

Echtzeitsysteme

Übungen zur Vorlesung

Fabian Scheler, Peter Ulbrich, Niko Böhm

Friedrich-Alexander-Universität Erlangen-Nürnberg
Lehrstuhl Informatik 4 (Verteilte Systeme und Betriebssysteme)
www4.informatik.uni-erlangen.de

26. Oktober 2009

Wer?

► **Vorlesung:**

Prof. Dr. Wolfgang Schröder-Preikschat

(wosch@informatik.uni-erlangen.de,
www4.informatik.uni-erlangen.de/~wosch)

► **Übung:**

Fabian Scheler

(scheler@informatik.uni-erlangen.de,
www4.informatik.uni-erlangen.de/~scheler)

Peter Ulbrich

(ulbrich@informatik.uni-erlangen.de,
www4.informatik.uni-erlangen.de/~ulbrich)

Niko Böhm

(boehm@informatik.uni-erlangen.de,
www4.informatik.uni-erlangen.de/~boehm)

Wann und wo?

- ▶ Vorlesung: Montag 14:15 - 15:45 H4
- ▶ Übung:
 - ▶ Tafelübung:

Montag	10:00 - 12:00	0.031
Dienstag	14:00 - 16:00	00.156
Donnerstag	12:00 - 14:00	00.156
 - ▶ Rechnerübung
 - ▶ Donnerstag 10:00 - 12:00 02.151a NON-Graphik
 - ▶ nach Vereinbarung in der manlobbi

Prüfungsmöglichkeiten

Diplomstudiengänge Informatik, IuK, Mechatronik, Maschinenbau ...

unbenoteter Schein – alle Studiengänge

- ▶ ausreichende Bearbeitung aller Übungsaufgaben
 - ▶ Aufgaben der erweiterten Übungen
- ▶ regelmäßige und aktive Teilnahme am Übungsbetrieb
- ▶ ggf. Rücksprache über die Scheinvergabe (bei der ungenügender Bearbeitung einer oder mehrerer Übungsaufgaben)

benoteter Schein – Informatik

- ▶ unbenoteter Schein
- ▶ Rücksprache (Termin per eMail vereinbaren!)

Prüfung – Informatik, **ben. Schein** – IuK, Mechatronik, ...

- ▶ mündliche Prüfung
- ▶ Inhalt ist Vorlesung **und** Übung
- ▶ Einzelprüfung (Termin per eMail vereinbaren!)

Prüfungsmöglichkeiten

Bachelor/Master-Studiengänge ...

unbenoteter Schein

- ▶ nicht mehr relevant
- ▶ jedes Modul schließt mit einer **benoteten** Leistung!

benoteter Schein

- ▶ Vorlesung (2,5 ECTS)
 - ▶ mündliche Prüfung über die Vorlesung
- ▶ Vorlesung + grundlegende Übungen (5 ECTS)
 - ▶ Bearbeitung der Aufgaben für die grundlegenden Übungen
 - ▶ mündliche Prüfung
- ▶ Vorlesung + erweiterte Übungen (7,5 ECTS)
 - ▶ Bearbeitung der Aufgaben für die grundlegenden Übungen
 - ▶ Bearbeitung der Aufgaben für die erweiterten Übungen
 - ▶ mündliche Prüfung
- ▶ Allgemein
 - ▶ Termin für die mündliche Prüfung per eMail vereinbaren!
 - ▶ Entscheidung für eine Variante ist **verbindlich**!

Anmeldung/Ressourcen/Fragen

- ▶ **Anmeldung über das Waffel:**

`https://waffel.informatik.uni-erlangen.de/signup/?course=52`

- ▶ **Folien, Hinweise, Aufgabenstellungen, Dokumentation ... zu Vorlesung und Übung:**

- ▶ **Im Internet**

`www4.informatik.uni-erlangen.de/Lehre/WS09/V\_EZS/`

- ▶ **Im CIP-Pool**

`/proj/i4ezs`

- ▶ **Subversion URL**

`https://www4.informatik.uni-erlangen.de:8088/i4ezs/<group\_name>`

- ▶ **Fragen an die Mailingliste**

`i4ezs@immd4.informatik.uni-erlangen.de`

Übungsaufgaben

Zielsetzung

- ▶ grundlegende Übungsaufgaben
 - ▶ Verständnis der Funktionsweise
 - ☞ Entwicklung von Anwendungen/Testfällen
 - ☞ Beobachtung von Eigenschaften dieser Anwendungen
- ▶ erweiterte Übungsaufgaben
 - ▶ Verständnis der Funktionsweise und der Implementierung
 - ☞ siehe oben
 - ☞ Implementierung von Teilen eines Echtzeitbetriebssystems

Themen

Typische in Echtzeitsystemen relevante Aufgabenstellungen:

- ▶ Schedule-Table Scheduler
- ▶ MLQ-Scheduler
- ▶ Aperiodic Task Support
- ▶ Thread Synchronization
- ▶ Deferrable Server

Aufgabenstellungen

grundlegende Übungen

Vorgaben sind (kleine) Echtzeitbetriebssysteme

- ▶ vollständig $\leadsto$ bauen nicht aufeinander auf

Aufgaben sind *Systembeschreibungen* und *Fragen*

- ▶ in Textform $\leadsto$ PDF
- ▶ Bearbeitung erfolgt teilweise
 - ▶ als Implementierung
 - ▶ Erstellen einer Anwendung zur Systembeschreibung
 - ▶ Beobachtung/Messung bestimmter Phänomene
 - ▶ schriftlicher Beantwortung von Fragen
 - ▶ Welche Probleme treten bei der Implementierung auf?
 - ▶ Warum sind gewisse Phänomene zu beobachten?
 - ▶ Beantwortung der Fragen innerhalb der Testanwendung

Aufgabenstellungen

erweiterte Übungen

Vorgaben sind Gerüste:

- ▶ Sie sollen bei der Orientierung behilflich sein!
- ▶ Sie sind keine **starre Schablonen**!

Aufgabe ist es diese Gerüste funktional zu vervollständigen

- ▶ Implementierung bestimmter Betriebssystemmechanismen

Schnittstellen sind **verbindlich**!

- ▶ Diese werden in der Tafelübung explizit angesprochen!

Gruppenarbeit und Abgabe

Gruppen

- ▶ Die Bearbeitung der Aufgaben erfolgt in 2er-Gruppen
- ▶ jeweils ein Übungspartner implementiert
 - ▶ die funktionale Aufgabenstellung
 - ▶ entsprechende Testfälle
 - ▶ damit teilweise bereits die Aufgabenstellungen der grundlegenden Übungen
- ▶ gemeinsame Beantwortung der Verständnisfragen
- ▶ Gruppeneinteilung per Mail an
`scheler@informatik.uni-erlangen.de`

Abgabe

Erstellen eines entsprechenden Subversion-Tags durch

```
make abgabe EXERCISE=aufgabe{1,2,3,4,5}
```

Übungsaufgaben - Ablauf

1. Ausgabe der Vorgabe/Aufgabenstellung (Montag 16:30)
2. Besprechung der Aufgabenstellungen
3. während der Bearbeitung (nur erweiterte Übungen)
 - ▶ beliebig viele Abgaben möglich
 - ▶ Abgaben werden getestet, das Ergebnis per Mail mitgeteilt
 - ▶ Referenztestfälle werden **nicht** herausgegeben
4. Abgabe ist ca. 2,5 Wochen später (immer Freitags 12:00)
5. Abgaben werden getestet bzw. korrigiert
 - ▶ alle Testfälle erfolgreich $\leadsto$ Aufgabe erfolgreich
 - ▶ Testfälle fehlgeschlagen $\leadsto$ Fehlerbehebung
 - ▶ Herausgabe der Referenztestfälle
6. erneuter Test der Abgabe (nur erweiterte Übungen)
 - ▶ 1 Woche später, sofern die Abgabe erneuert wurde
 - ▶ alle Testfälle erfolgreich $\leadsto$ Aufgabe erfolgreich
 - ▶ Testfälle fehlgeschlagen $\leadsto$ Aufgabe nicht erfolgreich
 - ▶ Test-Arena

Übungsaufgaben - Test-Arena

1. Testen der Testfälle gegen eine Referenzimplementierung
 - ▶ Referenzimplementierung gilt als fehlerfrei
 - ▶ Sind die Testfälle *sinnvoll*?
2. Testen: jeder gegen jeden
 - ▶ jede Implementierung wird gegen jeweils alle Testfälle aller anderen Gruppen getestet
3. Ergebnis:
 - ▶ Gruppe abc findet n Fehler in der Abgabe der Gruppe xyz
 - ▶ Rangliste - Sortierung nach irgendeinem Kriterium, z.B.:
bei anderen gefundene Fehler – eigene Fehler