Betriebssysteme (BS)

VL 1 – Einführung

Volkmar Sieh / Daniel Lohmann

Lehrstuhl für Informatik 4
Verteilte Systeme und Betriebssysteme

Friedrich-Alexander-Universität
Erlangen Nürnberg

WS 23 – 18. Oktober 2023

https://sys.cs.fau.de/lehre/ws23/bs

Vorwort

Die Lehrveranstaltung ist grundsätzlich für alle Studiengänge offen. Sie verlangt allerdings gewisse Vorkenntnisse. Diese müssen nicht durch Teilnahme an den Lehrveranstaltungen von I4/I16 erworben worden sein.

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 Betriebsystembau machen
- **Verstehen** der technologischen Hardware-Grundlagen
  - PC-Technologie verstehen und einschätzen können
  - Schwerpunkt: x86-/x86_64-Architektur

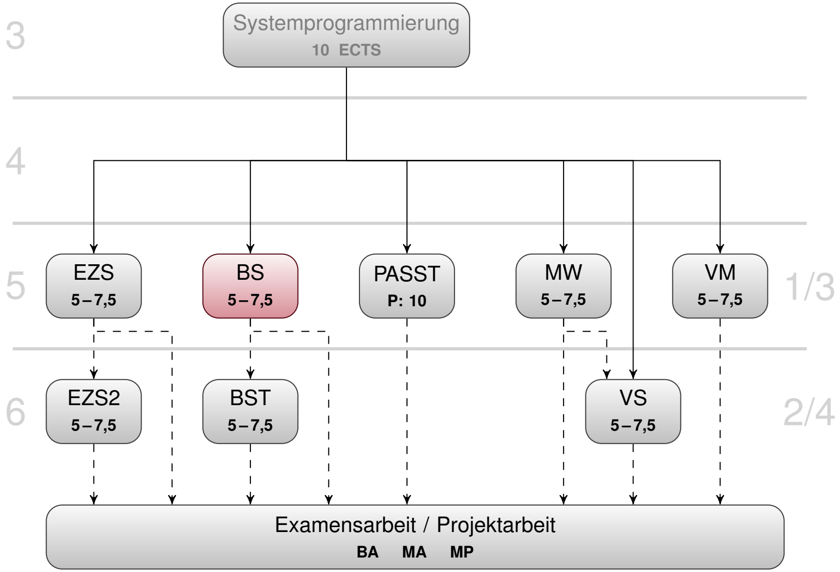
Voraussetzungen

- Rechnerarchitektur, **Systemprogrammierung**
- C / **C++**, Assembler (x86/x86_64)
- 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)



Veranstaltungsbestandteile (5 – 7,5 ECTS Punkte)

VL – Vorlesung **2,5**

Vorstellung und detailierte 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

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

Tafelübung

Raum 0.031 (Aquarium)

- **Ein** Termin
  - Mi, 16:15 – 17:45
- Übungsaufgaben sind in 2er-Gruppen zu bearbeiten

Rechnerübung

Raum 01.153 (WinCIP)

- Zwei Termine
  - Mi, 14:00 – 16:00
  - Fr, 12:00 – 14:00
- Abgabe der Übungsaufgaben

Zusatzseminar (freiwillig)

Raum 0.031 (Aquarium)

- Drei Einzeltermine
  - Mi, 08.11., 22.11. und 06.12. um 16:15 Uhr
  - Zeitschlitz der Tafelübung

Terminübersicht Wintersemester 2023

KW	Mi 12-14	Mi 14-16	Mi 16-18	Fr 12-14	Raum
42	VL_1	RÜ			0.031
43	VL_2	RÜ	$Ü_1$	RÜ	
44				RÜ	0.031
45	VL_3	RÜ	Sem_1	A_1	0.031
46	VL_4	RÜ	$Ü_2$	RÜ	
47	VL_5	RÜ	Sem_2	A_2	01.153
48	VL_6	RÜ	$Ü_3$	RÜ	01.153
49	VL_7	RÜ	Sem_3	A_3	
50	VL_8	RÜ	$Ü_4$	RÜ	
51	VL_9	RÜ		A_4	
02	VL_{10}	RÜ	$Ü_5$	RÜ	
03	VL_{11}	RÜ		A_5	
04	VL_{12}	RÜ	$Ü_6$	RÜ	
05	VL_{13}	RÜ		A_6	
06	VL_{14}		$Ü_7$		

Beteiligte Personen

Dozent Vorlesung

Volkmar Sieh

Übungsleiter

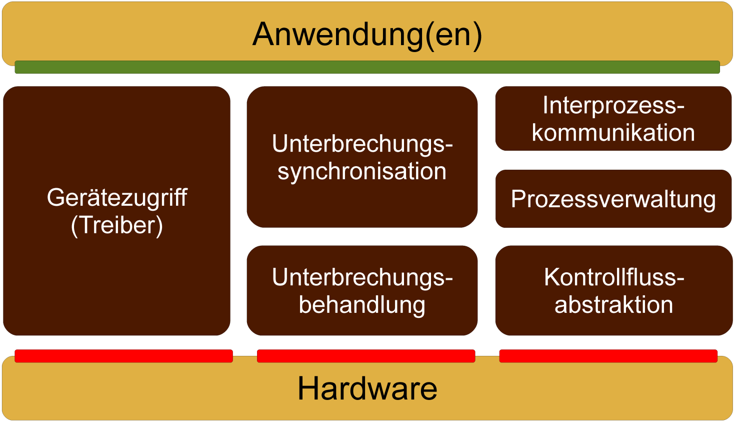
Phillip Raffeck

Dustin Nguyen

Maximilian Ott

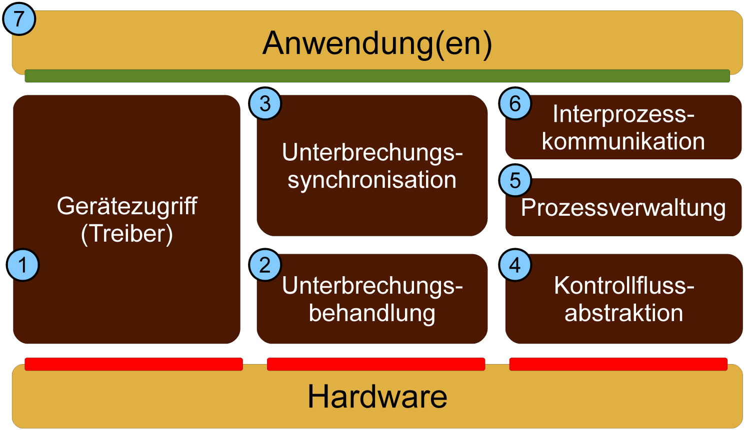
Paul Bergmann

Aufbau eines Betriebssystem



Themenübersicht Übung

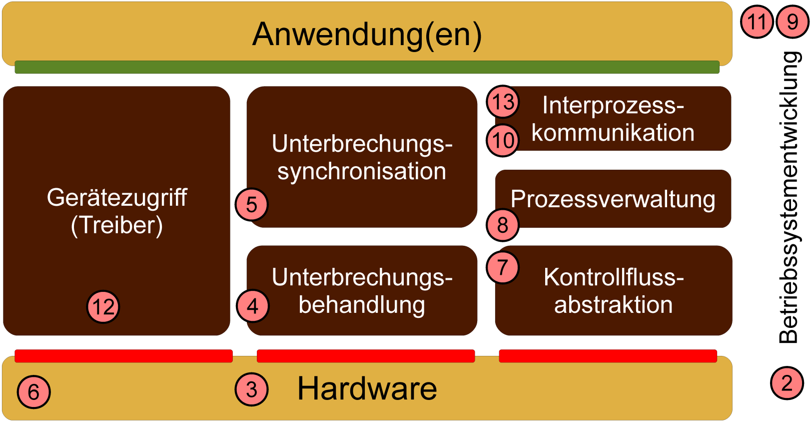
Am Beispiel von: OOStuBS, MPStuBS



Betriebssystementwicklung

Themenübersicht Vorlesung

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



Verzahnung von Vorlesung und Übungsaufgaben

KW 42	VL_1 *Einführung*	$Ü_0$ *Einführung*
KW 43	VL_2 *BS-Entwicklung*	$Ü_1$ *Ein-/Ausgabe*
KW 44	VL_3 *IRQs (Hardware)*	
KW 45	VL_4 *IRQs (Software)*	
KW 46	VL_5 *IRQs (SoftIRQ)*	$Ü_2$ *IRQ-Behandlung*
KW 47	VL_6 *IRQs (Synchronisation)*	
KW 48	VL_7 *Intel IA-32*	$Ü_3$ *IRQ-Synchronisation*
KW 49	VL_8 *Koroutinen und Fäden*	
KW 50	VL_9 *Scheduling*	$Ü_4$ *Fadenumschaltung*
KW 02	VL_{10} *Architekturen*	$Ü_5$ *Zeitscheiben-Scheduling*
KW 03	VL_{11} *Fadensynchronisation*	
KW 04	VL_{12} *Gerätetreiber*	$Ü_6$ *Fadensynchronisation*
KW 05	VL_{13} *IPC*	
KW 06	VL_{14} *Ausblick*	$Ü_7$ *“Eine Anwendung” (opt.)*

VL_2 Betriebssystementwicklung (oft ein harter Kampf!)

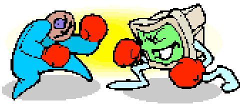
- Erste Schritte

  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

VL_{3-5} 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$_6$ 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_7 Das x86 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$_8$ Programmfäden

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

VL$_9$ Ablaufplanung

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

VL_{10} 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
  - ...

VL_{11} Fadensynchronisation

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

VL$_{12}$ 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

VL_{13} Interprozesskommunikation (IPC)

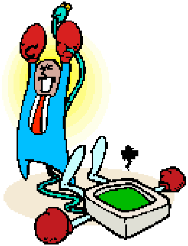
- Grundsätzliches
  - Wechselwirkung $\rightleftharpoons$ Synchronisation
  - implizite und explizite Synchronisation
- Abstraktionen jenseits von *Semaphore*
  - gemeinsamer und verteilter Speicher
  - Fern- und Nahaufrufe
- Dualität nachrichtenbasierter und prozeduraler Systeme
  - konkrete Beispiele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

VL_{14} Zusammenfassung und Ausblick

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
  - ...

Das Endziel. . .



Viel Spaß!