Rahim Rahmani (rahim@dsv.su.se)
Division of SAS
Department of Computer and Systems Sciences
Stockholm University

SVAR TILL TENTAMEN I DATORSYSTEM, HT2013

Tentamensdatum: 2013-10-30

Tentamen består av totalt 16 uppgifter. Del **A** består av 11 uppgifter och Del **B** av 5 uppgifter.
Totalt antal poäng: 30 p

Tillåtna hjälpmedel: Inga

Skrivhänvisningar:

- Skriv **läsbart och tydligt** för att undvika feltolkningar.
- **Motivera dina Svar**, ev. tabeller och beräkningar som används för att nå svaret ska också finns i lösning.
- ***Ofullständigt*** motiverade svar kan **INTE** ge poäng

Information om **betygsskala** kommer att läggas ut på kursenssidan i ileran.

LYCKA TILL!

Del A

Frågorna 1 till 3 är multiple choice frågor. välj rätt alternativ på multiple choice frågorna. Endast ett alternativ är rätt. Fler angivna svar beaktas ***inte.***

1. Vilken instruktion används för att addera värdet i två register och sedan spara resultatet i ett tredje register?. (1p)
   a) addi
   b) add
   c) addia

Svar: alternativ b är rätt

2. Vilken instruktion används för att lägga en minnesadress i ett register? (1p)
   a) movia
   b) addi
   c) movi

Svar: alternativ a är rätt

.

3. Vilken instruktion används för att skriva ett ord till minnet? (1p)
   a) stw
   b) stwio
   c) ldw

Svar: alternativ a är rätt

4. Beräkna paritetsbiten för följande strängar, både med jämn och udda paritet: (2p)

Bitsträng								Jämna	Udda
1	1	0	0	1	0	1	0	0	1
1	0	1	1	1	0	1	0	1	0
1	1	1	1	1	1	1	1	0	1
0	0	0	1	0	0	0	0	1	0

5. Konvertera talet 9.25 till binära talbasen. (1p)
   Svar: 1001,01

6. Konvertera $128E_{16}$ till decimal talbasen. (1p)
   Svar: 4750

7. Skriv sanningstabellen för AND, OR och NOT. (1p)

X AND Y		
X	Y	X.Y
0	0	0
1	0	0
0	1	0
1	1	1

X OR Y		
X	Y	X+Y
0	0	0
1	0	1
0	1	1
1	1	1

NOT	
E	$\bar{E}$
0	1
1	0

8. Under de senaste åren har processorns prestanda utvecklats exponentiellt. Prestandautvecklingen har både berott på att processortillverkarna har kunnat öka processorns klockpuls men också på grund av innovationer av processorns arkitektur. Ge exempel på två innovationer av processorns arkitektur som har ökat processorns prestanda under senaste åren. (2p)

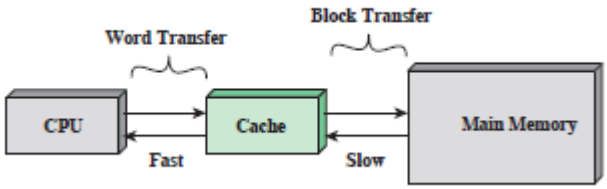
   Svar: Två exempel är Pipelines och Cacheminne. Flera exempel finns i kapitel 2 i kursboken.

9. Förklara vilken funktion har en buss i ett datorsystem. (1p)

Svar: En buss tillhandahåller kommunikation mellan olika enheter i ett datorsystem genom att varje enhet har tillgång till gemensamma ledningar för data, kontroll och adressering.

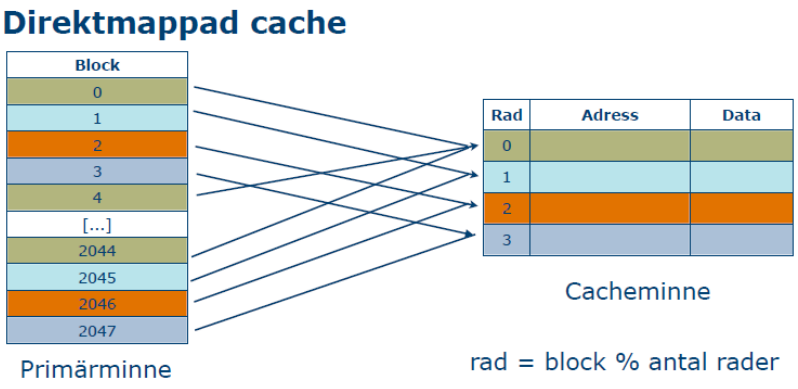
10. Förklara:
    a) vilken funktion har cache minne? (1p)

    Svar: Cacheminne har sin funktion mellan processorn och primärminnet och det sitter i praktiken alltid inbyggt i processorn. Även det strävar efter att öka processorns presentanda genom att utnyttja minnesreferensernas tidslokalitet, och rumslokalitetet.



    b) beskriva mappningsfunktionen i Direktmappat cacheminne. (1p)

    Svar: Se bilderna 10-12 från föreläs.7 (F7) även nedan exempel (Obs, bara exempel är inte fullständigt svar)

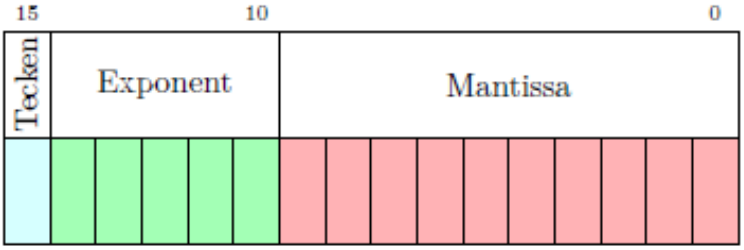


11. Förklara skillnaden mellan maskable Interrupt och nonmaskable Interrupt? (1p)
Svar: En maskable interrupt kan ignoreras av systemet medan en nonmaskable Interrupt kan inte ignoreras utan måste tas om hand.

Även mer om de olika Interrupt kan hittas i kap. 4 och F4 bilder 37-39.

Del B

1. Flyttal är ett sätt att representera stora, små och rationella tal. I nedanstående uppgift används standarden IEEE 754 för 16-bitars flyttal. Se bilden nedan om flyttals binärt lagring.



Beräkna operation C*D. Svara med ett 16-bitars binär flyttal enligt bilden ovan och IEEE 754.
Flyttal C : 0 10100 1001000000
Flyttal D : 0 10110 0101000000 (4p)

Formeln för att få ut det decimala värdet *v* ur ett flyttal:
$v=(-1)^{\text{teckenbit}} \times 2^{\text{exponent-15}} \times (1, \text{mantissa})_2$

OBS: ofullständigt svar kan **INTE** ge poäng dvs tabeller och beräkningar som används för att nå svaret ska också finns i lösningen.

Svar: Vi använder formeln från uppgiften för att få ut de värden vi behöver.
Flyttal **C:** Teckenbiten är 0, vilket get ett positivt tal. Exponenten är 10100, vilket ger 20. Eftersom formeln ger 20-15 som den faktiska exponenten får vi att exponenten är 5. Enligt formeln har vi också mantissan 1,1001.
Flyttal **D**: Tecken är 0, så att talet är positiv. Exponenten är 10110, vilket ger 22. Ur formeln får vi 22-15=7. Mantissan bli 1,0101 enligt formeln.
För att multiplicera de två talen använder vi formeln $(2^{x} \times C) \times (2^{y} \times D) = 2^{x+y} \times (C \times D)$.
Det ger i vårt fall $2^{5+7} \times (1,1001 \times 1,0101)$. 5+7 =12, och vi har därmed vår nya exponent.

Vi multiplicerar mantissorna från talen **C** och **D** och får följande:

```
          1, 1 0 0 1
        * 1, 0 1 0 1
        ----------------------
          1  1 0 0 1
        0  0  0 0 0
      1 1 0  0 1
    0 0 0 0  0
+ 1 1 0 0 1
---------------------------------------
1   0, 0 0 0 0 1 1 0 1
```

Utifrån den framräknade exponenten och mantissan kan vi konstruera ett nytt flyttal. Eftersom mantissan just nu är 10,0001101 behöver vi skifta talet så att vi instället får 1,000001101. Eftersom vi skiftar ett steg åt höger måste vi öka exponenten med ett och får därmed 13. För att räkna ut de nya exponent bitarna ökar vi 13 med 15 och får 28,

$28_{10} = 11100_2$, vilket ger våra fem bitar för exponenten.

Mantissan har redan 1, enligt formeln så i bitarna för mantissan lägger vi in 0000011010. Eftersom talet fortfarande är positiv bli tecken biten 0.

Flyttal C * Flyttal D = 0 11100 0000011010

2. Förklara vad som menas med en instruktion pipeline? (1p)

   Svar: Instruktion pipeling innebär att man gör det möjligt för processorn att utföra flera instruktioner samtidigt genom att köra flera fetch-decode-execute samtidigt. Tex man kan fetcha en instruktion på en pipeline samtidigt som man decodar en annan instruktion på en annan pipeline.

3. Rita av följande tabell, och sätt ett kryss för att markera vilka av funktionerna **1-5** som hanteras av protokoll **A-D**. Det kan vara inget, ett eller flera kryss i en kolumn, och i en rad. (2p)

	A: Ethernet	**B**: IP	**C**: UDP	**D**: TCP
1: Felupptäckande kod & Automatisk omsändning				X
2: Flödesstyrning				X
3: Congestion avoidance (Trafikstocknings-hantering)				X
4: Felupptäckande kod och bortkastande av felaktiga packet	X	X	X	
5: Kollisionshantering för Länk användning	X			

4. Beskriv e-postmeddelandets väg från klient via e-postservrar till rätt mottagarklient, och vilka mekanismer som gör att det ”hittar rätt”. Avsändar- och mottagarklienten tillhör inte samma e-postserver. Klienten sitter på samma subnet som sin e-postserver. Epostservrarna tillhör inte samma subnet. Beskrivningen ska mycket kort nämna samtliga TCP/IP-nivåer. Använd begreppen e-postadress, DNS, SMTP, POP3, TCP-segment, portnummer, IP-adress, MAC-adress, router och ARP. (4p)

   Här ska ni besvara frågan utifrån TCP/IP skiktning (Application, Transport, Internet och Physical (network Access)). Det ska DNS-uppslagningen och använding av pop3, SMTP och MAC etc. se F 13.

5. Ett cacheminne har plats för total 512 bytes och varje rad är 16 bytes lång. Minnet är direktmappat och en adress är 32 bitar lång. Förutsatt att cacheminnet är tomt från början, beräkna för varje instruktion nedan om det blir en cacheträff eller en cachemiss. Du kan också anta att instruktionerna sker sekventiellt och att data som läggs till i en uppgift finns kvar till nästa. (5p)

```
1   movia r8,     0xBEDA12C4
2   ldw r10,  0( r8 )
3   ldw r11, 16( r8 )
4.  stw r10, 32( r8 )
5.  ldw r10,  48(r8)
6. ldw r10,  64(r8)
```

OBS: ofullständigt svar kan **INTE** ge poäng dvs tabeller och beräkningar som används för att nå svaret ska också finns i lösningen.

Svar: 512 bytes stort och 16 bytes per rad
512/ 16= 32 rader i minnet
För att representera 16 bytes (0-15) behövs 4 bitar.
För att representera 32 rader (0-31) behövs 5 bitar
Hela adressen (32 bitar) minus 4 minus 5:
32-(4+5)=23 bitar är TAG

1. Cacheminnet är tomt -> cache miss, 0xBEDA12C4 + 0=

2. Från 0xBEDA12**C4**

C4$_{16}$ = 11000100$_2$ så att 0xBEDA12**C0 ---** 0xBEDA12**CF**
På rad 11000 , tom rad cache miss

3. Från 0xBEDA12**D4**

D4$_{16}$ = 11010100$_2$ så att 0xBEDA12**D0 ---** 0xBEDA12**DF**
På rad 11010 , tom rad cache miss

4. Från 0xBEDA12**E4**

E4$_{16}$ = 11100100$_2$ så att 0xBEDA12**E0 ---** 0xBEDA12**EF**
På rad 11100 , tom rad cache miss

Sedan du räknar på samma sätt för 5 och 6.