Lunds Universitet
LTH

Exempeltentamen
Datorteknik, EIT070,

Skrivtid: xx.00-xx.00

Tillåtna hjälpmedel: Inga.

Maximalt antal poäng: 50 poäng
För betyg 3 krävs 20 poäng
För betyg 4 krävs 30 poäng
För betyg 5 krävs 40 poäng

Alla lösa blad skall vara samlade i omslagsarket.
Inlämnade uppgifter skall vara försedda med uppgiftens nummer.
Lösningarna skrivs in i nummerordning.
Skriv namn på varje ark.
Omslagsarket skall vara fullständigt ifyllt med inskrivningsår, namn och personnummer.
Kryssa för lösta uppgifter och ange antalet inlämnade blad.

Uppgift 1.

- Totalt 10 poäng.
- Beskriv och motivera lösning tydligt (omotiverade svar = poängavdrag).
- Dina antaganden ska tydligt framgå

a) Förklara vad en kompilator är och vad den används till.

Svar: En kompilator är ett datorprogram som utifrån en programtext skrivet i ett högnivåspråk som till exempel C, skapar ett motsvarande lågnivåprogram kapabelt att utföra de aktiviteter som programtexten beskriver, alltså ett slags översättare. Normalt genererar en kompilator maskinkod,

b) Förklara sambandet mellan klockfrekvens och klockcykeltid.

Svar: f=1/T. En processor med en frekvens på 1 MHz som utför en instruktion varje klockcykel utför således 1.000.000 instruktioner per sekund.

c) Ge exempel på vad som påverkar prestandan i en dator förutom klockfrekvens.

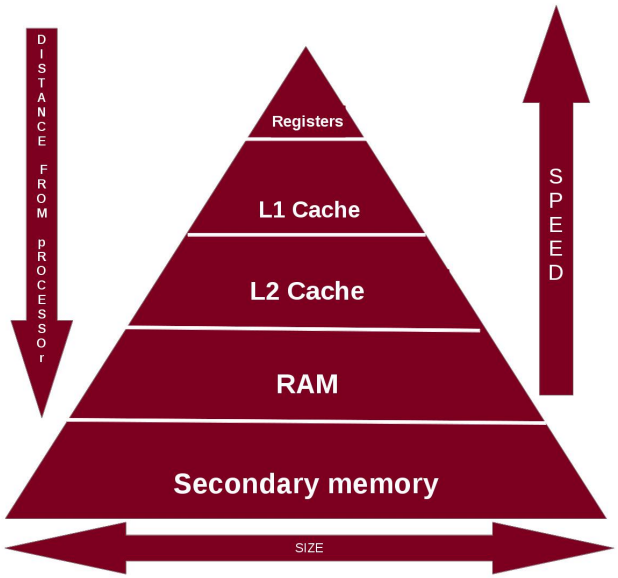
Svar: Algoritmer påverkar hur många instruktioner som behövs, programmeringsspråk påverkar antal instruktioner som behövs, kompilatorn påverkar hur många maskininstruktioner som behövs, kompilatorn påverkar prestanda av minnesaccess genom att använda mycket register, hårdvaran påverkar tid för varje instruktion.

Uppgift 2.

- Totalt 10 poäng.
- Beskriv och motivera lösning tydligt (omotiverade svar = poängavdrag).
- Dina antaganden ska tydligt framgå

a) Beskriv en minneshierarki.

Svar:



Typiska värden är:
Processor registers: 8-32 registers (32 bitar -> 32-128 bytes)
accesstid: få ns, 0-1 klockcykler
On-chip cache memory (L1):32 till 128 Kbytes, accesstid = ~10 ns, 3 klockcykler
Off-chip cache memory (L2):128 Kbytes till 12 Mbytes, accesstid = 10-tal ns, 10 klockcykler
Main memory:256 Mbytes till 4Gbytes, accesstid = ~100 ns, 100 klockcykler
Hard disk:1Gbyte tid 1Tbyte, accesstid = 10-tal milliseconds, 10 000 000 klockcykler

b) Beskriv likheter och olikheter av datatyperna:
Unsigned char och signed char

Svar: Båda kräver 1 byte i lagringsutrymme. Skillnaden är i hur tal representeras. För unsigned char kan tal mellan 0 och 255 lagras. För signed char kan tal mellan -128 och 127 lagras.

c) Förklara Little Endian och Big Endian.

Svar: Om t ex ett tal som kräver två bytes (byte 1 och byte 2) ska lagras så kan byte 1 lagras innan byte 2, eller tvärtom.

Uppgift 3.

- Totalt 15 poäng.
- Beskriv och motivera lösning tydligt (omotiverade svar = poängavdrag).
- Dina antaganden ska tydligt framgå

a) Förklara varför det är bra med cacheminne.

Svar: Cacheminne är för program där det uppstår lokalitet i tid och rum. Lokalitet är t ex när samma instruktion (och instruktioner i dess närhet) används flera gånger. Används en instruktion endast engång per exekvering är cacheminne inte bra.

b) I cacheminnen, vad är skillnaden mellan direktmappning och associativmappning?

Svar: I ett cacheminne med direktmappning kan en byte bara hamna i en specifik cacherad. Det underlättar implementation och man slipper ha så kallade ersättningsalgoritmer. Men, det minskar flexibilitetet i att ha data/instruktioner i cacheminne. Associativmappnng är det mest flexibla. Data/instruktioner kan hamna i vilken cacherad som helst. Det krävs algoritmer för att för att bestämma vilken cacherad som ska ersättas vid en miss. Det tar också längre tid vid sökning efter data eftersom alla cacherader måste sökas igenom.

c) Antag ett cacheminne på 64 byte som använder direktmappning och där blockstorleken/cacheline är på 8 byte.
    1. Om processorn läser på adress C000EAFCH, vilka minnesceller (vilka adresser) kommer då att läsas in i cacheminnet?

       Svar: C00EAF8 – C000EAFFH (OBS, man läser in 8 byte i taget och det gäller att hålla kolla på hur dessa ligger.)

2. Efter att processorn läst på adress C000EAFCH, fortsätter processorn att läsa adresserna enligt tabellen nedan i turordning. Ange för varje läsning om det är en träff (hit) eller miss (miss), där följande gäller för varje fall:
    i. Fall 1: blockstorleken 8 byte och direktmappning
    ii. Fall 2: blockstorleken 16 byte och direktmappning
    iii. Fall 3: blockstorleken 8 byte och associativmappning

Svar: (Det kluriga i uppgiften är att hålla koll på vilka block som läses. Om man läser på en adress så blir det naturligtvis inte alltid den adressen och ett antal byte med högre adress)

	Fall 1	Fall 2	Fall 3
C000EB00H	MISS	MISS	MISS
C000EB04H	TRÄFF	TRÄFF	TRÄFF
C000EB08H	MISS	TRÄFF	MISS
C000EB0CH	TRÄFF	TRÄFF	TRÄFF
C000EB80H	MISS	MISS	MISS
C000EB84H	TRÄFF	TRÄFF	TRÄFF
C000EB00H	MISS	MISS	TRÄFF
C000EB88H	MISS	MISS	MISS
C000EB8CH	TRÄFF	TRÄFF	TRÄFF
C000EB90H	MISS	MISS	MISS

Uppgift 4.

- Totalt 15 poäng.
- Beskriv och motivera lösning tydligt (omotiverade svar = poängavdrag).
- Dina antaganden ska tydligt framgå

a) Förklara följande adresseringsmoder:
   a. Direktadressering (Direct addressing)
   b. Omedelbaraddressering (Immediate addressing)
   c. Minnesindirektadressering (Memory indirect addressing)
   d. Registeradressering (Register addressing)
   e. Register indirektadressering (Register indirect addressing)

Svar:
(a) I direktadressering så pekar operanden ut en adress i minnet
(b) I omedelbaradressering ligger operanden i instruktionen
(c) I minnesinderektadressering ligger operanden i en adress som pekas ut av den adress som finns i instruktionen
(d) I registeradressering ligger operanden i det register som pekas ut i instruktionen
(e) I registeradressering pekas ett register ut i instruktionen. Det registret innehåller en adress i minnet där operanden ligger.

b) Vilken/vilka av adresseringsmoderna ovan resulterar i minst konflikter i en pipeline (förklara och motivera)?

Svar: För att minska antalet konflikter gäller det att minska antalet minnesaccesser till RAM/primärminne. Adresseringsmoder som har minst accesser till minnet är: omedelbaradressering och registeradressering.

c) Beskriv vad som händer om följande instruktion exekveras i en pipeline:

ADD R4, X //R4 <- R4 + X

Svar: Om pipelinen består av FI, DI, CO, FO, EI, WO, händer följande. I FI kommer instruktionen hämtas från minnet. I DI avkodas instruktionen så att processorn vet vad som ska göras. I CO beräknas vilka operander som är inblandade. I detta exempel är det register R4 och minnesplatsen X. I FO hämtas operanderna, dvs innehållet i register R4 och innehållet på minnesplats X. I EI beräknas additionen mha ALUn. I WO skrivs svaret av beräkningen i register R4.

d) När uppkommer kontrollkonflikter i en pipeline?

Svar: Kontrollkonflikter uppstår på grund av hopp. Normalt (inga hopp) så hämtas instruktionerna i den ordning de ligger i minnet. Det blir i en pipeline så att man påbörjar arbetet med en instruktion men den är fel eftersom man ska göra ett hopp.

e) Förklara hur delayed branching används om programmeraren skrivit detta:

MUL R3, R4	R3<-R3*R4
SUB #1, R2	R2<-R2-1
ADD R1, R2	R1<-R1+R2
BEZ TAR	Branch om zero
MOVE #10, R1	R1<-10

Svar: Vid villkorliga hopp (t ex BEZ) uppkommer en kontrollkonflikt. Ett enkelt sätt är att fördröja pipelinen så att 1 eller flera efterföljande instruktioner inte exekveras. Man gör så kallade STALL. Problemet är att prestandan går ner eftersom man slösar bort tid. För att undvika detta slöseri med tid kan man exekvera en instruktion som alltid ska exekveras. I detta fall kan man exekvera MUL R3, R4. Man placerar då den instruktionen efter BEZ TAR men i och med att man har en pipeline så kommer MUL R3, R4 att exekveras oavsett om hoppet tas eller inte.