Aufgabenblatt 10

Ausgabe 04/01/2010, Abgabe bis 11/01/2010 12:00

Name(n):
Matrikelnummer(n):
Übungsgruppe:

Aufgabe 10.1 Adressierung (3+3+3+3+3 Punkte)

Welcher Wert steht nach Ausführung der folgenden Befehle im Akkumulator einer 1-Adress-Maschine?

a) `LOAD IMMEDIATE 20`
b) `LOAD DIRECT 20`
c) `LOAD INDIRECT 20`
d) `LOAD DIRECT 30`
e) `LOAD INDIRECT 30`

Der Hauptspeicher enthält jeweils diese Werte:

Adresse	Inhalt
20	50
30	40
40	60
50	70

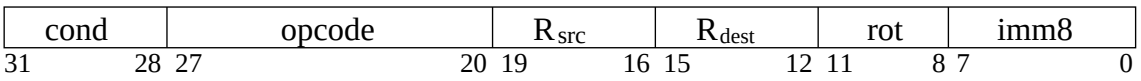
Aufgabe 10.2 Befehlskodierung (15 Punkte)

Entwerfen Sie eine Befehlskodierung, um alle der folgenden Befehle in 36-bit Befehlsworten unterzubringen:

- 7 Befehle mit zwei 15-bit Adressen und einer 3-bit Registernummer
- 500 Befehle mit einer 15-bit Adresse und einer 3-bit Registernummer
- 50 Befehle ohne Adressen oder Registerangaben

Aufgabe 10.3 Darstellung von Immediate-Operanden (4+4+4+4+4 Punkte)

Die für eingebettete Systeme und Mobilgeräte sehr beliebte 32-bit ARM-Architektur verwendet die folgende Darstellung von Immediate-Operanden für die arithmetischen Befehle:



Der bei diesen Befehlen verwendete Immediate-Wert ergibt sich aus folgendem Ausdruck:

Immediate-Wert $= \mathrm{imm8} \cdot 2^{(2*\mathrm{rot})}$

mit $0 \leq \mathrm{imm8} \leq 255$ und $0 \leq \mathrm{rot} \leq 12$.

Überlegen Sie sich die jeweilige 12-bit Kodierung der folgenden Immediate- Werte, oder begründen Sie, warum ein Wert nicht dargestellt werden kann:

a) 251
b) 259
c) 1233125376 $(= 2^{23} * 147)$
d) 2684354560 $(= 2^{31} + 2^{29})$
e) 573440 $(= 2^{14} * 35)$

Aufgabe 10.4 Befehlsformate (40+10 Punkte)

Vergleichen Sie 0-, 1-, 2- und 3-Adress-Maschinen, indem Sie für jede Maschine ein Programm zur Berechnung des folgenden Ausdrucks schreiben:

$W = (A * B + C)/(D - E * F)$

Die verfügbaren Befehle der entsprechenden Maschinen sind:

Dabei bedeutet M jeweils eine 16-bit Speicheradresse, während X,Y,Z entweder für eine 16-bit Speicheradresse oder eine 4-bit Registernummer kodieren.

0-Adress-Maschine (TOS "top of stack")

Mnemonic	Bedeutung
Push M	(push; TOS = M)
Pop M	(M = TOS; pop)
ADD	(tmp = TOS; pop; TOS = tmp + TOS)
SUB	(tmp = TOS; pop; TOS = tmp - TOS)
MUL	(tmp = TOS; pop; TOS = tmp * TOS)
DIV	(tmp = TOS; pop; TOS = tmp / TOS)

1-Adress-Maschine

Mnemonic	Bedeutung
LOAD M	(Akku = M)
STORE M	(M = Akku)
ADD M	(Akku = Akku + M)
SUB M	(Akku = Akku - M)
MUL M	(Akku = Akku * M)
DIV M	(Akku = Akku / M)

2-Adress-Maschine

Mnemonic	Bedeutung
MOV X,Y	(X = Y)
ADD X,Y	(X = X + Y)
SUB X,Y	(X = X - Y)
MUL X,Y	(X = X * Y)
DIV X,Y	(X = X / Y)

3-Adress-Maschine

Mnemonic	Bedeutung
MOV X,Y	(X = Y)
ADD X,Y,Z	(X = Y + Z)
SUB X,Y,Z	(X = Y - Z)
MUL X,Y,Z	(X = Y * Z)
DIV X,Y,Z	(X = Y / Z)

Die 0-Adress-Maschine verfügt über einen unbegrenzten Stack, die 1-Adress-Maschine über einen Akkumulator, und die 2-Adress- und 3-Adress-Maschinen über je 16 Universalregister.

a) Schreiben Sie (möglichst kurze) Programme für die Berechnung von $W = (A*B+C)/(D-E*F)$ auf den vier Maschinen.

b) Wenn die Befehlskodierung jeweils 8-bit für den Opcode verwendet (und natürlich 16-bit für eine Speicheradresse bzw. 4-bit für eine Registernummer), wieviele Bits werden dann für jedes der obigen vier Programme benötigt?

Welche Maschine hat also die kompakteste Kodierung (gemessen an der Programmgröße in Bits) für dieses Programm?