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

SVAR TILL TENTAMEN I DATORSYSTEM, VT2013

Tentamensdatum: 2013-03-21

Tentamen består av totalt 16 uppgifter. Del **A** består av 11 uppgifter och
Del **B** av 5 uppgifter.
Totalt antal poäng: 36 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!

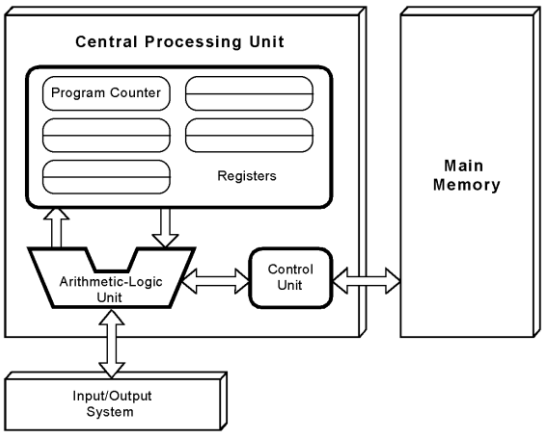
Del A

1. Beskriv vad innebär att ”fetch an instruction”. (1p)

   Fetch an instruction innebär att CPU hämtar en instruktion antigen i ett register eller i primärminnet för att sedan kunna exekvera den.

2. Beskriva Von Neumann-modellen även förklara syftet med: (3p)
   a) Arithmetic-Logic Unit (ALU).
   b) Program Counter (PC).
   c) Control Unit (CU).

   Von Neumann-modellen för instruktion exekvering i CPU. Control Unit hämtar instruktion genom att få info om deras adresser register av PC. Instruktionen dekodas och skickas till ALU för att genomföra de aritmetiska beräkningarna.



   ALU: Är den aritmetiska och logiska enheten. Utför beräkningar av operander.
   PC: Anger nästa instruktion som ska exekveras
   CU: Hämtar instruktioner från register / Main Memory & dekodar dem.

3. Konvertera talet $137B_{16}$ till decimal talbasen. (1p)
   $137B_{16}$ => 0001 0011 0111 1011 $_2$
   Svar: 4987_{10}

4. Konvertera talet 0**.**8125 till binära talbasen. (1p)

$$
\begin{array}{rl}
0.8125 & \\
-\ 0.5000 & = 2^{-1} \times 1 \\
\hline
0.3125 & \\
-\ 0.2500 & = 2^{-2} \times 1 \\
\hline
0.0625 & \\
-\ \quad 0 & = 2^{-3} \times 0 \\
\hline
0.0625 & \\
-\ 0.0625 & = 2^{-4} \times 1 \\
\hline
0 &
\end{array}
$$

0.8125_{10} *= 1/2 + 1/4 + 1/8 + 1/16*

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

AND			**OR**			**NOT**	
X	Y	X.Y	X	Y	X+Y	X	X`
0	0	0	0	0	0	0	1
0	1	0	0	1	1	1	0
1	0	0	1	0	1		
1	1	1	1	1	1		

6. Förklara: (2p)
   a) vad är en ”bus cycle” ?

En ”bus cycle” är tiden mellan två ”ticks” för bussklockan. ”Bus cycle” är alltså den tiden man har på sig att överföra data mellan CPU:n och primärminnet.

Begreppet ”bus cycle” står i sig själv för själva mellanrummet mellan två ”tickningar” som sker hos en bus clock. På så sätt kan man genom en bus cycle se hur mycket tid som krävs för att utföra olika typer av skrivning eller läsning emellan lagringsenheten(primärminnet) och central processing unit(CPU:n). Så under varje bus cycle skickas data via en bus snabbt och effektiv mellan dessa två enheter.

   b) skillnaderna mellan databussar, adressbussar och kontrollbussar.

Data som skickas via bussar förekommer i olika former och hanterar olika typer av saker. Det är även så att bussar inte alltid är ”point-to-point” utan används även ofta av flera olika enheter. På grund av detta har det utvecklats specifika ”lines” som alla hanterar en viss typ av data. Det finns tre huvudbussar som främst förekommer vilket är; databussar, adressbussar och kontrollbussar: - ***Databussar*** erhåller själva datainformationen som ska skickas ifrån punkt A till punkt B. - ***Adressbussar*** däremot erhåller information om vart dessa punkter finns lokaliserade, vilket är nödvändigt för att datorn ska kunna avgöra vart data förväntas att antingen läsas/skrivas ifrån. - ***Kontrollbussar*** är oerhört viktiga då de avgör vilka enheter som har tillåtelse till användande av bussen, samt registrerar även bussanvändarens syfte till användningen av bussen. Denna buss behandlar även viktiga kommandon som skickas ifrån CPU:n. Den hanterar till exempel förfrågningar ifrån enheter, eventuella avbrott, och synkroniseringssignaler av datorns klocka(clock).

7. Beskriv gränssnittet mellan CPU och I/O – Specificera delvis funktionalitet i I/O. (1p)

   Gränssnitt mellan CPU och I/O som bestämmer hur de olika enheterna kommer att kommunicera. Detta kan tex ske genom programmed I/O (CPU väntar på instruktioner från I/O).
   interrupts driven I/O (att Interrupts kontroller skickar interrupt signal till CPU när det finns instruktioner att hämta). Mer information kan hittas i föreläs. 8 bilderna 4-7.

8. Förklara fördelar och nackdelar av två cache minnes skrivningspolicy. (2p)

   De två olika cacheminnesskrivningspolicy som finns är write back och write through.
   Write through – Den här policyn innebär att eventuella förändring av information uppdateras hos cacheminnet såväl som i primärminnet vart eftersom någon förändring sker. Det uppenbara fördelen med detta är att information ständigt blir uppdaterad hos båda enheterna vilket gör att den befintliga informationen i cache stämmer överens med primärminnet och gör att data blir ”aktuell” och riktig. Den stora nackdelen är däremot att det kräver extra tid och arbete för datorn att uppdatera cache såväl som primärminnet, vilket gör att systemet blir långsammare.
   Write back – I denna policy uppdateras de olika blocken i primärminnet endast när block behöver rensas ur cacheminnet. Detta medför en nackdel som innebär att primärminnets data inte alltid kanske stämmer överens med den data som erhålls i cache, vilket kan resultera i ett förlorande av informationen om en processkrasch inträffar. Fördelen med denna skrivningspolicy är däremot att den är mycket snabbare än write-through-policyn då den inte ständigt behöver lägga ned arbete på att uppdatera minnet vid varje cache-uppdatering.

   Även se bilderna 20-22 i Föreläs. 7

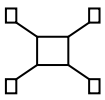
9. Vilka faktorer är avgörande för en processors prestanda? (2p)
   Den första viktiga faktorn som har en oerhört står påverkan på en dators prestanda är själva ***huvudprocessorn(CPU).*** Eftersom denna enhet är den enheten som utför alla program och operationer som inkluderar bland annat beräkning(som sker i ALU:n) och hantering av olika typer av data. En av de viktigaste delarna i en processor är den så kallade ***processorkärnan***, som är den delen i processorn som utför själva beräkningarna av data. Därefter ju fler integrerade processorkärnor en processor erhåller, desto flera beräkningar kan exekveras parallellt, vilket därefter ökar datorns prestanda.
   En annan viktig faktor är en dators prestanda är dess ***klockfrekvens*** som mäts i MHz. En klocka i datorn har en så kallad huvudklocka-signal som tickar ifrån 0 till 1 och därefter 1 till 0. Detta främst för att upprätthålla en slags synkron och se till att olika slags exekveringar sker i relativt synkroniserade intervaller. Sparregister måste till exempel invänta nästa ”tick” innan de kan ladda data. Därmed är det logiskt att ju snabbare klockfrekvens en dator har, ju snabbare och bättre kan den prestera. Under varje ”tick” sker olika typer av exekveringar och sparande, och allt måste hinnas med under samma klockslag. Dock innebär detta att klockan måste anpassas så att det utförandet som tar längst tid hinner att utföras, vilket är något som måste tas hänsyn till och blir ett hinner för att kunna sätta en allt för hög klockfrekvens.
   Arkitekturen på en dator och dess komponenter kan också vara en avgörande faktor för prestandan. Om enheter till exempel har kortare avstånd när det gäller kommunikation eller extra komponenter som påskyndar eventuell kommunikation eller exekveringar kan prestandan ökas avsevärt då en kortare färdväg ger snabbare exekvering och kortare

”väntetider” hos vissa utföranden. Snabbare tillgång till minne, snabbare läsning av minne, snabbare kommunikation är exempel på detta som därefter bidrar till en bättre prestanda.

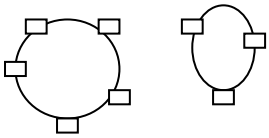
10. Vad är ett distribuerade system? Beskriv de Nyckel karakteristiska för ett distribuerade System. (2p)

    Detta innebär att det är ett flertal parallellt arbetande delsystem som genom utbyte av kommunikation emellan varandra via ett datornätverk tillsammans utför en gemensam uppgift. I vissa fall med distribuerade system är det så att man vill underlätta en viss typ av bearbetning genom att fördela denna till olika enheter. Detta sker oftast i en klient-server relation där centrala servrarna(server-system) distribuerar ut delproblem till andra sammanslutna datorer(klient-system). Beräkningen av dessa problem hos klient-datorerna utförs med oerhört låg prioritet, och kapaciteten till beräkning används bara om datorerna inte används för något annat. Efter att respektive klient-dator har löst sin del av problemet så skickar den tillbaka resultatet till central-servern och mottar eventuellt ett nytt problem att lösa. Eftersom så många datorer på detta sätt kan arbeta tillsammans och snabbt samt effektiv lösa stora problem så blir det hela jämförbart med prestandan hos en superdator.

11. I början av 1970-talet hade ett företag en eller maximalt två datorer. Dessa var isolerade öar och det fanns ingen kommunikation mellan dem. Dessa datorer kallades ”Main frames”[1]. Fenomenet med isolerade öar återkom även senare i och med att företagen fått fler och mindre datorer sammankopplade i nätverk [2]. Hur kunde problemet uppstå igen? Ge några olika orsaker till de isolerade öarna.



1.

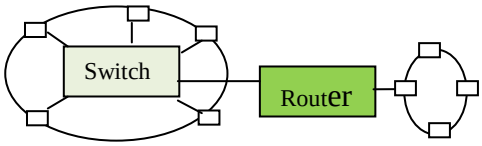


2.

a. Vilken revolutionerande teknik kunde så småningom lösa problemet med de isolerade öarna i bild 2? (1p)

Internet protokoll

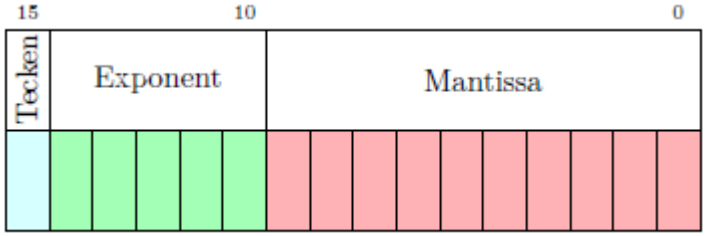
b. Beskriv denna tekniks delar. (2p)



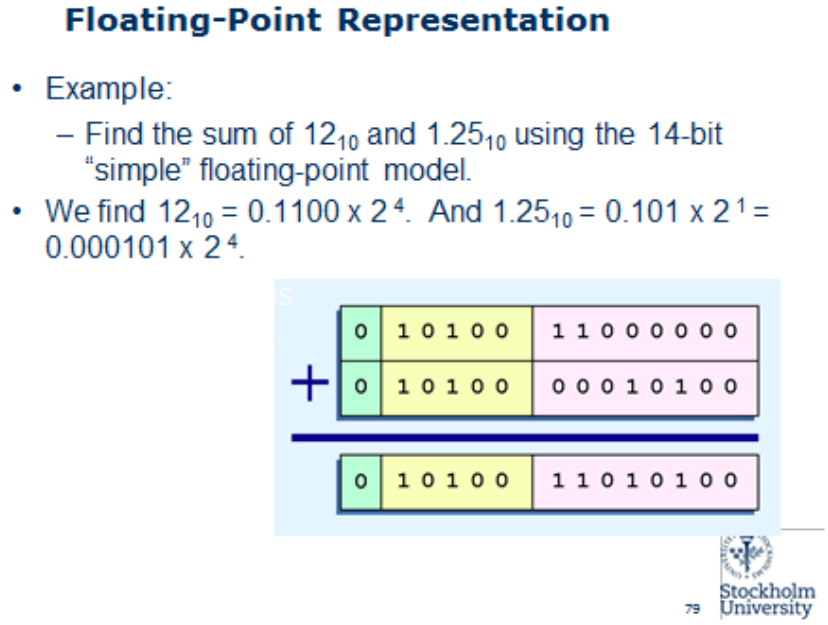
Med hjälp av IP och IP adressering kan information(i form av segmenter som kallas packet) från en enhet i ett nätverk skickas vidare via switchen som är kopplar till lokala nätet. Packeten sedan skickas vidare från switchen till routern. Routern är gateway mellan lokala nätet och Internet. Routern kommer att lista mottagares nätid från ip adressen och efteråt skickar vidare packeten till mottagare.

Del B

1. Flyttal är ett sätt att representera stora, små och rationella tal. I nedanstående uppgift används flyttal för 16-bitars flyttal.



Beräkna operation A+B . .
och A= 12_{10} och B= 1.25_{10} . (3p)



Normaliser : 12_{10} = 0,1100 x 2^4
$1{,}25_{10}$ = 0,101 x 2^1 = 0,000101 x 2^4

2. IP-nät indelas i fyra stycken adressklasser. Beskriv de fyra IP-adressklasser (bild 110 För. 13-14)

Classify and Define IPv4 Addresses

- Identify the historic method for assigning addresses and the issues associated with the method

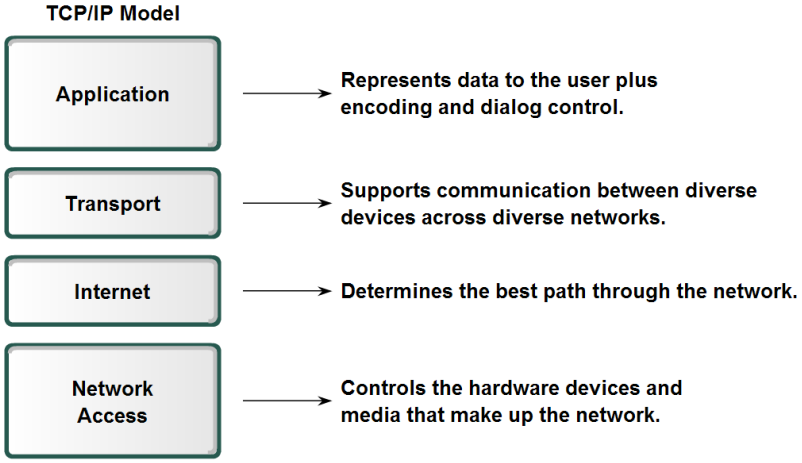
IP Address Classes

Address Class	1st octet range (decimal)	1st octet bits (green bits do not change)	Network(N) and Host(H) parts of address	Default subnet mask (decimal and binary)	Number of possible networks and hosts per network
A	1-127**	00000000-01111111	N.H.H.H	255.0.0.0	128 nets (2^7) 16,777,214 hosts per net (2^24-2)
B	128-191	10000000-10111111	N.N.H.H	255.255.0.0	16,384 nets (2^14) 65,534 hosts per net (2^16-2)
C	192-223	11000000-11011111	N.N.N.H	255.255.255.0	2,097,150 nets (2^21) 254 hosts per net (2^8-2)
D	224-239	11100000-11101111	NA (multicast)		
E	240-255	11110000-11111111	NA (experimental)		

** All zeros (0) and all ones (1) are invalid hosts addresses.

Stockholm University

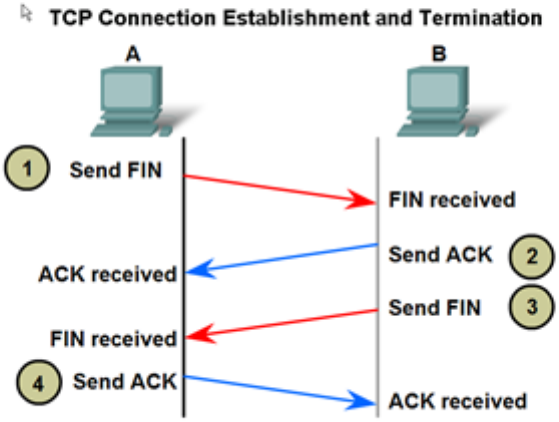
a. TCP/IP modellen inspirerade OSI-modellen till att delas upp i skikt, i själva verket har TCP/IP sin egen skiktning namnge vilka.



b. Namnge och beskriv TCP funktioner. (bilder 66-67 & 69-71, För.13-14)

Application and Operation of TCP Mechanisms

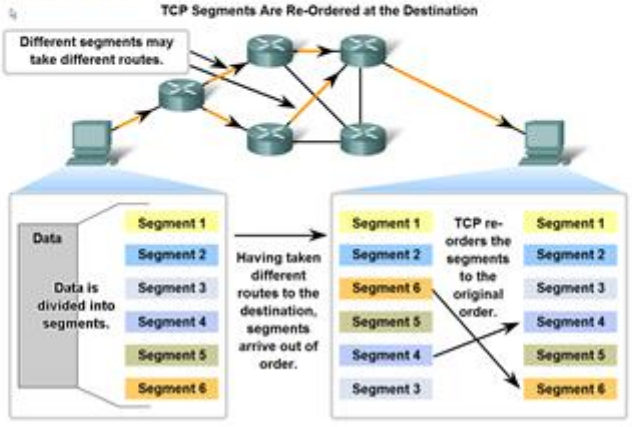
- Trace the steps in the handshake in the establishment of TCP sessions



Stockholm University

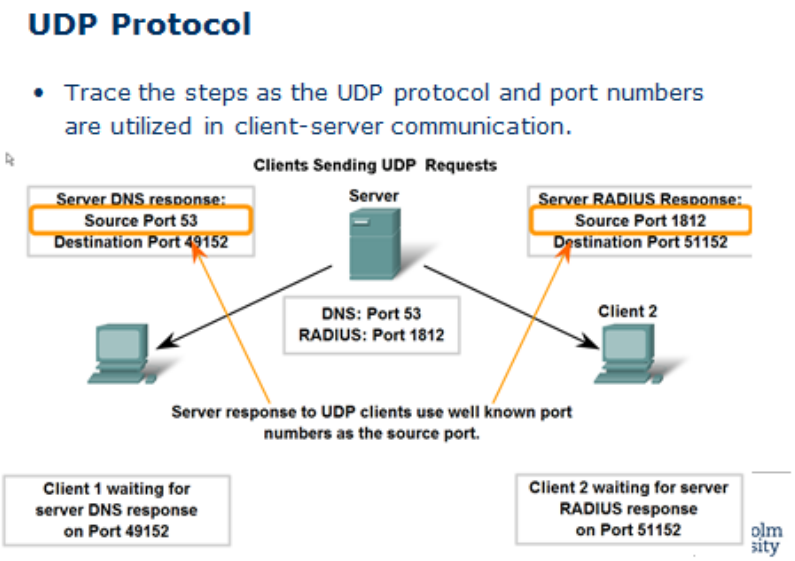
Managing TCP Sessions

- Describe how TCP sequence numbers are used to reconstruct the data stream with segments placed in the correct order



Stockholm University

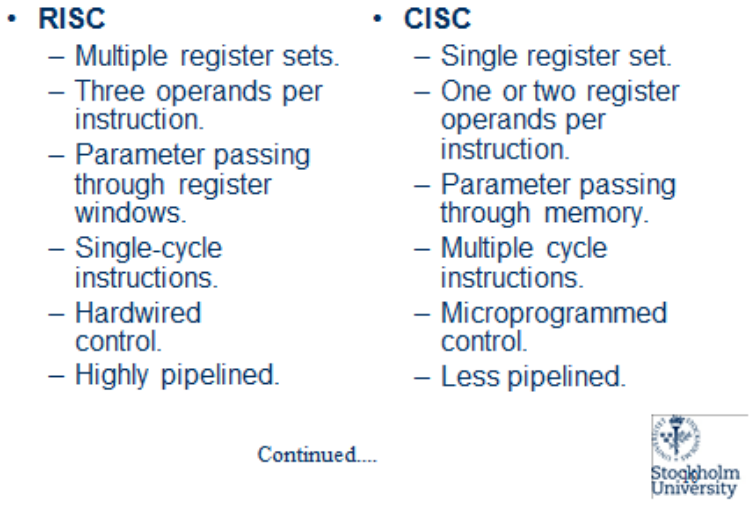
c. Beskriv med ett exempel hur DNS-uppslagningarna (DNS-queries) utförs. (4p)



3. Namnge och Beskriv tekniker för hantering av branches i pipelines. (2p)

Två olika tekniker för att hantera branchar vid pipelining är ”branch prediction” och ”delayed branch”.
Delayed branch, man läser in branch instruktionen men eftersom den inte tar effekt på en gång pga. pipelinen, så kan man lägga in en annan instruktion efter branch instruktionen. Den instruktionen kommer då att exekveras innan branch instruktionen tar effekt.
Branch prediction, man försöker förutsäga vilken instruktion som kommer att behövas härnäst, om en branch kommer att ske eller inte. Om man lyckas förutsäga branchen korrekt så har man inte slösat bort några processorcykler.

4. (a) Beskriv skillnader mellan RISC och CISC processorer. (2p)



- **RISC**
  - Simple instructions, few in number.
  - Fixed length instructions.
  - Complexity in compiler.
  - Only LOAD/STORE instructions access memory.
  - Few addressing modes.
- **CISC**
  - Many complex instructions.
  - Variable length instructions.
  - Complexity in microcode.
  - Many instructions can access memory.
  - Many addressing modes.

Stockholm University

(b) Vi har följande avsnitter ur två assemblerprogram.

*Avsnitt ur assembler program **1***

```
mov ax, 10
mov bx, 5
mul bx, ax
```

*Avsnitt ur assembler program **2***

```
        mov ax, 0
        mov bx, 10
        mov cx, 5
Begin   add ax, bx
        loop Begin
```

Du ska ange antal ”clock cycles” för respektive avsnitter ur assemblerprogram 1 och 2. (2p)

- Consider the program fragments:

CISC

```
mov ax, 10
mov bx, 5
mul bx, ax
```

RISC

```
        mov ax, 0
        mov bx, 10
        mov cx, 5
Begin   add ax, bx
        loop Begin
```

- The total clock cycles for the CISC version might be:
  `(2 movs x 1 cycle) + (1 mul x 30 cycles) = 32 cycles`
- While the clock cycles for the RISC version is:
  `(3 movs x 1 cycle) + (5 adds x 1 cycle) + (5 loops x 1 cycle) = 13 cycles`
- With RISC clock cycle being shorter, RISC gives us much faster execution speeds.

Stockholm University

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, bestäm 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. (4p)

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

512/16=32 rader i cachen. För att adressera 0-31 behöver man 5 st bitar till radadressen. Man behöver 4 bitar för att kunna adressera 16 bytes. Av 32 bitar har man då kvar 32-5-4=23 bitar som hör till tagen.

Instruktion 2: ldw r10, 0(r8)

0xBEDA12C4 + 0_{10} = 0xBEDA12C4: adress i primärminnet.

0x2C4 = 001|01100|0100_2, vilket ger rad 12_{10} i cacheminnet, det kommer att bli en cachemiss då cachen från början är tom.

0xBEDA12C4 + $0F_{16}$ = 0xBEDA12D3: sista adressen i primärminnet som läggs in i cacheminnet.

Man lägger in det som finns på adress 0xBEDA12C4 - 0xBEDA12D3 i primärminnet på rad 12_{10} i cacheminnet.

Instruktion 3: ldw r11, 16(r8)

0xBEDA12C4 + 10_{16} = 0xBEDA12D4: adressen i primärminnet.

0x2D4 = 001|01101|0100_2, vilket ger rad 13_{10} i cacheminnet. Det kommer också att bli en cachemiss då rad 13 i cacheminnet är tomt.

0xBEDA12D4 + $0F_{16}$ = 0xBEDA12E3: sista adressen i primärminnet som läggs in i cacheminnet.

Man lägger in det som finns på adress 0xBEDA12D4 - 0xBEDA12E3 i primärminnet på rad 13_{10} i cacheminnet.

Instruktion 4: stw r10, 32(r8)

0xBEDA12C4 + 20_{16} = 0xBEDA12E4: adressen i primärminnet.

0x2E4 = 001|01110|0100_2, vilket ger rad 14_{10} i cacheminnet. Det kommer att bli en cachemiss då cachen är tom på rad 14.

0xBEDA12E4 + $0F_{16}$ = 0xBEDA12F3: sista adressen i primärminnet som läggs in i cacheminnet.

Man lägger in det som finns på adress 0xBEDA12E4 - 0xBEDA12F3 i primärminnet på rad 14_{10} i cacheminnet.