Name: Matrikel-Nr.: Studiengang:

Klausur Betriebssysteme und Sicherheit, 06.08.2015

1	2	3	4	5	$\sum$
6	9	9	9	9	42

Aufgabe 1 – Sicherheit

6 Punkte

a) Nennen Sie die drei (Haupt-)Schutzziele in der Datensicherheit! **1 P**

b) Nennen Sie einen Nachteil symmetrischer gegenüber asymmetrischen Verschlüsselungsverfahren! **1 P**

Alice und Bob besitzen jeweils ein Paar RSA-Schlüssel: Alice den privaten Schlüssel D_A und den öffentlichen Schlüssel E_A, Bob entsprechend D_B und E_B. Gegeben sind die Funktionen $enc : (Msg, Key) \rightarrow Crypt$ zum Verschlüsseln, $dec : (Crypt, Key) \rightarrow Msg$ zum Entschlüsseln, $sig : (Msg, Key) \rightarrow Sign$ zum Signieren und $ver : (Msg, Sign, Key) \rightarrow \{0, 1\}$ zum Verifizieren einer Signatur.

c) Bob schickt Alice eine *verschlüsselte* Nachricht. Beschreiben Sie den Vorgang vom Klartext bei Bob bis zum entschlüsselten Klartext bei Alice mit Hilfe der vier Funktionen und Schlüssel! **2 P**

d) Alice schickt Bob eine *signierte* Antwort. Beschreiben Sie den Vorgang vom unsignierten Klartext bei Alice bis zum verifizierten Klartext bei Bob mit Hilfe der vier Funktionen und Schlüssel! **2 P**

Aufgabe 2 – Verklemmungen **9 Punkte**

In einem System stehen insgesamt 7 Bandlaufwerke bereit. Diese werden von verschiedenen laufenden Prozessen (A,B,C,D) für die Langzeitarchivierung von Daten genutzt. Jeder Prozess benötigt dabei ein Bandlaufwerk während seiner gesamten Laufzeit. Je nach zu verarbeitendem Datenaufkommen sind zu verschiedenen Zeitpunkten aber noch zusätzliche Laufwerke nötig. Es spielt dabei keine Rolle welches Laufwerk ein Prozess zugewiesen bekommt; das Schreiben muss allerdings in einem Stück geschehen, da andernfalls Daten verloren gehen. Um dies zu verhindern, wird jedes Bandlaufwerk immer genau einem Prozess exklusiv zur Verwendung zugewiesen bis dieser es ans System zurückgibt.

Die Prozesse benötigen maximal 4 (A), 2 (B), 6 (C) bzw. 5 (D) Bandlaufwerke gleichzeitig. Um eine möglichst hohe Auslastung der verfügbaren Laufwerke zu gewährleisten, soll ein Laufwerk einem Prozess nur dann bereitgestellt werden, wenn und solange er dieses benötigt.

a) Zeigen Sie, dass es im beschriebenen System zu Verklemmungen kommen kann! **2 P**

b) Um Verklemmungen vorzubeugen wird der Bank-Algorithmus eingesetzt. Aktuell sind den Prozessen 1 (A), 0 (B), 1 (C) und 3 (D) Laufwerke zugeordnet. Nacheinander fordern nun die Prozesse C und D je ein zusätzliches Laufwerk an. Wie reagiert das System auf diese Anforderungen? Wenden Sie zur Beantwortung der Frage den Bank-Algorithmus an! **4 P**

c) Kann es im Rahmen des Bank-Algorithmus dazu kommen, dass der Vorgänger eines sicheren Zustandes ein unsicherer Zustand ist? Begründen Sie Ihre Antwort! **1 P**

d) Nennen Sie eine Alternative zur Verwendung des Bank-Algorithmus sowie je einen Vor- und Nachteil dieser Alternative gegenüber dem Bank-Algorithmus! **2 P**

Aufgabe 3 – Synchronisation

9 Punkte

Folgender C-Code sei gegeben:

```
1 int x = 0;
2
3 int inc(void)
4 {
5   x++;
6 }
```

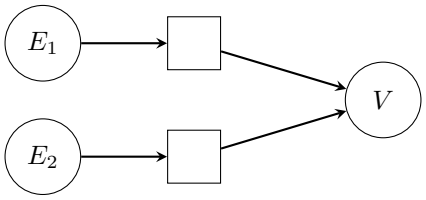
Die Funktion `inc()` werde von zwei Threads gleichzeitig aufgerufen.

a) Erläutern Sie an diesem Beispiel den Begriff der Wettlaufsituation (englisch „Race Condition“)! Nennen Sie mögliche Ergebnisse nach Ausführung der Funktion durch beide Threads! **2 P**

b) Nennen Sie vier Techniken, die zu einem deterministischen Ergebnis führen! Erwähnen Sie für jede Technik, ob diese mit Busy Waiting arbeitet!

HINWEIS: Die Funktionalität von `inc()` soll erhalten bleiben; ebenso die Parallelausführung der Funktion durch zwei Threads. **4 P**

Ein Erzeuger-Verbraucher-System bestehe aus zwei Erzeugern E_1 und E_2, die wiederholt jeweils einen einelementigen Puffer befüllen. Diese Puffer werden durch einen gemeinsamen Verbraucher V geleert. Dabei kann V die Puffer in beliebiger Reihenfolge bearbeiten. Insbesondere kann V arbeiten, sobald einer der beiden Puffer gefüllt ist.

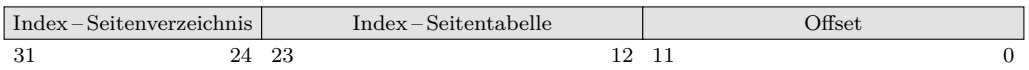


c) Geben Sie Implementierungen für Erzeuger und Verbraucher in Pseudocode an, bei der kein Busy Waiting auftritt und die maximale Parallelität gewährleistet! **3 P**

Aufgabe 4 – Speicher

9 Punkte

Eine hypothetische 32-bit Architektur verwende ein 2-stufiges Schema zur Adressübersetzung mit folgender Aufteilung der virtuellen Adressen:



Die Einträge des Seitenverzeichnisses und der Seitentabellen sind jeweils 32 Bit lang. Dabei wird Bit 0 genutzt um gültige Einträge zu kennzeichnen (*Present-Bit*). Bit 1 kennzeichnet schreibbare Einträge (*Writable-Bit*). Bei einem Zugriff werden diese Bits sowohl im Seitenverzeichnis-Eintrag als auch im Seitentabellen-Eintrag geprüft. Es müssen also z. B. beide Writable-Bits gesetzt sein, damit ein Schreibzugriff gelingt. Die Seitengröße beträgt 4 KiByte.

a) Wie viele Einträge hat das Seitenverzeichnis und wie viele hat eine Seitentabelle? Wie viele Rahmen werden für das Seitenverzeichnis bzw. eine Seitentabelle benötigt? **2 P**

Ein Programm möchte den physischen Speicher `0x40004000` – `0x40005FFF` für seinen Code und den physischen Speicher `0x70006000` – `0x70006FFF` sowie `0x70012000` – `0x70012FFF` als Stack verwenden. Für das Programm soll der Code im (virtuellen) Speicherbereich `0x00002000` – `0x00003FFF` liegen und nur lesbar sein. Der Stack soll schreibbar im Bereich `0x01000000` – `0x01001FFF` (virtuell) liegen.

Das Seitenverzeichniss liegt an der physischen Adresse `0x10000000`, für die Seitentabellen soll der physische Speicherbereich `0x10002000` – `0x10009FFF` verwendet werden.

b) Geben Sie die nötigen Einträge in Seitenverzeichnis und Seitentabellen, sowie die Adressen der Seitentabellen an, um diese Abbildung herzustellen! **6 P**

Seitenverzeichnis

Addr.:	`0x10000000`
0	
1	
2	
3	
4	

Seitentabelle

Addr.:	
0	
1	
2	
3	
4	

Seitentabelle

Addr.:	
0	
1	
2	
3	
4	

Zu einem späteren Zeitpunkt werden der folgende zusätzliche Eintrag im Seitenverzeichnis sowie die dargestellte Seitentabelle erstellt:

Seitenverzeichnis

Addr.:	`0x10000000`
0	
1	
2	
3	`0x1000A003`
4	

Seitentabelle

Addr.:	`0x1000A000`
0	
1	
2	`0x10004003`
3	
4	

c) Stellt diese Abbildung eine Gefahr für das System dar? Begründen Sie Ihre Antwort! **1 P**

Aufgabe 5 – Scheduling **9 Punkte**

Gegenstand dieser Aufgabe ist die Einplanung einer Task-Menge T mit Tasks der Form $X = (p, e)$, beschrieben durch Periodenlänge p und konstante Ausführungszeit e. Die relative Deadline d ist gleich der Periodenlänge ($d = p$). Der Scheduler des Systems kann laufende Jobs jederzeit unterbrechen, der dabei auftretende Overhead wird ignoriert. Dazu werde folgendes Beispiel betrachtet: $T = \{A = (12, 5), B = (3, 1), C = (6, 3), D = (4, 2)\}$.

Die Tasks sollen auf einem System mit zwei gleichartigen Prozessoren P_0 und P_1 ausgeführt werden, wozu sich prinzipiell zwei Möglichkeiten anbieten – Partitionierung und globales Scheduling.

Zunächst werde folgende Partitionierung verwendet: Die Tasks A und B werden dem Prozessor P_0 zugeordnet und nur auf diesem ausgeführt; analog laufen C und D nur auf Prozessor P_1. Auf jedem der beiden Prozessoren erfolgt das Scheduling mithin wie in einem Einprozessor-System. Dabei soll das Scheduling auf der Basis von statischen Prioritäten erfolgen.

a) Prüfen Sie, ob die Task-Menge T auf diese Weise einplanbar ist! Nehmen Sie diese Prüfung rein rechnerisch vor, soweit dies möglich ist und begründen Sie Ihr Vorgehen! **5 P**

Falls erforderlich, können Sie diesen Bereich zum Zeichnen von Ablaufplänen nutzen.

Alternativ entfallen Partitionierung und statische Prioritätszuordnung, die Tasks werden stattdessen global nach EDF eingeplant. Dabei wählt der Scheduler des Systems den Job, der auf einem der beiden Prozessoren laufen soll, gemäß dem Ihnen bekannten EDF-Schema aus. Das hat zur Folge, dass die Jobs ein und derselben Task je nach Situation auf dem einen oder auf dem anderen Prozessor ausgeführt werden können.

b) Ermitteln Sie zunächst die Periodenanfänge (und damit auch Deadlines) aller relevanten Jobs für die Dauer der ersten Hyperperiode! Markieren Sie sie unter dem untenstehenden Diagramm in der Form *Job/Ausführungszeit*, wie für die ersten beiden Einträge bereits beispielhaft vorgegeben. **1 P**

c) Zeichnen Sie nun im nachfolgenden Diagramm den resultierenden Ablaufplan für die Dauer der ersten Hyperperiode ein! **3 P**