GSS Abgabe 06

Carolin Konietzny, Paul Bienkowski, Julian Tobergte, Oliver Sengpiel

2. Juli 2015

1.1 a) Wie groß ist eine Seite?

Da unsere virtuellen Adressen 16 bit breit sind und wir 4 bit für die Adressierung der 16 Seiten in der Seitentabelle brauchen, haben wir ein Offset von 2^{12} bit Breite. Eine Seite enthält also 8 KB.

c) Welche Physikalischen Adressen ergeben sich?

i) 0x0fe8

ii) page fault

iii) page fault

iv) 0x8101

d) Wahl einer optimalen Seitengröße:

Größere pages führen dazu, dass große Bereiche des Speichers ungenutzt bleiben, kleinere pages führen zu größeren page tables und zu mehr over- head beim laden/speichern von pages.

e) Optimale page size:

Die optimale Größe einer Seite ergibt sich aus der Formel

$$p = \sqrt{2se}$$

mit s als Größe des Durchschnittlichen Prozesses und e als Wortbreite des Systems.
In diesem Fall also $\sqrt{2^{26}}$.

3. a)

```
Semaphor W, Mutex;
int NumberOfActiveReaders;

function Writer::processWriter() {
    while (NumberOfActiveReaders > 0) {
        wait();
    }

    W.lock();
    writeData();
    W.free();
}

function Reader::processReader() {
    Mutex.lock();
    NumberOfActiveReaders++;
    // only the first reader that appears needs to allocate the resource
```

```
    if (NumberOfActiveReaders == 1) {
        W.lock();
    }
    Mutex.free();

    readData();

    Mutex.lock();
    NumberOfActiveReaders--;
    // the last reader to finish frees the resource
    if (NumberOfActiveReaders == 0) {
        W.free();
    }
    Mutex.free();
}
```

b) Reader werden hierbei bevorteiligt, denn wenn bereits mind. ein Reader läuft, kann ein weiterer sofort starten. Startet während dessen read-Vorgang wiederum ein weiterer Reader, bleibt die Resource geschützt. Somit können bei „ungünstigem" Timing eine Reihe von Readern durchlaufen, ohne dass ein Writer, der u.U. schon länger auf Freigabe der Resource wartet, an die Reihe kommt.