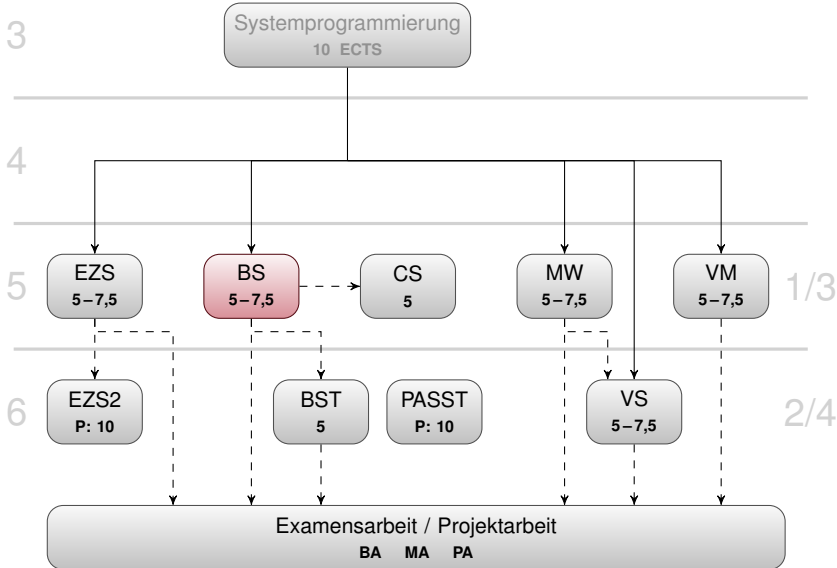
- Rechnerarchitektur, **Systemprogrammierung**
- C / **C++**, Assembler (x86)
- Ein gewisses Maß an **Durchhaltevermögen**
- Freude an systemnaher und **hardwarenaher Programmierung**

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

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



Einpassung in den Musterstudienplan (Bachelor/Master)



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**



- **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



Tafelübung

Raum 0.031 (Aquarium)

- Zwei Termine zur Auswahl
 - Fr, 12:15 – 13:45 *oder* Mo, 14:15 – 15:45
- Übungsaufgaben sind in 2er-Gruppen zu bearbeiten

Rechnerübung

Raum 0.01 (Huber-CIP)

- Zwei Termine zur Auswahl
 - Mo, 14:00 – 18:00 *und* Fr, 10:00 – 12:00

Erweiterte Rechnerübung

Raum 0.01 (Huber-CIP)

- Bei Bedarf (wird angekündigt)
 - Mi, 12:00 – 16:00



Terminübersicht Wintersemester 2018

Woche	Mo 14-16	Mo 16-18	Mi 12-16	Do 12-14	Fr 10-12	Fr 12-14	Raum
15.10.				VL ₁		Ü ₀	0.031
22.10.	Ü ₀	RÜ	eRÜ	VL ₂	RÜ	Ü ₁	
29.10.	Ü ₁	RÜ	eRÜ				0.031
05.11.		eRÜ	eRÜ	VL ₃	RÜ	Ü ₂	0.01
12.11.	Ü ₂	RÜ	eRÜ	VL ₄	RÜ	A ₁	
19.11.	A ₁	RÜ	eRÜ	VL ₅	RÜ	Ü ₃	0.01
26.11.	Ü ₃	RÜ	eRÜ	VL ₆	RÜ	A ₂	
03.12.	A ₂	RÜ	eRÜ	VL ₇	RÜ	Ü ₄	
10.12.	Ü ₄	RÜ	eRÜ	VL ₈	RÜ	A ₃	
17.12.	A ₃	RÜ	eRÜ	VL ₉	RÜ	Ü ₅	
07.01.	Ü ₅	RÜ	eRÜ	VL ₁₀	RÜ	A ₄	
14.01.	A ₄	RÜ	eRÜ	VL ₁₁	RÜ	Ü ₆	
21.01.	Ü ₆	RÜ	eRÜ	VL ₁₂	RÜ	A ₅	
28.01.	A ₅	RÜ	eRÜ	VL ₁₃	RÜ	Ü ₇	
04.02.	Ü ₇	RÜ	A ₆	VL ₁₄	RÜ	A ₆	



Beteiligte Personen

Dozent Vorlesung

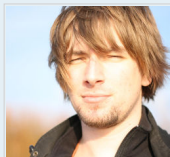


Volkmar Sieh

Übungsleiter

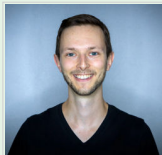


Andreas Ziegler

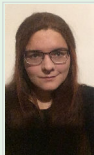


Bernhard Heinloth

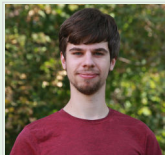
Betreuer Rechnerübung



Harald Böhm



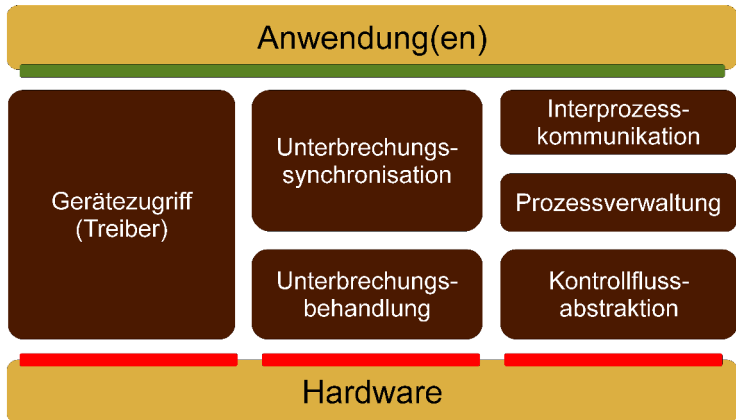
Judith Hemp



Jakob Wagner

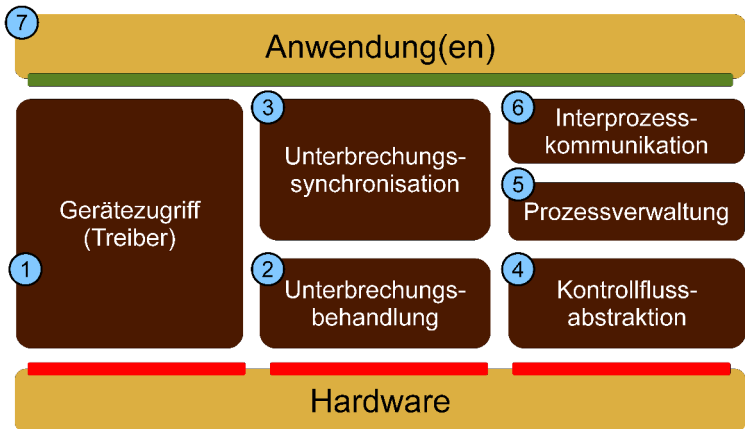


Aufbau eines Betriebssystem



Themenübersicht Übung

Am Beispiel von: OOSTuBS, MPStuBS

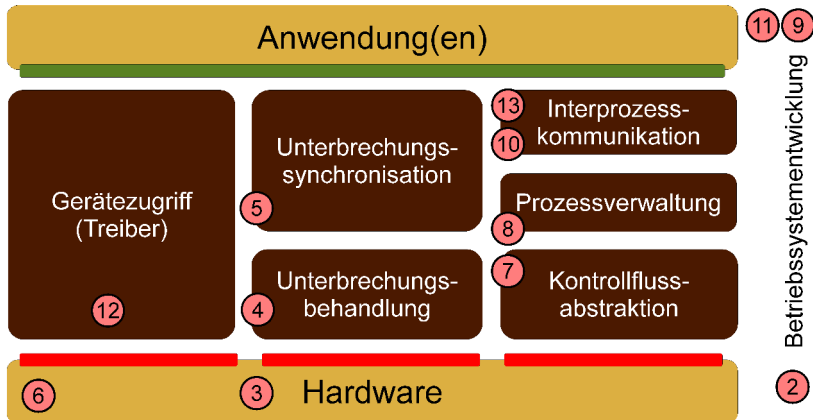


Betriebssystementwicklung

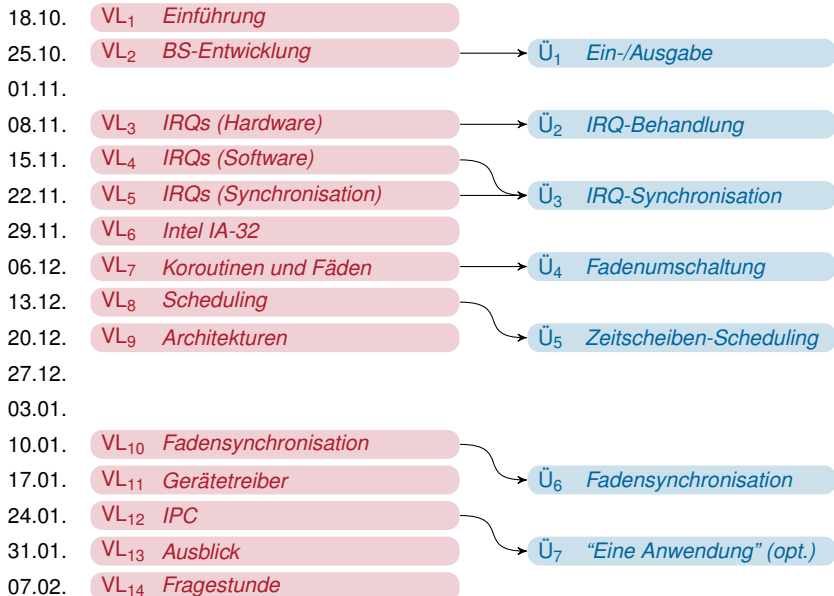


Themenübersicht Vorlesung

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



Verzahnung von Vorlesung und Übungsaufgaben



■ 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



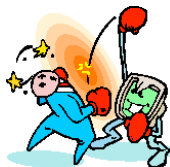
- im Prinzip
 - Unterbrechungen, *Traps* und Ausnahmen
 - Vektortabellen
 - geschachtelte Unterbrechungen
 - *spurious interrupts*
- beim PC
 - CPU und APIC
 - Unterbrechungen in Multiprozessorsystemen
- Behandlung im Betriebssystem
 - Kopplungsfunktion
 - Zustandssicherung



- 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



- 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



- 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



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



Was ihr noch alles fragen wolltet... ;-)





Viel Spaß!