Hintergrundinformationen zur Vorlesung
Praktische Informatik I
WS 06/07

AG Betriebssysteme, FB3

Rechneraufbau und Rechnerstrukturen

Version1.2 vom 1.11.2006

1 Register

MAR	Memory Adress Register	Speicheradress–Register für die Adresse des nächsten anzusprechenden Speicherplatz
MBR	Memory Buffer Register	Puffer–Register für die Kommunikation mit dem Speicher
A	Akkumulator	General Purpose Register für alle anfallenden Aufgaben
MR	Multiplier Register	Multiplikator (Aufnahme von Multiplikationsergebnissen)
L	Link Register	Einstelliges Register für Überträge und Ähnliches
IR	Instruction Register	Befehlsregister, enthält den aktuellen Befehl
PC	Program Counter	Befehlszähler, enthält die Adresse des nächsten Befehls

- Der Akkumulator A enthält die Register MR und L, d.h. MR und L sind Erweiterungen des Akkus. Aus diesem Grund zerstört ein Schreiben auf MR den Inhalt von L und A (wichtig für Multiplikation, s.u.).
- Die Register A (und damit auch MR und L) und MBR sind Teil des Datenprozessors, ebenso das Rechenwerk ALU (Arithmetic Logical Unit).
- Die Register MAR, IR und PC sind Teil des Befehlsprozessors, ebenso wie der Dekodierer zum Entschlüsseln der Befehle und das Steuerwerk zur Steuerung der Ausführung.

2 Mikro-Instruktionen

MAR	←	PC	Inhalt von PC (Adresse) wird nach MAR gebracht
MBR	←	<MAR>	Inhalt der Speicheradresse, die im MAR steht, wird nach MBR gebracht
MBR	←	A	Inhalt des Akkumulators in das MBR laden
<MAR>	←	MBR	Inhalt von MBR auf die Adresse speichern, auf die MAR zeigt
IR	←	MBR	Instruction Register erhält Befehl aus MBR
PC	←	PC + n	PC um n erhöhen
PC	←	MBR	PC auf Adresse setzen, die im MBR steht
MAR	←	MAR+n	Adresse in MAR auf Inhalt + n setzen
MBR	←	MAR	Inhalt des MAR (Adresse) in das MBR bringen
PC	←	MAR	PC auf Adresse im MAR setzen
A	←	A + MBR	Berechnung im Akkumulator, Addition
A	←	$-$A	unäres Minus
A	←	$\neg$A	logische Negation
MR	←	A * MBR	Multiplikation
A	←	MBR	Konstante in Akku laden
A	←	A < 0	logischer Vergleich mit Konstanten, Ergebnis im Akku
MAR	←	MBR	Umladen von Adressen

Beim Maschinencode werden die tatsächlichen Register der CPU angegeben.

3 Pseudo-Assembler

LOAD x	Lade den Inhalt von Adresse *x* in den Akkumulator
STORE x	Speichere den Inhalt des Akkumulators in der Speicherzelle mit der Adresse *x*
ADD x	Addiere den Wert an Adresse *x* zum Inhalt des Akkumulators
SUB x	Subtrahiere den Wert von Adresse *x* vom Inhalt des Akkumulators
JMPNEG x	Springe zur Marke *x*, wenn der Inhalt des Akkumulators < *0*
JMPEQ x	Springe zur Marke *x*, wenn der Inhalt des Akkumulators = *0*
JMP x	Springe zur Marke *x*
NOT	logische Negation - bitweises Komplement des Wertes im Akkumulator
NEG	unäres Minus angewendet auf den Wert im Akkumulator.
HALT	Beendet das Programm