



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

Tutors:
Uwe Dannowski
Gerd Liefänder
Espen Skoglund

System Architektur

Freiwillige Zwischenprüfung

WS 1999/2000

10. 12. 1999

Nachname	Vorname	Matrikelnummer

- Bitte tragen Sie zuerst auf allen Klausurblättern Ihren Namen, Vornamen und Ihre Matrikelnummer ein, auch auf dem Konzeptblatt.
- Die Prüfung dauert 60 Minuten und besteht aus 5 Aufgaben auf 6 Seiten und einem Konzeptblatt.
- Sie hätten die Prüfung mit mindestens 20 Punkten von maximal 60 erreichbaren Punkten bestanden.
- Es sind keinerlei Hilfen erlaubt!
- Die Prüfung gilt als nicht bestanden, wenn Sie bei einem aktivem oder passiven Betrugsversuch erwischt werden.
- Wenn Sie zusätzliches Konzeptpapier benötigen, verständigen Sie bitte die Klausuraufsicht.
- Bitte machen Sie eindeutig klar, was Ihre endgültige Lösung zu den jeweiligen Teilaufgaben ist. Wir bewerten keine Teilaufgabe, zu der mehr als eine Lösung vorgeschlagen wird.

Die folgende Tabelle wird von uns bei der Korrektur ausgefüllt.

Aufgabe	1	2	3	4	5	Total
Erreichbare Punkte/Aufgabe						60
Erreichte Punkte/Aufgabe						
					Note:	

Nachname	Vorname	Matrikelnummer

Aufgabe 1

(Zum Aufwärmen, 12 Punkte)

Einige der folgenden Aussagen sind wahr, einige falsch. Markieren Sie den Wahrheitswert mit einem Kreuz an der richtigen Stelle.

Einige der folgenden Aussagen sind unvollständig, ergänzen Sie die fehlenden Stellen!

1. Unabhängig von der Schedulingstrategie können *spin locks* in einem Einprozessorsystem zum wechselseitigen Ausschluß (*mutual exclusion*) für sehr kurze kritische Abschnitte (*short critical sections*) verwendet werden.

Wahr:

Falsch:

2. Eines der Ziele beim Zeitscheibenverfahren (*time-slice mechanism*) ist es zu verhindern, daß ein Thread den Prozessor monopolisieren kann.

Wahr:

Falsch:

3. Das Transaktionskonzept sollte die ACID-Eigenschaften erfüllen, wobei die Abkürzungen ACID folgende Bedeutung im Englischen haben:

A =

C =

I =

D =

4. Das normale Zweiphasensperrprotokoll erzeugt maximale Nebenläufigkeit (*concurrency*), „konflikt serialisierbare“ (*conflict serializable*) und wiederaufsetzbare (*recoverable*) Pläne (*schedules*), kann aber zu Verklemmungen (*deadlocks*) führen.

Wahr:

Falsch:

5. Tragen Sie die vier notwendigen Bedingungen für eine Verklemmung (*deadlock*) ein!

1.

2.

3.

4.

6. Das Buddysystem kann sowohl zum internen (*internal*) als auch zum externen (*external*) Verschnitt (*fragmentation*) führen.

Wahr:

Falsch:

7. Nur in Speicherverwaltungssystemen mit verzögerter Freispeichervereinigung (*lazy reunification*) macht eine Speicherbereinigung (*garbage collection*) Sinn.

Wahr:

Falsch:

8. Sie geben einen Speicherblock *rb* der Größe 2^k in einem Buddysystem frei. Um herauszufinden, ob der Buddy von *rb* auch frei ist, müssen Sie die Freiliste der Speicherblöcke der Größe 2^k durchsuchen.

Wahr:

Falsch:

Nachname	Vorname	Matrikelnummer

Aufgabe 2 (Wechselseitiger Ausschluß = mutual exclusion, 4+4+4 Punkte)

Wechselseitiger Ausschluß ist verknüpft mit kritischen Abschnitten (*critical sections*) innerhalb von Threads. Nehmen Sie an, daß die betrachteten Threads zu einer Applikation mit eigenem Adreßraum (*address space*) gehören.

1. Beschreiben Sie so genau wie möglich, was Sie unter einem kritischen Abschnitt auf Applikationsebene (*application level*) verstehen!

2. Welche 3 wesentlichen Anforderungen (*requirements*) sollte eine Lösung zum Problem des wechselseitigen Ausschlusses anbieten?

.....
.....
.....

3. Analysieren Sie, ob die folgende Softwarelösung auf Anwendungsebene (*application level solution*) für die zwei Threads eine korrekte Lösung darstellt, die alle Anforderungen (*requirements*) der Teilaufgabe 2 erfüllt!

```

var flag: array[0..1] of boolean;
{thread 0}
repeat
  while flag[1] = true do {nothing};
  flag[0] := true;
  {critical section}
  flag[0] := false;
  {remainder section}
forever

{initially flag[0]=flag[1]=false}
{thread 1}
repeat
  while flag[0] = true do {nothing};
  flag[1] := true;
  {critical section}
  flag[1] := false;
  {remainder section}
forever

```

Nachname	Vorname	Matrikelnummer

Aufgabe 3

(Verklemmung = deadlock, 4 + 4 + 8 Punkte)

1. Zeigen Sie, warum die zirkulare Wartebedingung (circular wait) nur eine notwendige, aber keine hinreichende Bedingung für das Eintreten einer Verklemmung darstellt.
2. Warum führt eine unsichere (*unsafe*) Betriebsmittelsituation nicht notwendigerweise zu einer Verklemmung?
3. In einem System mit 5 Threads $\{T_1, T_2, T_3, T_4, T_5\}$ und 4 Betriebsmitteltypen (resource types) $\{r_1, r_2, r_3, r_4\}$ liege die folgende Betriebsmittelsituation vor. Stellen Sie fest, ob das System noch sicher oder bereits unsicher ist. Geben Sie nach jedem Algorithmenschritt den Vektor der nicht belegten Betriebsmitteln sowie die Menge der noch nicht terminierten Threads an!

$R = (3,2,4,1)$ ist der Vektor der pro Betriebsmitteltyp im System vorhandenen Betriebsmittel. C ist die Matrix der Gesamtanforderungen (*maximal resource requirement*) und A ist die Matrix der gerade belegten Betriebsmittel (*currently allocated resources*).

$$\begin{array}{cccc}
 3 & 2 & 0 & 0 \\
 2 & 0 & 4 & 0 \\
 C = 1 & 1 & 2 & 1 \\
 2 & 2 & 2 & 0 \\
 1 & 1 & 1 & 0
 \end{array}
 \quad
 \begin{array}{cccc}
 0 & 0 & 0 & 0 \\
 1 & 0 & 0 & 0 \\
 A = 0 & 1 & 1 & 1 \\
 1 & 0 & 2 & 0 \\
 0 & 0 & 1 & 0
 \end{array}$$

Nachname	Vorname	Matrikelnummer

Aufgabe 5 (Speicherverwaltung = memory management, 4+4+2 Punkte)

1. Charakterisieren Sie das Randkennzeichnungsverfahren (boundary tag) hinsichtlich der in der Vorlesung angegebenen orthogonalen Entwurfparameter!

2. Zählen Sie möglichst viele Vorteile des virtuellen Speichers (*virtual memory management*) gegenüber einer nicht virtuellen Speicherverwaltung auf!

3. Zählen Sie Nachteile des virtuellen Speichers auf.

Nachname	Vorname	Matrikelnummer