FACH Hochschule Aachen

BSK/ARBK SS-04 Oß

(Fachbereich 5, Elektrotechnik und Informationstechnik)

Zuname:	Vorname:	Matr.-Nr.:

Klausur "Betriebssystemkonzepte" sowie "Architektur von Rechnersystemen und" "Betriebssystemkonzepte"

Fach-Nummer:	5552(BSK) / 5310(ARBK) Fachprüfung
Termin:	Donnerstag, 7. Oktober 2004, 13.30-16.30 Uhr
Prüfer:	Prof. Dr.-Ing. Martin Oßmann

Hinweise:

Bearbeitungszeit:	180 Minuten (3 Zeitstunden)
Hilfsmittel:	Taschenrechner mit einzeiligem Display ohne Datenbankfunktion Ein zweiseitig selbstbeschriebenes A-4 Blatt
Vollständige Klausur:	6 Aufgaben

Die Aufgaben direkt nur auf den Aufgabenblättern oder Leerbögen bearbeiten. Lösungen ohne erkennbaren und leserlichen Lösungsweg sind wertlos. Jedes Blatt mit Name und Matrikelnummer versehen ! Deckblatt unterschreiben !

Zur Kenntnis genommen:

_____

Unterschrift

Korrekur:

1	2	3	4	5	6	Summe	Note

(Max. 100 Punkte)

Aufg. 1 (20 Punkte)

Eine Festplatte verfügt über 1 Oberfläche und 400 Spuren. Pro Spur hat die Festplatte 600 Cluster und die Gesamtkapazität beträgt ca. 2 GByte. Es wird FAT32 (d.h. 32-Bit große FAT-Einträge) verwendet.

a) Wieviele Cluster hat die Festplatte genau ? Wie groß ist ein Cluster ? Was ist die genaue Größe der Speicherkapazität der Festplatte in k-Bytes (1k Byte=1024 Byte)?

b) Im Hauptverzeichnis werden 64 Bytes pro Eintrag reserviert. Es können maximal 10000 Dateien im Hauptverzeichnis stehen. Wieviele Cluster belegt das Hauptverzeichnis.

c) Wieviele Cluster bleiben zur Datenspeicherung frei, wenn 16 Boot-Cluster, 3 FATs und ein Hauptverzeichnis angelegt werden ? Wieviel Prozent der Festplattenkapazität sind also Overhead ?

d) Es wird nun auf die vorher leere Platte eine 1 Byte große Datei mit Namen aaa.dat und danach eine 10000 Byte große Datei mit Namen bbb.dat geschrieben. Freie Cluster werden jeweils vom Beginn der FAT aus gesucht. Skizzieren Sie, wie nach Schreiben dieser zwei Dateien die FAT aussieht:

Adresse	Inhalt
0	reserviert
1	reserviert
2	
3	
4	
5	
6	
7	
8	
9	
10	

FAT zu d

Adresse	Inhalt
0	reserviert
1	reserviert
2	
3	
4	
5	
6	
7	
8	
9	
10	

FAT zu e

Adresse	Inhalt
0	reserviert
1	reserviert
2	
3	
4	
5	
6	
7	
8	
9	
10	

Fat zu f

e) Danach wird die Datei aaa.dat gelöscht und eine 17000 Bytes große Datei ccc.dat geschrieben. Wie sieht nun die FAT aus ?

f) Danach wird die Datei bbb.dat gelöscht und eine 4000 Bytes große Datei ddd.dat geschrieben. Wie sieht nun die FAT aus ?

Es liest jeweils nur ein Kopf auf der Oberfläche. Die Festplatte arbeite mit einer Umdrehungsrate von 1500 rpm. Ein Spurwechsel dauert 20ms.

g) Wieviele Bits hat eine Spur ? Welche Datenrate (Bit pro Sekunde) liefert der Lesekopf maximal ?

h) Wie lange dauert es im günstigsten Fall, die gesamte Festplatte zu lesen ? Geben Sie genau an, wie sie zu Ihrem Ergebnis gelangen !

i) Warum ist eine hohe (externe) Fragmentierung bei Festplatten unerwünscht ?

j) Ein Betriebssystem bekommt den Namen einer Datei auf einer Festplatte. Wie ermittelt das Betriebssystem, wo der erste Sektor der Datei steht ? Wo stehen die dazu notwendigen P¿arameter und welche sind das ?

Aufg. 2 (20 Punkte)

i) Ein Rechner verwaltet seinen Speicher mittels einstufigem Demand Paging. Er besitzt eine Memory Management Unit MMU welche den Speicherzugriff durchführt. Die Seitentabelle kann 16 Einträge aufnehmen. Der physikalische Speicher kann 8 Rahmen aufnehmen. Die MMU ist mit einem Assoziativspeicher (TLB) mit 3 Einträgen ausgestattet. im TLB wird LRU (Least Recently Used) als Ersetzungsstrategie angewandt.

Am Anfang sind keine Seiten im Speicher eingelagert.

In der gegebenen Reihenfolge wird auf die folgenden Seitem zugegriffen:

3,4,2,3,1,2,5,2,3

i-a) Geben Sie an wie sich der Inhalt des TLB zeitlich entwickelt. Notieren Sie den Inhalt in folgender Tabelle:

RU									
LRU									

i-b) Wieviele Seitenrahmen befinden sich nach den Zugriffen im Speicher (Begründung !) ?

i-c) Welche Treffer-Rate erzielt der TLB bei diesen Zugriffen ?

i-d) Ohne TLB-Treffer dauert der Speicherzugriff 10ns. Bei einem TLB Treffer dauert er 3ns. Wie hoch muß die TLB-Trefferrate sein, damit die mittlere Zugriffszeit mit TLB halb so groß ist wie ohne TLB ?

i-e) Wie groß ist die Seitengröße, wenn der physikalische Speicher 4096 Worte umfasst ?

i-f) In einer Seitentabelle wird üblicherweise ein R-Bit ("Recently-Bit") gespeichert. Wann wird es gesetzt ? Wann wird es zurückgesetzt ?

i-g) In einer Seitentabelle wird üblicherweise ein M-Bit ("Modified-Bit") gespeichert. Wann wird es gesetzt ? Wann wird es zurückgesetzt ?

ii) Nun wird zur Verwaltung eines 16M gorssen Speichers das Buddy-System verwaltet. Die kleinste Einheit ist 1M gross. Segmente werden bei niedrigen Adressen beginnend gesucht.

Es erfolgen die folgenden Anforderungen/Freigaben in der gegebenen Reihenfolge:

Prozess	Art	Grösse
A	Anforderung	3M
B	Anforderung	0.8M
C	Anforderung	5M
A	Freigabe	
D	Anforderung	1.6M
B	Freigabe	

ii-a) Nach jeder Anforderung/Freigabe skizziere man, wie der Speicher aufgeteilt ist ! Weiter gebe man dabei an, wieviel Speicher dann jeweils noch allokierbar ist, und wieviel Speicher durch interne Fragmentierung ingesamt verloren gegangen ist. Durch interne Fragmentierung verlorenen Speicher markiere man in der Skizze besonders !

ii-b) Wieviel Speicher kann bei diesem System nmaximal durch interne Fragmentierung verloren gehen. Beschreiben Sie, wie es dazu kommen kann !

ii-c) Wann sind zwei Segmente "Buddys" im Sinne des Buddy-Systems ?

Aufg. 3 (15 Punkte)

Ein Betriebssystem vergibt die Rechenzeit an Prozesse mittels Warteschlangen. Das Scheduling wird von einem Timer-Interrupt durchgeführt, der jede Sekunde auftritt. Der erste Interrupt passiert bei t=0 Sekunden.

Es gibt zwei Warteschlangen, eine für Aufträge mit hoher Priorität (H), eine für niedrige Priorität (N). Innerhalb der Warteschlangen wird die Vergabe mittels Round-Robin durchgeführt. Eine Zeitscheibe ist dabei 1 Sekunde lang und diese Zeitscheibe kann nicht unterbrochen werden. Benutzt ein Prozess nicht die gesamte Zeitscheibe, wird erst nach Ablauf der Zeitscheibe gescheduled. Der Prozessor bearbeitet nur dann Prozesse niedriger Priorität, wenn die Warteschlange mit hoher Priorität leer ist.

Es sind die folgenden Prozesse zu bearbeiten !

Prozess	Ankunft/s	Rechenzeit/s	Prorität
A	0.3	2	N
B	1.5	3	N
C	4.5	3	H
D	6.2	2	H
E	11.5	1	N

a) Zeichen Sie das Gantt-Chart für die Prozessorvergabe.

b) Geben Sie die Wartezeit und die Turn-Around Zeit für die Prozesse C,D und E an.

c) Wird bei diesem Verfahren bei beliebigen Prozessen jeder Prozess mit endlicher Rechenzeit irgendwann fertig (Begründung) ?

d) Die obigen Prozesse werden nun gemäß FIFO nicht preemptiv gescheduled. Zeichen Sie das Gantt-Chart für diese Prozessorvergabe !

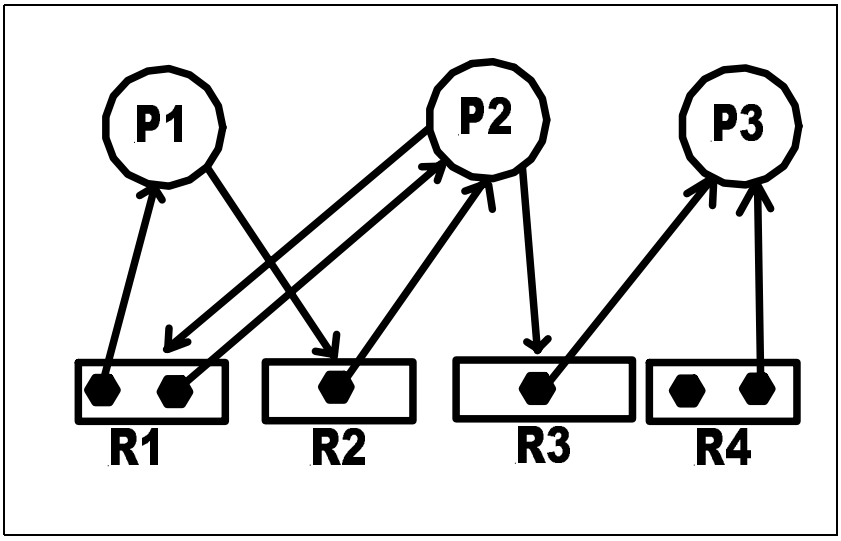
e) Warum bekommen Interrupt-Prozesse in Betriebssystemen im allgemeinen eine hohe Priorität ?

f) Warum führt man dynamisch veränderliche Prioritäten ein ? Wann erhalten Interaktive Prozesse einen "Prioritäts-Boost" ?

g) Beim Scheduling wird der "Kontext" eines Prozesses gerettet. Was versteht man unter dem "Kontext" ?

Aufg. 4 (10 Punkte)

Gegeben ist der folgende Betriebsmittelgraph:



a) Man stelle die zu dieser Situation gehörige Anforderungsmatrix $R = (r_{ij})$ und Belegungsmatix $C = (c_{ij})$ sowie den Betriebsmittelrestvektor A und den Betriebsmittelvektor E auf !

b) Liegt ein Deadlock vor ? Begründen Sie Ihre Antwort mit Hilfe des Algorithmus zur Deadlockerkennung ! Nach jedem Schritt des Algorithmus sind die neuen Matrizen und Vektoren anzugeben ! Welche Prozesse sind am Deadlock beteiligt ?

c) Wieviele und welche Zyklen enthält der Betriebsmittelgraph aus a) ? Geben Sie für jeden Zyklus die Prozesse/Betriebsmittelklassen in der Durchlaufungsreihenfolge an.

d) Ein Betriebssystem vergibt die Betriebsmittel wie folgt: Jeder Prozess muss alle Betriebsmittel, die er benötigt, "auf einenen Schlag" anfordern. Dann erhält er entweder alle Betriebsmittel, oder wird blockiert, bis er alle erhält. Man begründe, warum bei diesem Verfahren kein Deadlock entstehen kann ! Man begründe, warum dieses Verfahren im Allgemeinen nicht praktikabel ist !

e) Wie kann man es erreichen, dass ein Prozess bei einem Deadlock abgebrochen und später neu gestartet werden kann ?

Aufg. 5 (15 Punkte)

Ein Betriebssystem wird in einer C-ähnlichen Programmiersprache programmiert. Bereits implementiert sind die folgenden Routinen:

$void \quad DISABLE()$

Sperrt alle Interrupts.

$void \quad ENABLE()$

Gibt die Interrupts wieder frei.

$void \quad SLEEP(int \quad n)$

Der aktulle Prozess wird "schlafen gelegt". Der INT-parameter n gibt die Nachricht n an, auf welche der Prozess wartet, um wieder aufgeweckt zu werden. Es können mehrere Prozesse auf die gleiche Nachricht warten.

$void \quad WAKEUP(int \quad n)$

Ein schlafender Prozess, der auf Nachricht n wartet, wir aufgeweckt.

a) Erklären Sie im Detail, was die UP und DOWN Funktion bei zählenden Semaphoren machen !

b) Programmieren Sie in einer C-ähnlichen Sprache die UP und $DOWN$ Funktion mit Hilfe der oben gegebenen bereits implementierten Funktionen. Als Argument sollen die beiden Funktionen UP und $DOWN$ die Adresse der Semaphorvariablen s verwenden.

c) Welchen Vorteil haben Semaphore gegenüber "busy waiting" ?

d) Warum ist es ungünstig, Interrupts zu sperren ?

e) Für welche Aufgaben werden "Timer-Interrupts" verwendet ?

f) Erklären Sie, aufgrund welcher Ereignisse in einem Betriebssystem ein Prozess seinen Zustand von "blocked" nach "ready" wechselt !

Zuname:	Vorname:	Matr.-Nr.:

Aufg. 6 (20 Punkte)

Gegeben ist das folgende M32 Assemblerprogramm:

```
 LINE     RAM          CODE                          TEXT
01 00000000H :                                    ORG  0H

02 00000000H :  21002001H ________H               MOV  R1,4

03 ________H :  12002002H FFFFFFFFH               SUB  R2,0FFFFFFFFH

04 ________H :  F3000022H ________H               PRT  @TXT

05 ________H :  31000020H ________H               JMP  OK

06 ________H :  F3000022H ________H               PRT  @TXT

07 ________H :  21004203H ________H        OK     MOV  R3,3(R2)

08 ________H :  21002204H 00000010H               MOV  R4,@STOP

09 ________H :  21000440H 00000000H               MOV  0(R0),R4

10 ________H :  F2000000H                  STOP   HALT

11 ________H :  00000041H 00000042H        TXT    DW   "A",042H

12 ________H :  ________H ________H               DW   043H,0

13 ________H :                             VVV    DS   1

14 ________H :  01000000H                  UUU    NOP

15 ________H :                                    END
```

Leerstellen (jeweils ein 32 Bit-Wort) mit ________H markiert !!

a) In der Spalte RAM steht normalerweise die Adresse, unter welcher der Assembler Code und Daten abspeichert. Ergänzen Sie alle Leerstellen dieser Spalte.

b) Ergänzen Sie ebenfalls die in den Spalten CODE vorhandenen Leerstellen !

c) Geben Sie den Inhalt der Symboltabelle nach dem Assemblerlauf an !

d) Geben Sie den Inhalt der Register R1 bis R4 nach dem Programmlauf (gestartet durch RESET nach dem Assemblieren) an.

e) Das Programm wird assembliert und dann (nach RESET) gestartet. Geben Sie für die ersten 5 Speicherzugriffe jeweils die Adresse und die gelesenen bzw. geschriebenen Daten (hexadezimal) an !

f) Welche Ausgabe erzeugt das Programm ?

g) Nach dem ersten Programmlauf wird das Programm mittels RESET und dann RUN erneut gestartet. Welche Ausgabe erzeugt das Programm nun und warum ?

h) Warum unterscheiden CPUs zwischen USER- und SYSTEM-Modus ?

i) Wozu dient das LIMIT Register in einer Memory-Management-Unit (z.B. beim M32 Rechner) ? Erläutern Sie seine Funktion im Detail !

Zuname:	Vorname:	Matr.-Nr.: