

Betriebssysteme (BS)

VL 1 – Einführung

Volkmar Sieh / Daniel Lohmann

Lehrstuhl für Informatik 4
Verteilte Systeme und Betriebssysteme

Friedrich-Alexander-Universität
Erlangen Nürnberg

WS 22 – 20. Oktober 2022

<https://sys.cs.fau.de/lehre/ws22/bs>



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.



- **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: x86-Architektur



Voraussetzungen

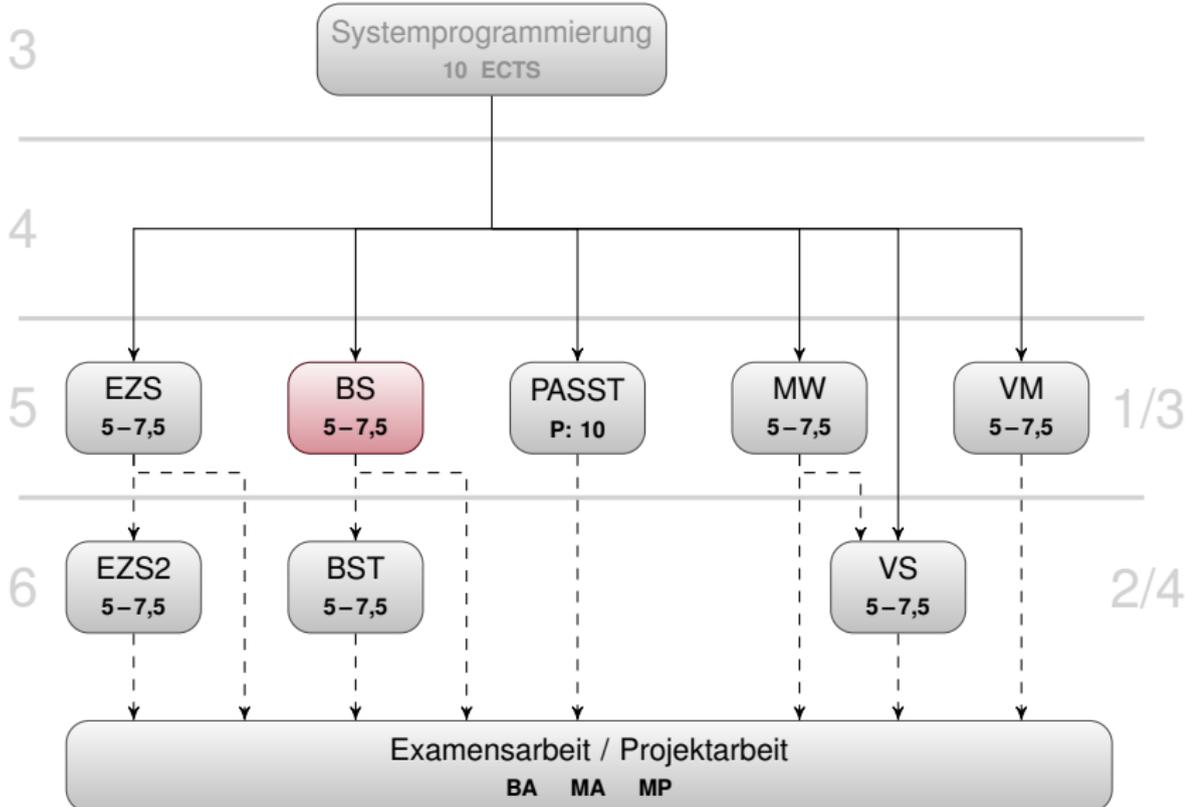
- Rechnerarchitektur, **Systemprogrammierung**
- C / **C++**, Assembler (x86)
- Ein gewisses Maß an **Durchhaltevermögen**
- Freude an systemnaher und **hardwarenaher Programmierung**

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 Übungsaufgaben
- Abnahme alle 14 Tage

oder

EÜ – Erweiterte Übung

5

- Übung **MPStuBS**
- 6 erweiterte Aufgaben
- Abnahme alle 14 Tage



- **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)

- **Ein Termin**
 - Di, 08:15 – 09:45
- Übungsaufgaben sind in 2er-Gruppen zu bearbeiten

Rechnerübung

Raum 01.153 (WinCIP)

- **Zwei Termine**
 - Mi, 14:00 – 18:00 *und* Fr, 12:00 – 16:00
 - davon je die letzten 2h als „Überlaufzeitschlitz“
- Abgabe der Übungsaufgaben

Zusatzseminar (freiwillig)

Raum 0.031 (Aquarium)

- **Drei Einzeltermine**
 - Di, 08.11., 22.11. und 06.12. um 08:15 Uhr
 - Zeitschlitz der Tafelübung



Terminübersicht Wintersemester 2022

KW	Di 8-10	Mi 14-16	Do 14-16	Fr 12-14	Raum
42			VL ₁		0.031
43	Ü ₁	RÜ	VL ₂	RÜ	
44		RÜ	VL ₃	RÜ	0.031
45	Sem ₁	RÜ	VL ₄	A ₁	0.031
46	Ü ₂	RÜ	VL ₅	RÜ	
47	Sem ₂	RÜ	VL ₆	A ₂	01.153
48	Ü ₃	RÜ	VL ₇	RÜ	01.153
49	Sem ₃	RÜ	VL ₈	A ₃	
50	Ü ₄	RÜ	VL ₉	RÜ	
51		RÜ		A ₄	
02	Ü ₅	RÜ	VL ₁₀	RÜ	
03		RÜ	VL ₁₁	A ₅	
04	Ü ₆	RÜ	VL ₁₂	RÜ	
05		RÜ	VL ₁₃	A ₆	
06	Ü ₇		VL ₁₄		



Dozent Vorlesung



Volkmar Sieh

Übungsleiter



Phillip Raffeck



Dustin Nguyen



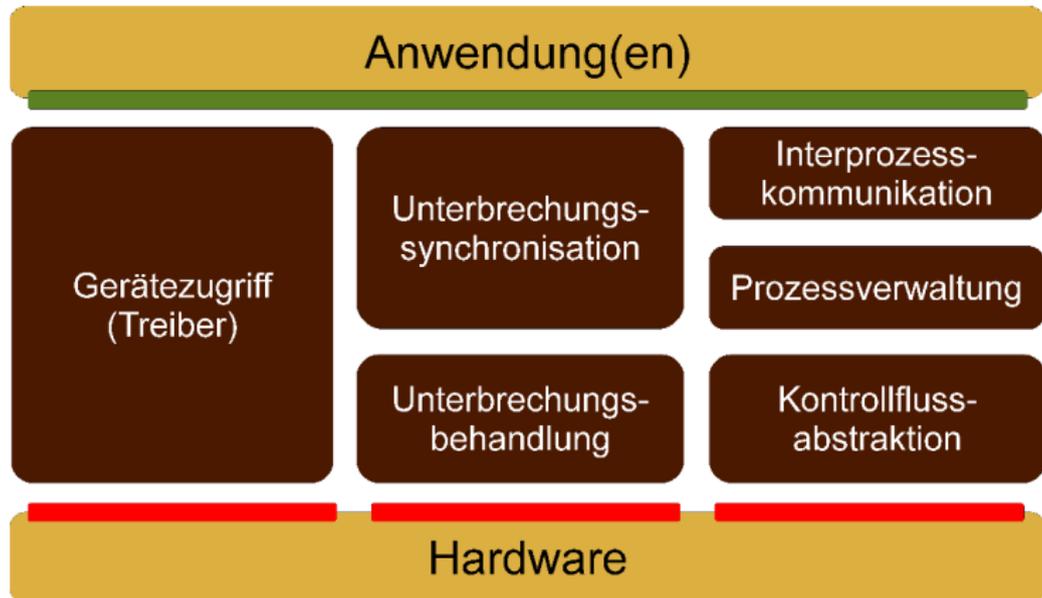
Bernhard Heinloth



Paul Bergmann

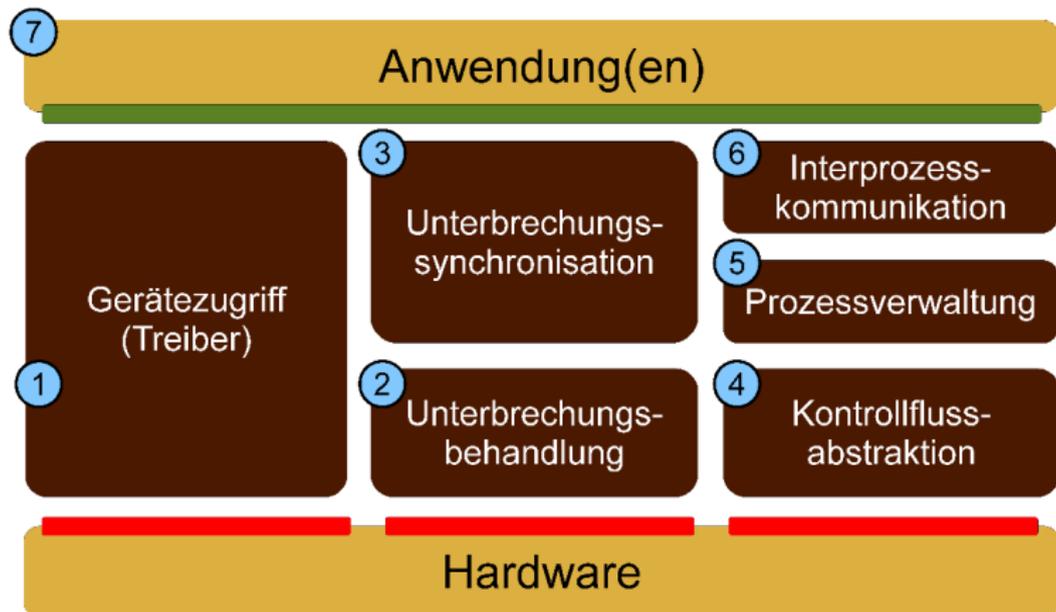


Aufbau eines Betriebssystems



Themenübersicht Übung

Am Beispiel von: OOSTuBS, MPStuBS

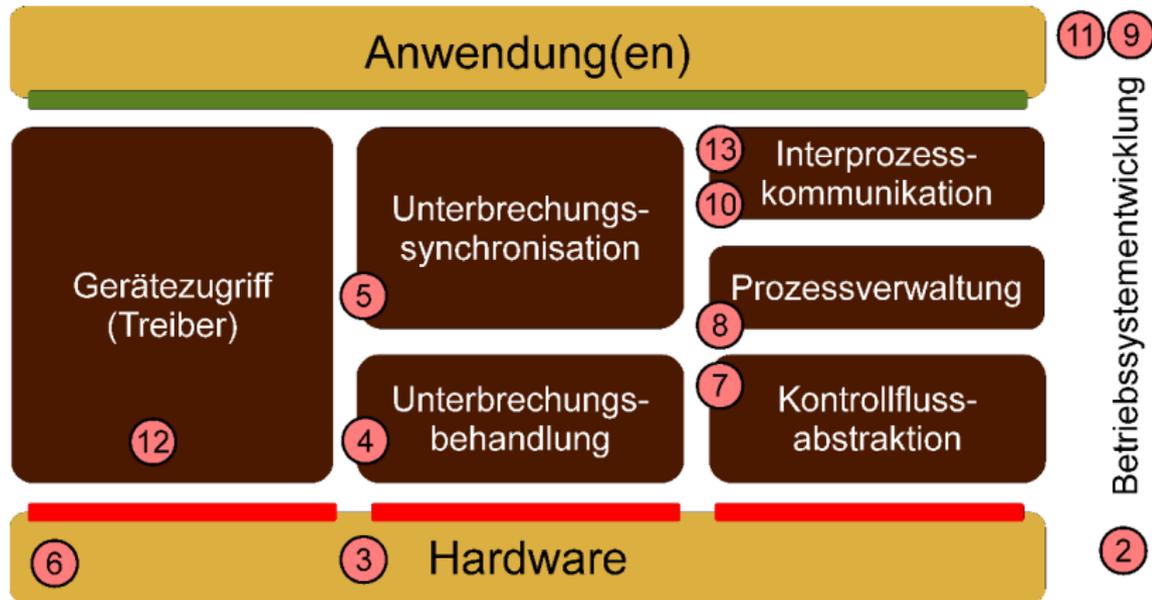


Betriebssystementwicklung

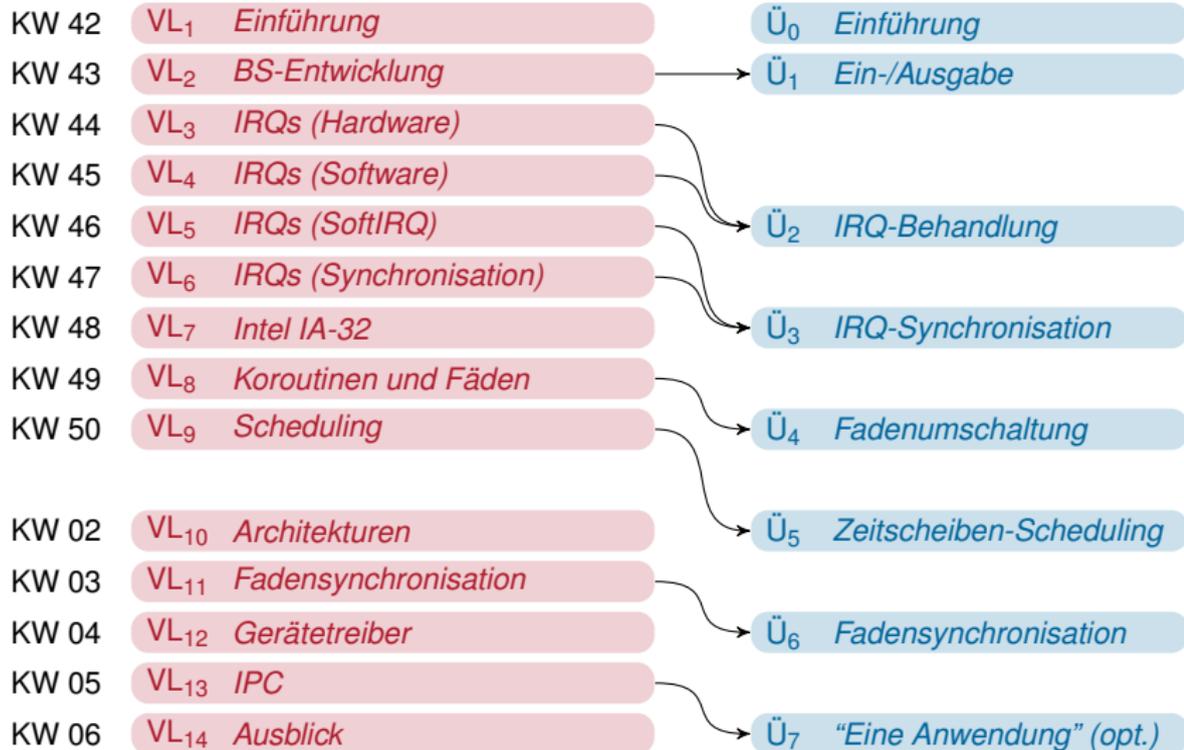


Themenübersicht Vorlesung

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



Verzahnung von Vorlesung und Übungsaufgaben



■ 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
 - ...





Viel Spaß!

