WS 2005/2006	Übungen zu 18.003 Bachelor Modul IP7 Rechnerstrukturen (RS) Teil 2	Aufgabenblatt 2.1
LV 18.004	J. Zhang	Abgabe: 16.01.06

Aufgabe 2.1.1 [10 Punkte] Zuordnung von Begriffen:
Ordnen Sie die Begriffe aus der untenstehenden Liste den Fragen zu. Benutzen Sie dazu die vorangestellten Buchstaben. Jeder Begriff sollte nur einmal vorkommen.

a	Abstraktion	k	DRAM (dynamic random access memory)
b	Assembler	l	ALU (arithmetic logic unit)
c	Binärzahl	m	Instruktion
d	Bit	n	Instruction Set Architecture
e	Cache	o	Speicher
f	CPU (central processor unit)	p	Betriebssystem
g	Compiler	r	Prozessor
h	Rechnerfamilie	s	Halbleiter
i	Steuerwerk	t	Transistor
j	Datenpfad	u	VLSI (very large-scale integrated circuit)

2.1.1.1 [0.5]: Spezielle Abstraktion, die die Hardware der low-level Software zur Verfügung stellt.

2.1.1.2 [0.5]: Aktiver Teil eines Computers, der die Instruktionen eines Programmes abarbeitet. Addiert Zahlen, testet Zahlen usw.

2.1.1.3 [0.5]: Andere Bezeichnung für Prozessor.

2.1.1.4 [0.5]: Ansatz beim Entwurf von Hard- und Software. Hierarchisches Vorgehen, wobei höhere Hierarchie-Stufen Details niedriger Hierarchie-Stufen verbergen.

2.1.1.5 [0.5]: Zahl zur Basis 2.

2.1.1.6 [0.5]: Binärziffer.

2.1.1.7 [0.5]: Implementationen identischer Instruction Set Architectures, die zeitgleich auf dem Markt verfügbar sind und sich in Preis und Leistungsfähigkeit unterscheiden .

2.1.1.8 [0.5]: Einheit des Prozessors, die die arithmetischen Operationen ausführt.

2.1.1.9 [0.5]: Teil des Prozessors, der die Steuersignale für Datenpfad, Speicher, I/O-Geräte usw. in Abhängigkeit des auszuführenden Befehles generiert.

2.1.1.10 [0.5]: Hardware, die eine Instruction Set Architecture implementiert.

2.1.1.11 [0.5]: Spezielles Kommando/Befehl an einen Rechner.

2.1.1.12 [0.5]: Integrierte Schaltungen, aus denen üblicherweise der Hauptspeicher besteht.

2.1.1.13 [0.5]: Begriff der Technik, die es erlaubt, millionen von Transistoren auf einem Chip zu fertigen.

2.1.1.14 [0.5]: Einheit, die das Programm enthält, wenn es ausgeführt wird. Die Einheit enthält auch die erforderlichen Daten.

2.1.1.15 [0.5]: Als elektrisch gesteuerter Ein-Ausschalter einsetzbar.

2.1.1.16 [0.5]: Programm, das die Ressourcen eines Rechners zum Nutzen weiterer laufender Programme verwaltet.

2.1.1.17 [0.5]: Programm, das die symbolische Darstellung von Instruktionen in die binäre Repräsentation transformiert.

2.1.1.18 [0.5]: Programm, das von einer höheren Programmiersprache in die Assemblersprache übersetzt. wird.

2.1.1.19 [0.5]: Kleiner, schneller Speicher, der als Puffer für den Hauptspeicher dient.

2.1.1.20 [0.5]: Material, das im üblichen Sinne kein guter elektrischer Leiter ist.

Aufgabe 2.1.2 [25 Punkte] Grundelemente und Grundprinzipien eines Computers: Fertigen Sie eine Skizze an, anhand derer Sie die Grundprinzipien eines Computers einem Laien erklären können. Verwenden Sie für die Skizze Grundelemente wie CPU, Speicher, I/O-Geräte, Systembus usw. Die Erläuterungen sollten max. eine Seite umfassen.

Aufgabe 2.1.3 [20 Punkte] Ebenen eines Computers: Andrew S. Tanenbaum unterscheidet fünf Modell-Ebenen eines Rechners. Die unterste Ebene, die Ebene 0, wird durch die Digitale Logik gebildet, dann folgt die Register-Transfer-Ebene (RT-Ebene), dann die Instruction-Set Architecture (ISA), die Ebene der Betriebssysteme, die Ebene der Assemblersprache und üblicherweise letztlich die Ebene 5 mit den problemorientierten Sprachen. So könnte es z.B. vorkommen, dass einige Instruktionen auf der Ebene der Betriebssystemmaschine identisch sind mit entsprechenden Instruktionen auf der ISA-Ebene. Diese Instruktionen würden dann nicht vom Betriebssystem, sondern im Falle einer mikroprogrammierten Maschine direkt vom Mikroprogramm und im Falle eines hartverdrahteten Steuerwerkes direkt durch dieses interpretiert. Kann dieser Fall vorkommen? Wenn Ja, begründen Sie, ob bzw. warum es sinnvoll ist und wenn Nein, begründen Sie, warum es nicht möglich ist.

Aufgabe 2.1.4 [15 Punkte] Moores Law: Der MC 86000 verfügte bei einer Chipfläche von etwa $44\,mm^2$ über ca. 68000 Transistoren. Wieviele Transistoren ließen sich 1 1/2 Jahre nach Erscheinen des 68000 mit der dann aktuellen Technik auf der gleichen Fläche integrieren? Was bedeutet das für die Strukturgrößen?

Aufgabe 2.1.5 [10 Punkte] Adressierung:
Welcher Wert steht nach Ausführung der folgenden Befehle im Akkumulator einer 1-Adress Maschine?

2.1.5.1 [2]: LOAD IMMEDIATE 20
2.1.5.2 [2]: LOAD DIRECT 20
2.1.5.3 [2]: LOAD INDIRECT 20
2.1.5.4 [2]: LOAD DIRECT 30
2.1.5.5 [2]: LOAD INDIRECT 30

Der Hauptspeicher enthält jeweils diese Werte:

Adresse	Inhalt
20	40
30	50
40	60
50	70

Aufgabe 2.1.6 [30 Punkte] Befehlsformate:

2.1.6.1 [20]: 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 16 Universalregister.

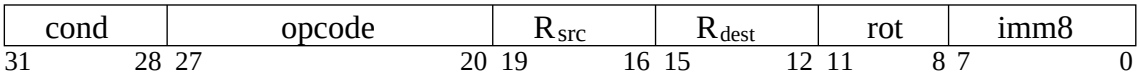
2.1.6.2 [10]: Wenn die Befehlskodierung jeweils 8-bit für den Opcode verwendet (und natürlich 16-bit für eine Speicheradresse bzw. 4-bit für eine Registernummer), wieviele Bits werden dann für jedes der obigen vier Programme benötigt?

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

Aufgabe 2.1.7 [15 Punkte] Befehlskodierung:
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 2.1.8 [15 Punkte] Darstellung von Immediate-Operanden:
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 $= imm8 \cdot 2^{(2*rot)}$

mit $0 \leq imm8 \leq 255$ und $0 \leq rot \leq 12$.

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

2.1.8.1 [3]: 253

2.1.8.2 [3]: 257

2.1.8.3 [3]: 1233125376 $(= 2^{23} * 147)$

2.1.8.4 [3]: 2684354560 $(= 2^{31} + 2^{29})$

2.1.8.5 [3]: 573440 $(= 2^{14} * 35)$

Aufgabe 2.1.9 [20 Punkte] Adressierungsmechanismus:
Entwerfen Sie einen Adressierungsmechanismus, der eine beliebige Menge von 32 Adressen (die nicht aufeinanderfolgen müssen) in einem Adressraum von insgesamt 4 GB (32 Bits) mit nur 5-Bits im Befehlswort kodiert.