

Solution Examination System Architecture march 2009

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

1. „Unterstreichen Sie diejenigen der folgenden Adressbereiche, die von Pure-User-Level-Threads (PULTs) einer mehrfädigen Applikation gemeinsam genutzt werden!“
“Underline those of the following address ranges that can be jointly used by pure-user-level threads (PULTs) of a multi-threaded application.”

User-Code

User-Heap

~~User-Stack~~

Kernel-Stack

2. „Unterstreichen Sie diejenigen unter den folgenden Anforderungen einer gültigen Lösung eines wechselseitigen Ausschlussproblems (*mutual exclusion*), die notwendige Anforderungen darstellen.“
“Underline those of the following requirements of a valid solution of a mutual exclusion problem that are necessary requirements.”

Begrenztes Warten/bounded waiting

~~Leistung/performance~~

Fortschritt/progress

~~Skalierbarkeit/scalability~~

3. „Nehmen Sie an, dass DIMMUNIX das erste Auftreten einer bestimmten Verklemmung (deadlock) entdeckt hat. Welche der vier notwendigen Bedingungen für das wiederholte Auftreten der gleichen Verklemmung wird zukünftig vermieden?“
“Assume DIMMUNIX has detected the first occurrence of a specific deadlock. Which of the four necessary conditions for the repeated occurrence of the same deadlock will be avoided in the future?”

Circular wait

4. „Das Betriebsmittelvergabeprotokoll „Nicht verdrängbarer kritischer Abschnitt“ vermeidet Verklemmungen auf Einprozessorsystemen.“
“The resource allocation protocol “non preemptive critical section” avoids deadlocks on single-processor systems.”

Korrekt

~~inkorrekt~~

5. „In einem System, das nur den Aktivitätstyp „Prozess“ unterstützt, muss im Fall einer Blockierung im Systemaufruf auch ein Adressraumwechsel ausgeführt werden.“
“In a system that only supports the activity type process, you must perform an address space switch in case of a blocking within a system call.”

Korrekt

~~inkorrekt~~

6. „In einem System mit virtuellem Speicher, welches nur Seiten mit Standardgröße unterstützt, kann es im Zusammenhang mit einem Seitenersetzungsverfahren auf Verlangen (*demand paging*) zu keinem externen Verschnitt (*fragmentation*) bei der Verwaltung des physischen Hauptspeichers kommen.“
“In a system with virtual memory that only supports pages of standard size, there can be no external fragmentation with regard to demand paging concerning the physical main memory.”

Korrekt

~~inkorrekt~~

7. „Bei DIMMUNIX kann es innerhalb einer mehrfädigen (*multi threaded*) Applikation aus $n \gg 1$ Kernel-Level Threads (KLTs) an einem bestimmten Mutex nur einmal zu einer Verklemmung (*deadlock*) kommen.“

“With `DIMMUNIX` in a multi-threaded application with $n \gg 1$ kernel-level-threads (KLTs) a deadlock at a specific mutex can occur only once.”

korrekt

inkorrekt

8. „Zwei mehrfädige Linux Applikationen, von denen eine aus PULTs und die andere aus KLTs besteht, können mittels Kommunikation (*IPC*) oder mittels gemeinsamen Speicher (*shared memory*) zusammenarbeiten.“

“Two multi-threaded Linux applications, one consisting of PULTs, the other consisting of KLTs, can work together by communication (*IPC*) or by shared memory.”

korrekt

inkorrekt

9. „Ihr Raid-System habe acht Platten. RAID 1 bietet Redundanz zu niedrigen Kosten.“

“Your RAID system has eight disks. RAID 1 offers redundancy at low cost.”

korrekt

inkorrekt

Aufgabe 2 / Question 2

(3 + 2 + 1 + 6 Punkte/marks)

1. „Als Linux Benutzer haben Sie mehrere Möglichkeiten, eine Datei zu löschen. Beispielsweise schreiben Sie ein Shell-Skript und benutzen das Kommando `rm` oder Sie schreiben ein C-Programm und benutzen den Systemaufruf (*system call*) `unlink`. Vergleichen Sie und bewerten Sie beide Methoden hinsichtlich ihres Aufwands an a) Laufzeit, b) Zahl benötigter Systemaufrufe, c) Zahl benötigter Adressräume!“

“As a Linux user you have multiple options to delete a file. For example you write a shell-script and use the command `rm` or you write a C-program and use the system call `unlink`. Compare and evaluate both methods with respect to their overhead concerning a) runtime, b) number of system calls, c) number of needed address spaces.”

a) Interpreting is more time consuming than executing opcode

b) Script: `fork` + `exec` for creating child shell, `fork` + `exec` for running `rm` binary, call of `unlink`

C-Prg: `fork` + `exec` to load and run C program, call to `unlink`

=> Script requires two additional syscalls

c) Script: 2 Address spaces: child shell + `rm`

C-Prg: 1 Address space (for the C-program)

2. „Der Taskzustand einer zweifädigen Applikation aus zwei PULTs sei rechnend. Einer der PULTs sei ebenfalls im PULT-Zustand rechnend, während der andere im PULT-Zustand blockiert sei. Geben Sie vier verschiedene Gründe an, weswegen dieser PULT blockiert sein könnte.“

“The task state of a two-threaded application consisting of two PULTs is running. One of its PULTs is in the PULT-state running, whereas the other one is in the PULT-state blocked. Enumerate four different reasons why this PULT can be blocked.”

`p()` operation on a user level semaphore

monitor function with a user level conditional wait

synchronous receive_message (user level IPC)

wait_signal wait for signal of another PULT of the task

Remark: All functions leading to a PULT state blocked must be library functions without any support from the kernel (1 bonus mark)

3. „Abgesehen von Stillstand- oder Verklemmungssituationen, welcher sonstige **gravierende Nachteil** ist mit **Spinlocks auf Applikationsebene** verknüpft?“
*“Despite the fact that spinlocks might lead to live- or deadlocks, what other **major disadvantage** might you encounter with **spinlocks at application level**?”*

Wasting CPU cycles, i.e. inefficiency or low performance

4. „Betrachtet werde eine Schwimmstaffel aus drei Schwimmern, bei der zweite Schwimmer erst starten darf, wenn der Erste fertig ist, und der Dritte erst starten darf, wenn der Zweite fertig ist. Die unten angedeuteten Schwimmer seien korrekt als **KLTs** erzeugt worden. Zur Synchronisation werden folgende Pseudocodefunktionen vorgeschlagen, die eine an der Kernschnittstelle angebotene „**schwache Semaphore SEM0**“ verwenden. Begründen Sie, warum der Vorschlag keine gültige Lösung darstellt! Modifizieren Sie den Vorschlag, so dass er gültig wird. Sie können die Korrekturen im Pseudocodevorschlag anbringen.“
“Regard a relay of three swimmers, whereby the second can only start when the first one has finished, as well as the third swimmer can only start when the second one has finished. The below mentioned swimmer in pseudo code have been correctly created. To synchronize the relay the below mentioned pseudo code has been proposed, using a weak semaphore SEM0 at the kernel API. Explain why the proposal is not a valid solution. Modify the proposal to get a valid solution. You can add the corrections within the pseudo code.”

```
/* global data declarations */
weak_semaphore SEM0 = 0;                               /* No initial Signal */
weak_semaphore SEM1 =0; (2)

swimmer1(){
    swim();
    V(SEM0);
}

swimmer2(){
    P(SEM0);
    swim();
    V(SEM1);
}

swimmer3(){
    P(SEM1);
    swim();
}
```

Begründung/Reasoning: Both swimmers (swimmer2 and swimmer3) compete for SEM0, no scheduling guarantee which one will get the SEM0 first

Aufgabe 3 / Question 3**(4 + 2 + 2 + 2 + 2 Punkte/marks)**

1. „Gegeben seien folgende beiden Kernel-Level Threads einer zweifädigen Applikation, die den unten angegebenen Code ausführen. Kann es bei gleichzeitiger Ausführung dieser beiden Threads zu einer Verklemmung kommen? Begründen Sie Ihre Ansicht!“
“Assume the following two Kernel-level threads of a two-threaded application whose code is given below. Suppose both threads are executed concurrently, can there be a deadlock? Explain your opinion.”

```
global spinlock_t A,B,C = free;
```

Thread 1	Thread 2
<pre>procedure update_data(...) begin lock(A) // update some data unlock(A) lock(B) lock(C) // update some more data unlock(B) unlock(C) end</pre>	<pre>procedure update_data(...) begin lock(B) // update some data lock(C) // update some more data unlock(B) lock(A) // update even more data unlock(C) unlock(A) end</pre>

Verklemmung/Deadlock?

Ja/yes:—**Nein/no:**

Begründung/explanation: **B & C are requested in the same order, i.e. no circular wait (2), A only nested in Thread T2, i.e. no hold & wait**

2. „Erläutern Sie das Konzept „**adaptiver Lock**“ und diskutieren Sie dessen **Vor- und Nachteile!**“
*“Explain the concept of an **adaptive lock** and discuss its **advantages and disadvantages.**”*

In front of an adaptive lock an activity spins for a while ($t > 1$ cycles), but after spinning it will be blocked

+ **limits the inefficient busy waiting time on a CPU**

- **If always blocking, you lose t avoidable CPU cycles by lazy blocking**

3. „Warum verwendet die **Planungsstrategie**(*scheduling policy*) „**Multi-Level Feedback**“ **unterschiedlich lange Zeitscheiben**(*time slices*)? Begründen Sie Ihre Aussage!“
*“Why does the **scheduling policy** “**Multi-Level-Feedback**” use **time slices of different length**? Explain your answer.”*

Begründung/explanation: **Short response-/turnaround time of short/interactive jobs, few switching overhead of long running compute bound jobs**

4. „**Wozu** dient die **Weitergabe von Tickets**(*ticket donation*) beim **Lotteriescheduling**?“
*“**For what purpose** does **lottery scheduling** offer **ticket donation**?”*

Assume a client wants a service from a server: if it donates its tickets to the server, it will increase the servers probability to be scheduled, resulting in a shorter response from the server

5. „Welche Informationen enthalten die **Signaturen** von **DIMMUNIX**, mit denen eine **zweite Verklemmung** vermieden werden kann?“
 “What information do the *signatures* of **DIMMUNIX** contain to avoid a *second deadlock*?”

Call stacks of the threads involved in a deadlock, (including the involved mutexes and the IP addresses of the lock_mutex calls

Aufgabe 4 / Question 4

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

1. „Anhand welches Kriteriums trifft der optimale Seiteneretzungsalgorithmus seine Entscheidung, welches Seiten-/Kachelpaar **<page, page frame>** ersetzt werden soll?“
 “Based on which *criterion* does the *optimal page replacement algorithm* take its *decision* which pair **<page/page frame>** shall be *replaced*?”

Choose the pair that will not be accessed for the longest period of time in the future .

2. „**Messungen** in einem **dynamischen** System haben ergeben, dass Applikationen der Klasse **K1** besser mit **4 KB-Seiten** und Applikationen der Klasse **K2** besser mit **16-KB Seiten** bearbeitet werden können. Der für Benutzeradressräume vorgesehene Hauptspeicher sei 2 GByte groß. Welches Speicherverfahren verwenden Sie? Begründen Sie Ihre Entscheidung!“
 “*Measurements* within a *dynamic* system show that applications of class **K1** execute better with pages of size **4 Kbyte**, whereas applications of class **K2** work better with pages of size **16 Kbyte**. The size of the usable memory for the user address spaces is 2 Gbyte. What memory manager do you use? Explain your choice.”

A1: Buddy-System, no internal fragmentation because requested memory units have sizes of powers of 2 (2^{12} or 2^{14} byte)

A2: Memory manager with two block sizes (4 and 16 KB), allocating 4 KB frames from low addresses and 16 KB frames from high addresses (or vice versa)

A3: Hierarchical memory manager , top level handles 16 KB frames, lower level deals with 4 KB blocks in 16 KB frames

(Only one of these alternative managers A1, A2, A3)

3. „Die Heapverwaltung einer mehrfädigen Applikation verwendet das Randkennzeichnungs-verfahren (*boundary tag manager*) mit sofortiger Wiedervereinigung (*eager reunification*). Kein Thread werde abgebrochen, sondern terminiere ordnungsgemäß, d.h. er hat nicht mehr benötigte Heapobjekte vorher freigegeben. Benötigen Sie eine Speicherbereinigung (*garbage collection*)? Begründen Sie Ihre Ansicht!“
 “The heap management of a multi-threaded application uses the *boundary tag manager* with *eager reunification*. No thread will be cancelled, each thread terminates properly, i.e. it has released no longer needed heap objects beforehand. Do you need a *garbage collection*? Explain your answer.”

No garbage collection is required: Objects that are no longer needed are released by the application, and adjacent free units are always reunified as early as possible (during release) due to eager reunification.

4. „Gegeben sei ein Zweiprozessorsystem mit **software-kontrollierten TLBs**. Zwei Prozesse P1 und P2 haben einen **gemeinsamen Adressbereich** bestehend aus einer Seite. Jeder der beiden Prozesse rechnet stets auf dem gleichen Prozessor, dabei rechnet P1 auf CPU1 und P2 auf CPU2. Wird von bzw. auf diese gemeinsame Seite mehrfach pro Prozess gelesen und geschrieben, wie **oft muss dabei mindestens** auf den Seitentableneintrag der gemeinsam benutzten Seite zugegriffen werden?“
 „Assume a two processor system with **software controlled TLBs**. Two processes P1 and P2 have a **shared address region** consisting of a single page. Each of the two processes runs always on the same CPU, whereby P1 runs on CPU1 and P2 runs on CPU2. Assume each of the processes issues several reads and writes from or to the shared page, how **often at least do you have** to access the page table entry of that page?“

Starting with empty TLBs, the first access (read or write) on CPU1 requires loading the PTE into its TLB (1st PTE access). The same is true on CPU2 (2nd access).

The first write additionally requires writing the PTE to update the dirty bit (at least 1 additional access). In total, at least 3 accesses are required.

If both CPUs initially read a clean PTE, the first write on each CPU will cause the dirty bit to be ‘updated’ in the PTE, thus requiring at least 4 accesses.

5. „Manche Architekturen bieten ein **Cache-Disable Bit** an. Erklären Sie, wofür man dieses Kontrollbit benutzen kann! **Wann** und **wer** verändert dieses Kontrollbit?“
 “Some architectures support a **cache-disable-bit**. Explain for what purpose this control bit can be used. **When** and **who** modifies this control bit?“

A set cache-disable-bit prevents memory contents from being buffered in the cache.

It is required for memory-mapped I/O and access to buffers that are modified via DMA (updates might not be noticed by the cache, thus causing stale data to be read from the cache).

The OS kernel (typically a device driver) sets this bit.

Aufgabe 5 / Question 5

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

1. „Welchen **minimalen Wert** hat der **Zähler der Hardlinks** in der **Inode** eines Linux EXT2 **Verzeichnisses (directory)**?“
 “What is the **minimal value** of the **counter of hardlinks** in the **inode** of a Linux EXT2 **directory**?“

Minimaler Wert/Minimal value: **2**

2. „Welche **Informationen** enthält ein **Verzeichniseintrag (directory entry)** im Linux EXT2-Dateisystem?“
 “Which **information** does a **directory entry** of the Linux file system EXT2 contain?“

File Name

Inode-Number

3. „Bei einer erweiterbaren Hashdatei wird stets die **aktuelle Generationszahl** **gen_{max}** in der Hashfunktion verwendet. Wie kann man mit **gen_{max}** ein Datenelement finden, obwohl dieses Datenelement zuvor mit einer kleineren Generationszahl in einen Datencontainer abgebildet worden war?“
 “In an extensible hash file, you always use the **current generation number** **gen_{max}**. How can you find a data element with **gen_{max}**, even though this data element was mapped beforehand to its data container with a smaller generation number?“

