

Lösung (Solution) Klausur (Examination) WS 2005/2006, 15. September 2006

Aufgabe1/Question 1 (Zum Aufwärmen/Warm up, 3 + 3 + 6 Punkte/marks)

1. „Unterstreichen Sie die Speicherverwaltungen aus den folgenden drei Beispielen, welche **innerhalb des Linuxkerns** verwendet werden!“
 “*Underscore those memory managers out of the following three examples that are used **within the Linux kernel.***”

a) <u>Stack</u>	b) <u>Buddy System</u>
c) <u>Slab Allocator</u>	

2. „Zählen Sie **drei charakterisierende Eigenschaften** von **Kern(modus)threads** (*kernel (mode) threads*) auf!“
 “*Enumerate **three characteristic properties** of **kernel (mode) threads.***”

<u>Always executed in kernel mode</u>	<u>Not assigned to any process/task</u>
<u>TCB in kernel/microkernel</u>	<u>No user-level stack</u>
<u>Scheduled by kernel scheduler</u>	<u>Already created at boot time</u>

3. „Ohne **prioritätenbasierte Ablaufplanung** (*priority based scheduling*) kann man **nicht** zwischen **E/A-** und **CPU-intensiven** Prozessen unterscheiden.“
 “*Without **priority based scheduling** you can **not distinguish** between **I/O- and CPU-intensive processes.***”

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

4. „Sieht man von **invertierten Seitentabellen** ab, dann hat **jede definierte virtuelle Seite** eines aktivierten Prozesses einen **entsprechenden Seitentabelleneintrag** (*PTE*).“
 “*If one refrains from **inverted page tables**, then **each defined virtual page** of an activated process has a **corresponding page table entry** (*PTE*).*”

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

5. „Bevor ein **TLB-Fehler** (*TLB miss*) an der **Seite** (*page*) p_j auftreten kann, muss zuvor für diese Seite p_j ein **Seitenfehler** (*page fault*) behandelt worden sein.“
 “*Before a **TLB miss** at **page** p_j can occur, a **page fault** has been to be handled for this **page** p_j .*”

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

6. „In einem **strikt prioritätsorientierten System mit Verdrängung** (*preemption*) ist **garantiert**, dass alle Threads, die darauf warten, ihren kritischen Abschnitt zu betreten, **höchstens gleiche** oder **kleinere** Priorität besitzen, als der Thread, der sich gerade im kritischen Abschnitt befindet.“
 “*In a **strict priority oriented system** with **pre-emption** it is **guaranteed** that all threads waiting to enter their critical section do have **at least an equal or a smaller** priority value compared to the priority of the thread in the critical section.*”

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

7. „Sobald ein System hinsichtlich seiner Betriebsmittelnutzung in den **unsicheren** Zustand (*unsafe*) gelangt ist, wird eine **Verklemmung** (*deadlock*) eintreten.“
 “*Whenever a system enters the state **unsafe** (concerning its resource usage) a **deadlock** will occur.*”

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

8. „Nur mit dem **Randkennzeichnungsverfahren** (*boundary tag system*) kann man im Rahmen der Operation `mem_release()` das frei gewordene Stück mit einem benachbarten freien Stück **effizient** verschmelzen.“
 “*Only with the **boundary tag system** can you **efficiently** reunify the released free memory piece with a neighboring free memory piece in the operation `mem_release()`.*”

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

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

1. „Erläutern Sie möglichst **präzise** und **vollständig** den Begriff **Prozess!**“
*“Explain as **precisely** and **completely** as possible the term **process.**”*
A process is a single threaded application or system task, i.e. it provides the corresponding environment for the application or system program to be executed as a scheduling entity. The needed information is collected within the PCB in the kernel, the PCB typically includes the following resource related information:
Address space, and needed, or reserved, or used system resources, e.g. open files, ...
A PCB might contain additional scheduler related information:
Priorities, start time, deadline, recent or expected execution time,

2. „Zählen Sie **mindestens acht PCB-Attribute** eines **Linux-** oder **Unix-Prozesses** auf!“
*“Enumerate at least **eight PCB-attributes** of a **Linux** or **Unix** process.”*

PID	PPID
UID	GUID
State(running,...)	Initial Priority (Nice Value)
Dynamic Priority	Pointer to open file table
Pointer to address regions	Used CPU time

3. „Begründen Sie präzise und vollständig, warum die Linux-Systemarchitekten die **Prozesskontrollblöcke (PCB) im Linux-Kern (kernel)** implementiert haben!“
*“Justify precisely and completely why the system-architects of Linux have implemented the **process control blocks (PCB) inside the Linux kernel.**”*

PCBs contain relevant scheduling information that allows the kernel scheduler to favour or to disadvantage the processes. They contain system critical information and should not be manipulated by users, otherwise every scheduling policy is obsolete. PCB also contain the kernel stack, and other state of interrupt handlers, thus they must be resident, i.e. never paged out etc. which is easy to install, when you place them into the kernel)

4. „Begründen Sie präzise und vollständig, warum Systemarchitekten das **Threadmodell** eingeführt haben **und** geben Sie hierfür ein **einleuchtendes Beispiel** an!“
*“Justify precisely and completely why system-architects have introduced the **thread model** and give a **plausible example** for this.”*

Creating concurrent applications with processes is more expensive, i.e. you need a new address space, you must cross the user/kernel boundary (Not with PULTS). Interaction and sharing of resources with threads is cheap and efficient, where as with processes it is at least more expensive and slower, if possible at all.
Webserver with one worker thread per request + one dispatcher thread

Aufgabe 3 / Question 3 :**(1 +1 +1 + 3 + 6 Punkte/marks)**

1. „Welche **Ablaufplanungsstrategie** (*scheduling policy*) **ohne Verdrängung** (*without preemption*) **optimiert** auf einem **Einprozessorsystem** die **mittlere Verweilzeit** (*turnaround time*)?“
 “Which *scheduling policy without pre-emption optimizes the average turnaround time in a single processor system?*”

Shortest Processing time First (SPT)

2. „Geben Sie für obige Ablaufplanstrategie ein **konkretes Beispiel** an, bei dem diese Strategie **mit Verdrängung besser** arbeiten würde!“
 “For the above mentioned policy, give a *concrete example* where this policy *with pre-emption* would work better.”

Assume a running long-runner creating a new short process

3. „Welche Kenngröße muss a priori bekannt sein, damit man beide obigen Ablaufplanungsstrategien **effektiv** einsetzen kann?“
 “What characteristics have to be known a priori in order to use *effectively* both above mentioned scheduling policies?”

Execution Time

4. „Beschreiben Sie eine Situation und eine Konfiguration, in der es aus Sicht der Ablaufplanung günstiger ist, einen gerade frei gewordenen Prozessor leer laufen zu lassen, obwohl noch Anwenderprozesse/KLTs in der Bereitwarteschlange (ready queue) vorhanden sind!“
 “Describe a situation and a configuration whereby it is better from the viewpoint of the scheduler to run a free processor idle even though there are application processes/KLTs in the ready queue.”

Assume a SMP with (logically) heterogeneous CPUs.**1) Inappropriate CPU becomes free and does not find a fitting ready KLT****2) Slow CPU becomes free, but soon the fastest CPU will be free, too. Then it might be better to wait for the fast CPU!**

4. „Sie sollen für ein **Zweiprocessorsystem** die **FCFS-Ablaufplanungsstrategie** so **exakt** und **strikt** wie möglich implementieren. Erläutern Sie mittels einer **detaillierten** Programmskizze oder per Beschreibung, was Sie u.U. alles im **Rahmen der Unterbrechungsbehandlung** „**Synchrone Ein-/Ausgabe beendet**“ für die Durchsetzung dieser strikten FCFS-Strategie tun müssen! Hinweis: Ein sehr alter Prozess/KLT könnte sehr lange wegen E/A blockiert gewesen sein. Falls es noch andere Stellen im System gibt, an denen Sie wegen der strikten FCFS tätig werden sollten, dann skizzieren Sie dies ebenfalls.“
 „For a *double processor system* you have to implement the FCFS scheduling policy as *exactly* and *strictly* as possible. Describe by a detailed program draft or by text what you possibly have to do concerning this strict FCFS-scheduling in the *context of the interrupt handler “end of synchronous I/O”*. Hint: A very old process/KLT could have been blocked

due to this I/O for a very long time. If there are additional places affected by dealing with the strict FCFS, then draft these places.”

If FCFS is implemented correctly, currently the oldest non blocked processes are running. Whenever a process is unblocked (e.g. due to end of I/O), the kernel scheduler has to check whether the unblocked process is older than one of the two running processes. If not, insert the unblocked process according to its start-time, else pre-empt the youngest running process and insert it according to its start-time.

Furthermore, whenever any process does a request (e.g. for a resource) insert this request-block into the corresponding waiting-queue according to the start-time of the process.

Aufgabe/Question 4

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

- „Erläutern Sie **mögliche Nachteile** einer **globalen Seitenersetzungsstrategie** (*global page replacement policy*).“
*“Explain **potential disadvantages** of a **global page replacement policy**.”*
A page-fault in process P_i might page-out a still used page of another process P_j thus reducing the system performance in total. (Circular page stealing is possible.) The WS of the individual processes are not regarded.
- „Manche Systeme implementieren **gemeinsam genutzte Code- bzw. Datenseiten** so, dass diese Seiten **nicht ersetzt werden dürfen**, solange mindestens noch einen Thread/Prozess diese benötigen könnte. Diskutieren Sie Vor- und Nachteile dieser Idee!“
*“Some systems implement **shared code or data pages** such that these pages are **not replaced** as long as at least one thread/task might need them. Discuss pros and cons of this idea.”*
A reference count is sufficient to save shared pages, however, a user creates task of a working thread and its dummy partner thread, sharing all pages, thus none of these pages will ever be paged-out as long as this application is activated.
- „Erklären Sie zuerst **Bedeutung** und **Zweck** des „**Pinned-Bits**“. Erläutern Sie dann, **wofür** man dieses Pinned-Bit **anwenden** kann.“
*“First explain **meaning** and **purpose** of the pinned-bit. Then describe for what you can **apply** this pinned-bit.”*
The pinned-bit informs the pages, that the corresponding page is pinned to its current page frame, i.e. it will not be page out. This bit is used for DMA, that often can only address physically, thus their buffers are pinned to specific page frames. Another application are very important real-time processes, once started their important address ranges have to be resident, otherwise they can not meet their deadlines.
- „Ein System bietet **4 KB Standard-** und **4 MB-Superseiten** an. Wie müssen Anwendungen beschaffen sein, damit diese **effizienter** mit den Superseiten statt mit Standardseiten ausgeführt werden können?“
*“A system offers a **4 KB standard page** and a **4 MB super page**. How do applications have to be constituted that they are executed **more efficiently** with super pages than with standard pages?”*
The application must contain a huge address region , that is addressed randomly.

5. „Aus welchen Feldern würde ein von ihnen favorisierter TLB-Eintrag für ein Mehrprogrammssystem bestehen? Geben Sie für **jedes TLB-Eintragsfeld** an, **warum** und **ob** es **notwendig** bzw. nur **wünschenswert** ist.“

*“What fields would your favorite TLB-entry have? Justify and describe for each TLB-entry field **why**, and **if it is necessary** or only **desirable**.”*

Entry	Description	Required/Optional
Virtual page no	Find matching PTE	r
Physical page frame no	To complete address translation	r
Address space ID	Prevents flushing TLB when AS switching takes place	o
Valid bit	Indicates whether TLB entry is valid for address translation	r
Dirty bit	Whether page frame has been modified in the very past	o
Global bit	Whether AS identification must be used or not, e.g. all kernel pages are global!	o
Write enable bit	Used to write protect a page	o
Other access right bits	Used to control to page	o
Kernel-/user bit	Controls whether applications try to access kernel pages	o

Aufgabe 5/ Question 5:

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

1. „**Worin** unterscheiden sich in der Ausgabe des **ps -aux** Kommandos **Kerndämonen-einträge** von den **übrigen Prozesseinträgen**?“

*“How do **kernel daemon entries** differ from the entries of the **other processes** in the output of the **ps -aux** commands?”*

[entries are in]
PPID = 1

daemons don't have own memory

2. „Erklären Sie den Begriff **temporäre Datei** (*temporary file*) und geben Sie **mindestens eine typische Anwendung** von temporären Dateien an!“

*“Explain the notion **temporary file** and indicate **at least one typical application** of temporary files.”*

If you want to store information only for a very fixed amount of time, you use temporary files, instead of normal files. Typically these files are deleted when the creating process terminates. Typical applications: 1. Compiler, 2. Locking

3. „Einige Systeme bieten **vorausschauendes Lesen** (*read ahead*) an. Diskutieren Sie Vor- und Nachteile dieses Konzepts!“
 “*Some systems offer **read ahead**. Discuss advantages and disadvantages of this concept*”.

If you access a sequential huge data file strict sequentially, then read ahead might improve the turnaround time of this application. However, sometimes this application might only use the first parts of the sequential file. When you have read ahead the complete sequential file many blocks of the file system cache are spoiled for nothing, thus other applications might suffer a lot.

4. „Erklären Sie präzise und vollständig die Unterschiede zwischen **Hard-** und **Softlinks!**“
 “*Explain precisely and completely the differences between **hard-** and **soft-links**.*”

A soft-link is a file, containing a pathname, thus its content can even point to a file not yet existing or not yet mounted. Thus a soft-link can cross the file-systems. A hard-link is bound to the file-system and is just another name for a file in this file-system.

5. „Viele Systeme unterscheiden zwischen **nicht privilegierter** und **privilegierter** Software. Warum? Zählen Sie mindestens **jeweils zwei typische** Beispiele auf!“
 “*Many systems distinguish between **non-privileged** and **privileged software**. Why? Enumerate at least **two typical examples** for each.*”

The main reasons are robustness and security.

Parts of the kernel have to be protected from user access (e.g. PCB, page tables etc.) in order to keep the policy of system, otherwise a user might forge the system to be his slave. Some system software has to use privileged instructions, that are forbidden in user mode.

Furthermore security and protection aspects have to be regarded, e.g. file should only be accessible the authorized users, e.g. to preserve confidentiality

Example of privileged software:

kernel, OS-server, device driver, administrative tool

Example of non privileged software:

most applications, web browser, calculator, real player