Halmstad University
School of Information Science, Computer and Electrical Engineering
Tomas Nordström, CC-lab

TENTAMEN

Datorteknik (DO2005)

D1/E1/Mek1/Ö1

Datum: 2012-05-23
Tid och plats: 9:00 – 13:00 i sal T158.

Hjälpmedel: Kompendium "Tentahjälpmedel Datorteknik", Version 2012-05-21, A4-format, 72 sidor.

Kompendiet ska lämnas tillbaka efter tentan!

Maxpoäng: 60p
Betyg 3 24p
Betyg 4 36p
Betyg 5 48p

Frågor under tentamen: Tomas Nordström: 035/167334 eller 070/2888632.

Notera att för alla uppgifter gäller att assemblerkod ska skrivas för den mikroprocessor som har används vid laborationerna, en ATMEL ***SAM3U***, vilken i sin tur innehåller en ***ARM Cortex M3*** processor. Utdrag ur datablad och annat referensmaterial som behövs för att lösa tentauppgifterna finns i kompendiet "Tentahjälpmedel Datorteknik".

Notera även att ***all kod som skrivs måste vara väl kommenterad*** för att ge full poäng.

Datorteknik (DO2005)
Tentamen 2012-05-21

Uppgift 1. (5p)

(0.5p/rätt, felaktigt alternativ = –0.5p. Dock lägst noll poäng. Max 5p)

Kombinera en term i den vänstra kolumnen med tillhörande fras(er) i den högra kolumnen.

l. Instruktionsregister	a. signal från givare omvandlas till lämplig form för datorbearbetning
2. Refresh	b. här finns styrkoder för instruktioner
	c. används vid adressering
3. A/D-omvandling	d. DRAM
	e. krets för seriell in-utmatning
4. ALU	f. alfanumerisk kod
	g. här hamnar instruktioner för avkodning
5. Startbit	h. pekar på nästa instruktion
	i. här sker beräkningar
6. RISC	j. kan delas upp i flera tillstånd
	k. en instruktionsuppsättning
7. ASCII	l. programmerad in-utmatning
	m. assemblator
8. Baud	n. PROM
	o. asynkron seriell överföring
9. Cache	p. krets för parallell in-utmatning
	q. snabbt minne nära processorn
10. Thumb	r. handskakning
	s. ”load/store” arkitektur
	t. statusregister
	u. enhet för signaleringshastighet
	v. assemblerinstruktion
	x. digital-analog omvandling

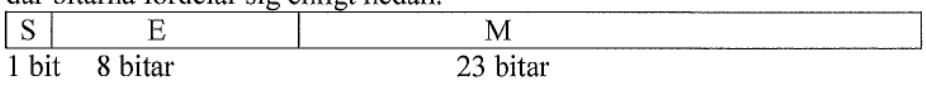
Uppgift 2. (4p)

Du har fått en datafil med innehållet 1010 6A93 40F0 0000
tolkat i hexadecimal kod. Du tar reda på att de 16 första bitarna är ett positivt heltal. Därefter följer 2 st 8-bitars 2-komplementtal och slutligen ett 32-bitars flyttal

a) Översätt det positiva heltalet till ett decimaltal.
b) Översätt de två 2-komplementtalen till decimaltal.
c) Översätt flyttalet till decimaltal.
Ledning: Flyttalet är kodat i IEEE-standarden får enkel precision. För denna gäller

$$(-1)^S \cdot 1.M \cdot 2^{E-127}$$

där bitarna fördelar sig enligt nedan.



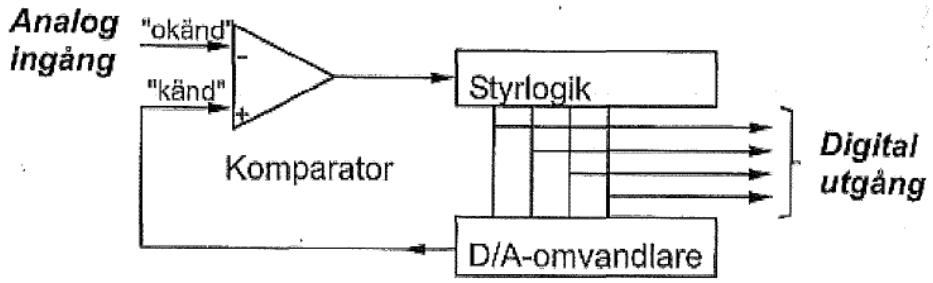
d) Om strängen ”Hej Hopp!” lagras som ASCII, vilka värden (i hex) motsvarar detta?

Uppgift 3. (4p)

I denna uppgift förutsätts 8 bitars ordlängd och 2-komplementkodning. Ange svaren på följande additioner, dels i binär form och dels som decimaltal. Sätt också ut värdet får Carry (C), Overflow (V) och Negate (N) och redovisa som tabellen nedan.

Hex	Binärt	Decimalt	Carry	Overflow	Negate
a) 7E+ 2A					
b) BF+Cl					

Uppgift 4. (4p)



Ovanstående figur visar en 4-bitars A/D-omvandlare av typen successiv approximationsomvandlare.

a) Antag att den "okända" analoga ingången har ett värde $8/16 < U_{in} < 9/16$ av fullt utslag. Vad blir det digitala värdet $<x_3\ x_2\ x_1\ x_0>$efter omvandling? Motivering med ett tidsdiagram som visar omvandlingen, med förklarande text, krävs.
b) Hur många klockpulser kräver denna omvandling?
c) Hur många bitar skall en AD-omvandlare ha for att upplösningen ska vara l% av största värdet?

Uppgift 5. (2p)

Vid dokumentation av en subrutin i assembler vilka delar/egenskaper bör åtminstone vara beskrivna för att dokumentationen ska vara acceptabel.

Uppgift 6. (5p)

Skriv ett program som omvandlar tal som ligger på adress 0x2000 till dess absolutvärden.
Talet skall sparas på adress 0x3000 och framåt. Talen avslutas med talet 0.

Ex.

Adress	Före omvandling	Adress	Efter omvandling
0x2000	1234	0x3000	1234
0x2004	3	0x3004	3
0x2008	-345	0x3008	345
0x200C	24	0x300C	24
0x2010	-99	0x3010	99
0x2014	0	0x3014	0

Uppgift 7. (3p)

I en vanlig ARM kod kan vi finna följande instruktion:

```
STMFD R13!,{R0-R4, LR}
```

a) Förklara vad instruktionen gör!
b) Var i koden är det mest sannolikt att denna typ av instruktion dyker upp?
c) Anta att R13 = 0x20004000 innan instruktionen. Vad är innehållet i R13 efter att ha exekverat instruktionen?

Uppgift 8. (3p)

Anta man vill läsa in talet 0x12345678 till R1. Detta går att göra på tre sätt. Skriv kod för att göra detta

a) Genom att nyttja assemblerdirektiv EQU samt instruktionen LDR
b) Genom att nyttja assemblerdirektiv DC32 samt instruktionen LDR
c) Genom att bara nyttja instruktionen LDR

Uppgift 9. (6p)

Man brukar ofta prata om 5 steg i instruktionscykeln för en RISC processor.

a) Namnge vilka steg som finns och tala om vad som sker i respektive steg med högst en mening var.
b) Om följande instruktioner finns i minnet ifrån adress 0x20000000

```
TEST  LDR R9,[R6,#5]
      SUBS R1, R1, R9
      BNE TEST
```

Förklara vad som händer i alla stegen av instruktionscykeln för de tre instruktionerna. Ange speciellt vad som finns på adressbussen under de olika stegen. Anta att R6 har värdet 0x20001234 innan dessa instruktioner och att en von Neumann arkitektur används (med gemensamt instruktions och data minne) utan pipelining.

Uppgift 10. (4p)

a) Förklara, kortfattat, vad som menas med löpandebandprincipen (pipelining) inom datorteknik.
b) Med ”superscalar”-teknik så utförs flera instruktioner samtidigt. Vad begränsar möjligheterna att använda ”superscalar”-tekniken?

Uppgift 11. (6p)

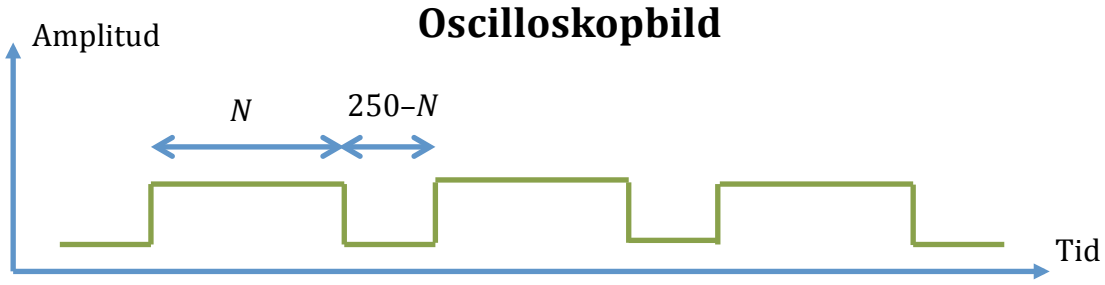
a) Skriv den kod som behövs för att konfigurera SYSTICK i den processors som labsystemet använder (ARM Cortex M3 / ATMEL SAM3U) så att avbrott sker 25000 gånger per sekund, anta att processorns klockfrekvens är 5MHz.
b) Förklara allt vad som händer i processorn då ett avbrott ifrån SYSTICK kommer.
c) Skriv även den kod som behövs för att lägga till en `SYSTICK_Handler` i den aktuella avbrottsvektorn. Notera att avbrottsvektorn har flyttats till SRAM av en bootloader/debugger så du måste först ta reda på var vektorn ligger i minnet innan den modifieras!

Uppgift 12. (6p)

a) Skriv den kod som behövs för att konfigurera PIO PortB som utgångar utan pull-up motstånd på bit1 och bit0.
b) Skriv den kod som behövs för att konfigurera PIO PortA som ingångar med pull-up på bit19 och bit18. Konfigurera dessa ingångar också så att de ger avbrott då en knapp kopplad till dessa ingångar trycks in (och drar signalen låg).
c) Skapa en avbrottsrutin `PIOA_InteruptHandler` som tar hand om knappnertryckningsavbrottet. Rutinen ska i sin tur anropa en hanteringsrutin `ButtonHandle` med R0 som inparameter (R0=2 om knappen till bit19 har tryckts ner; R0=1 om knappen till bit18 har tryckts ner; R0=3 om båda knappar tryckts ner).

Uppgift 13. (8p)

a) Skriv ett program som genererar en PWM (pulsbreddsmodulerad) utsignal på PortB (bit0). Sätt bit0 hög *N* gånger och låg 250–*N* gånger, jämför med figuren nedan. Talet *N* (0<=*N*<=250) som styr PWM signalen antas vara lagrat i minnet på adress `N_VAL`.
Periodtiden för PWD signalen måste vara sådan att om detta styr en lampa så får den inte fladdra (dvs > 50 Hz). Därför är SYSTICK initierats så att den genererar ett avbrott (anropar `SYSTICK_Handler`) 25000 gånger per sekund. Skriv den kod som behövs i `SYSTICK_Handler` så att datorn genererar den önskade PWM signalen.



b) Vi vill självfallet även kunna ändra talet värdet på *N* som styr PWM signalen (och lagras på adress `N_VAL`) och valt att göra detta genom knapptryckningar. Skriv den kod som gör att *N* ökas om upp-knapp (bit18) eller minskas om nerknapp (bit19) trycks ner. Om båda trycks ner ska *N* återgå till sitt startvärde = 128. Du kan anta att koden i uppgift 10 finns och att du bara behöver implementera `ButtonHandle`. Notera att `SYSTICK_Handler` måste kunna anta att *N* håller sig inom sina gränser (0<=*N*<=250).
c) Då det blir väldigt många knapptryckningar för att komma till ändlägena skulle man vilja ha en repeat-funktion. Modifiera rutinerna ovan så att en sådan funktion skapas. Om man håller en knapp nedtryckt i minst en sekund *N*-värdet därefter börja uppdateras 2 gånger per sekund.