

# Überblick

## Systemprogrammierung

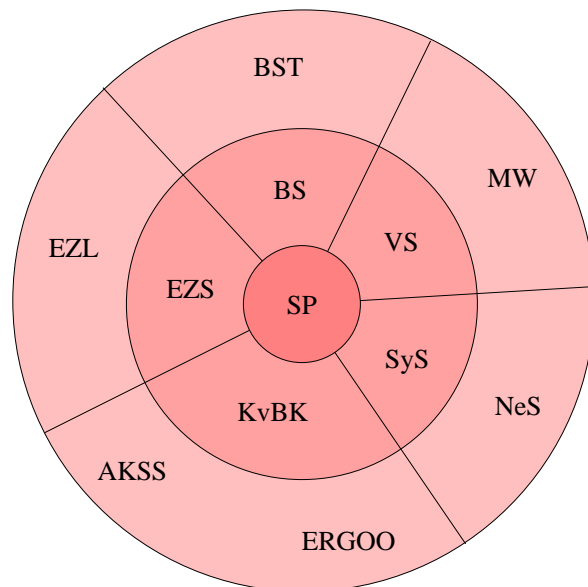
Lehrveranstaltungskonzept

19. April 2010

### Lehrveranstaltungskonzept

Einordnung  
Studiengänge  
Lernziele und Lehrinhalte  
Voraussetzungen  
Vorlesung und Übung  
Leistungsnachweise  
Kontakt

## Lehre@I4



## Lehre@I4: *post SP* — Aufbau und Spezialisierung

BS	Betriebssysteme	V/Ü
VS	Verteilte Systeme	V/Ü
EZS	Echtzeitsysteme	V/Ü
SyS	Systemsicherheit	V/Ü
KvBK	Konzepte von Betriebssystemkomponenten	PS

BST	Betriebssystemtechnik	V/Ü
MW	Middleware	V/Ü
EZL	Echtzeitsystemlabor	IV
NeS	Netzwerksicherheit $\mapsto$ I7	V/Ü
AKSS	Ausgewählte Kapitel der Systemsoftware	HS
ERGOO	Erlangen Research Group on Distributed Objects and Operating Systems	KO

Integrierte Lehrveranstaltung: 8 SWS  $\mapsto$  10 ECTS

$$\text{Termine} \left\{ \begin{array}{ll} \text{Vorlesung} & 2 \\ \text{Übung} & 1 \\ \text{Rechner} & 1 \end{array} \right\} \equiv 4 \times 1,5 = 6 \text{ Zeitstunden wöchentlich}$$

## Vor-/Nacharbeit

- ▶  $N$  Stunden wöchentlich:  $0 \leq N \leq (162 - X)$
- ▶  $X \ll 162$  ist das Zeitstundenäquivalent anderer „Pflichten“

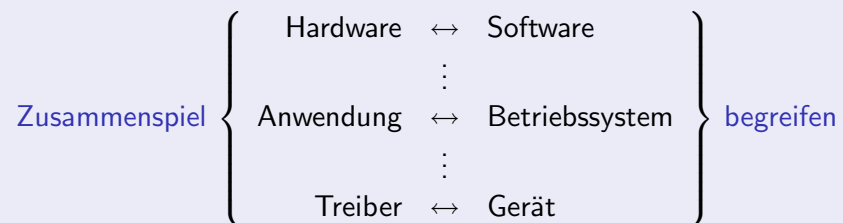
$$\text{Arbeitsaufwand} \sim 30 \times V + 15 \times \ddot{U} + 15 \times P$$

Präsenzzeit 120 h, davon 90 h Kernzeit  $\approx$  6 Wochenstunden

Eigenstudium  $1,5 \times$  Präsenzzeit = 180 h  $\approx$  12 Wochenstunden

## Lernziele

Vorgänge innerhalb von Rechensystemen *ganzheitlich* verstehen:



Programmierung von **Systemsoftware** (in Grundzügen) selbst erleben

- ▶ im Kleinen praktizieren  $\Rightarrow$  **Dienstprogramme**
- ▶ im Großen erfahren  $\Rightarrow$  **Betriebssysteme**

## Kernfach der Informatik

## Diplom/Bachelor

- ▶ Informatik, IuK, CE
- ▶ Wirtschaftsinformatik

## Lehramt Informatik

- ▶ Gymnasium



## Lehrinhalte

**Vorlesung** — Vorstellung und detaillierte Behandlung des Lehrstoffs

- ▶ Organisation (der Systemsoftware) von Rechensystemen
- ▶ Grundlagen von Betriebssystemen
- ▶ maschinennahe Programme

**Übung** — Vertiefung, Besprechung der Übungsaufgaben, Tafelübungen

- ▶ Systemprogrammierung in C
- ▶ Systemprogramme, -aufrufe, -funktionen von UNIX

**Praktikum** — Rechnerarbeit: Programmierung, Fehlersuche/-beseitigung

- ▶ UNIX (Linux), CLI (*shell*), GNU (gcc, gdb, make), vi. . .

## Inhaltsüberblick

### Teil A $\mapsto$ C-Programmierung

1. Organisation
2. Einführung in C
3. Programm  $\mapsto$  Prozess (UNIX)

### Teil B $\mapsto$ Überblick

4. Einleitung
5. Rechnerorganisation
6. Betriebsarten
7. Abstraktionen (UNIX)
8. Zwischenbilanz

### Teil C $\mapsto$ Vertiefung

9. Einplanung
10. Einlastung
11. Synchronisation
12. Verklemmungen
13. Adressräume
14. Arbeitsspeicher
15. Dateisysteme



## Vorlesungsbetrieb und Lehrmaterialien

### Vorlesungstermine ab KW 19

Montag	10:15 – 11:45	0.031
Dienstag	10:15 – 11:45	0.031

### C/UNIX Einführung

- ▶ 6 Vorlesungstermine + 1
- ▶ ab heute: KW 16 – 18

Handzettel (engl. *handout*) sind verfügbar wie folgt:

1. [www4.informatik.uni-erlangen.de/Lehre/SS10/V\\_SP](http://www4.informatik.uni-erlangen.de/Lehre/SS10/V_SP)
2. Gutscheinverkauf zum Bezug von Folienkopien: 1 EUR Schutzgebühr
  - ▶ die Kopien werden vor der Vorlesung ausgegeben

Fachbegriffe der Informatik (Deutsch  $\leftrightarrow$  Englisch)

- ▶ [www.babylonia.ork.uk](http://www.babylonia.ork.uk)
- ▶ [www.inf.fu-berlin.de/inst/ag-ss/montagswort](http://www.inf.fu-berlin.de/inst/ag-ss/montagswort)
- ▶ [www.aktionlebendigesdeutsch.de](http://www.aktionlebendigesdeutsch.de)

## Erforderliche Grundkenntnisse



### Algorithmen & Datenstrukturen $\mapsto$ Grundlagen der Programmierung

- ▶ Datentypen, Kontrollkonstrukte, Prozeduren
- ▶ statische und dynamische Datenstrukturen
- ▶ „Programmierung im Kleinen“

### Technische Informatik $\mapsto$ Grundlagen der Rechnerorganisation

- ▶ „von Neumann Architektur“
  - ▶ Operationsbefehle, Befehlsoperanden, Adressierungsarten
  - ▶ Unterbrechungssteuerung (Pegel kontra Flanke)
  - ▶ Assemblerprogrammierung
- ▶ CPU, DMA, FPU, IRQ, MCU, MMU, NMI, PIC, TLB

## Ergänzende Literatur

-  Brian W. Kernighan and Dennis MacAlistair Ritchie.  
*The C Programming Language*.  
 Prentice-Hall, Inc., second edition, 1988.
-  Jürgen Nehmer and Peter Sturm.  
*Systemsoftware: Grundlagen moderner Betriebssysteme*.  
 dpunkt.Verlag GmbH, zweite edition, 2001.
-  Abraham Silberschatz, Peter Baer Galvin, and Greg Gagne.  
*Operating System Concepts*.  
 John Wiley & Sons, Inc., sixth edition, 2001.
-  Andrew Stuart Tanenbaum.  
*Structured Computer Organization*.  
 Prentice-Hall, Inc., fourth edition, 1999.

## Übungsbetrieb

**Tafelübung:** Beginn in KW 17 (Montag, 26.04.)

- ▶ Anmeldung über **WAFFEL**<sup>1</sup> (URL siehe Webseite von SP)
  - ▶ Freischaltung erfolgt nach der Vorlesung, heute gegen 12 Uhr
- ▶ Übungsplätze werden FCFS<sup>2</sup> vergeben
  - ▶ unterbelegte Termine werden ggf. gestrichen
  - ▶ überbelegte Termine erhalten ggf. mehr Ressourcen
- ▶ Übungsaufgaben sind teilweise in Zweiergruppen zu bearbeiten
  - ▶ Übungspartner müssen für dieselbe Tafelübung angemeldet sein

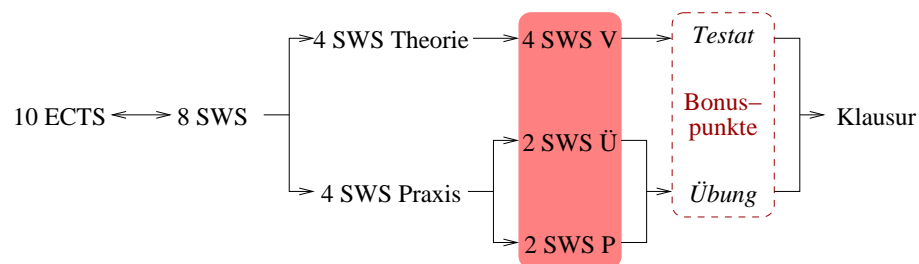
**Rechnerübung:** Anmeldung ist nicht erforderlich

- ▶ die Teilnahme ist optional, je nach Bedarf
  - ▶ es können auch mehrere Termine pro Woche wahrgenommen werden
- ▶ Übungsleiter stehen bei Fragen zur Verfügung

<sup>1</sup>Abk. für Webanmeldefrickelformular Enterprise Logic

<sup>2</sup>Abk. für engl. *first come, first served*, Einplanung nach Ankunftszeit

## Studien- und Prüfungsleistungen



1. **Klausur:** schriftliche Prüfung von 120 Minuten
  - ▶ Einbringung von Bonuspunkten zur Notenverbesserung möglich
2. **Bonuspunkte:** bei erfolgreicher Übungsteilnahme
  - ▶ ergeben sich anteilig aus Testat und Bewertung der Übungsaufgaben

## Bedeutung von Tafel- und Rechnerübungen

**Tafelübungen**  $\leadsto$  „*learning by exploring*“

- ▶ Besprechung der Übungsaufgaben, Skizzierung von Lösungswegen
- ▶ Vertiefung des Vorlesungsstoffes, Klärung offener Fragen

**Rechnerübungen**  $\leadsto$  „*learning by doing*“

- ▶ selbständiges Bearbeiten der Übungsaufgaben am Rechner
- ▶ Hilfestellung beim Umgang mit den Entwicklungswerkzeugen
- ▶ der Rechner ist **kein Tafelersatz**, die Betreuung verläuft eher passiv

*Der, die, das.  
Wer, wie, was?  
Wieso, weshalb, warum?  
Wer nicht fragt, bleibt dumm!*



## Testat und Schein

Testat  $\sim$  Ex  $\mapsto$  **Miniklausur:** optional, „beliebiger“ Vorlesungstermin

- ▶ geprüft wird Stoff von Vorlesung und Übung
  - ▶ Fragen zu **Teil A** und **Teil B** der Vorlesung
  - ▶ Trockenübung in der Programmiersprache C
- ▶ Mehrfachauswahl (engl. *multiple choice*), 45 Minuten

**Schein:** obligatorisch, erfolgreiche Teilnahme an Übung und Praktikum

- ▶ 50 % der Punkte aus „Übungsaufgaben + Ex“ sind zu erreichen
- ▶ Punkte darüberhinaus gehen in die Bonuspunktberechnung ein

**Bonuspunkte für die Klausur ansammeln**

- ▶ max. 10 % der Klausurpunkte erreichbar  $\approx$  12 Punkte
- ▶ Erfahrungswert:  $\varnothing$  5.5 Klausurpunkte pro Notenstufe 0.3
- ▶ Verbesserung der Klausurnote um bis zu zwei Stufen möglich

## Bonuspunkte und Klausur

**Bonuspunkte:** nur auf Basis des (unbenoteten) Scheins aus SS 2010

- ▶ beeinflussen die angelegte Notenskala nicht, werden jedoch bei bestandenen Klausuren auf die Klausurpunkte aufaddiert
- ▶ können die Note einer bestandenen Klausur verbessern, nicht jedoch den Ausschlag zum Bestehen der Klausur geben
- ▶ Erreichen der Bestehensgrenze muss also immer mit regulären Klausurpunkten erfolgen

**Klausur:** Termin steht noch nicht fest

- ▶ Struktur analog Testat, zusätzlich Fragen zu **Teil C** der Vorlesung

Präsenz und **aktive Mitarbeit** machen die Klausur „leicht“

- ▶ Programme im Team entwickeln, aber selbst zum Laufen bringen

## Fragen...

# 42

## [www4.informatik.uni-erlangen.de/](http://www4.informatik.uni-erlangen.de/)\*

### Dozenten

- ▶ Jürgen Kleinöder (~jklein)
- ▶ Wolfgang Schröder-Preikschat (~wosch)

### Mitarbeiter

- ▶ Michael Stilkerich (~stilkerich)
- ▶ Peter Ulbrich (~ulbrich)

### Tutoren

- ▶ Christoph Erhardt, Jens Schedel

