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

Vorwort

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.

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: Intel x86 / IA-32

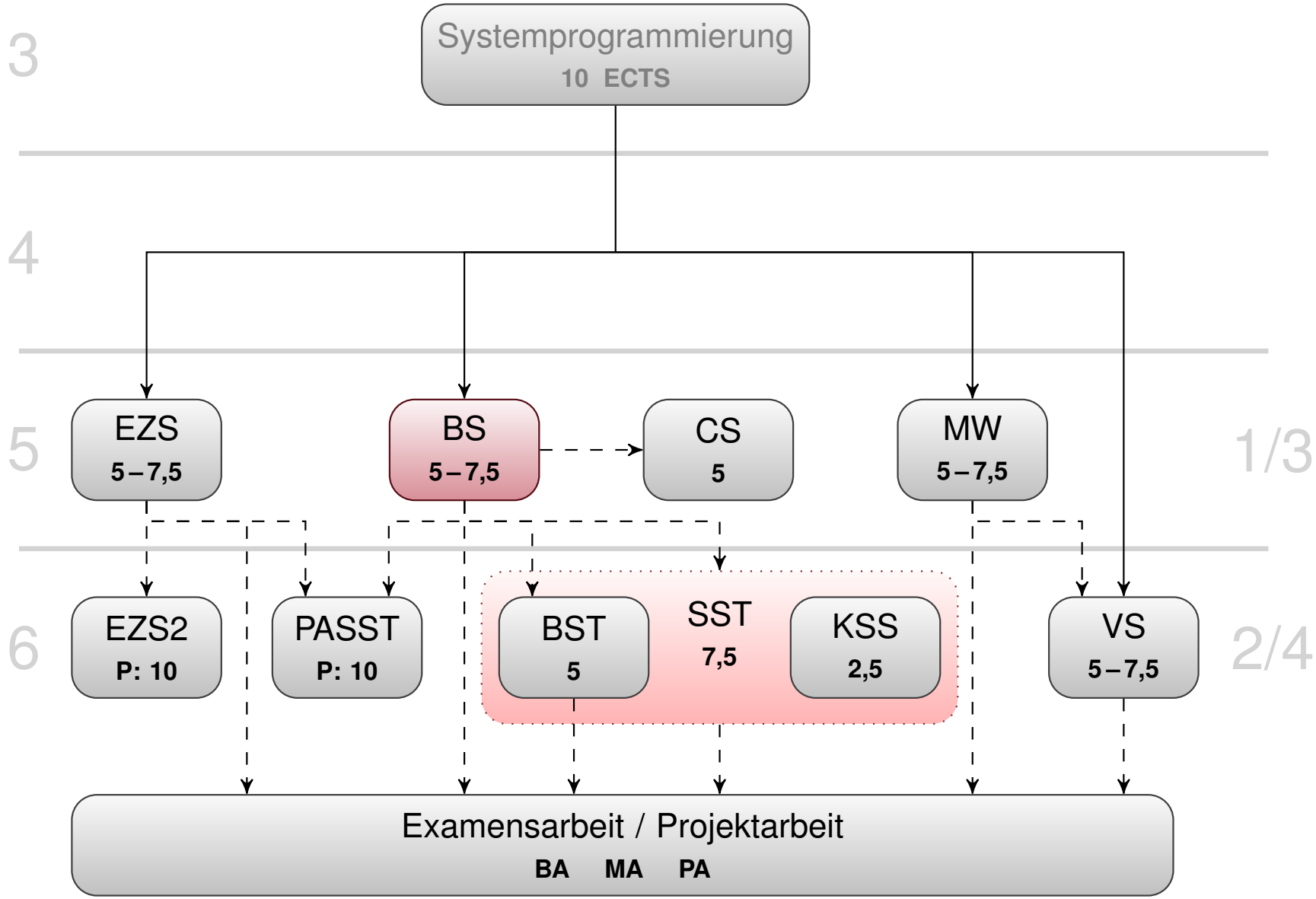
Voraussetzungen

- 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)



Veranstaltungsbestandteile

(5 – 7,5 ECTS Punkte)

VL – Vorlesung 2,5

Vorstellung und detailierte 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

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_1					0.031
19.10.	VL_2	$Ü_1$		$Ü_1$		00.151
26.10.	VL_3	$Ü_2$	RÜ	$Ü_2$	RÜ	0.031
02.11.	VL_4	A_1	RÜ	A_1	RÜ	0.01
09.11.	VL_5	$Ü_3$	RÜ	$Ü_3$	RÜ	0.01
16.11.	VL_6	A_2	RÜ	A_2	RÜ	
23.11.	VL_7	$Ü_4$	RÜ	$Ü_4$	RÜ	
30.11.		A_3	RÜ	A_3	RÜ	
07.12.	VL_8	$Ü_5$	RÜ	$Ü_5$	RÜ	
14.12.	VL_9	A_4	RÜ	A_4	RÜ	
21.12.						
11.01.	VL_{10}	$Ü_6$	RÜ	$Ü_6$	RÜ	
18.01.	VL_{11}	A_5	RÜ	A_5	RÜ	
25.01.	VL_{12}	A_6	RÜ	A_6	RÜ	
01.02.	VL_{13}	$Ü_7$	RÜ	$Ü_7$	RÜ	

Beteiligte Personen

Dozenten Vorlesung

Daniel Lohmann

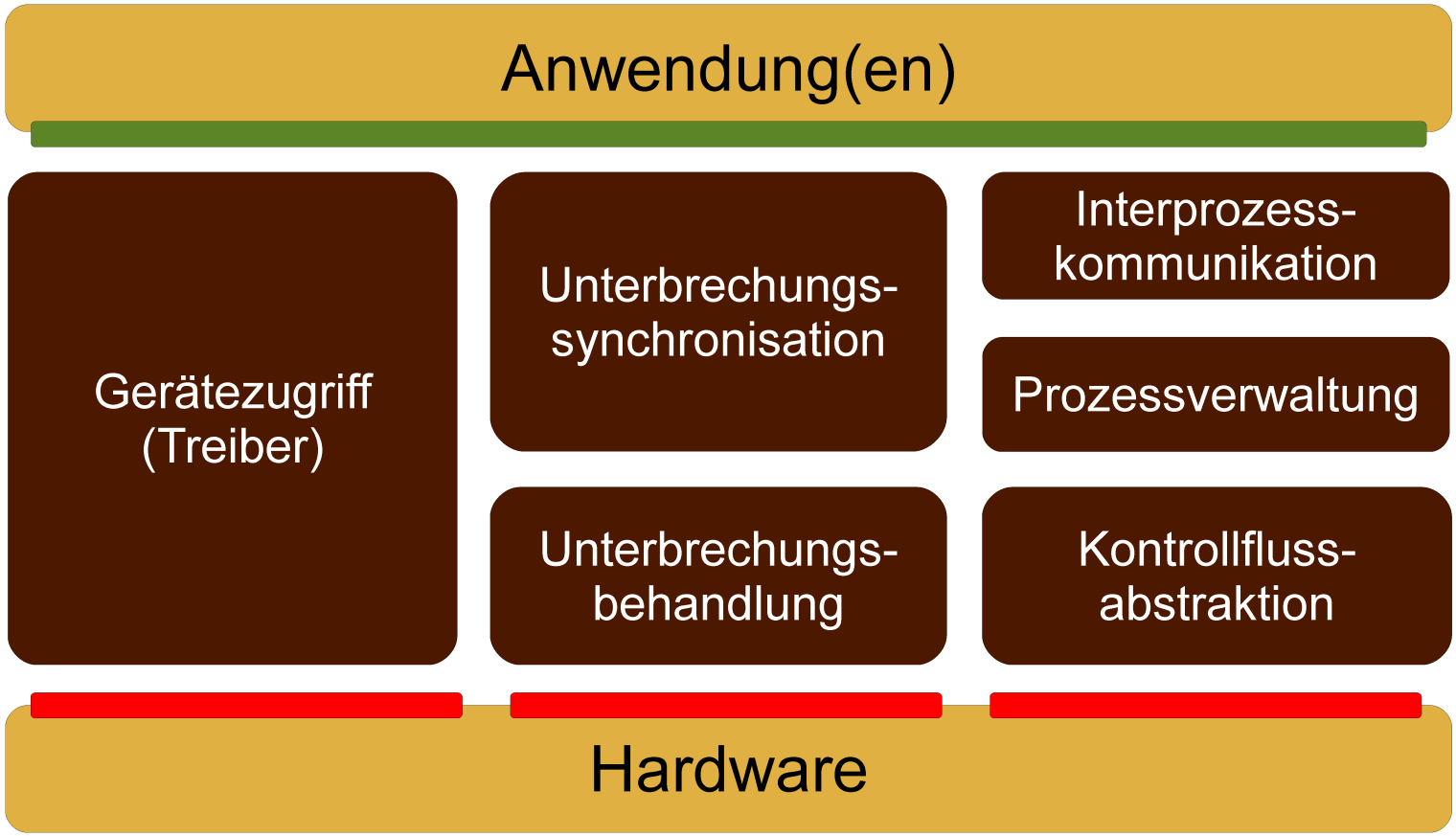
Tafel- und Rechnerübung

Christian Dietrich

Valentin Rothberg

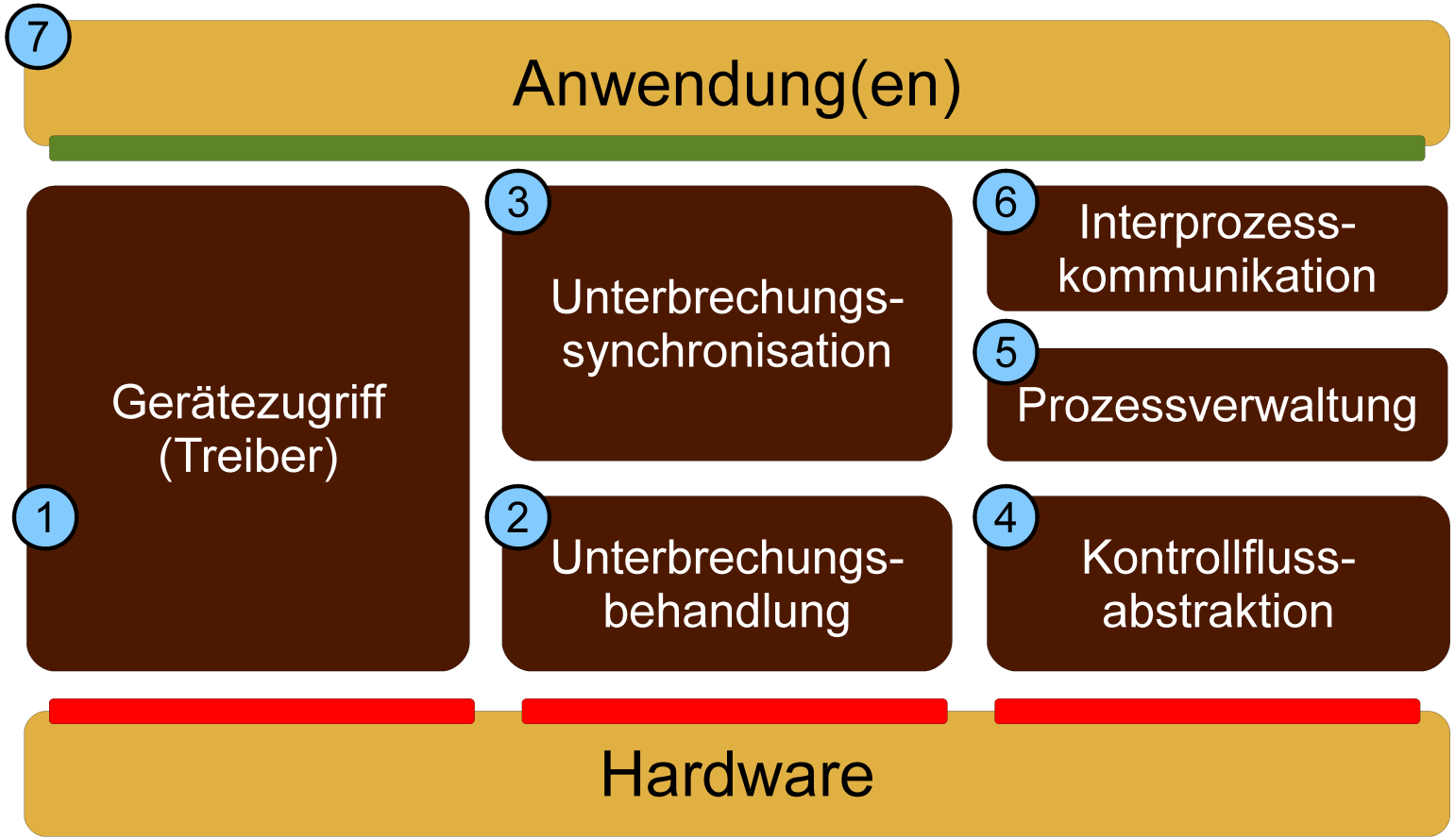
Sebastian Maier

Aufbau eines Betriebssystem



Themenübersicht Übung

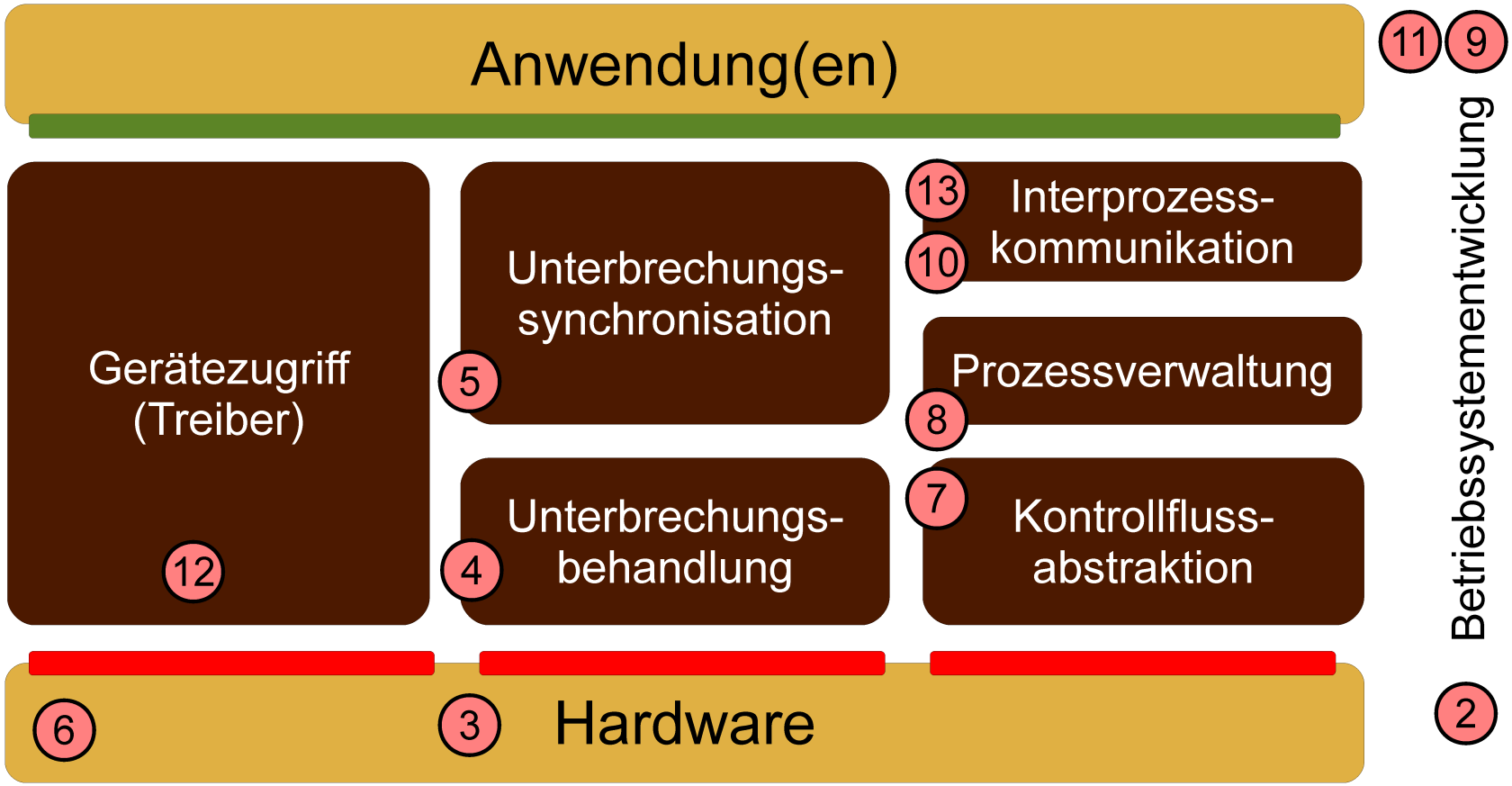
Am Beispiel von: OOStuBS, MPStuBS



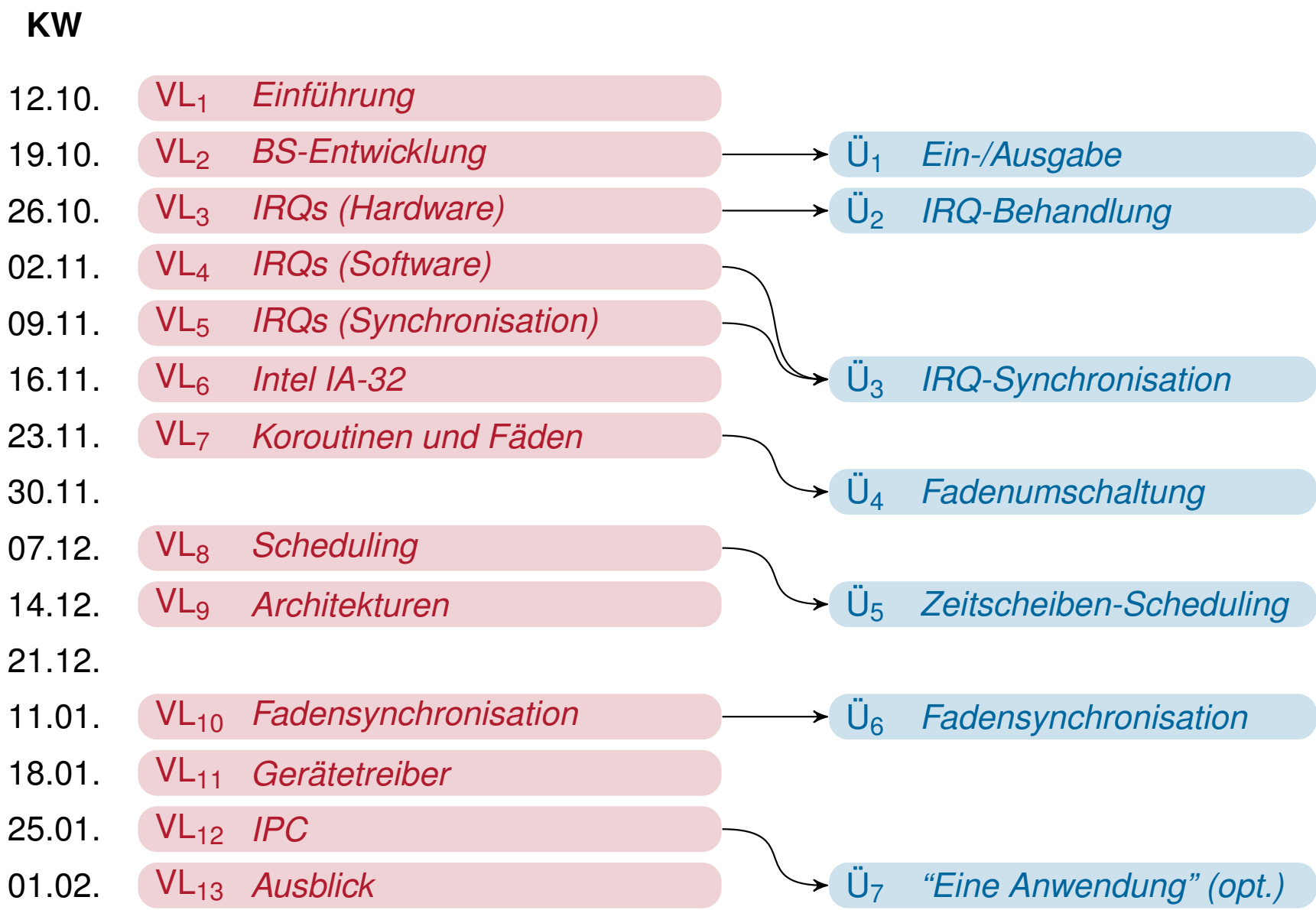
Betriebssystementwicklung

Themenübersicht Vorlesung

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



Verzahnung von Vorlesung und Übungsaufgaben



VL_2 Betriebssystementwicklung (oft ein harter Kampf!)

- Erste Schritte

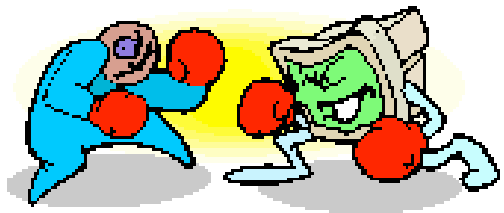
  Wie bringt man sein System auf die Zielhardware?

  - 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_{3-4} 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

VL_5 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_6 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_7 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_8 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_9 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_{10} 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_{11} 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_{12} Interprozesskommunikation (IPC)

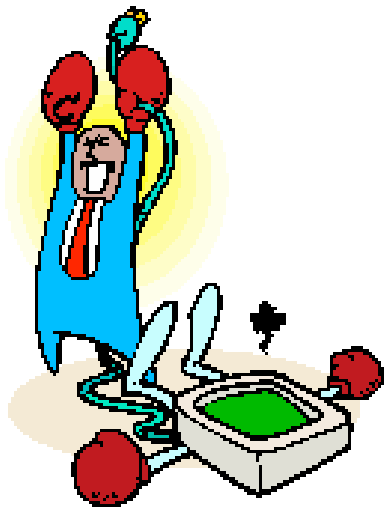
- Grundsätzliches
  - Wechselwirkung $\rightleftharpoons$ Synchronisation
  - implizite und explizite Synchronisation
- Abstraktionen jenseits von *Semaphore*
  - gemeinsamer und verteilter Speicher
  - Fern- und Nahaufrufe
- Dualität nachrichtenbasierter und prozeduraler Systeme
  - konkrete Beispiele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

VL_{13} 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!