

6. „In einem System, dessen multi-threaded Anwendungen nur aus PULTs bestehen, bewirkt jeder **blockierende Systemaufruf** (*system call*), der zum Blockieren führt, einen **Task-** oder **Prozesswechsel**.“

*“In a system whose multi-threaded applications only consist of PULTS, every **blocking system call**, that involves a blocking, invokes a **task or process switch**.”*

korrekt

inkorrekt

7. „**Datensätze** einer **erweiterbaren Hashdatei** können **sequentiell** gelesen werden.“
*“Data records of an extensible hash file can be read **sequentially**.”*

korrekt

inkorrekt

8. „Zur **Reduzierung der Plattenarmbewegungen** auf einem **Notebook** wird häufig verwendete Dateisysteminformation auf den **mittleren Zylindern** abgelegt.“

*“To **reduce the movement of the disk arm** on a notebook frequently used file system information is stored in cylinders in the middle of the track range.”*

korrekt

inkorrekt

Aufgabe/Question 2

(4 + 2 + 6 Punkte/marks)

1. „Erläutern Sie möglichst **vollständig**, welche **Konsequenzen** sich ergeben können, wenn ein PULT blockiert!“

*“Explain as **completely** as possible what **consequences** might arise when a PULT blocks.”*

- a) PULT blocking at user level causes the user level scheduler to look for another ready PULT (*as long as the kernel scheduler allows the task to go on running*)
 b) PULT blocking at kernel level (*due to a blocking system call*) ⇒ task will be blocked completely, *but PULT is still running*

2. „Threads werden durch **Systemdaten**(strukturen) unterstützt. Worin unterscheiden sich KLTs und PULTs hinsichtlich dieser **Systemdaten**?“

*“Threads are supported by **system data** (structures). How do KLTs and PULTs differ concerning these **system data**?”*

PULT TCB is located in user address space of the task and managed by user level library. PULT TCB might be tailored ⇒ contains less attributes than a KLT TCB. All PULTS share 1 KLT, kernel knows KLTs but not PULTS.

KLT TCB is located in the kernel, and is managed by the kernel. KLT is a compromise, thus must contain all potential attributes.

3. „Welches **Systemarchitekturproblem** trat im Rahmen der „Pathfinder“-Marsrerkundung auf und wie wurde dieses Problem schließlich gelöst?“
*“What **system architecture problem** did arise during the mars pathfinder mission and how was this problem solved?”*

Priority Inversion, detected by system shut down due to time outs in high priority process. Update system with an enhanced patch, introducing priority inheritance in the locking mechanism.

4. „Vergleichen Sie die **Eigenschaften** der **vier** in der Vorlesung behandelten **Betriebsmittelbelegungsprotokolle** (*resource allocation protocols*), welche das in Frage 2.3 angegebene Problem lösen können!“
*“Compare the **characteristics** of the **four** in the lecture mentioned **resource allocation protocols** which can solve the problem mentioned in question 2.3.”*

| RAP/Characteristic | Waiting Time | Deadlock free | No a priori knowledge |
|---------------------------|--|---------------|-----------------------|
| NPCS | <i>Longest lower priority CS</i> | Yes | No |
| Priority Inheritance | <i>Min (n, m) lower priority processes</i> | No | No |
| Priority Ceiling | <i>Longest lower priority CS</i> | Yes | Yes |
| Stack based Prio. Ceiling | <i>Longest lower priority CS</i> | Yes | Yes |

Aufgabe 3 / Question 3 :

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

1. „Zählen Sie die **vier Anforderungen** einer **Lösung** für ein **kritisches Abschnittsproblem** auf und geben Sie jeweils eine **knappe Erläuterung** für jede der Anforderungen! “
*“Enumerate the **four requirements** for a **solution** of a **critical section problem** and give a **short explanation** for each requirement.”*

Exclusiveness (at most 1 KLT in its CS), otherwise no exclusive critical section

Portability (no assumptions on speed or numbers of CPUs or on scheduling policy), otherwise race conditions

Progress (no KLT running outside its CS prevents another KLT from entering its CS) otherwise too restricted protocols

Bounded Waiting (if KLT is waiting on the CS, then this KLT will eventually enter its CS), otherwise a process may starve in front of its critical section

2. „**Analysieren** Sie die unten angegebenen Funktionen **enter_cs** bzw. **exit_cs** darauf hin, welchen den in 3.1 notierten Anforderungen sie **nicht** genügen!“

“Analyze the following function **enter_cs** and **exit_cs** which of the requirements of question 3.1 they do **not** satisfy.”

```
/* enter_cs          */
DO
    reg := MyThreadId; /* ∃ no thread with MyThreadId =0 */
    xchg (SpinLock,reg); /* atomic exchange operation */
    /* exchanges mem/cache variable SpinLock and reg */
UNTIL reg = 0 OD;
/* end of enter_cs  */
```

```
/* exit_cs          */
SpinLock := 0;
/* end of exit_cs  */
```

No portability (*on a single-processor system*) with strict priority scheduling the high priority process might never enter its critical section ⇒ system life lock

Bounded waiting (*on all systems*), i.e. a process can wait for a long time in front of its CS, as long as there are competitors

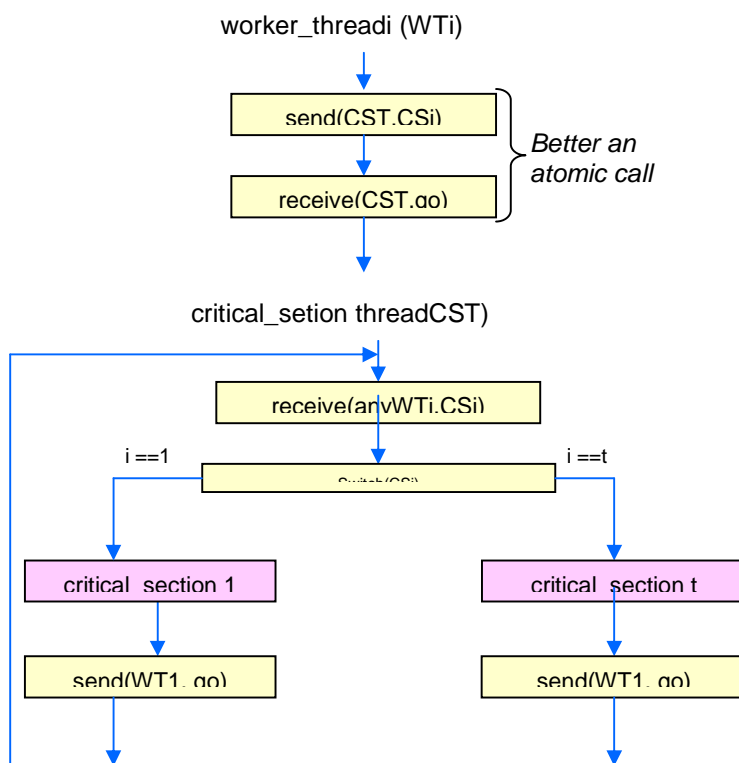
3. „**Verbessern** Sie die Implementierung der Funktion **enter_cs** aus 3.2 so, dass sie in einem **Mehrprozessorsystem effizienter** läuft.“

„Improve the implementation of the function **enter_cs** of 3.2 so that it works **more efficiently** on a **multi-processor system**.“

On SMPs there might be a mutual cache ping-pong effect within the do loop, because the atomic instruction invalidate the caches.

```
/* enter_cs          */
DO
    while SpinLock != 0 do {}; {Reading from main memory only once}
    reg := MyThreadId;
    xchg(SpinLock,reg)
UNTIL reg = 0 OD ;
/* end of enter_cs  */
```

4. „Skizzieren Sie entweder als Ablaufdiagramm oder als C-ähnliche Programm-schablone, wie Sie mittels **synchroner Kommunikationsoperationen (mit direkter Adressierung der Kommunikationspartner)** einen **wechselseitigen Ausschluss** für **$k > 1$ KLTs** einer multi-threaded Applikation herstellen können. Beachten Sie dabei, dass **jeder kritische Abschnitt CS_i** ein **verschiedenes Programm** ausführt.“
 “Outline either as a flow-diagram or as a C-like program template, how you would implement **mutual exclusion** for **$k > 1$ KLTs** of a multi-threaded application using **synchronous communication operations (with direct addressing of the communicating partners)**. However, regard that **each critical section CS_i** executes a **different program**.”



Aufgabe/Question 4

(3 + 3 + 6 Punkte/marks)

1. „Welche **Entwicklungen** haben die Autoren des Artikels „*Practical, transparent OS support for Superpages*“ **motiviert**?“
 “Which **developments** have **motivated** the authors of the article “*Practical, transparent OS support for Superpages*”?”

Modern micro processors supported different page sizes (see Intel).

RAM sizes grow faster than cost efficient TLBs, thus decreasing TLB coverage

Working sets of most applications has grown significantly in the past \Rightarrow increased # of TLB misses

Many TLB misses require access to main memory even if related info is already in the on chip caches (\Rightarrow degrading performance is even more expensive)

2. „Wie haben die Autoren diese **transparente Betriebssystemunterstützung** für **Superseiten** realisiert?“

*“How did the authors implement this **transparent OS support for superpages?**”*

Reservation based allocation of physical page frames, (i.e. in case of a page fault the OS tries to find a free contiguous memory region that is able to host the maximal desired future super-page around that current page)

Fragmentation control (trade off between large super pages and wasting RAM if no longer needed)

Promotion, if a certain % of a super page is already mapped, map the rest of it

Demotion (opposite of promotion, i.e. When to cut down a former super page into smaller fragments)

Eviction, if memory pressure is too high, and a super pages was active for a while

3. „Gegeben sei ein **seiten-basierter virtueller Speicher** (*paged virtual memory*), der mittels einer **invertierten Seiten-Kacheltabelle** implementiert ist. **Wie** und **wo** kann man die **Information** hinterlegen, wo auf der Platte die gerade **nicht im Hauptspeicher abgebildeten Seiten** liegen? “

*“Given a **paged virtual memory** being implemented via an **inverted page-table**. **How** and **where** can you implement the **information** where the currently **not mapped pages** are located on disk?”*

Via a **region data structure** per task/process whereby each region describes a contiguous address range within the corresponding address space. An entry in this region structure is the **starting address** on the disk, i.e. the offset of individual pages within each region can be calculated.

4. „Erklären Sie den Unterschied zwischen einem **Soft-TLB** und einem **Hard-TLB**.“

*“Explain the difference between a **soft-TLB** and a **hard-TLB**.”*

In case of a Hard-TLB-miss the **MMU parses the page-table** ⇒ restricts the page table model

In case of a Soft-TLB-miss the MMU raises an **exception, its handler pares the page-table**

5. „Was versteht man unter dem **Vorwärtsquäleffekt** (*thrashing*)? **Wie** kann er bemerkt werden? **Was** kann man darauf hin dagegen tun?“

*“What is **thrashing**? **How** can it be detected? **What** can be done to combat it?”*

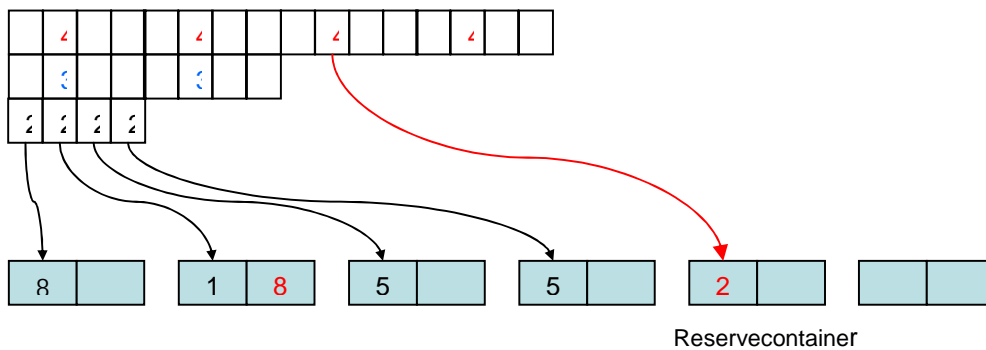
Effect: Too many concurrent tasks/processes have been activated ⇒ all (at least many) resident sets are too small ⇒ system is busy with paging in and out instead of executing applications.

Detection: Very slow progress of applications

Combat: Reduce multi-programming degree

Aufgabe 5/ Question 5:**(2 + 4 + 6 Punkte/marks)**

- „Gegeben sei eine **erweiterbare Hashdatei** (*extensible hash file*), deren Datencontainer **nur bis zu zwei Datensätze** (alle Datensätze sind gleich groß) enthalten. **Woran** kann dies liegen? Zutreffende Antworten unterstreichen!“
 “Given an *extensible hash file*, whose data containers only contain **up to two records** (all records are of same size). **How** can that happen? Underline the fitting answers.”
 - Container zu klein (*container too small*)
 - Ungünstige Hashfunktion (*bad hash function*)
 - Datensatz zu groß (*record too large*)
 - Verbesserte Zugriffszeit (*improved lookup*)
- „Füge in eine erweiterbare Hashdatei (à la Teilaufgabe 5.1) **Datensätze** mit folgenden **Schlüsseln** ein: **8, 17, 25, 50, 51** und **81**. Initial sei als Generationszahl 2 gegeben, so dass damit der initiale Basisvektor auf vier noch leere Datencontainer zeigt (siehe unten).“
 “Insert into an extensible hash-file (of type 5.1) **records** with following **keys: 8, 17, 25, 50, 51, and 81**. Initially the generation number is 2, i.e. the initial base vector has pointers to 4 empty data containers (see below).”



- „Können **alle Datensätze** einer erweiterbaren Hashdatei **ohne Kenntnis der Schlüsselwerte gelesen** werden? **Begründen** Sie Ihre Ansicht!“
 “Can **all data records** of an extensible hash file be **read without knowing the values of the keys**? **Explain your answer.**”

Yes, use the current base vector to get all data containers \Rightarrow you can get all data records

- „Erläutern Sie die **Vor- und Nachteile** von **RAID 0** im Vergleich zu einer teuren, großen Platte (vom Typ **SLED**)!“
 “Explain the **advantages and disadvantages** of **RAID 0** compared to a single large expensive disk (of type **SLED**).

+ : less expensive, improved reading due to concurrent disk requests or improved I/O bandwidth

- : less availability (each disk drive can break down)