

Betriebssysteme (BS)

VL 1 – Einführung

Daniel Lohmann

Lehrstuhl für Informatik 4
Verteilte Systeme und Betriebssysteme

Friedrich-Alexander-Universität
Erlangen Nürnberg

WS 12 – 17. Oktober 2012

http://www4.informatik.uni-erlangen.de/Lehre/WS12/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

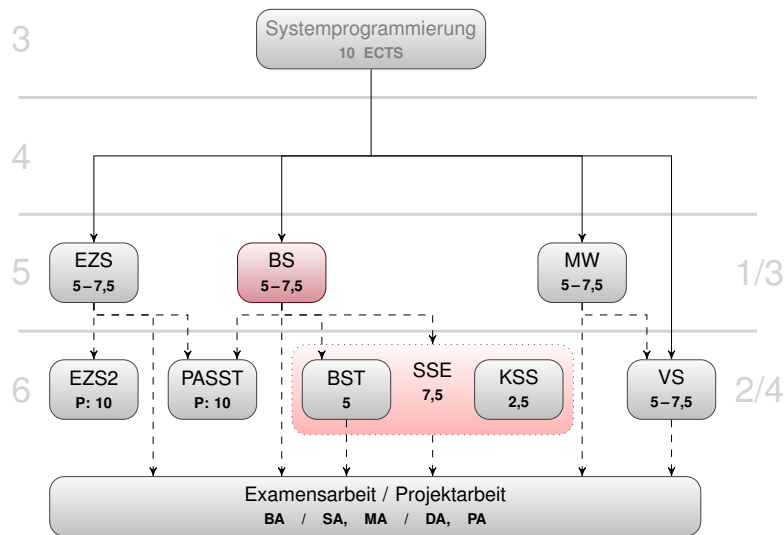
- Rechnerarchitektur, **Systemprogrammierung**
- C / **C++**, Assembler (x86)
- Ein gewisses Maß an **Durchhaltevermögen**
- Freude an systemnaher und **hardwarenaher Programmierung**

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

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



Einpassung in den Mutterstudienplan (Bachelor/Master)



Veranstaltungsbestandteile

(2,5 – 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**

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

Übungsbetrieb

Übung

(0.85 / K2, Abgaben in 0.01)

- Zwei Termine zur Auswahl
 - Mi, 16:15 – 17:45 (0.85) oder Do, 14:15 – 15:45 (K2)
- Übungsaufgaben sind in 2er-Gruppen zu bearbeiten
- Anmeldung über **WAFFEL** (URL siehe Webseite)
 - Freischaltung erfolgt nach der Vorlesung, heute im Tagesverlauf

Rechnerübung

(0.01)

- Zwei Termine zur Auswahl
 - Mo, 14:15 – 15:45 oder Di, 14:15 – 15:45
- Betreuer können auch jederzeit direkt angesprochen werden

Terminübersicht Wintersemester 2012

KW	Mo 14-16	Di 14-16	Mi 12-14	Mi 16-18	Do 14-16	Raum
15.10.			VL ₁			0.031
22.10.			VL ₂	Ü ₁	Ü ₁	0.85 / K2
29.10.			VL ₃	RÜ	RÜ	0.01-142
05.11.	RÜ	RÜ	VL ₄	Ü ₂	Ü ₂	
12.11.	RÜ	RÜ	VL ₅	A ₁	A ₁	0.01-142
19.11.	RÜ	RÜ	VL ₆	Ü ₃	Ü ₃	
26.11.	RÜ	RÜ	VL ₇	A ₂	A ₂	
03.12.	RÜ	RÜ		Ü ₄	Ü ₄	
10.12.	RÜ	RÜ	VL ₈	A ₃	A ₃	
17.12.	RÜ	RÜ	VL ₉	Ü ₅	Ü ₅	
07.01.	RÜ	RÜ	VL ₁₀	A ₄	A ₄	
14.01.	RÜ	RÜ	VL ₁₁	Ü ₆	Ü ₆	
21.01.	RÜ	RÜ	VL ₁₂	A ₅	A ₅	
28.01.	RÜ	RÜ	VL ₁₃	Ü ₇	Ü ₇	
04.02.			VL ₁₄	A ₆	A ₆	



Beteiligte Personen

Dozenten Vorlesung



Daniel Lohmann



Wolfgang Schröder-Preischat

Tafel- und Rechnerübung



Benjamin Oechlein



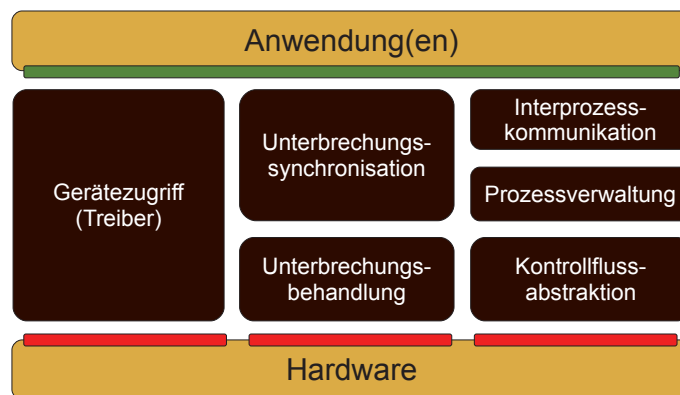
Gabor Drescher



Daniel Danner

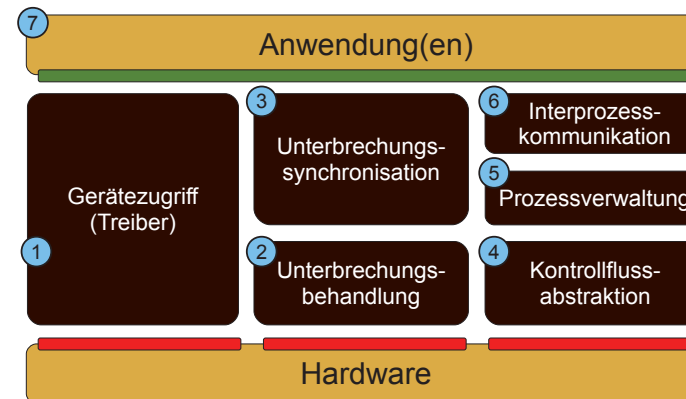


Aufbau eines Betriebssystems



Themenübersicht Übung

Am Beispiel von: OOSTuBS, MPStuBS

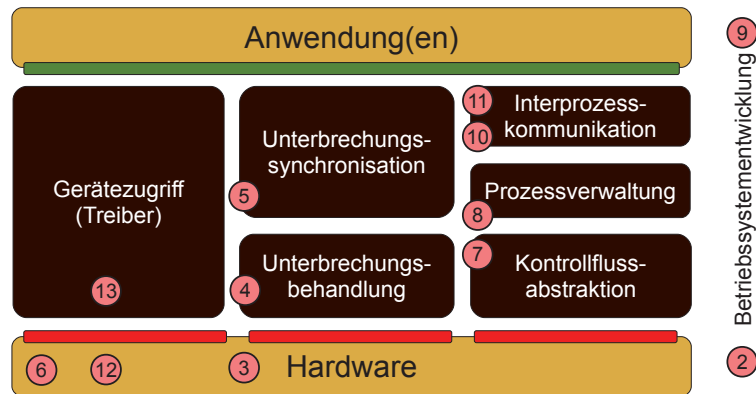


Betriebssystementwicklung



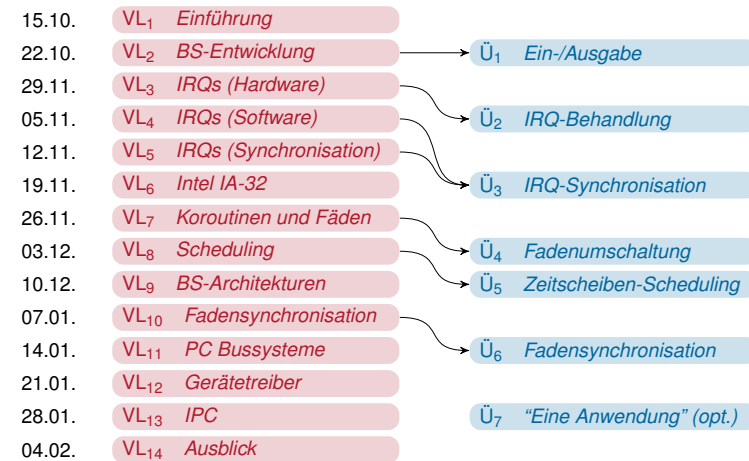
Themenübersicht Vorlesung

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



Verzahnung von Vorlesung und Übungsaufgaben

KW



VL₂ Betriebssystementwicklung (oft ein harter Kampf!)

- Erste Schritte
 - Wie bringt man sein System auf die Zielhardware?
 - Übersetzen und Linken für "nakte Hardware"
 - Bootvorgang
- Testen und Fehlersuche
 - Was tun, wenn das System nicht reagiert?
 - "printf"-Debugging
 - Simulatoren
 - Debugger
 - Remote debugging
 - Hardwareunterstützung



VL₃₋₄ 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



VL₅ 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



VL₆ 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



VL₇ 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



VL₈ 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



VL₉ Architekturen

- 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
 - . . .



VL₁₀ Fadensynchronisation

- 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



VL₁₁ Interprozesskommunikation (IPC)

- 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



VL₁₂ PC-Bussysteme

- Herkunft und Architektur
 - ISA und die Folgen
 - Programmiermodell
- Lokale Bussysteme
 - PCI und PCI Express
 - AGP
 - InfiniBand und HyperTransport
 - . . .
- E/A-Bussysteme
 - USB, Firewire
 - SCSI, SATA
 - . . .



VL₁₃ Geräteprogrammierung

- 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



VL₁₄ Zusammenfassung und Ausblick

- Zusammenfassung des Lernstoffes
- Diskussion der Evaluationsergebnisse
- Tipps und Hinweise für die Prüfung
- Ausblick



Das Endziel...



Viel Spaß!

