

EDA432 Digital och datorteknik IT

INN790 Digital och datorteknik GU

Tentamen

Måndag 23 oktober 2006, kl. 08.30 – 12.30 i V-salar

Examinatorer

Rolf Snedsböl, tel. 772 1665

Kontaktpersoner under tentamen

Som ovan

Tillåtna hjälpmedel

Häftet

Instruktionslista för FLEX

och

Instruktionslista för CPU12

I dessa får rättelser och understrykningar vara införda, inget annat.

Tabellverk och miniräknare får ej användas!

Allmänt

Siffror inom parentes anger full poäng på uppgiften. **Full poäng kan fås om:**

- redovisningen av svar och lösningar är läslig och tydlig. **OBS!** Ett lösningsblad får endast innehålla redovisningsdelar som hör ihop med en uppgift.

- din lösning ej är onödigt komplicerad.
- du motiverat dina val och ställningstaganden
- redovisningen av en hårdvarukonstruktion innehåller funktionsbeskrivning, lösning och realisering.
- redovisningen av en mjukvarukonstruktion i assembler är fullständigt dokumenterad, d v s är redovisad både i strukturform (flödesplan eller pseudopråk) och med kommenterat program i assemblerspråk, om inget annat anges i uppgiften.

Betygsättning

För godkänt slutbetyg på kursen fordras att både tentamen och laborationer är godkända. Tentamen ger slutbetyget (max 50p):

$20p \leq \text{betyg 3} < 30p \leq \text{betyg 4} < 40p \leq \text{betyg 5}$

Lösningar

anslås på kursens www hemsida (EDA432).

Betygslistan

anslås såsom anges på kursens hemsida (EDA432).

Granskning

Tid och plats anges på kursens hemsida (EDA432).



Uppgift 1 *Talomvandling, koder, aritmetik och flaggor.*

I uppgift a-d nedan används 5-bitars tal. $X = 01101$ och $Y = 11001$.

- a) Visa med penna och papper hur räkneoperationen $R = X - Y$ utförs i en dator (i en ALU). (1p)
- b) Ange sedan flaggbitarna N, Z, V, C. (1p)
- c) Tolka bitmönstren R, X och Y som tal *utan* tecken och ange dess decimala motsvarighet. Vilken(vilka) flaggbit(ar) anger om resultatet är korrekt vid tal utan tecken? (1p)
- d) Tolka bitmönstren R, X och Y som tal *med* tecken och ange dess decimala motsvarighet. Vilken(vilka) flaggbit(ar) anger om resultatet är korrekt vid tal med tecken? (1p)
- e) Studera bitmönstren 11011100_2 och 01010100_2 . Kan bitmönstren representera följande:

- 1) NBCD-tal
- 2) ett negativt tal på tvåkomplementsform
- 3) ett tal på tecken beloppsform
- 4) ett Graykodat dataord med udda paritetsbit
- 5) en operationskod för FLEX-processorn
- 6) en operationskod för CPU12-processorn

(Ge ditt svar i tabellform enligt:)

	11011100	01010100
1	Ja/Nej	Ja/Nej
2		
osv...		

(2p)

- f) Om det är möjligt, skriv det decimala talet 32 som ett 6-bitars tvåkomplementstal. Argumentera! (1p)

Uppgift 2 *Digitalteknik - Småfrågor*

- a) Man behöver en 2-ingångs OR-grind men har bara 2-ingångs AND-grindar och 2-ingångs XOR-grindar. Hur kopplar man upp OR-grinden med dessa? (2p)
- b) Ange funktionstabellen och excitationstabellen för en JK-vippa. (2p)
- c) Den booleska funktionen $f(x,y,z) = xy' + xz' + y'z'$ är given. Ange denna på konjunktiv minimal form och konjunktiv normal form. (3p)
- d) Rita grindnätet för en en-bits halvadderare adderare som utför $x_i + y_i$. Utsignalerna är s_i och c_{i+1} . (2p)
- e)

Uppgift 3 *Digitalteknik - Konstruktion*

- a) Konstruera ett kombinatoriskt nät med fyra insignaler (x, y, z, w) och en utsignal (f) enligt följande: Utsignalen $f=1$ för $3 \leq (xyzw)_2 \leq 9$. För alla andra värden på insignalen är $f=0$. Insignalerna bildar ett binärtal $(xyzw)_2 \in [0,15]_{10}$.

Gör en minimal lösning och rita upp det kombinatoriska nätet.

Använd NAND/NAND-logik (disjunktiv form). Du har tillgång till NAND-grindar med valfritt antal ingångar (Inga andra grindar får förekomma i din lösning). (4p)

- b) Konstruera och rita upp en räknare som har räknarsekvensen **000,110,010,100,011**, 000,110,010,100,011 etc. på de tre utsignalerna $Q_2Q_1Q_0$. Du kan bortse från hur räknaren startas. Använd T-vippor. Du har tillgång till vanliga grindar (AND, NAND, OR, NOR) med valfritt antal ingångar, samt INVERTERARE. (6p)

Uppgift 4 Styrenhet och dataväg för FLEX.

- a) I tabellen nedan visas RTN-beskrivningen för EXECUTE-sekvensen för en **instruktion** för FLEX-processorn. NF i tabellens sista rad anger att nästa tillstånd (state) skall vara det första i FETCH-sekvensen. Rita en tabell där du anger State nr (0..4) och Styrsignaler. Endast styrsignaler = 1 skall anges. Du kan utelämna RTN-beskrivningen i din tabell (2p)

State	RTN-beskrivning	Styrsignaler (=1)
0	PC → MA, PC+1 → PC, S-1 → S	
1	M → T	
2	S → MA	
3	PC → M, T → R	
4	R → PC, NF	

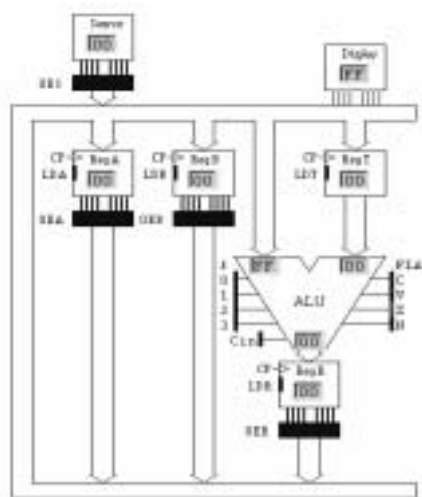
- b) Förklara i ord vad instruktionen ovan utför i varje klockcykel. Ange sedan instruktionen med assemblerspråk för FLEX-processorn (Ex: **LDA Adr**). (2p)

- c) Ange de styrsignaler som krävs för att utföra operationerna enligt nedanstående RTN-beskrivning:

RTN-beskrivning: $2A - 4B \rightarrow A$

Använd den enkla datavägen till höger och ge ditt svar i tabellform liknande den ovan.

Förutsätt att register A och B innehåller de data som skall beräknas. Register B får inte ändras. Använd så få tillstånd som möjligt.



(4p)

Uppgift 5 Småfrågor rörande FLEX-processorn

- a) Ett 24-bitars tal P är lagrad på adresserna A5, A6 och A7 med mest signifikanta byten på adress A5. Skriv en instruktionssekvens som utför $P+1 \rightarrow P$ (inkrementerar P). Ange även maskinkoden för denna instruktionssekvens. Instruktionssekvensen skall ha startadress 40_{16} i minnet. (5p)

- b) Studera nedanstående instruktionssekvens.

```
LDA    #Q
CMPA  #135
BLO   Hopp
NOP
```

För vilka värden på Q $[0,255_{10}]$ kommer hoppet att utföras? (Resonera) (2p)

- c) Hopp-instruktionen i uppgift 5b) byts ut mot BLT. För vilka värden på Q $[0,255_{10}]$ kommer hoppet att utföras. (Resonera) (2p)

Uppgift 6 *Assemblerprogrammering av CPU12-processorn*

Vid simulatorpassen och i labbet använde du stömbrytarna (ML4 INPUT) och sifferindikatorn (ML4 OUTPUT).

Du skall skriva ett program som utför en fördröjning – och visar fördröjningen på en sifferindikator. Inställd värde på strömbrytarna anger hur lång fördröjningen är. Du har tillgång till en subrutin DELAY1s som utför en fördröjning på 1 s.

Du skall skriva ett programsekvens som

- 1) läser strömbrytarna (Inport, 8 bitar, som anger antal sekunders fördröjning)
- 2) maskar fram bit $[b_3, b_0]$ av inporten (för att maximera fördröjningen till 15s)
- 3) och visar detta värde $[0, F_{16}]$ på sifferindikatorn och
- 4) därefter väntar inställd värde (Hoppa till DELAY1s ett visst antal gånger) och
- 5) slutligen släcker sifferindikatorn

Du har tillgång till en tabell med segmentkoder och följande defenitioner:

Inport	EQU	qqqq	Adress för inport (Strömbrytare)
Utport	EQU	zzzz	Adress för utport (Sifferindikator)
SegCode	FCB	xx,yy,zz,etc	Tabell med segmentkoder för $[0, F]$
DELAY1s	EQU	www	Startadress för Delay-rutinen som utför 1s fördröjning

(7p)

Bilaga 1 - Assemblerspråket för mikroprocessorn CPU12.

Assemblerspråket använder sig av de mnemoniska beteckningar som processorkonstruktören MOTOROLA specificerat för maskininstruktioner och instruktioner till assemblern, så som pseudoinstruktioner eller assemblerdirektiv. Pseudoinstruktionerna framgår av följande tabell:

Direktiv	Förklaring
ORG N	Placerar den efterföljande koden med början på adress N (ORG för ORiGin = ursprung)
L RMB N	Avsätter N bytes i följd i minnet (utan att ge dem värden), så att programmet kan använda dem. Följden placeras med början på adress L. (RMB för Reserve Memory Bytes)
L EQU N	Ger label L konstantvärdet N (EQU för EQUates = beräknas till)
L FCB N1, N2	Avsätter i följd i minnet en byte för varje argument. Respektive byte ges konstantvärdet N1, N2 etc. Följden placeras med början på adress L. (FCB för Form Constant Byte)
L FDB N1, N2	Avsätter i följd i minnet ett bytepar (två bytes) för varje argument. Respektive bytepar ges konstantvärdet N1, N2 etc. Följden placeras med början på adress L. (FDB för Form Double Byte)
L FCS "ABC"	Avsätter en följd av bytes i minnet, en för varje tecken i teckensträngen "ABC". Respektive byte ges ASCII-värdet för A, B, C etc. Följden placeras med början på adress L. (FCS för Form Caracter String)

Bilaga 2 - ASCII-koden.

0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1	$b_6b_5b_4$ $b_3b_2b_1b_0$
NUL	DLE	SP	0	@	P	`	p	0 0 0 0
SOH	DC1	!	1	A	Q	a	q	0 0 0 1
STX	DC2	"	2	B	R	b	r	0 0 1 0
ETX	DC3	#	3	C	S	c	s	0 0 1 1
EOT	DC4	\$	4	D	T	d	t	0 1 0 0
ENQ	NAK	%	5	E	U	e	u	0 1 0 1
ACK	SYN	&	6	F	V	f	v	0 1 1 0
BEL	ETB	'	7	G	W	g	w	0 1 1 1
BS	CAN	(8	H	X	h	x	1 0 0 0
HT	EM)	9	I	Y	i	y	1 0 0 1
LF	SUB	*	:	J	Z	j	z	1 0 1 0
VT	ESC	+	;	K	[Ä	k	{ä	1 0 1 1
FF	FS	,	<	L	\Ö	l	ö	1 1 0 0
CR	GS	-	=	M]Å	m	}å	1 1 0 1
S0	RS	.	>	N	^	n	~	1 1 1 0
S1	US	/	?	O	_	o	RUBOUT (DEL)	1 1 1 1