# Praktikum 6 (10 Punkte) – Assembler

Von Tim Steven Meier und Fabian Husemann

## Aufgabe 1

```
asm volatile(
    "MOV r3,%0\n\t"
    "CMP r3, #128\n\t"
    "ITE eq\n\t"
    "MOVEQ r3, #1 \n\t"
    "ADDNE r3, r3\n\t"
    "MOV %[value],r3\n\t"

    : [value] "r+" (number0)
  :
  : "r3", "cc", "memory"
  );
```

| Befehl | Taktzyklus |
|---|---|
| mov | 1 |
| cmp | 1 |
| ite | 1 |
| moveq | 1 |
| addne | 1 |

Der Assemblerteil benötigt 6 Taktzyklen für jedes durchlaufen des loop().

## Aufgabe 2

```
asm volatile(
  "MOV r4, %[fib] \n\t"
  "loopfibo:\n\t"
  "LDR r5, [r4, #1]\n\t"
  "LDR r6, [r4, #0]\n\t"
  "ADD r6, r5\n\t"
  "STR r6, [r4, #2]\n\t"
  "ADD r4, #1\n\t"
  "LDR r6, [r4, #0]\n\t"
  "ADD r7, #1\n\t"
  "CMP r7, %[index]\n\t"
  "IT ne\n\t"
  "BNE loopfibo"
```

```
    :
    :[fib] "r" (fibData), [index] "r" (lastFiboIndex)
    :"r4","r5","r6","r7","cc","memory"
    );
```

| Befehl | Taktzyklus |
|---|---|
| mov | 1 |
| ldr | 2 |
| add | 1 |
| str | 2 |
| cmp | 1 |
| it | 1 |
| bne | 1 + P |

Der Assemblerteil benötigt 11 * (14 + P) +1 Taktzyklen.

Der "memory" - Tag bewirkt, dass die Register die alten Werte am Ende wieder laden.