

## Lösung: System Architektur Klausur 25. März 2008

### Aufgabe 1 / Question 1 (Zum Aufwärmen/Warm up, 2 + 2 + 1 + 1 + 6 Punkte/marks)

1. „An welchen **vier Systemzielen** (*system objectives*) sind Nutzer eines Dienstleistungsrechnersystems am meisten interessiert?“  
 “For users of a central computer service system, what are the **four** most interesting **system objectives**?”

**Customizability**

**Performance**

**Quality of Service**

**Security & Protection**

2. „Zählen Sie **zwei** Möglichkeiten auf, wie man **Parameter** eines **Systemaufrufs** (*system call*) aus der Anwenderenebene in die geschützte Kernebene übergeben kann!“  
 “Enumerate **two** possibilities how to hand over **parameters** of a **system call** from application level to the protected kernel level.”

**Register**

**User-Stack**

3. „Wollte man **Verklemmungsverhinderung** (*deadlock prevention*) implementieren, welche der vier notwendigen Verklemmungsbedingungen würde man dann angreifen?“  
 “In case you want to implement **deadlock prevention**, which of the four necessary deadlock conditions would you attack?”

**No-Circular-Wait-Condition**

Einige der folgenden Aussagen sind korrekt, einige sind 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.

4. „Beim **Halbierungsverfahren** (*buddy system*) kann es sowohl zum **internen** (*internal*) als auch zum **externen** (*external*) **Verschnitt** (*fragmentation*) kommen.“  
 “In a **buddy system** there might be **internal** as well as **external fragmentation**.”

korrekt

inkorrekt

5. „Der **virtuelle Adressraum** eines **IA64 Rechners** ist **größer** als der korrespondierende **physische Adressraum**.“  
 “The **virtual address space** of an **IA64 machine** is **larger** than the corresponding **physical address space**.”

korrekt

inkorrekt

6. „Das **Auffinden** eines **bestimmten Datensatzes** (*record*) in einer **sehr großen B\*-Baum-indexierten sequentiellen** Datei ist **optimal** bezüglich der Zahl der Plattenzugriffe.“

*“Looking up a specific record in a huge B\*-tree indexed sequential file is optimal with respect to the number of disk accesses.”*

korrekt

inkorrekt

7. „Der **trap** oder **int** Befehl, mit dessen Hilfe man bei einem Systemaufruf (*system call*) vom Nutzermodus in den Kernmodus gelangen kann, ist ein normaler, d.h. **nicht privilegierter** Prozessorbefehl.“

*“The trap or int instruction that can be used to switch from user mode to kernel mode during a system call is a normal, i.e. non-privileged processor instruction.”*

korrekt

inkorrekt

8. „In einem Dateisystem kann es **keinen externen Verschnitt** (*external fragmentation*) geben, da alle Speichereinheiten, d.h. die Plattenblöcke, die gleiche Größe haben.“

*“There can be no external fragmentation in a file system, because all storage units, i.e. the disk blocks, have the same size.”*

korrekt

inkorrekt

9. „Damit ein **reiner Anwenderthread** (PULT) einen anderen PULT der **gleichen Task aufwecken** kann, muss die **umgebende Task** im **Zustand rechnend** sein.“

*“To allow a pure user-level thread (PULT) to wakeup another PULT of the same task, its surrounding task must be in the state running.”*

korrekt

inkorrekt

10. „Ein **fairer PULT-Scheduler** **garantiert**, dass in einem Mehrprogrammsystem (*multi programming*) **jede Anwendung** einen **fairen CPU-Anteil** erhält.“

*“A fair PULT-scheduler guarantees that each application gets a fair share of the CPU in a multiprogramming system.”*

korrekt

inkorrekt

## Aufgabe 2 / Question 2

(1 + 3 + 2 + 4 + 2 Punkte/marks)

1. „Welches **Prinzip** der **Systemstrukturierung** hat Dijkstra benutzt, um sein **THE System** zu entwerfen?“

*“Which principle of structuring systems did Dijkstra use to design his THE system?”*

**Layered Model (Schichtenmodell)**

2. „Beschreiben Sie die **wesentlichen Eigenschaften** einer **komponenten-basierten** oder **server-orientierten Systemarchitektur!**“

*“Describe the **fundamental characteristics** of a **component based** or **server oriented system architecture.**”*

**A server-oriented architecture (SOA) is more robust, because a failure within one server only crashes this server, all other services can still survive, no unwanted side effects of a buggy server. In a macro-kernel the same failure might lead to a system crash.**

**A SOA needs fast IPC, otherwise it's too slow compared to a macro-kernel based system.**

**A SOA can be extended quite easily, just add a new server, that's it.**

**A SOA is easier to maintain due to less complexity per server.**

3. „Erklären Sie, **warum das Parallelisieren** von Anwendungen (*applications*) auch auf einem **Einprozessorsystem Vorteile** bringen kann!“

*“Explain why **parallelizing applications** can be **advantageous** even on a **single-processor.**”*

**When some of the KLTs invoke synchronous I/O from time to time or must wait for results, some other KLT is calculating, the kernel scheduler could switch to a KLT of the same task thus speeding up the turnaround time of the whole task.**

4. „Diskutieren Sie die **zwei Aspekte**, weswegen man **virtuelle Maschinen** bei der **Entwicklung** von **Betriebssystemen** einsetzen kann.“

*“Discuss the **two aspects** for which **virtual machines** can be used in **OS development.**”*

**Virtual machines support the possibility to develop and test new system features, while the users still work with the old ones. Booting the new system is faster, and no additional hardware is needed.**

**Virtual machines also support specific OS versions that might be the only port of a specific application-system or a data base. With virtualization, each legacy application can be run on top of its preferred operating system in a virtual machine, while current applications can be executed on an up-to-date version or OS in another virtual machine—all on the same hardware. This approach allows the OS developers to add/change/improve the OS API while the administrators can still run the software requested by their users.**

5. „Unterstreichen Sie diejenigen der unten angegebenen **Makrokern**-Datentypen, die **nicht** Bestandteil eines **modernen Mikrokerns** sind.“

*“Underline those of the below mentioned **macro-kernel** data types that are not part of a **modern micro-kernel.**”*

PULT control block

KLT Kernel stack

Bitmap for free-/allocated disk blocks

KLT control block

**Aufgabe 3 / Question 2****(2 + 2 + 2 + 6 Punkte/marks)**

1. „Erläutern Sie den Unterschied zwischen einer **kritischen Region** (*critical region*) und einem **kritischen Abschnitt** (*critical section*)!“

*“Explain the difference between a **critical region** and a **critical section**.”*

**A critical section CS is a code path within a parallel application (either part of a thread or a process) that has to be executed exclusively concerning all other related CSs of the same critical region. (Typically a CS code accesses common data -either global data of a task or part of a shared memory object-, or uses an exclusive resource)**

**A critical region consists of all related critical CSs within a parallel application.**

2. „Erklären Sie den Begriff „**geschachtelter Monitor** (*nested monitor*)“! Unter welchen Umständen kann es bei der Verwendung geschachtelter Monitore zu **Verklemmungen** (*deadlocks*) kommen?“

*“Explain the notion of a **“nested monitor”**! Under which circumstances can the use of nested monitors lead to a **deadlock**?”*

**A nested monitor consists of at least two different software monitors, whereby we distinguish between an outer (or lower) and an inner (or higher) monitor. They are nested when inside a monitor function of the outer monitor another monitor function of the inner monitor is called.**

**Deadlocks can arise when in a monitor function of the inner monitor a function of the outer monitor is called.**

3. „Beschreiben Sie die Eigenschaften, die eine **gültige Lösung** eines wechselseitigen Ausschlussproblems besitzen muss, damit sie die **Portabilitätsanforderung** erfüllt.“

*„Describe the characteristics that a **valid solution** for a mutual exclusion problem must have in order to fulfill the requirement **portability**.“*

**Solution must be valid independent of number of CPUs and of scheduling policy**

4. „Sie müssen ein **1:n-„AND“-Kanalobjekt** (*broadcast communication channel*) im Kern implementieren. Welche Gestaltungsparameter können Sie hierbei noch frei wählen? **Diskutieren** Sie deren **Auswirkungen** auf die Nutzung dieses Objekt durch KLTs!“

*“You must implement a **1:n-“AND”-broadcast communication channel** in the kernel. What design parameters can you still chose arbitrarily? **Discuss their implications** on the usage of this communication object by KLTs.”*

**Fixed design parameters:**

- a) **Indirect addressing**
- b) **Connectionless**
- c) **Data transfer must be via memory as long the messages are different in size. Data transfer can only be in registers if two constraints are met: 1. Short messages and 2. Synchronous send, whereby the sender is blocked until the last receiver has consumed the message. Using a kernel channel you must deal with copy-in and copy-out.**

**Selectable design parameters:**

- a) **Synchronization:** good choice asynchronous sender & synchronous receiver, but there is the danger of flooding the kernel. With a synchronous send a sender might be blocked until the latest receiver has called receive. However, you could also say, that the sender might proceed when the first receiver has consumed the message, or you can say that when x% of all attached receiver have consumed the message you will wakeup the “semi-synchronous” sender.
- b) **Deterministic number of receivers,** i.e. at run-time the number of attached receiver stays constant. But you can also allow dynamic attaching, but then you must solve the problem, how long a message should stay in message-queue, if some attached receiver is blocked for a very long time. Furthermore you can chose, whether a receiver is allowed to consume the next message if a previous message is still in the message queue, because another lazy receiver has not yet done its job.
- c) **Messages can be tagged thus enabling a selective broadcast,** i.e. only receivers with the tag can receive the message, others have to wait for an appropriate message.

**Aufgabe 4 / Question 4****(4+ 2 + 2 + 4 Punkte/marks)**

1. „Zählen Sie **jeweils zwei unterschiedliche** Stellen in einem modernen Mikrokern bzw. in einem traditionellen Makrokern (aber nicht im Mikrokern) auf, an denen die Funktion **Thread\_Switch** aufgerufen werden kann!“  
*“Enumerate **two different** places in both kernel types, i.e. micro respectively macro kernel (but not in a micro kernel) where the function **Thread\_Switch** can be called.”*

Mikrokern (*micro kernel*)**send\_message****receive\_message**Makrokern (*macro kernel*)**read\_file****allocate\_resource**

2. „Weswegen muss ein Systemarchitekt in der Lage sein, **temporär Unterbrechungen** (*interrupts*) **ausmaskieren** können? Erklären Sie den **allgemeinen Verwendungszweck** und geben Sie dann noch ein **ganz konkretes Fallbeispiel** an!“  
*“Why must a system architect be able to **mask interrupts temporarily**? Explain the **general purpose** and then give a **specific example**.”*

**In the kernel there might be a very critical path (i.e. accessing the global ready queue) that should run exclusively, i.e. without any interruption. Another example is a semaphore, i.e. within a p-Operation testing the semaphore value you do not want to be interrupted by a time-slice interrupt, otherwise the semantics of an atomic p-operation is no longer valid.**

3. „Erklären Sie, welche Aktionen bei einem **Systemaufruf** (*system call*) direkt von der **Hardware** und welche von der **Software** ausgeführt werden.“  
 “*Explain what actions are executed directly by hardware respectively by the software during a system call.*”

**HW does the mode switch including setting of kernel mode-bit & saving the user-level context either onto the current kernel-stack or into shadow registers  
 Jump to the start address specified via the index of the “interrupt-vector table”**

**SW extracts the necessary parameters from the user-stack or from shadow registers & calls the corresponding system call handler, masking out maskable interrupts.**

4. „Welche **Metadaten** gehören zu einer **B\*-Baum-indizierten sequentiellen Datei**?“  
 “*Which meta data belong to a B\*-tree-indexed sequential file?*”

**Header of a doubled-linked list of all the leave nodes of the B\*-tree enabling sequential access.**

**Root of B\*-tree enabling direct access via further inner nodes.**

**Inner nodes containing the corresponding maximal index of the subtree respectively the leave node, i.e. the data container**

**All other user and file related information, i.e. file owner, time of last modifying access, access rights etc.**

5. „Welche **Einträge** enthält sogar ein „leeres“ ext2-Verzeichnis?“  
 “*Which entries does even an “empty” ext2 directory contain?*”

**.. = current directory**

**.. = parent directory**

## Aufgabe 5

(2 + 2 + 2 + 2 + 2 + 2 Punkte)

1. „Charakterisieren Sie das Speicherverfahren **Slab-Allocator** hinsichtlich **vier orthogonaler Entwurfparameter!**“  
 “Characterize the **slab allocator** according to **four orthogonal design parameters.**”

**No internal and no external fragmentation**

**Size of memory object per slab is constant**

**General allocation and release sequence**

**Separated data structure to manage the slabs**

**No (or very lazy) reunification, as long as the slab is used**

**Allocation policy ~ last release first reused**

2. „Wie viele Seitenfehler (*page faults*) können auf einem **CISC-Rechner** während der Ausführung eines „Zwei Operanden Befehls“ auftreten? Begründen Sie Ihre Antwort!“  
 “*How many page faults on a CISC-machine can occur during the execution of a two-operand instruction? Explain your answer!*”

**With no indirect addressed operands 4(6), because a CISC instruction might need two pages and each of the two operand might be on a separate page, (as well as each operand)**

3. „Angenommen in einem System mit **transparenter Superseitenunterstützung** tritt **Speicherdruck** auf. Wie kann man darauf reagieren?“  
 “*Suppose in a system with transparent support for super pages there will be memory pressure. How can you react?*”

**You can always swap out a disturbing task, but you can also demote super pages, i.e. you chose the most promising super page and swap it out, if modified or just declare it no longer mapped and try to satisfy the current page fault, however only using smaller super page-frames or even the standard page frame.**

4. „Angenommen die Hardware bietet die **üblichen drei Kontrollbits** **r**(ead),**w**(rite),**(e)x**(ecute) an, um falschen Zugriff auf seine Hauptspeicherkacheln zu erkennen. Warum verwenden einige Systemarchitekten, deren System kompatibel zu früheren Systemversion sein muss, obiges **x-Bit** nicht zum Schutz ihrer Codeabschnitte an?“  
 “*Suppose the hardware offers the usual access control bits r(ead), w(rite), and (e)x(ecute), to detect illegal access of its main memory page frames. Why do some system architects, whose system must be compatible with earlier system versions, not use this x-bit to protect their code sections?*”

**An old application using self modifying code must still be runnable on the system**

5. „Ihr System läuft ins **Thrashing** hinein. Welche der aktuell eingelagerten Tasks/Prozesse würden Sie **auslagern**? **Begründen** Sie Ihre Ansicht!“  
 “*Assume your system is thrashing. Which of the currently swapped-in tasks/processes would you would swap out? Reason your choice.*”

**The task with the longest remaining processing time, the increased turnaround time is relatively small.**

**The task with largest resident set, because then the remaining task can profit most.**

6. „Wie lautet der **Linux Systemaufruf** (*system call*), mit dessen Hilfe ein **neuer KLT** zur Laufzeit erzeugt werden kann? Der als KLT auszuführende Code muss **eine bestimmte syntaktische** Eigenschaft haben. Welche?“  
 “*What is the name of the Linux system call by which a new KLT can be created? The code that will be executed as this new KLT must has a certain syntactical property. Which one?*”

**clone()                    the code must be a function (procedure)**