

University of Karlsruhe
System Architecture Group
Prof. Dr. Jochen Liedtke

Tutors:
Christian Ceelen
Uwe Dannowski
Kevin Elphinstone
Felix Hupfeld
Gerd Liefländer
Espen Skoglund

System Architektur (*Architecture*)

Klausur (*Examination*)

WS 2000/2001, 20. April 2000

- Bitte tragen Sie zuerst auf allen Klausurblättern Ihren Namen, Vornamen und Ihre Matrikelnummer ein, auch auf den Konzeptblättern. *Please enter your last name, first name and matriculation number on each page (including used and unused draft pages).*
- Die Prüfung dauert 60 Minuten und besteht aus 5 Aufgaben auf 11 Seiten und zwei Konzeptblättern. *You have 60 minutes to complete your answers. The examination consists of 5 questions on 11 pages. You have received two additional blank pages for drafts, etc.*
- Die Prüfung ist mit mindestens 20 Punkten von 60 erreichbaren Punkten bestanden. *You pass the examination by obtaining at least 20 marks out of the possible 60 marks.*
- Es sind keinerlei Hilfen erlaubt! *No additional means are allowed!*
- Die Prüfung gilt als nicht bestanden, wenn Sie versuchen, aktiv oder passiv zu betrügen. *You fail the examination if you try to cheat actively or passively.*
- Wenn Sie zusätzliches Konzeptpapier benötigen, verständigen Sie bitte die Klausuraufsicht. *If you need more draft pages please notify one of the supervisors.*
- Bitte machen Sie eindeutig klar, was Ihre endgültige Lösung zu den jeweiligen Teilaufgaben ist. Teilaufgaben mit mehreren Lösungen oder mit widersprüchlichen Teilen werden mit **0 Punkten** bewertet. *Make sure that it is absolutely clear what your final solution is for each subquestion. Subquestions with multiple solutions or with contradicting parts are void: 0 marks.*

Die folgende Tabelle wird von uns ergänzt! *The below table will be completed by us.*

Aufgabe/Question	1	2	3	4	5	Total
Erreichbare Punkte <i>Possible marks</i>	12	12	12	12	12	60
Erreichte Punkte/ Obtained marks						
Note/Grade:						

Nachname/ <i>Last name</i>	Vorname/ <i>First name</i>	Matrikelnummer/ <i>Matriculation number</i>

Aufgabe/Question 1 (Zum Aufwärmen/*Warm up*, 3 + 3 + 1 + ... + 1 Punkte/*marks*)

1. „Zählen Sie **drei** verschiedene **Typen** von **Sicherheitsbedrohungen** für Systeme auf und geben **jeweils** Sie **ein konkretes Beispiel** an!“

*“Enumerate **three** different **types of security threats** for systems and give **a concrete example for each** of them.”*

a)

.....

b)

.....

c)

.....

2. „Zählen Sie **drei** wesentliche **Abstraktionen** der Systemarchitektur auf!“

*“Enumerate **three** basic **abstractions** of system-architecture.”*

a)

b)

c)

Einige der folgenden Aussagen sind korrekt, einige inkorrekt. **Unterstreichen** Sie „korrekt“, wenn die Aussage korrekt ist, unterstreichen Sie „inkorrekt“, wenn die Aussage inkorrekt ist.

*Some of the following statements are correct, some are incorrect. **Underline** “korrekt” if the statement is correct; underline “inkorrekt” if the statement is incorrect!*

3. „Java bietet zur **Synchronisation** von Threads das **Konzept Semaphore** an.“

*“To **synchronize** threads, **Java** offers the **concept of semaphores**.”*

korrekt

inkorrekt

Fortsetzung von Aufgabe 1 / Question 1 continued: (1+1+1+1+1 Punkte/marks)

4. „Ein NFS-Dateidienstgeber (*server*) ist zustandslos (*stateless*).“
“*An NFS-file-system server is a stateless server.*”
korrekt inkorrekt
5. „Skalierbarkeit ist ein wesentliches Merkmal von Clusterarchitekturen.“
“*Scalability is a essential characteristic of cluster-architectures.*”
korrekt inkorrekt
6. „Auf einem System mit einem **Mikrokern der zweiten Generation** sind abgesehen vom Mikrokern **alle sonstigen Systeminstanzen auslagerbar**.“
“*On a system based upon a second generation μ -kernel, apart from the μ -kernel all other system tasks can be swapped out.*”
korrekt inkorrekt
7. „Ein wesentlicher **Nachteil jeder zentralisierten Lösung** in einem verteilten System ist ihre **eingeschränkte Leistung**.“
“*One important drawback of any centralized solution within an distributed system is its limited performance.*”
korrekt inkorrekt
8. „Ein Thread, der **unendlich lange** auf das Eintreffen einer Nachricht wartet, wobei er eine **synchrone IPC-Operation *receive_message()*** benutzt hat, **ist verklemmt (*deadlocked*)**.“
“*A thread waiting forever for an incoming message having used a synchronous IPC-operation receive_message() is deadlocked.*”
korrekt inkorrekt

Nachname/ <i>Last name</i>	Vorname/ <i>First name</i>	Matrikelnummer/ <i>Matriculation number</i>

Aufgabe/Question 2**(5 + 2 + 5 Punkte/marks)**

1. „Gegeben sei ein symmetrisches Mehrprozessorsystem (SMP) aus 4 Prozessoren, die alle über den Systembus an den gemeinsamen Hauptspeicher (*main memory*) angeschlossen sind. Pro Prozessor existiere ein L1- und ein L2-Pufferspeicher (*cache*). Auf Prozessor CPU₄ läuft ein Schachprogramm mit einer großen Arbeitsmenge (*working set*), die nicht vollständig in den L2-Cache paßt. Dieses Schachprogramm läuft unabhängig von allen sonstigen Anwendungen. Drei kooperierende Threads mit jeweils mittelgroßer Arbeitsmenge (*working set*), d.h. jede von ihnen paßt vollständig in einen L2-Cache, laufen echt parallel auf den Prozessoren CPU₁ bis CPU₃. Diese drei kooperierenden Threads synchronisieren sich mittels der unten skizzierten „**Spinlockimplementierung**“:

“Given a symmetric multi-processor system (SMP) consisting of 4 processors operating on a shared main memory via the system-bus. L1 and L2 caches exist per processor. Processor CPU₄ executes a chess program with a large working set that does not fit into its L2 cache. This chess program runs independently of all other applications. Three cooperating threads with medium sized working sets, i.e. each working set fits into an L2-cache, run in parallel on processors CPU₁ through CPU₃. The three cooperating threads synchronize by means of the **spinlock implementation** described below:

DO

```
reg := MyThreadId ;
```

```
xchg (SpinLock, reg) {special atomic exchange operation}
                    {exchanges mem/cache variable SpinLock and reg}
```

UNTIL reg = 0 **OD**;

```
...
... critical section ...
...
```

SpinLock := 0

```
{SpinLock ≠ 0 ⇔ another Thread is in its critical section}
{SpinLock = 0 ⇔ critical section is free }
```

Das Schachprogramm läuft auf diesem SMP-System **deutlich langsamer** als auf einem Einprozessorsystem. **Modifizieren** Sie die **Implementierung** des Spinlock, so daß das **Schachprogramm nicht mehr so stark behindert** wird!“

*The chess program runs **substantially slower** in this SMP-system than on a single-processor system. **Modify** the **spinlock implementation** so that the **chess program is not as hampered as before.**”*

Fortsetzung von Aufgabe 2 / Question2 continued:**(2 + 5 Punkte/marks)**

2. „Analysieren Sie die **maximale Wartezeit** für einen der drei kooperierenden Threads vor seinem kritischen Abschnitt, wenn die verbesserten Spinlocks aus Teilaufgabe 2.1 verwendet werden.“

*“Analyze the **maximal waiting-time** for one of the three cooperating threads in front of its critical region using the improved spin-locks of subquestion 2.1.”*

Maximale Wartezeit (*maximal waiting time*):

Begründung (reasons):

.....

.....

3. „Diese Teilaufgabe ist unabhängig vom konkreten Beispiel aus Teilaufgabe 2.1. und 2.2. Das Betriebssystem bietet an seiner Anwendungsprogrammierschnittstelle (**API**) **Spinlocks** und **binäre Semaphore** an. Welche **Probleme** können auftreten, wenn ein Programmierer einer „vielfädigen Applikation“ (*multi-threaded application*) einen Spinlock statt einer binären Semaphore zur Lösung des **wechselseitigen Ausschluß** verwendet.

Hinweis: Diese Applikation soll auf Ein- und Mehrprozessorsystemen lauffähig sein; ferner soll sie unabhängig von einer spezifischen Schedulingstrategie sein. “

*“This subquestion is independent of the specific example in subquestions 2.1. and 2.2. The operating system offers **spinlocks** as well as **binary semaphores** at its **API**. What **problems** might arise, when a programmer of a multi-threaded application uses a spinlock instead of a binary semaphore in order to solve **mutual exclusion**?*

Hint: *This application has to run on single- and multi-processor systems; furthermore it should be independent of a specific scheduling policy.”*

“Sie können auch die Rückseite benutzen.“ “You can also use the reverse side.”

Nachname/ <i>Last name</i>	Vorname/ <i>First name</i>	Matrikelnummer/ <i>Matriculation number</i>

Aufgabe 3 / Question 3**(3 + 3 + 6 Punkte/marks)**

1. „Analysieren Sie das folgende Programm! Stellt es eine **korrekte Lösung** für das **Produzent-/Konsumentproblem** mit einem **zyklischen Puffer** der **Kapazität b** dar? Begründen Sie Ihre Meinung!

Hinweis: Der Puffer sei als **zusammenhängendes Feld (array)** realisiert, zwei Zeiger unterstützen das Einfügen und Entfernen von Datenelementen (*items*) in den und aus dem Puffer. “

“*Analyze the following program. Is it a correct solution for the bounded producer-consumer problem using a b-slot cyclic buffer? Clarify your conclusion.*”

Hint: *The buffer is an array, two pointers support inserting and removing items into and out of the buffer.*”

```
lock: semaphore := 1;      {mutual exclusion on the buffer}
empty_slots: semaphore := b;
full_slots: semaphore := 0;
```

```
{producer code}
```

```
{consumer code}
```

```
repeat
  produce item;
  P(lock);
  P(empty_slots);
  insert item;
  V(full_slots);
  V(lock);
forever
```

```
repeat
  P(lock);
  P(full_slots);
  remove item;
  V(empty_slots);
  V(lock);
  consume item;
forever
```

.....

.....

.....

2. „Wenn man in obiger Programmskizze die Semaphore **lock** nebst ihren Schnittstellenoperationen **P(lock)** und **V(lock)** ganz wegläßt, wie sieht dann das Analyseergebnis aus? (Gleiche Pufferrimplementierung wie in 3.1.)“

“*What’s the result of the analysis, if you leave out the semaphore **lock** as well as its interface-operations **P(lock)** and **V(lock)** in the above program? (Same buffer implementation as in 3.1.)*”

.....

.....

Fortsetzung von Aufgabe 3 / *Question 3 continued:*

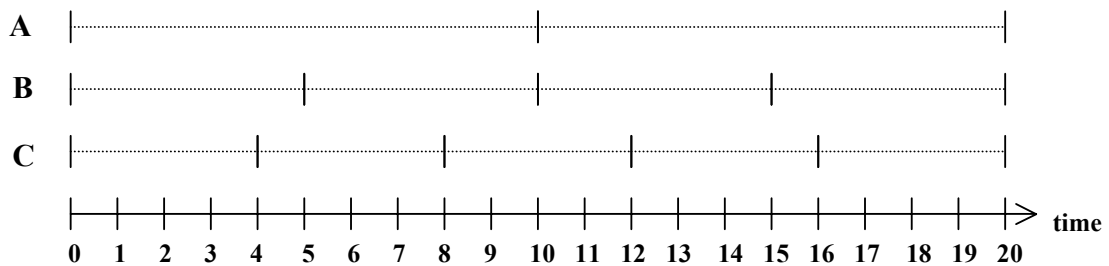
(6 Punkte/marks)

3. „Die Algorithmen „**Rate-Monotonic (RM)**“ und „**Earliest Deadline First**“ (EDF) sind zwei im Echtzeitbereich populäre nicht verdrängende (*nonpreemptive*) Schedulingstrategien. Beschreiben Sie beide Verfahren! Illustrieren Sie deren Wirkungsweise am folgenden Schedulingproblem aus drei periodischen Echtzeitthreads! Sie können annehmen, daß Threadumschaltungen (*thread switches*) unendlich schnell sind.“

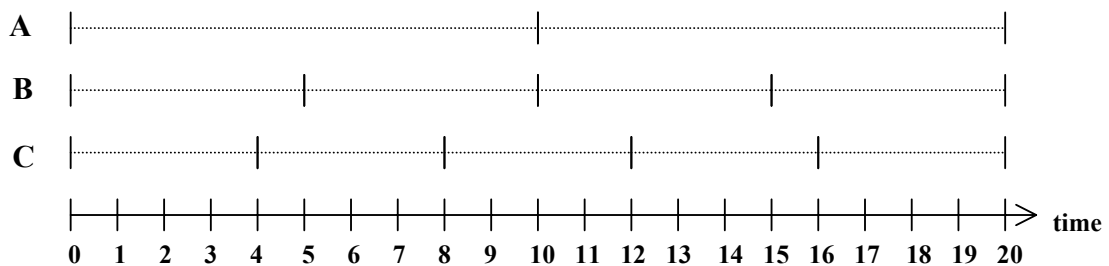
“Rate monotonic (RM) and earliest deadline first (EDF) are two popular nonpreemptive scheduling policies for real-time systems. Describe both algorithms. Illustrate your answer by showing how each of them would schedule the following set of threads. You may assume that thread switches are instantaneous.”

Thread	Exakte Bearbeitungszeit pro Periode (Requires exactly per period)	Periodenlänge (every)
A	2 ms	10 ms
B	3 ms	5 ms
C	1 ms	4 ms

Rate-Monotonic:



Earliest Deadline First:



Nachname/<i>Last name</i>	Vorname/<i>First name</i>	Matrikelnummer/ <i>Matriculation number</i>

Nachname/<i>Last name</i>	Vorname/<i>First name</i>	Matrikelnummer/ <i>Matriculation number</i>

Nachname/<i>Last name</i>	Vorname/<i>First name</i>	Matrikelnummer/ <i>Matriculation number</i>

Einverständnis/Agreement:

Ich stimme zu, daß mein Klausurergebnis in einer durch ein Passwort geschützten Ergebnisliste im Web bekanntgegeben wird. Die Liste wird Einträge der Form: **Matrikelnummer, erzielte Punkte und Note** enthalten.

*I agree that my results of this examination will be published on the web in a password protected web page. The web page will contain entries of the form: **matriculation number, obtained marks, and grade.***

Unterschrift/*Signature*: