Karlsruher Institut für Technologie

Tutorium 42, #5

Max Göckel– *uzkns@student.kit.edu*

Institut für Theoretische Informatik - Grundbegriffe der Informatik

Speicher

Man kann sich den Speicher als eine Art Tabelle vorstellen. Die linke Spalte sind die Adressen, rechte Spalte die Daten an der jeweiligen Adresse.
Bei einer Tabelle wird immer nur ein Zustand gespeichert, nicht eine "Referenz" auf die Tabelle.

Bsp: Speicher m_0

Stelle	Wert
00	010
01	000
10	111
11	100

Speicheroperationen: Schreiben

Schreiben in den Speicher mit:

Definition

- $memwrite : Val^{Adr} xAdrxVal \rightarrow Val^{Adr}$.
  - $(m, a, v) \mapsto m'$

Dies schreibt den Wert **v** an Adresse **a** von Speichertabellenzustand **m** und gibt den veränderten Zustand **m'** zurück.

Speicheroperationen: Lesen

Lesen aus dem Speicher mit:

Definition

- $memread : Val^{Adr} \, xAdr \to Val$
  - $(m, a) \mapsto m(a)$

Dies liest aus der Adresse **a** von Zustand **m** aus und gibt diesen Wert zurück.

Speicheroperationen: Verkettung

Da die Operationen einen Rückgabewert haben, können wir sie Anstelle von konkreten Werten in anderen Operationen einsetzen.

Beispiel: Wert aus Stelle 00 auslesen und in 01 einsetzen.

m_1:

Stelle	Wert
00	010
01	000
10	111
11	100

Speicher: Aufgabe

Wie sieht m_3 nach der folgenden verketteten Operation auf Tabelle m_2 aus?

- $m_3 = memwrite(m_2, 11, memread(memwrite(m_2, 10, 000), 00))$

m_2:

Stelle	Wert
00	010
01	000
10	111
11	100

Speicher: Lösung

Wie sieht m_3 nach der folgenden verketteten Operation auf Tabelle m_2 aus?

- $m_3 = memwrite(m_2, 11, memread(memwrite(m_2, 10, 000), 00))$

m_3:

Stelle	Wert
00	010
01	000
10	111
11	010

Die MIMA: Aufbau

- Steuerwerk
  - Holt Befehle aus dem Speicher, decodiert sie und steuert die Ausführung
- Rechenwerk
  - Führt arithmetische/logische Operationen aus
- Speicher
  - Speichert Daten an Adressen, besteht aus Programm- und Datenspeicher
  - Adressen 20Bit, Werte 24Bit
- Register
  - Speichern je ein Wort oder eine Adresse (20/24Bit)
- Bus
  - Verbindet die Komponenten

Die MIMA: ALU

Führt die vom Steuerwerk decodierten Befehle aus (logisch oder arithmetisch).

- X, Y: Eingabewerte
- Z: Ausgabe der verarbeiteten Werte

Die MIMA: ALU-Operationen

Bitweise Operationen der ALU:.

Name	Operation in Worten
ADD	Zahl 1 + Zahl 2
AND	1 $\Leftrightarrow$ beide Eingabwerte 1 sind (logische $\wedge$)
OR	1, wenn min. 1 wert 1 ist (logisches $\vee$)
NOT	Jeder wert wird invertiert (Negierung)
XOR	1 wenn genau ein Wert 1 ist (exclusive or)

Die MIMA: Register

- Akku: Akkumulator (Zwischenspeicher)
- X, Y: ALU-Operanden
- Z: ALU-Ergebnis
- EINS: Die Konstante 1
- IAR: Instruktionsadressreg., Speichert den Speicherwert des nächsten Befehls
- IR: Instruktionsreg., Speichert die derzeitige Instruktion
- SAR: Speicheradressreg., (write-only) ruft im Speicher seinen wert auf
- SDR: Speicherdatenreg., Wert$\rightarrow$Speicher oder Speicher$\rightarrow$Wert

Die MIMA: Befehle

- Maschinenbefehle/Assemblersprache: Rechen- und Sprungbefehle
- Mikrobefehle: Bitfolgen, die an das Steuerwerk gesendet werden

Die Mikrobefehle werden in 7+ Takten abgearbeitet. Wie lange ein Takt (bzw. wann der nächste Takt beginnt) steuert ein Taktquarz im Steuerwerk.

Die MIMA: Befehle

- 1. - 5. Takt: Holphase. Die Stelle des Befehls kommt vom IAR, der Befehl wird in IR geladen
- 6. Takt: der Befehl wird vom Steuerwerk in die einzelnen Befehle decodiert
- 7.+ Phase: Die Befehle werden ausgeführt...

Die MIMA: Befehle

Auf einer Extra-PDF im ILIAS.

- Die Befehle werden aufsteigend durchnummeriert und dann abgearbeitet
- Alle Werte immer in binär aufschreiben!
- Es ist nicht möglich mehr als 20Bit lange Worte zu laden und es ist auch nicht möglich negative Zahlen zu laden
- Kommentare werden im Format *;<Kommentar>* geschrieben
- Jedes Assemblerprogramm endet mit einem HALT (Punktabzug!)

Die MIMA: Aufgabe

Was tut das folgende Programm?

```
LDC 1
STV 011
LDV 010
NOT
ADD 011
ADD 001
JMN here
LDC 1
STV 111 ;result
JMP end
here: LDC 0
STV 111 ;result
end: HALT
```

Die MIMA: Aufgabe

Schreibe ein Programm, dass den Wert an Adresse 001 verdreifacht.

Die MIMA: Lösung

Schreibe ein Programm, dass den Wert an Adresse 001 verdreifacht.

```
LDV 001
ADD 001
ADD 001
STV 001
HALT.
```

Die MIMA: Aufgabe

Schreibe ein Programm dass für ein $x \geq 0$ an Adresse adr *x div 2* ausführt und und das Ergebnis wieder in adr abspeichert.

Die MIMA: Lösung

Schreibe ein Programm dass für ein $x \geq 0$ an Adresse adr *x div 2* ausführt und und das Ergebnis wieder in adr abspeichert.

```
LDC 1
NOT
AND adr
RAR
STV adr
HALT.
```

Die MIMA: Aufgabe

Schreibe ein Programm, das für den Wert x an der Adresse *001* x mod 2 berechnet und das Ergebnis wieder in x speichert.

Die MIMA: Lösung

Schreibe ein Programm, das für den Wert x an der Adresse *001* x mod 2 berechnet und das Ergebnis wieder in x speichert.

```
LDC 1
AND 001
STV 001
HALT.
```