TI 3 Übung 4

Tobias Lohse, Sven Klaus & Luisa Castano
Tutor: Thomas Tegethoff, Do 8-10

Aufgabe 1

(a)

```c
#include <stdio.h>
int main(void)
{
    printf("main at: %p\n", main);
    while(1);
}
```

Die Ausgabe des obigen Programms ist immer `main at: 0x10f6d2f40` , auch wenn es mehrere male parallel gestartet wird.

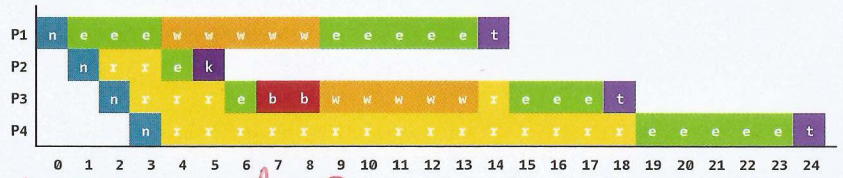
Dies verwundert nicht, da in 64bit UNIX Systhemen für jeden Prozess die vollen 16TB virtueller Addressraum reserviert werden.[1] Daher können mehrere Prozesse dieselben virtuellen Adressen haben. Der Linker in ELF Sythemen wie UNIX erstellt ausführbare Dateien im ET_EXEC Format, bei dem die Addresse des Entry-Points (main funktion) immer dieselbe ist.[2,3]

Wenn ein Prozess einen lesezugriff auf eine Addresse macht, schaut die MMU im TLB nach, wo (HD, RAM, Caches) die Seite mit der Addresse für diesen Prozess liegt und erhält eine physikalische Addresse. Wenn nötig werden die Daten aus HD, RAM oder einem niedrigeren Cache in den obersten Cache geschrieben, aus dem die Daten dann an der Prozess weiter gegeben werden.

Quellen:
[1]: http://www.princeton.edu/~unix/Solaris/troubleshoot/vm.html
[2]: http://stackoverflow.com/questions/3699845/how-is-it-that-main-function-is-always-loaded-at-the-same-address-whereas-variab
[3]: https://pax.grsecurity.net/docs/randexec.txt

(b)

(1)



Legende?

(2)

In Takt 7 und 8 befindet sich das System im Zustand "busy waiting", da P1 den Drucker benutzt und dadurch P3 blockiert ist, was die CPU blockt.

... 1/2

(3)

UNIX Systheme besitzen einen Befehl `killall` , welcher es erlaubt einen beliebigen Prozess abzubrechen. Dies kann nützlich sein, wenn sich zum Beispiel ein Programm aufgehängt hat. Dasselbe kann in der Kommandozeile mit `ctrl + C`

Aufgabe 2

Keine Codeabgabe!

```c
#include <stdio.h>
#include <stdlib.h>

{
    int pId;//ProzessID
    int aTime;//Ankunftszeit
    int sTime;//Ausfuehrungszeit
    struct pData *next;
    struct pData *prev;
};
typedef struct pData PROCESS;
typedef PROCESS *LINK;
int SIZE;

//Liest die Prozesse ein und erstellt die Liste
void readProcesses(LINK head)
{
    FILE *file = fopen("prcs.dat.txt", "r");
    if (file)
    {
```