Musterlösung Nachklausur

25.09.2015

Alle Punkteangaben ohne Gewähr!

- Bitte tragen Sie zuerst auf dem Deckblatt Ihren Namen, Ihren Vornamen und Ihre Matrikelnummer ein. Tragen Sie dann auf den anderen Blättern (auch auf dem Konzeptblatt) Ihre Matrikelnummer ein.
  Please fill in your last name, your first name, and your matriculation number on this page and fill in your matriculation number on all other pages (including the draft page).
- Die Prüfung besteht aus 11 Blättern: Einem Deckblatt und 10 Aufgabenblättern mit insgesamt 5 Aufgaben.
  The examination consists of 11 pages: One cover sheet and 10 sheets containing 5 assignments.
- Es sind keinerlei Hilfsmittel erlaubt!
  No additional material is 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 additional draft paper, please notify one of the supervisors.
- Bitte machen Sie eindeutig klar, was Ihre endgültige Lösung zu den jeweiligen Teilaufgaben ist. Teilaufgaben mit widersprüchlichen Lösungen werden mit 0 Punkten bewertet.
  Make sure to clearly mark your final solution to each question. Questions with multiple, contradicting answers are void (0 points).

Die folgende Tabelle wird von uns ausgefüllt! *The following table is completed by us!*

Aufgabe	1	2	3	4	5	Total
Max. Punkte	12	12	12	12	12	60
Erreichte Punkte						
Note						

Aufgabe 1: Grundlagen

Assignment 1: Basics

a) Welche Abstraktion wird vom Betriebssystem für das Multiplexen von CPUs bereitgestellt? **0.5 pt**

What abstraction does the operating system supply for multiplexing CPUs?

Lösung:

The operating system abstracts CPUs with processes and threads **(0.5 P)**. *Either process or thread is sufficient.*

b) Was versteht man unter der *Trap* Instruktion? Ist diese privilegiert? **1.5 pt**

What is the trap *instruction? Is is privileged?*

Lösung:

The trap *instruction switches the processor to privileged mode* **(0.5 P)** *and continues execution at an operating system defined location. It is used to implement system calls* **(0.5 P)** *and is thus not privileged* **(0.5 P)**.

c) Nennen Sie eine Betriebssystemfunktion für die üblicherweise kein Eintritt in den Kern notwendig ist. Begründen Sie Ihre Antwort. **1.5 pt**

Give an operating system function that typically does not require entering the kernel. Explain your answer.

Lösung:

`malloc` **(0.5 P)**. *The heap allocator is usually implemented in a user-space library and only calls into the kernel if the heap ran out of memory* **(1 P)**.

Another example: `GetTickCount`. *Receiving the number of ticks since the start of the system must be cheap and quick. The kernel therefore maps a page containing the current tick count at a known location into each process.*

Other answers are accepted if properly explained.

d) Bewerten Sie folgende Aussage: Bei einem Systemaufruf muss das Page Table Directory (CR3) ausgetauscht werden, um in den Kernadressraum zu wechseln. Begründen Sie Ihre Antwort. **2 pt**

Evaluate the following statement: On a system call, the page table directory (CR3) must be switched to transition to the kernel address space. Explain your answer.

Lösung:

The statement is wrong **(0.5 P)**. *The kernel address space is mapped into each process's address space* **(1.5 P)** *at the high end of the virtual address range. A switch of the CR3 register is thus not needed.*

e) Was versteht man unter einem *Trampolin für Systemaufrufe*? Wie stellt der Kernel dieses einem Prozess zur Verfügung? **3 pt**

What is a system call trampoline*? How does the kernel provide a process access to the trampoline?*

Lösung:

Some CPU architectures such as x86, provide different trap instructions depending on the manufacturer **(1 P)** *(Intel or AMD), namely* `sysenter` *and* `syscall`*. To make user-level programs compatible with all variants, the operating system places system call invocation code (including the trap instruction) for each variant in a separate code segment–the trampoline* **(0.5 P)***. The operating system chooses the right trampoline based on the detected CPU architecture variant (at boot time)* **(0.5 P)***.*

The selected trampoline is mapped into every process at a known location **(1 P)***.*

f) Nennen Sie eine CPU Ausnahme, die eine weitere Ausführung des Programms erlaubt. Welche Bedingungen müssen dabei erfüllt sein? **1.5 pt**

Give a CPU exception that allows continuing program execution. What conditions must be fulfilled in that case?

Lösung:

`page fault` **(0.5 P)***. The access must be permitted* **(1 P)***, which means the address space region must be valid and the access mode (read/write/execute) must be permitted in the current state (user/kernel).*

Also accepted: `un-aligned memory access` **(0.5 P)***. No conditions required* **(1 P)***.*

g) Unterstreichen Sie diejenigen Segmente, die *nicht* Teil einer ELF-Datei sind. Geben Sie eine kurze Begründung, warum diese nicht Bestandteil sind. **2 pt**

Underline those segments that are not *part of an ELF file. Shortly explain why these segments are missing.*

Lösung:

Data   *R/O-Data*   <u>*Heap*</u> **(0.5 P)**   *BSS*   *Text*   <u>*Stack*</u> **(0.5 P)**

Each thread requires a stack. Since threads are created dynamically during program execution, the ELF file cannot define the stack segments **(0.5 P)***. The same is true for the heap as it stores dynamically allocated data and the correct size cannot be determined at compile time* **(0.5 P)***.*

Total: 12.0pt

Aufgabe 2: Prozesse und Threads

Assignment 2: Processes and Threads

a) Welche der folgenden Aussagen sind richtig?
(falsches Kreuz: -0.5P, kein Kreuz: 0P, korrektes Kreuz: 0.5P) **2 pt**

Which of the following statements are correct?
(incorrectly marked: -0.5P, not marked: 0P, correctly marked: 0.5P)

korrekt/ *correct*	inkorrekt/ *incorrect*	
☐	■	Der Systemaufruf `exec()` erstellt einen neuen Prozess *The `exec()` system call creates a new process*
☐	■	Das PSJF-Schedulingverfahren garantiert, dass jeder Thread nach endlicher Wartezeit mindestens eine Zeitscheibe erhält *PSJF scheduling guarantees that each thread receives at least one timeslice after a finite waiting time*
■	☐	Der Konvoi-Effekt kann beim PSJF-Schedulingverfahren nicht auftreten *The convoy effect does not occur with PSJF scheduling*
■	☐	Der PCB enthält üblicherweise Informationen über geöffnete Dateien *The PCB typically contains information about open files*

b) Gegeben seien fünf Prozesse auf einem Einprozessorsystem mit den angegebenen Ankunftszeiten (0 = Start) und Burst-Zeiten. Vervollständigen Sie die untenstehenden Ablaufpläne für die Strategie *Shortest Job First (SJF)* sowie die Strategie *Round-Robin (RR)*. Ein Kasten im Zeitplan stellt eine Zeiteinheit dar. **4 pt**

Consider five processes on a uniprocessor system, with given arrival times (0 = start) and burst times. Complete the scheduling plans given below for the policy Shortest Job First (SJF) *and the policy* Round-Robin (RR). *A box in the scheduling plan represents one unit of time.*

Process	Arrival Time	Burst-Time
1	2.5	5
2	7.5	3
3	1.5	6
4	4.5	1
5	0	3

Lösung:

Shortest Job First (SJF)

5	*5*	*5*	*1*	*1*	*1*	*1*	*1*	*4*	*2*	*2*	*2*	*3*	*3*	*3*	*3*	*3*	*3*

Round-Robin (RR)

5	*5*	*3*	*5*	*1*	*3*	*4*	*1*	*3*	*2*	*1*	*3*	*2*	*1*	*3*	*2*	*1*	*3*

***-0.5 P** for each incorrect scheduling decision*

c) Berechnen Sie für den obigen *SJF-Ablaufplan* die Wartezeit aller Prozesse. **2 pt**

For the above SJF scheduling plan, *calculate the waiting time of each process.*

Lösung:

The waiting time for each process can be calculated as $t_{wait} = t_{finish} - t_{arrival} - t_{burst}$

Process	Finish time	Arrival time	Burst time	Waiting time
1	8	2.5	5	0.5
2	12	7.5	3	1.5
3	18	1.5	6	10.5
4	9	4.5	1	3.5
5	3	0	3	0

d) Berechnen Sie für den obigen *RR-Ablaufplan* die Tournaround-Zeit aller Prozesse. **2 pt**

For the above RR scheduling plan, *calculate the tournaround time of each process.*

Lösung:

The turnaround time for each process can be calculated as $t_{tournaround} = t_{finish} - t_{arrival}$

Process	Finish time	Arrival time	Turnaround time
1	17	2.5	14.5
2	16	7.5	8.5
3	18	1.5	16.5
4	7	4.5	2.5
5	4	0	4

e) Was versteht man unter *kooperativem Scheduling*? Geben Sie je einen Vor- und Nachteil von kooperativem Scheduling an. **2 pt**

What is cooperative scheduling*? Give an advantage and a disadvantage of cooperative scheduling.*

Lösung:

Cooperative scheduling means that the operating system does not forcibly preempt running threads **(0.5 P)**. *Instead, threads must relinquish the CPU voluntarily, for example by blocking for I/O or calling* `yield()` *explicitly* **(0.5 P)**.

The main disadvantage of cooperative scheduling is that uncooperative or malfunctioning threads can hog the CPU; the OS has no way of bringing such threads under control **(0.5 P)**.

Advantages of cooperative scheduling include a reduced scheduling overhead since threads can relinquish the CPU at a time that is convenient for them (e.g., when they must block for I/O anyway) and a simpler scheduler implementation since there is no need to implement preemption. **(0.5 P)**

Total: 12.0pt

Aufgabe 3: Koordination und Kommunikation von Prozessen

Assignment 3: Process Coordination and Communication

a) Was sind die vier notwendigen Bedingungen für einen Deadlock? **2 pt**

What are the four necessary conditions for a deadlock?

Lösung:

- *Mutual exclusion* **(0.5 P)**
- *Hold and wait* **(0.5 P)**
- *No preemption* **(0.5 P)**
- *Circular wait* **(0.5 P)**

b) Was sind *race conditions*? Wie können diese vermieden werden? **2 pt**

Explain the term race condition. *How can race conditions be avoided?*

Lösung:

Concurrent access to shared data by different threads **(0.5 P)** *can lead to wrong results.* **(0.5 P)**

Alternatively: Computed results depend on the execution order of involved concurrently running threads or processes. **(1 P)**

Protecting critical sections **(0.5 P)** *with synchronization mechanisms* **(0.5 P)** *avoids race conditions.*

c) Gegeben sei ein System, das nicht-blockierende (d.h. asynchrone) Interprozesskommunikation (IPC) unterstützt. Warum muss eine korrekte Implementierung dieses Systems das Erzeuger-Verbraucher-Problem *(producer-consumer-problem)* lösen?
Welche Designentscheidung für IPC würde die der Konstruktion des Systems zugrundeliegende Synchronisationsaufgabe vereinfachen? **3 pt**

Assume a system that supports non-blocking (i.e., asynchronous) inter-process communication (IPC). Explain why a correct implementation of this system must solve the producer-consumer problem.
Which design choice for IPC would simplify the synchronization task involved in building this IPC system?

Lösung:

Messages have to be stored in a ***buffer*** *because an arbitrary amount of time could pass between the non-blocking send and the corresponding receive operation. We cannot assume the initial copy of the message (in the sender's AS) to be available at receive time because the sender may destroy it once the non-blocking send operation succeeds. Operations on these buffers have to be synchronized.* **(1 P)**

If we choose to reserve a limited amount of memory for message buffers ***(bounded buffers)****, a sender has to wait once all buffers are occupied, thereby degrading the non-blocking send to a blocking send operation. Consequently, receivers that consume a message from a fully occupied buffer have to wakeup potential waiting senders. This scenario is the producer-consumer problem.* **(1 P)**

If we choose to let buffers grow indefinitely, which may result in all memory being consumed by message buffers (i.e., a bad idea), senders will never have to wait when posting a message. Thus, the communication system would not need to solve the complete producer-consumer problem. **(1 P)** *(You may alleviate the issue of unlimited memory use by having the sender or the receiver provide buffer space).*

d) Welche der notwendigen Bedingungen für eine gültige Lösung des Problems kritischer Abschnitte werden von einfachen Spinlocks erfüllt, welche nicht? Erläutern Sie jeweils warum.
Welcher andere Synchronisationsmechanismus erfüllt alle diese Bedingungen? **5 pt**

Which of the requirements for a valid solution of the critical section problem do simple spinlocks fulfill, which not? For each requirement explain why / why not.
State another synchronization mechanism that does fulfill all of these requirements.

Lösung:

Simple spinlocks provide mutual exclusion **(0.5 P)**. *They are implemented using atomic instructions that ensure that only one candidate process will retrieve the spinlock's state as free and subsequently change it to occupied.* **(1 P)**

Simple spinlocks do fulfill the progress *criteria* **(0.5 P)**. *A process can acquire the spinlock all by itself. Processes in their remainder section never access the spinlock variable and thus cannot interfere with another process trying to enter the critical section.* **(1 P)**

Simple spinlocks do not fulfill bounded waiting **(0.5 P)** *because the atomic operations used for busy waiting do not guarantee an order in which threads acquire the lock.* **(1 P)**

Strong semaphores (and e.g., ticket locks) do fulfill all requirements for a valid solution of the critical section problem. **(0.5 P)**

Total: 12.0pt

Aufgabe 4: Speicher

Assignment 4: Memory

a) Nennen sind die zwei Ursachen für das Auftreten von Seitenfehlern aus Sicht der MMU. Erläutern Sie jeweils kurz wie das Betriebssystem diese üblicherweise behandelt. **3 pt**

Give the two causes for page faults from the MMU's perspective. Briefly explain how the operating system typically handles each cause.

Lösung:

(a) No valid mapping for the accessed address **(0.5 P)***. The present/valid bit in the PTE is not set. If the access is directed to a valid memory region, a frame is loaded with appropriate contents and mapped to the page* **(0.5 P)***. The operation is restarted* **(0.5 P)**.

(b) Valid mapping, but access not permitted **(0.5 P)** *(e.g., write on read-only page). The process is terminated* **(1 P)***. Alternative answer: The exception is forwarded to the process.*

b) Gegeben sei ein System mit vier physischen Seitenrahmen. Vervollständigen Sie die Übersetzungstabelle für einen Prozess der auf virtuelle Seiten in der unten angegebenen Reihenfolge zugreift. Verwenden Sie den Clock-Algorithmus zur Seitenersetzung. Nehmen Sie an, dass der zirkuläre Puffer der Uhr in aufsteigender Reihenfolge der Rahmennummern sortiert ist (Rahmen 0, 1, 2, 3), die Hand der Uhr initial auf Rahmen 0 (unterstrichen) zeigt und das Referenzbit (R) für die Seite 0 gesetzt ist. **5 pt**

Consider a system with four page frames. Complete the mapping table for a process that accesses virtual pages in the order given below with clock page replacement. Assume the circular buffer of the clock to be in ascending order (i.e., frame 0, 1, 2, 3), the clock hand to be initially positioned at frame 0 (underlined), and the reference bit (R) for page 0 to be set.

Lösung:

Frames	$Pages(t_0)$	$Pages(t_1)$	$Pages(t_2)$	$Pages(t_3)$	$Pages(t_4)$	$Pages(t_5)$
0	$\underline{2}$	5 (R)	5 (R)	$\underline{5\ (R)}$	5	3 (R)
1	0 (R)	$\underline{0\ (R)}$	0	0	1 (R)	$\underline{1\ (R)}$
2	4	4	2 (R)	2 (R)	$\underline{2\ (R)}$	2
3	3	3	$\underline{3}$	4 (R)	4 (R)	4

Zugriffsfolge auf virtuelle Seiten/*Virtual page reference string*: **5 2 4 1 3**

Reference bit and clock position markings were not required. **-0.5 P** for each mistake. Correctly filling out the table for an algorithm that differs from the clock algorithm taught in the lecture was not accepted!

c) Geben Sie je ein Beispiel an, wann das Betriebssystem einzelne bzw. mehrere Einträge in einem TLB ohne Tagging invalidieren muss. **1 pt**

When does the operating system need to invalidate individual entries in a non-tagged TLB? When does the operating system need to invalidate multiple entries? Give one example each.

Lösung:

If the mapping (i.e., PTE) of a page has been changed **(0.5 P)** *(e.g., swapped page, change of access rights, new mapping, etc.), the corresponding TLB entry must be invalidated. Multiple / all entries must be invalidated on an address space switch* **(0.5 P)**.

d) Wie groß ist der maximal adressierbare virtuelle Speicher bei einer dreistufigen Seitentabelle mit 4 Einträgen in der oberste Seitentabelle und 512 Einträgen in den folgenden Stufen? Gehen Sie davon aus, dass jede Seite 4 KiB umfasst und der Speicher byte-weise adressierbar sein soll.
Wieviele Bits muss die virtuelle Adresse mindestens lang sein, um den gesamten Adressraum abzudecken? **1 pt**

What is the size of the maximally addressable virtual memory when using a three-level page table hierarchy with 4 entries in the top-most level and 512 entries in each following level? Assume each page to be 4 KiB in size and that the memory should be byte-addressable.
What is the minimal length (in bits) for a virtual address that can cover the entire address space?

Lösung:

`MaxAddressableVirtualMemory` $= (2^2 * 2^9 * 2^9) * 2^{12} = 2^{32} = 4\,GiB$ **(0.5 P)**
`MinVirtualAddressLength` $= 32$ **(0.5 P)**

e) Diskutieren Sie Vor- und Nachteile von Demand-Paging und Pre-Paging. **2 pt**

Discuss advantages and disadvantages of demand-paging and pre-paging.

Lösung:

Demand-paging only transfers what is needed **(0.5 P)** *and thus potentially uses less memory* **(0.5 P)**. *Pre-paging on the other side can improve I/O throughput by reading in larger chunks* **(0.5 P)** *while demand-paging produces many small requests (for individual pages). Pre-paging may, however, waste I/O bandwidth if pages are not needed* **(0.5 P)**.

Total: 12.0pt

Aufgabe 5: Hintergrundspeicher und Dateisysteme

Assignment 5: Secondary Storage and File Systems

a) Was versteht man unter Logical Block Addressing (LBA) zur Adressierung von Festplattensektoren? Nennen Sie einen Vor- und einen Nachteil gegenüber Cylinder Head Sector (CHS) Adressierung. Begründen Sie Ihre Antwort. **2 pt**

Explain the term logical block addressing (LBA) to address disk sectors. Give one advantage and one disadvantage compared to cylinder head sector (CHS) addressing. Explain your answer.

Lösung:

LBA (Logical Block Addressing) treats the whole disk as a single array of blocks **(1 P)**. *It is the prevalent technique today, because it hides disk geometry details and thus simplifies access* **(0.5 P)**. *However, LBA makes is difficult for operating systems to optimize disk accesses (e.g., to minimize seek times), because the physical placement of sectors on the disk is unknown* **(0.5 P)**.

b) Ein Verbund aus vier Festplatten kann für unterschiedliche RAID Level konfiguriert werden. Wie viele Festplatten können in der jeweiligen Konfiguration ausfallen, bevor Daten endgültig verloren gehen? Begründen Sie jeweils Ihre Antwort. **4 pt**

An array of four hard disks can be configured for different RAID levels. How many disks can fail in each configuration before data is permanently lost? Explain your answer.

Lösung:

RAID 0: 0 **(0.5 P)**, *stripping, no redundancy* **(0.5 P)**.
RAID 1: 3 **(0.5 P)**, *mirroring, all disks contain identical contents* **(0.5 P)**.
RAID 4: 1 **(0.5 P)**, *data is stored on 3 disks, parity information is stored on a separate disk, thus reconstruction is only possible if one disk fails* **(0.5 P)**.
RAID 5: 1 **(0.5 P)**, *same as RAID 4, but parity information is distributed across all disks* **(0.5 P)**.

c) Moderne Betriebssysteme verwenden für SSDs den `trim`-Befehl. Warum wird dieser Befehl nicht für HDDs verwendet? **1 pt**

Modern operating systems use the `trim` command for SSDs. Why is this command not used for HDDs?

Lösung:

In contrast to SSDs, HDDs do not require to release and explicitly erase data before the space can be overwritten or reused.

d) Nennen Sie vier Metadaten, die das Betriebssystem für eine geöffnete Datei halten muss. Welche der Informationen werden systemweit nur einmal, welche pro geöffneter Instanz gespeichert? **2 pt**

Give four pieces of metadata that the operating system has to store for an open file. Which information is stored once per system and which once per open file instance?

Lösung:

per system: *file-open counter* **(0.5 P)**, *information on cached file parts* **(0.5 P)**
per instance: *access mode* **(0.5 P)**, *file pointer* **(0.5 P)**

e) Erläutern Sie Unterschiede in der Implementierung von harten und symbolischen Links. Geben Sie jeweils ein Verwendungsbeispiel an. **3 pt**

Explain the implementation differences of hard and soft links. Give a usage example for each.

Lösung:

While hard and soft links both get a dedicated directory entry, hard links of the same file share the same inode **(0.5 P)**. *Creating a new hard link only increments a reference counter in the inode* **(0.5 P)**. *Hard links can be used for incremental backups* **(0.5 P)**.

Soft links, on the other side, are regular files **(0.5 P)** *that contain the absolute or relative path to the destination* **(0.5 P)**. *A directory can be moved to another disk or file system, but can still be accessed from the original location* **(0.5 P)**.

Total: 12.0pt