10.01.2026

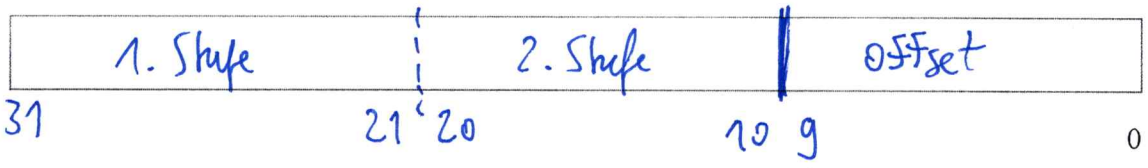
LK-Aufgabe 33

Paging

Ein Prozessor verwendet

- 32 Bit breite virtuelle Adressen mit
- einer Seitengröße von 1 KByte, $= 2^{10}$ Byte $\Rightarrow$ 10 Bit Offset $\Rightarrow$ 22 Bit Seitennr.
- 2-stufigem Paging, wobei alle Seitentabellen gleich groß sind, $\Rightarrow$ 11 Bit pro Komponente
- Seitentabelleneinträgen der Länge 4 Byte.

(i) Wie ist eine virtuelle Adresse aufgebaut (welche Bits der Adresse haben welche Bedeutung)?



Zeichnen Sie die Unterteilung hier ein und beschriften Sie die Abschnitte geeignet.

(ii) Wie viele Seitentabellen der verschiedenen Stufen gibt es? Wie groß sind diese Tabellen?

Stufe 1: 1 Tabelle
Stufe 2: 2^{11} Tabellen
hypothetisch für Stufen 3, 4:
Stufe 3: 2^{22} Tab.
4: 2^{33} Tab.

54 bit breite Adr.
10 bit Offs.
44 bit Seitennr.
4-stufiges Pag.

Größe Tab. = #Einträge · Größe Eintr.
$= 2^{11} \cdot 4$ Byte $= \underline{2^{13}}$ Byte $= 8$ KB

b) Wie viele innere Seitentabellen (Tabellen auf der zweiten Stufe) gibt es? Wie groß sind die äußere (Stufe 1) bzw. die inneren (Stufe 2) Seitentabellen?

34. Wahr oder falsch?

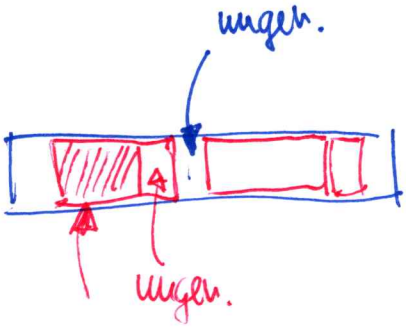
Betrachten Sie die folgenden Aussagen über Speicherverwaltung und entscheiden Sie für jede, ob sie wahr oder falsch ist:

X = falsch

a) Beim Paging werden Seitenrahmen (*frames*) auf Seiten (*pages*) abgebildet. → anders rum

b) Paging mit mehrstufigen Seitentabellen reduziert den Speicherverbrauch (im Vergleich zu einstufigem Paging). ✓

c) Paging lagert bei Speicherknappheit den Speicher eines oder mehrerer Prozesse vollständig auf Festplatte aus. Dieses Aus- und Wiedereinlagern wird auch *Swapping* genannt. nur einzelne Seiten!



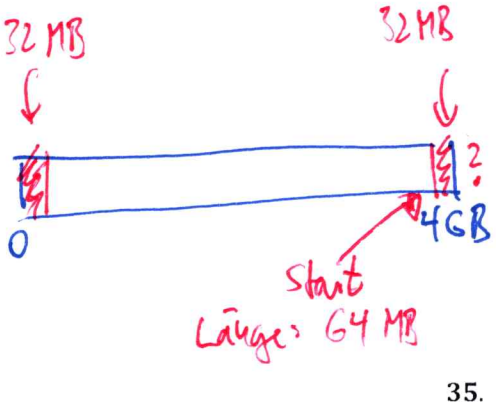
d) Paging gehört zu den Verfahren der *zusammenhängenden Speicherverwaltung*.

e) Die feste Partitionierung verursacht hohe *externe Fragmentierung*.

f) ** Beim Paging kann ein Prozess auch mehr virtuellen Speicher nutzen als der Rechner an physischem RAM enthält. ✓

g) ** Wenn ein Prozess einen *Page Fault* verursacht, weil er auf eine Adresse zugreift, deren Seite ausgelagert ist, wird der Prozess blockiert – es entsteht eine vergleichbare Wartezeit (I/O) wie beim Lesen eines Datenblocks aus einer Datei. ✓

h) Um einen mit `ptr = malloc(10*1024);` angeforderten Speicherbereich auf 20 KByte zu vergrößern, eignet sich der Befehl `ptr = realloc (ptr, 20*1024);`. → kann fehlschlagen



i) Segmentierung gehört zu den Verfahren der *zusammenhängenden Speicherverwaltung*. ✓

j) ** Bei Segmentierung auf einem 32-Bit-Intel-Prozessor mit maximaler Speicherausrüstung (4 GByte) kann ein 64 MByte großes Segment gebildet werden, das *gleichzeitig* aus den untersten 32 MByte RAM (`0x0000 0000-0x01FF FFFF`) und den obersten 32 MByte RAM (`0xFE00 0000-0xFFFF FFFF`) besteht. ✓

(Fragen mit ** sind anspruchsvoller.)

35. Begriffe

Erklären Sie in eigenen Worten *Shared Memory* und *Copy-on-Write*.

1	2	3	4	5	6	7	8	9			Σ

Die Bearbeitungszeit der Probeklausur ist 55 Minuten; für die richtige Klausur haben Sie 90 Minuten Zeit. Entsprechend hat diese Probeklausur reduzierten Umfang (ca. 60%). Bitte bearbeiten Sie alle Aufgaben. Es sind insgesamt 55 Punkte zu erreichen (richtige Klausur: 90 Punkte).

Tipp: Lesen Sie zunächst alle Aufgaben durch und entscheiden Sie, welche Fragen Sie am leichtesten beantworten können; starten Sie dann mit diesen Aufgaben.

Viel Erfolg!

1. Geschichte der Betriebssysteme (2 / 55 Punkte)

Erläutern Sie (in Stichworten), inwiefern frühe Rechnersysteme mit Kartenleser zum Namensgeber für die beim Scheduling verwendete Kategorie der Stapelverarbeitungssysteme sind.

FCFS

$\frac{2^{37}}{2^{11}} = 2^{37-11} = 2^{26}$

$K = 2^{10}$
$M = 2^{20}$
$G = 2^{30}$
$T = 2^{40}$

K, M, G, T

128 GB = 2^7

2^1 2
2^2 4
2^3 8
⋮
2^9 512

2. Dateisysteme (Indirektion) (5 / 55 Punkte)

Ein Unix-artiges Dateisystem arbeite mit folgenden Parametern:

- Größe des Dateisystems: 128 GByte = $2^7 \cdot 2^{30}$ B = 2^{37} B
- Größe eines Datenblocks: 2 KByte = 2^{11} B
- Indirektion: 3-fach
- Im Inode: 3 direkte Verweise, 3 einfach indirekte, 2 zweifach indirekte, 2 dreifach indirekte Verweise

$\frac{128\,G}{2\,K} = 64\,M = 2^6 \cdot 2^{20} = 2^{26}$

Berechnen Sie die nötige Größe einer Blockadresse (für die Speicherung in den Indirektionsblöcken), die Anzahl der Adressen pro Block und die maximale Dateigröße. (Zweierpotenzen 2^n sind nicht auszurechnen.)

$\#\text{Blöcke} = \frac{128\,GB}{2\,KB} = \frac{2^{37}}{2^{11}} = 2^{26} \Rightarrow$ 32 Bit (26, aufgerundet) für Blocknummer

$\frac{2\,KB}{4\,B} = 512$ Einträge pro Block

max. Dateigr. = # Adressen · Blockgröße

$= (3 + 3 \cdot 512 + 2 \cdot 512^2 + 2 \cdot 512^3) \cdot 2\,KB$

$= (7 + 3 \cdot 2^9 + 2 \cdot 2^{18} + 2 \cdot 2^{27}) \cdot 2 \cdot 2^{10}\,B > 2^{39}\,B = 512\,GB$

$$\frac{2\,GB}{64\,KB} = \frac{1\,GB}{32\,KB} = \frac{1024\,MB}{32\,KB}$$

$$= \frac{512\,MB}{16\,KB} = \frac{256\,MB}{8\,KB} = \frac{128\,MB}{4\,KB}$$

$$= \frac{64\,MB}{2\,KB} = \frac{32\,MB}{1\,KB} = 32\,K$$

$$\frac{32\,MB}{1024\,B} = \frac{16\,M}{512} = \frac{8\,M}{256}$$

$$\frac{4M}{128} \quad \frac{2M}{64} \quad \frac{1M}{32} \quad \frac{1024K}{32}$$

$$\frac{512K}{16} \quad \frac{256K}{8} \quad \frac{128K}{4} \quad 32K$$

$$\frac{2GB}{64KB} = \frac{2 \cdot 2^{30}}{2^6 \cdot 2^{10}} = \frac{2^{31}}{2^{16}} = 2^{15} = 32 \cdot 2^{10} = 32K$$

3. Prozesse und Threads (10 / 55 Punkte)

a) Mit welchem Signal können Sie einen Prozess so beenden, dass er noch Gelegenheit hat, offene Dateien zu schließen und sich somit „ordentlich" zu beenden? Als Referenz finden Sie nebenstehend die Liste der ersten 28 Signale.

TERM (15)

```
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) SIGILL
 5) SIGTRAP      6) SIGABRT      7) SIGBUS       8) SIGFPE
 9) SIGKILL     10) SIGUSR1     11) SIGSEGV     12) SIGUSR2
13) SIGPIPE     14) SIGALRM     15) SIGTERM     16) SIGSTKFLT
17) SIGCHLD     18) SIGCONT     19) SIGSTOP     20) SIGTSTP
21) SIGTTIN     22) SIGTTOU     23) SIGURG      24) SIGXCPU
25) SIGXFSZ     26) SIGVTALRM   27) SIGPROF     28) SIGWINCH
```

b) Zwei Unix-Prozesse A und B haben die Nice-Werte -10 (A) und +10 (B). Welcher hat die höhere Priorität?

A

c) Viele Betriebssysteme erlauben es dem Benutzer, eigene *Prozesse* zu suspendieren (bei Unix z. B. über das STOP-Signal); ein entsprechendes Feature für *Threads* fehlt aber. Warum?

i.d.R. nicht sinnvoll, Threads von außen anzusp.

d) Neben „suspendiert" (Aufgabe c) gibt es oft weitere Zustände, die nur *Prozesse* (und nicht *Threads*) annehmen können. Nennen und erläutern Sie (in Stichworten) ein Beispiel.

„swapped": Speicher wird von allen Threads dieses Prozesses benötigt, die sind also alle „swapped"

e) Nennen und erklären Sie jeweils einen Vorteil von User-Level-Threads (ULTs) gegenüber Kernel-Level-Threads (KLTs) bzw. von KLTs gegenüber ULTs.

(wann kann blockieren?)

4. Scheduler (8 / 55 Punkte)

Betrachten Sie die folgenden Aussagen und nehmen Sie zu jeder der Aussagen Stellung (Angabe von „richtig“ oder „falsch“ und kurze Begründung in Stichworten). auch bei richtigen

1. Kernel Level Threads und User Level Threads unterscheiden sich dadurch, dass der Scheduler (im Betriebssystem) User Level Threads nicht „kennt“, also auch nicht auswählen kann.

2. **Prioritätsinversion** bedeutet, dass ein mangelhaft arbeitender Scheduler Prozesse mit niedriger Priorität gegenüber solchen mit hoher Priorität bevorzugt.

3. **Lotterie-Scheduler** entscheiden durch Ziehen eines Loses, welcher Prozess als nächster laufen darf.

4. Ein **präemptiver** Scheduler ist nicht in der Lage, laufende Prozesse zu unterbrechen – er muss warten, bis der Prozess von sich aus die CPU „abgibt“, also in den Scheduler springt.

5. **Aging** bedeutet, dass ein bereiter Prozess dauerhaft vom Scheduler nicht ausgewählt wird, wodurch er ständig „altert“.

6. Der Begriff **Burst** bezeichnet eine CPU- bzw. eine I/O-Phase, also z. B. für CPU-Phasen den Zeitraum vom Aktivieren des Prozesses bis zum Deaktivieren durch den Scheduler oder Einleiten einer I/O-Aktion durch den Prozess.

7. Der Scheduler legt fest, wann die I/O-Blockade eines Prozesses aufgehoben wird (Übergang vom Zustand „blockiert“ in den Zustand „bereit“).

8. SJF ist eine präemptive Variante von SRT.