# Übung 11

Bearbeiter: Björn Oke Maas, Mihai Renea
Tutorium: Jonas Cleve, Do. 14 Uhr

## Aufgabe 1.

```
;Bearbeiter: Bj rn Oke Maas, Mihai Renea
;Tutorium:   Jonas Cleve, Do. 14 Uhr

global _start

BUFF_SIZE   equ 1024
BUFF_SIZE_R equ 1025    ;1 byte zur Null-Terminierung
STD_IN      equ 0
STD_OUT     equ 1
READ        equ 0
WRITE       equ 1
EXIT        equ 60

section .text
_start:
    sub     rsp, BUFF_SIZE_R    ;Buffer auf dem Stack anlegen
    mov     rsi, rsp            ;Buffer Adresse in RSI

    mov     rdi, STD_IN         ;Standard-in FP in RDI
    mov     rdx, BUFF_SIZE      ;Buffer Groesse in RDX
    mov     rax, READ           ;Syscall Read in RAX
    syscall

    xor     r8, r8
    mov     [rsi + rax], r8b    ;Gelesene String Null-Terminieren

    mov     rdi, STD_OUT
    mov     rdx, BUFF_SIZE_R
    mov     rax, WRITE
    syscall

    mov     rax, EXIT
    mov     rdi, 0              ;Exit success
    syscall
```

## Aufgabe 2: Superskalare Pipeline

1. Siehe Schaubild.

2. Wenn wir wieder von einer 5-stufigen Pipeline ausgehen, brauchen wir bei idealer Anordnung (ohne NOPs) mindestens 30 Takte zur Ausführung gegenüber 14 bei der superskalaren Pipeline. Der Speedup beträgt hier etwa 2, also ist die superskalare Pipeline doppelt so schnell.

3. Die verschiedenen Teile der EXE-Einheit arbeiten nicht ideal parallel. Die Befehle 8 und 9 sind zum Beispiel voneinander, aber nicht von anderen Befehlen abhängig und könnten vorgezogen werden. Reordering und begrenzt Forwarding würden die Ausführung beschleunigen.