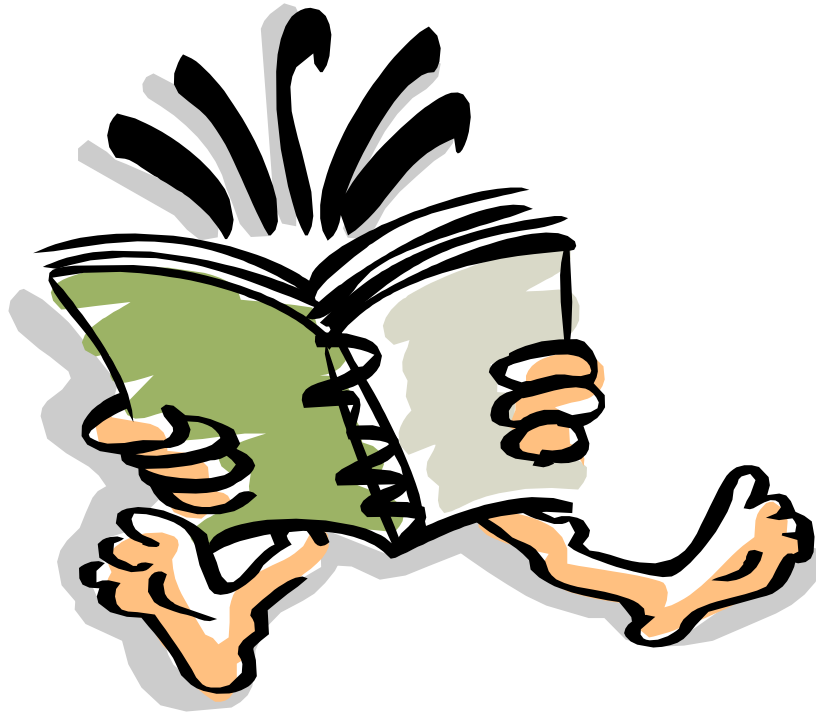




Tentamen

Mikrodatorteknik, 4p

2004-01-17

Skrivtid:

5 timmar

Ansvariglärare:

Benny Thörnberg

Hjälpmedel:

The 68000's Instruction Set

Miniräknare

OBS!

Skriv tydligt!

Skriv namn och kod på varje blad.

Skriv bara på en sida av varje blad.

Redovisa alla steg i beräkningar och konstruktioner.

Rita snygga figurer!

Lycka till,
/Benny

1. Vilka register finns det i processorn och vad används dessa till? 6p
2. Nämn och beskriv kortfattat två sätt att adressera data i ett cacheminne. 8p
3. Vilken uppgift har UDS* och LDS* ? 5p
4. Vad står DMA för samt beskriv hur DMA fungerar? 6p
5. Antag att register D1 innehåller \$00000099. Ange D1, C-flaggans samt Z- flaggans värde efter att instruktionerna nedan har exekverats. Observera att a till f är enskilda instruktioner och inte ett program. 6p

- a) EORI.B #\$66,D1
- b) ADDI.B #\$66,D1
- c) ADDI.B #\$67,D1
- d) SUBI.B #\$66,D1
- e) ASR.B #2,D1
- f) ROL.B #4,D1

6. Funktionsanrop i assembler. En funktion i C är deklarerade som följande:

```
void printChar(int a, int b);
```

Parametrarna a och b ska skickas till funktionen via stacken.

- a) Rita stackens innehåll efter hoppet till subrutinen printChar, precis innan första instruktionen i subrutinen exekverats. Stackpekaren pekar på adress \$7000 innan anropet. Anropet till funktionen i assembler görs enligt följande:

Funktionsanrop i C

```
PrintChar(a,b);
```

Funktionsanrop i assembler

```
MOVE.W a, -(A7)
MOVE.W b, -(A7)
BSR printChar
ADDA.L #4,A7
```

Stacken

6FF8	
6FF9	
6FFA	
6FFB	
6FFC	
6FFD	
6FFE	
6FFF	
SP → 7000	00

Rita inte på detta paper!!!

6p

- b) Vilken funktion har instruktionen ADDA.L #4,A7 ?

3p

7. Adresseringsmetoder: Vad kommer D2 och A3 att innehålla efter att instruktionen har exekverats. Innehållet i minnet är specificerat av tabellen nedan. Instruktionerna är oberoende av varandra (inget program). D2 har värdet \$00000000 och A3 \$00004004 innan varje instruktion exekveras.

- MOVE.L \$4004, D2
- MOVE.W 4(A3), D2
- MOVE.L -(A3), D2
- MOVE.B -(A3), D2
- MOVE.W -2(A3), D2
- MOVE.W #\$4004, D2
- MOVE.B (A3)+, D2
- MOVE.L (A3)+, D2

Adress (Hexadecimal)	Innehåll (Hexadecimal)
4000	77
4001	21
4002	C3
4003	FD
4004	00
4005	14
4006	99
4007	74
4008	A6
4009	AA

8p

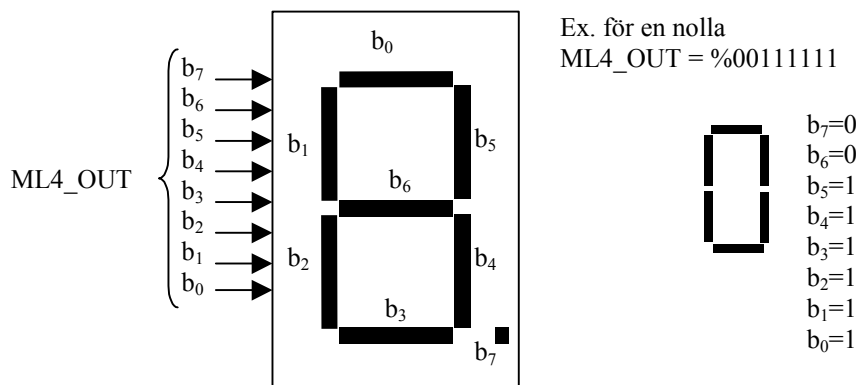
8. Assemblerprogrammering: Skriv en subrutin i 68000-assembler som skriver ut ett tal (0-15) till en display. På displayen presenteras talet som en hexadecimal siffra (0-F). Displayen är av samma typ som på labbkorten, dvs. en 7-segment display.

Displayen är ansluten till ML4_OUT (porten är definierad). Varje bit i ML4_OUT är ansluten till en lysdiod i 7-segment displayen. Kopplingen mellan bitarna i ML4_OUT och lysdioderna är definierad i figuren nedan (T ex. b_0 är kopplad till den översta vågräta dioden). b_7 är kopplad till punkten på 7-segmentdisplayen, så den sätter ni alltid till noll.

In: ML4_OUT = Talet som ska skrivas ut till 7-segmentdisplayen (tal mellan 0-16)

Ut: Inget ska returneras från subrutinen.

Exempel: Nedre byten på D0 = %00110000 ska ge en etta på displayen och ML4_OUT = %00111111 ger en nolla.

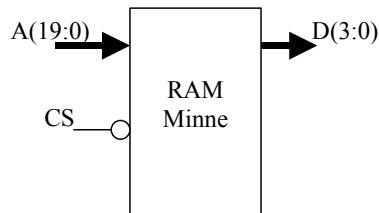


12p

9. Sätt samman en minnesbank som är konstruerad av flera mindre minnesobjekt (se komponenten i figuren). Signaler till minnesbanken ska vara A(adress), D(data) samt CS*. Minneskapslarna som används för att konstruera minnesbanken har samma signaler som den färdiga minnesbanken har. Där CS* signalen är aktivt låg. Du ska indikera vilka adress/data-signaler som ska kopplas till de olika minneskapslarna.

Minnesbanken ska vara 8Mbytes stort organiserat med en 16 bitars databuss.
Minnesbanken sätts samman med hjälp av minneskomponenter som har 4-bitars databuss och 20 bitars adressbuss.

10p



10. A) Konstruera en fullständig adressavkodare (fullständig adressavkodning) som realiserar adresskartan definierad i uppgiften. Implementera med hjälp av logiska grindar och avkodare (specificerad i Appendix)

<u>Komponent</u>	<u>Adressområde</u>
ROM1	00 0000 - 00FFFF
ROM2	010000 - 01FFFF
RAM	04 0000 - 07 FFFF
KOMP_1	80 0000 - 80 00FF
KOMP_2	81 0000 - 81 00FF
KOMP_3	82 0000 - 82 000F
KOMP_4	83 0000 - 83 000F

12p

B) Hur många bytes är definierat för varje komponent i adresskartan?

6p

11. Skriv en subrutin som konverterar ett 32-bitars binärtal till en textsträng med nollor och ettor.

In: D0 = Talet som ska omvandlas. A0 pekar på startadressen för strängen, dvs på det mest signifikanta bit-tecknet.

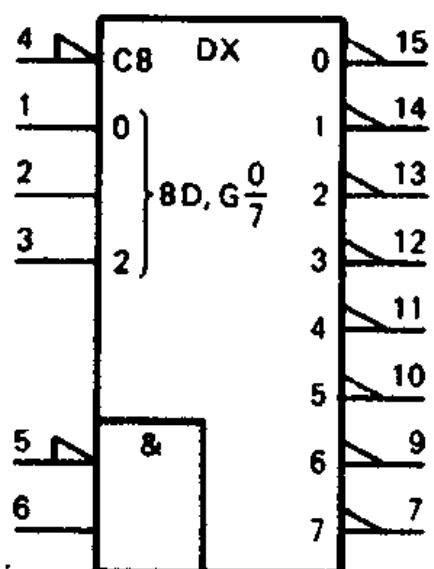
Ut: En teckensträng med ettor och nollor som avslutats med en binär nolla i den sista positionen (A0+32) för att indikera strängslut.

Exempel: Talet \$4323 Ska ge textsträngen "00000000000000000000100001100100011",0.

Ledning: Att omvandla från en siffra till ASCII tecknet för siffran görs genom att addera ASCII-koden för en nolla ('0' = \$30) och för en etta ('1' = \$31)

12p

Appendix



$V_{CC} = \text{Pin } 16$

GND = Pin 8