Übungen zu Vorlesung
Praktische Informatik I
Wintersemester 2000/01

Serie 2

Rechneraufbau und Rechnerstrukturen

nach Oberschelp/Vossen

Lektion 1: Arbeitsweise der CPU

Stichworte:

- CPU-Komponenten (Bild S. 232)
  - Datenprozessor - Befehlsprozessor
  - Adressbus - Datenbus
  - Decodierer/Steuerwerk
  - IR
  - MAR
  - PC
  - MR - L - A
  - ALU
  - MBR
- "fetch-execution" Zyklus
- Maschinencode - Mikroprogrammierung

Dazu soll folgender Mikrocode eingeführt werden:

`MAR`	$\leftarrow$	`PC`	MAR erhält Inhalt von `PC` (Adresse)
`MBR`	$\leftarrow$	`<MAR>`	`MBR` erhält Inhalt der Speicheradresse, die im `MAR` steht
`MBR`	$\leftarrow$	`A`	lade Inhalt des Akku in `MBR`
`MBR`	$\leftarrow$	`MAR`	lade Inhalt des `MAR` (Adresse) in `MBR`
`<MAR>`	$\leftarrow$	`MBR`	Speichere Inhalt von `MBR` in die Adresse auf die `MAR` zeigt
`IR`	$\leftarrow$	`MBR`	Instruktions-Register erhält Befehlt aus `MBR`
`PC`	$\leftarrow$	`PC+n`	`PC` erhöhen um `n`
`PC`	$\leftarrow$	`MBR`	`PC` auf Adresse im `MBR` setzen (Absolutadressierung)
`PC`	$\leftarrow$	`MAR`	`PC` auf Adresse im `MAR` setzen
`MAR`	$\leftarrow$	`MAR+n`	`MAR` auf Inhalt $+n$ setzen
`A`	$\leftarrow$	`A + MBR`	Akku-Berechnung, Addition
`A`	$\leftarrow$	`-A`	unäres Minus
`A`	$\leftarrow$	`¬A`	logische Negation
`MR`	$\leftarrow$	`A * MR`	Multiplikation
`A`	$\leftarrow$	`MBR`	Konstante in Akku laden
`A`	$\leftarrow$	`A < 0`	logischer Vergleich mit Konstanten, Ergebnis im Akku
`MAR`	$\leftarrow$	`MBR`	Umladen von Adressen

Anmerkung: `MR` ist üblicherweise eine Erweiterung des Akku, d.h. ein Schreiben auf `MR` zerstört den Inhalt von `L` und `A`.

Maschinencode:
Wie in der Vorlesung eingeführt.

`LOAD adr`	n	`LOAD`	Lade den Inhalt der Speicherdresse adr in den Akku `A`
	n+1	`<adr>`	
`STORE adr`	n	`STORE`	Speichere den Inhalt des Akku `A` im RAM an Adresse `adr`
	n+1	`<adr>`	
`NEG`	n	`NEG`	Negiere den Inhalt des Akku `A`
`ADD adr`	n	`ADD`	Addiere den Speicherinhalt der Adresse `adr` zum Inhalt des Akku `A` und platziere das Ergebnis wieder in `A`.
	n+1	`<adr>`	
`MULT adr`	n	`MULT`	Multipliziere den Speicherinhalt der Adresse `adr` zum Inhalt des Akku `A` und platziere das Ergebnis im Multiplikationsregister `MR`
	n+1	`<adr>`	
`JMPNEG adr`	n	`JMPNEG`	Wenn der Inhalt dess Akku `A` negativ ist, springe zur Adresse `adr` und lies den dort gespeicherten BEFEHL (da dürfen keine Daten stehen!) Andernfalls mache weiter mit dem nächsten Befehl (der 2 Worte hinter dem `JMPNEG`-Befehl steht)
	n+1	`<adr>`	
	...		
	`<adr>`	BEFEHL	
`HALT`	n	`HALT`	Programm hält an

Beispiele:

1. Die Umsetzung des Maschinenbefehls `LOAD x`, mit `x=20` in die Mikroinstruktionen; `PC` soll auf 0 stehen;

   Maschinenprogramm:

1	LOAD
2	20
...	
20	10

`MAR`	$\leftarrow$	`PC`	`MAR == 0`
`MBR`	$\leftarrow$	`<MAR>`	`MBR == Bitcode von LOAD`
`IR`	$\leftarrow$	`MBR`	`IR == Bitcode von LOAD`
Dekodieren			führt zu:
`MAR`	$\leftarrow$	`MAR + 1`	`MAR == 1`
`MBR`	$\leftarrow$	`<MAR>`	`MBR == 20`
`MAR`	$\leftarrow$	`MBR`	`MAR == 20`
`MBR`	$\leftarrow$	`<MAR>`	`MBR == 10`
`A`	$\leftarrow$	`MBR`	`A == 10`
`PC`	$\leftarrow$	`PC + 2`	`PC == 2`

   dabei bezeichnet `<X>` den Inhalt der Zelle an der Adresse, die in `X` steht.

2. Ein kleines Assemblerprogramm zur Realisierung von

```
if ( x < y )
    z = 1;
else
    z = 2;
```

(Das Beispiel sollte in der Vorlesung drangewesen sein)

Wie sieht das Assemblerprogramm aus (es arbeitet mit absoluten Adressen nicht mit Labeln)

```
<Adresse 0>   LOAD    101
              NEG
              ADD     100
              JMPNEG  12
              LOAD    103
              STORE   102
              HALT
              ...
<Adresse 12>  LOAD    103
              STORE   102
              HALT
```

Dabei sei die Speicherbelegung wie folgt:

100	reservierter Platz für x
101	reservierter Platz für y
102	reservierter Platz für z
103 ''1''	
104 ''2''	

Online-Übung:

a) Implementiere `ADD x` im Mikrocode,
Wirkung von `ADD x`: `A == A + <x>`

b) Implementiere `SWAP x y` im Maschinencode (mittels `load` und `store`),
Wirkung von `SWAP x y`: `<x>,<y> := <y>,<x>`

Lektion 2: Adressierung

Stichworte:

- Speicher
- Adressbus
- MAR
- MBR

Online-Übung: Erläutere den Zusammenhang zwischen der Größe des MAR (in bits), des MBR und einer Speicherzelle. Betrachte dann die folgenden "Design-Alternativen" für einen Rechner und bestimmen die, die nicht sinnvoll sind (Begründung):

Alternative	MAR-Göße in Bits	Speichergröße	Länge eines Speicherwortes (in Bits)
a	10	1024	8
b	10	1024	12
c	9	1024	10
d	11	1024	10
e	10	10	1024
f	1024	10	10

Lektion 3: Ein/Ausgabe-Einheit

zu lesen: Abschnitt 8.5 Oberschelp/Vossen
Stichworte:

- Schnittstellen-Treiber und Controller
- Polling/Scanning
- Interrupts
- DMA

Online-Übung:
Diskutiere die verschiedenen Ansätze für Controller im Vergleich.