

## Aufgabe 1/Question 1

1. „Welche der folgenden Ereignisse können dazu führen, dass ein „**Pure User-Level Thread**“ (PULT) in den **Threadzustand „blockiert (waiting)“** wechselt?“ (Unterstreichen Sie Zutreffendes!)  
*“Which of the following events can cause a **pure user-level thread (PULT)** to switch to the **thread state “waiting”**?” (Underline accordingly)*
  - a) **p( ) operation of a kernel semaphore**
  - b) **Blocking I/O system call**
  - c) User-level monitor procedure
  
2. „Zählen Sie **drei mögliche quantitative Leistungsmaße** (performance measures) des **CPU-Scheduling** auf.“  
*“Enumerate **three potential quantitative performance measures of CPU scheduling.**”*

<b>Througput</b>	<b>Turnaroudtime</b>	<b>Responsetime</b>
------------------	----------------------	---------------------
  
3. „Der **Kernstapel** (kernel stack) eines KLT ist **leer**, wenn der KLT im **Nutzermodus** (user mode) rechnet.“  
*“The **kernel stack of a KLT is empty** when the KLT is running in **user mode.**”*

<u>korrekt</u>	inkorrekt
----------------	-----------
  
4. „Bei **spärlich besetzten Adressräumen** ist der **Platzbedarf** einer **mehrstufigen Seitentabelle** (multi-level page table) **geringer** als derjenige einer **linearen Seitentabelle**.“  
*“For **sparsely occupied address spaces** the **storage requirement of a multi-level page table is less than that of a linear page table.**”*

<u>korrekt</u>	inkorrekt
----------------	-----------
  
5. „In **Linux** sind **Gerätetreiber** typische Vertreter **ladbarer Kernmodule**.“  
*“In **Linux device drivers** are a typical use case of **loadable kernel modules.**”*

<u>korrekt</u>	inkorrekt
----------------	-----------
  
6. „In einem **Einprozessorsystem** ist ein **Spinlock** auf **Anwenderebene** eine **gültige Lösung** eines **kritischen Abschnittproblems**.“  
*“In a **single processor system** a **spinlock used at application level** is a **valid solution of a critical section problem.**”*

korrekt	<u>inkorrekt</u>
---------	------------------
  
7. „**Prozessoraffinität** und **Migration** sind **unvereinbare** Ablaufplanungskonzepte (scheduling concepts).“  
*“**Processor affinity and migration** are **incompatible scheduling concepts.**”*

korrekt	<u>inkorrekt</u>
---------	------------------
  
8. „In Einprozessorsystemen mit hierarchischer und einmaliger Betriebsmittelnutzung ist das „**Non Preemptive Critical Section**“ Betriebsmittelzuordnungsprotokoll **verklemmungs-frei**.“  
*“In **single processor systems with a nested and exclusive resource usage**, the “**Non Preemptive Critical Section**” resource allocation protocol is **deadlock free.**”*

<u>korrekt</u>	inkorrekt
----------------	-----------

**Aufgabe 2/Question 2****(4 + 2 + 6 Punkte/marks)**

1. „Erläutern Sie möglichst **vollständig** die **Vorteile** des **PULT-Threadmodells!**“  
*“Explain as **completely** as possible the **advantages** of the **PULT-thread model.**”*

- + **Specific user level scheduling policy**
- + **Fast user level thread operations (no kernel entry/exit)**
- + **Smaller UTCB**
- + **Usable on systems without KLTs (portable)**
- + **Kernel can be kept more simple**
- + **(no real parallelism)**

2. „Annahme: Der **Kernumschalter** (*kernel scheduler*) überführt eine Applikationstask aus **p>1 PULTs** in den **Taskzustand** „**rechnend**“. Welche Situation herrscht dann vor, wenn **alle** p PULTs blockiert sind? Schließen Sie alle Signalbehandlungsroutinen (*signal handler*), die auf externe Ereignisse reagieren, aus.“  
*“Suppose the **kernel scheduler** transfers an application task -consisting of **p>1 PULTs**- into the **task state running**. Which situation occurs when **all** p PULTs are blocked? Ignore all signal handlers that react to external events.”*

**Deadlock or Livelock**

3. „Erläutern Sie das **hybride Threadmodell** und diskutieren Sie dessen Vor- und Nachteile!“  
*“Explain the **hybrid thread model** and discuss its **advantages** and **disadvantages!**”*

- + **p ≥ 1 “fellow” PULTs are mapped to one KLT**
- + **A PULT blocking in a syscall only blocks its “fellow” PULTs but not the entire task**
- + **You can use multiple CPUs concurrently (as long as it pays off)**
- + **Thread operations within the fellowship are still fast and might profit from a dedicated user level scheduler**
- **However, now both schedulers have to cooperate (e.g. via scheduler activations)**
- **Upcalls like scheduling activations leak a multi-layered architecture**
- **Within the kernel you have to provide and distinguish between TaskCBs and TCBs**
- **HL model more complicated**

4. „Zählen Sie mindesten jeweils ein **kommerzielles System** und ein **Forschungssystem** auf, das ein **hybrides Threadmodell** anbietet!“  
*“Enumerate at least one **commercial system** and one **research system** that provides a **hybrid thread model.**”*

**Solaris (Unix, Linux, Windows)****Scheduler Activations (NetBSD, K42, HThreads of Kansas University)**

**Aufgabe 3/Question 3 :****(2 +2 + 8 Punkte/marks)**

1. „Ein Mehrprozessorsystemkernablaufplaner (*SMP kernel scheduler*) kann entweder eine zentrale Bereitwarteschlange (*ready queue*) oder eine getrennte Bereitwarteschlange pro Prozessor verwenden. Erläutern Sie Vor- und Nachteile einer zentralen Bereitwarteschlange!“  
*“A multi-processor kernel scheduler can choose to use either a central or a per-processor ready queue. Discuss the pros and cons of using a central ready queue.”*  
**+ You don't need load balancing, each CPU takes the first ready KLT.**  
**+ The kernel scheduler can achieve efficiently a consistent policy (e.g. strict priority).**  
**- Enabling CPU-affinity is more complicated  $\Rightarrow$  assigning next KLT might be slower.**  
**- Does not scale very well, because the central ready queue might become a bottleneck with  $n \gg 1$  CPUs.**
2. „Der Kernablaufplaner in älteren Linux-Systemen basierte auf einem „Arbeitsklaualgorithmus“ (*work stealing*), um die Last unter den verschiedenen Prozessoren auszugleichen. Hierbei stiehlt ein Prozessor, der ansonsten leer laufen würde, einen Prozess aus einer fremden Bereitwarteschlange. Warum und in welchen Fällen erreicht dieser Arbeitsklaualgorithmus keine Ausgewogenheit der Arbeitslast?“  
*“The multi-processor scheduler in older Linux systems used a work-stealing algorithm to balance load between different processors. In this work-stealing algorithm a processor that falls idle steals a process from another read-queue. Why and in which cases does this work stealing algorithm fail to perform proper load balancing?”*  
**As long as there is at least one KLT on each CPU no work stealing will take place, i.e. suppose  $n-1$  long running KLTs are mapped to CPU0 ... CPU $n-1$ , whereas CPU $n$  is busy with  $m \geq 1$  KLTs.**
3. „Erläutern Sie das Konzept **Co-Scheduling** und geben Sie hierfür **typische Anwendungen** an!“  
*„Explain the concept of co-scheduling and state typical applications of this concept.”*  
**KLTs of one multi-threaded application (or cooperating application processes) are mapped to  $m > 1$  CPUs at the same time.**  
**KLTs of a numerical solution with low interaction on shared memory that can profit from concurrent execution.**
4. „Warum bieten manche Systemarchitekten nur eine **synchrone:synchrone IPC** an der **Kernschnittstelle ihres Betriebssystems** an?“  
*“Why do some system architects offer only synchronous:synchronous IPC at the kernel interface of their operating system?”*  
**No buffering in the kernel required.**  
**To avoid a logical bomb within the kernel, i.e. prevent denial of service attacks with nonsense messages that exhaust the kernel message pool.**
5. “Skizzieren Sie, wie man dem Anwenderprogrammierer **asynchrone:synchrone IPC** zur Verfügung stellen kann, wenn ihr Betriebssystem an der Kernschnittstelle nur synchrone:synchrone IPC anbietet!“  
*„Outline how you can offer asynchronous:synchronous IPC to the application programmer, when your operating system only offers synchronous:synchronous IPC at its kernel interface.”*  
**Install a message pool of  $s$  slots with the AS of the sender.**  
**Per slot install a separate message handler thread that buffers each asynchronous message until the receiver wants to get it.**

## Aufgabe 4/Question 4

(2 + 1 + 2 + 1 + 6 Punkte/marks)

1. „Erläutern Sie die Vor- und Nachteile einer **lokalen Seitenersetzungsstrategie** (*local page replacement policy*).“

“*Explain the advantages and disadvantages of a local page replacement policy.*”

- + Offers better suited and faster control over each application’s resident set.
- + Suppose you got an appropriate resident set for a (multi-threaded) application, then you might predict the turnaround time of the application quite accurately.
- Defining the appropriate resident set size might be a problem. If the user is allowed to define it, he/she will always overestimate this size.
- Thrashing can occur when resident set size  $\ll$  working set size.

2. „Warum ist die Kachel- bzw. Seitengröße typischerweise eine **Zweierpotenz**?“

“*Why is the size of a page frame or page typically a power of two?*”

- In the PTE you need fewer bits for storing the frame number.
- You can concatenate the offset with the frame number.

3. „Einige Systeme offerieren **hardwareunterstütztes Seitentabellendurchlaufen**, andere bevorzugen eine Softwarevariante. Diskutieren Sie die Vor- und Nachteile der Hardwarevariante.“

“*Some systems offer hardware supported page-table walking, others prefer software walking. Discuss the pros and cons of the hardware variant.*”

- + HW PTW is faster, but needs a specific page table structure.
- Thus it is less flexible and it is expensive for huge AS layouts.

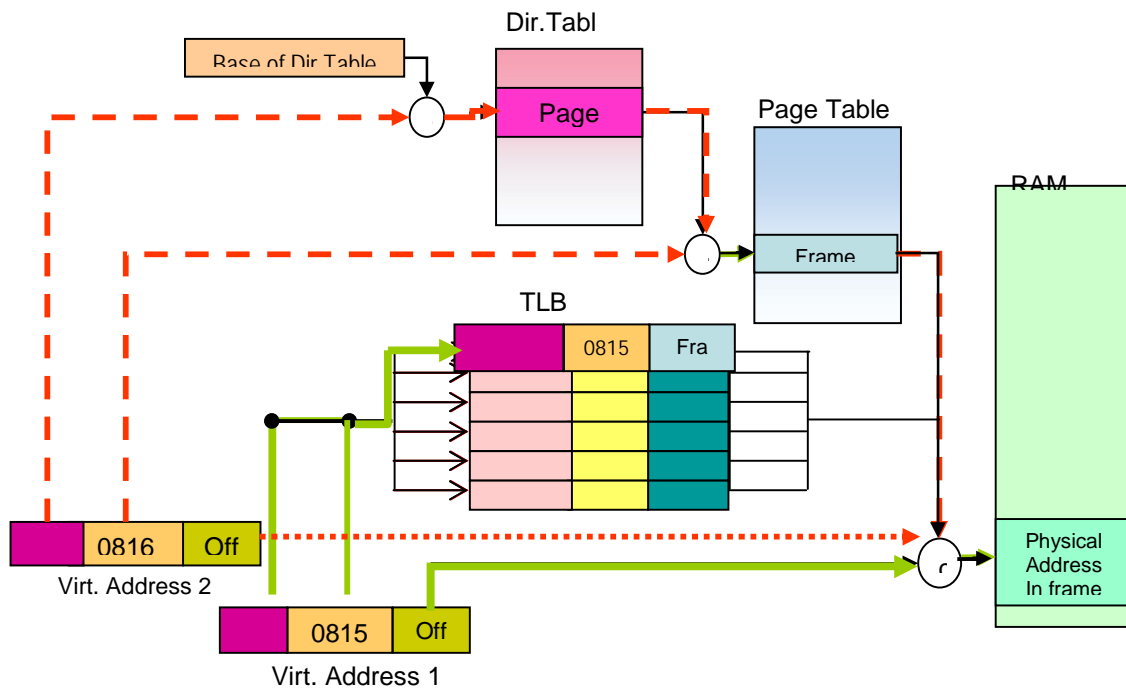
4. „Erläutern Sie, **wann** und **warum** „Copy-on-Write“ in einem Linux System angewendet wird!“

“*Explain when and why “copy-on-write” is used in a Linux system.*”

**With each `fork()` you do not need to copy all data regions of the parent, that might be exchanged soon after if the child calls `execve()`, i.e. if it loads a new program image.**

5. „Skizzieren Sie, wie bei einer **zweistufigen** Seitentabelle mit **TLB** die **Adress-transformation** von virtueller zu physischer Adresse funktioniert. Aus der Skizze sollte ferner erkenntlich sein, wie ein **TLB-Miss** und ein **Seitenfehler** erkannt werden kann.“

“*Outline a draft, how address transformation from virtual to physical address works with two-level paging and a TLB. The draft should indicate how a TLB-Miss and how a page-fault can be detected.*”



You need a third virtual address that has an entry in the page table (PTE) with a missing frame number.

Textual solution:

You need a register providing the base address of the directory table.

In case of a TLB hit there is a TLB entry containing the pair:

<dir. + page number, physical frame number>

You only have to concatenate the offset of the logical address with the frame number.

In case of a TLB Miss there is no corresponding entry in the TLB, thus you have to parse both tables to get the desired frame number that you can concatenate again.

In case of a page fault there is no frame number within the page table, i.e. you have to start the paging algorithm to map the corresponding page from disk to RAM.

**Aufgabe 5/ Question 5:**

1. „Gegeben sei eine **erweiterbare Hashdatei** (*extensible hash file*), die **200** Datencontainer besitzt. **Wie viele Einträge** muss der zugehörige Basisvektor **mindestens** enthalten?“  
 “Given an *extensible hash file* that holds **200** data containers. **How many entries** must the according base vector have **at least**?”  
**256**
2. „Aus welchen **Feldern** (*fields*) bestehen die **Einträge** des Basisvektors einer erweiterbaren Hashdatei?“  
 “What are the *fields* of an *entry* of the base vector of an *extensible hash file*?”

**Generation number**

**logical block address of the data container**

3. „Wie würden Sie die **Metadaten** einer erweiterbaren Hashdatei **ergänzen**, so dass **gleichzeitig möglichst viele konfliktfreie Dateioperationen** möglich sind, während konfliktträchtige Dateioperationen serialisiert werden?“  
 “How would you *extend* the *meta data* of an *extensible hashing file* in order to allow as **many concurrent non-conflicting file operations** as possible, whereas *conflicting file operations* are *serialized*?”

**Hierarchical locking protocol, i.e. a central mutex for the base vector and per container a reader/writer lock.**

**Inserting a new element requires the top lock and all reader/writer locks. This also solves the extension of the base vector.**

**Deleting an element or writing an element only temporarily requires the central lock and additionally the write-lock of the container. Having gotten the writer-lock you can release the central lock.**

**Reading also temporarily requires the central lock and then the read lock of the corresponding container.**

4. „Analysieren Sie in der folgenden Programmskizze zur Lösung des Leser-/Schreiber-problems, ob die **vier notwendigen** Anforderungen und weitere **wünschenswerte Eigenschaften** einer gültigen Lösung von kritischen Abschnittproblemen erfüllt werden. Nehmen Sie an, dass beide Prozessarten die gleichen Prioritätswerte besitzen und dass die Semaphore **access streng** ist.  
 “Analyze in the following program draft of a solution of the reader-writer problem, whether the **four necessary requirements** and **additional desirable properties** are met to solve critical section problems. Suppose that both process types have the same priority values and that the semaphore **access** is **strict**.”

```
var semaphore mutex = 1;
var semaphore access = r; // r = maximal number of readers
process reader (integer i=1 ... r) //up to r readers {
  do
  {
    p(access);
    // ... reading ...
    v(access);
    // ... other operations
  }}

process writer (integer i=1 ... w) //up to w writers {
  do
  {
    p(mutex);
    for k = 1 ... r do p(access);
    // ... writing ...
    for k = 1 ... r do v(access);
```

7

```
v(mutex);  
// ... other operations  
}}
```

**Solution is not valid, because bounded waiting for writers is not guaranteed.**  
**Readers are favored, can overtake writers (especially with many readers and few writers).**  
**Writers can overtake writers (because mutex is not strict).**