Wiederholung OS

a) Wie viel Speicher kann man theoretisch mit einem einzigen Outer-Page-Table Eintrag adressieren, wenn das Computer System 8 KB Seiten benutzt und für die Inner-Page-Table 10 Bits benutzt werden?

b) Gegeben sei demand-paging mit virtuellem Adressraum von 16 MB, physikalischem Speicher von 1MB, page size von 4KB, und einem TLB mit 4 Einträgen.
Ein Prozess wird gestartet, dessen Pagetable nach folgendem Mapping arbeitet:
- „virtual page“ i wird auf „physical frame“ i+8 gemappt.
Der Prozess generiert die angegebene Sequenz von Adresszugriffen:

Hex	**Binär**	**Dezimal**
0F0FF7	1111 0000 1111 1111 0111	987127
0251FD	0010 0101 0001 1111 1101	152061
0322A0	0011 0010 0010 1010 0000	205472
0AF570	1010 1111 0101 0111 0000	718192
0AF231	1010 1111 0010 0011 0001	717361
0251FD	0010 0101 0001 1111 1101	152061
01FFFF	0001 1111 1111 1111 1111	131071
0AF570	1010 1111 0101 0111 0000	718192

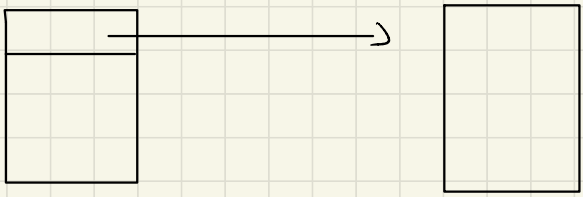
Beschreiben Sie den Inhalt des TLBs am Ende der Sequenz, angenommen, TLB ist am Anfang leer und innerhalb des TLBs findet FIFO Replacement statt.

c) Eine Bierkiste bietet Platz für 16 Flaschen.
Maximal 4 Personen können gleichzeitig in die Kiste greifen.
Schreiben Sie ein Programm in Pseudocode, welches den Zugriff für beliebig viele Personen, sowohl beim Herausnehmen, als auch beim Zurückstellen von Flaschen aus einer Kiste regelt. Das Programm soll so effizient wie möglich sein, d.h. unnötige Wartezeiten sollen vermieden werden.
Hinweise:

- Jede Person die eine Flasche zum Trinken genommen hat, will diese danach auch wieder zurückstellen. (Ablauf also: Flasche nehmen, trinken, zurückstellen)
- Beim Zurückstellen ist nicht notwendigerweise ein Platz in der Kiste frei. (dies ist explizit sicherzustellen).

d) In einem 32bit System, 2 Level Paging, 10 + 10 + 12, ein Prozess mit 24 MB, (4 MB Code, 16 MB Daten und 4 MB Stack), befindet sich im virtuellen Speicher zwischen 1GB und 3GB. Der Code und Datenblock befinden sich nacheinander ab der Adresse 0x40 00 00 00. Der Stack befindet sich an der Adresse 0xBF C0 00 00.
d1) Wie viele Einträge hat man in dem Page Directory?
d2) Welcher Index haben die Einträge? (Hex oder Dezimal)

a)



$\Rightarrow$ 10 Bit $\Rightarrow 2^{10}$ mögliche Einträge

$\Rightarrow 2^{10} \cdot 8\,KB = 2^{10} \cdot 8 \cdot 2^{10} B = 8 \cdot 2^{20} B = 8\,MB$

b)

TLB

1111 0000	1111 1000
0010 0101	0010 1101
0011 0010	0011 1010
1010 1111	1011 0111

$\Leftarrow$

0001 1111	0010 0111

Achtung: Adressraum von 16 MB

$2^4\ 2^4\ 2^4 \mid 2^4\ 2^4\ 2^4 = 16\,MB$

↳ in TLB Tabelle müssen theoretisch noch überall 4 führende Nullen vorhanden

c)

```
semainit(Kiste, 4)
semainit(Flasche, 16)
semainit(Plätze, 0)

Person() {

    P(Kiste)
    P(Flasche)
    V(Plätze)
    V(Kiste)

    // trinken

    P(Kiste)
    P(Plätze)
    V(Kiste)

}
```

⇒ Deadlock möglich

```
semainit(Kiste, 4)
semainit(mutex_voll, 1)
semainit(mutex_platz, 1)
Plätze = 0                    voll = 16
place_found = false
got_drink = false
Person() {

    while(!got_drink)

        P(Kiste)
        P(mutex_voll)
        if voll > 0
            voll--
            V(mutex_voll)
            P(mutex_plätze)
            plätze++
            V(mutex_plätze)
            got_drink = true
        else
            V(mutex_voll)
        V(Kiste)
        // trinken
        while(!place_found)
            P(Kiste)
            P(mutex_plätze)
            if plätze > 0
                plätze--
                place_found = true
            V(mutex_plätze)
            V(Kiste)
```

⇒ ohne Deadlock aber Busy Waiting

d)

d) In einem 32bit System, 2 Level Paging, 10 + 10 + 12, ein Prozess mit 24 MB, (4 MB Code, 16 MB Daten und 4 MB Stack), befindet sich im virtuellen Speicher zwischen 1GB und 3GB. Der Code und Datenblock befinden sich nacheinander ab der Adresse 0x40 00 00 00. Der Stack befindet sich an der Adresse 0xBF C0 00 00.
d1) Wie viele Einträge hat man in dem Page Directory?
d2) Welcher Index haben die Einträge? (Hex oder Dezimal)

d1) Wie viele Pages werden benötigt?

Page size: $2^{12} = 4 \cdot 2^{10} = 4\,KB$

Code + Daten: $\frac{20 \cdot 2^{20}}{4 \cdot 2^{10}} = 5 \cdot 2^{10}$

Stack: $\frac{4 \cdot 2^{20}}{4 \cdot 2^{10}} = 2^{10}$

$\Big\} \; 5 \cdot 2^{10} + 2^{10} = 6 \cdot 2^{10}$

$\Rightarrow$ Es werden 6 Einträge in der Outer Page Table benötigt.

d2) 40 0

0100 0000 0000
0100 0001 00

0001 0000 0100

0x 100
256

0x 1 0 4
260

BFC
1011 1111 1100

0010 1111 1111

0x 2FF
511