CHALMERS

Institutionen för data- och informationsteknik
Avdelningen för datorteknik

TENTAMEN

KURSNAMN	**Digital- och datorteknik**
PROGRAM:	**Mekatronikingenjör (samt data- och elektroingenjör)** **Åk 1/ lp 2 (1)**
KURSBETECKNING	**LEU431**
EXAMINATOR	**Lars-Eric Arebrink**
TID FÖR TENTAMEN	**2011-04-26 kl 8.30 – 12.30**
HJÄLPMEDEL	**Av institutionen utgiven** **”Instruktionslista för FLEX-processorn” (INS1)** **Tabellverk eller miniräknare får ej användas.**
ANSV LÄRARE: **Besöker tentamen**	**Lars-Eric Arebrink, tel. 772 5718** **vid flera tillfällen**
ANSLAG AV RESULTAT	**Resultatlistor anslås senast 2011-05-10 på kursens hemsida. Granskning av rättning på institutionen 2011-05-10 och 2011-05-11 kl 12.30-13.00.**
ÖVRIG INFORM.	**Tentamen omfattar totalt 60 poäng.** **Onödigt komplicerade lösningar kan ge poängavdrag.** **Svar på uppgifter skall motiveras.**
BETYGSGRÄNSER.	**Betyg 3: 24 poäng** **Betyg 4: 36 poäng** **Betyg 5: 48 poäng**
SLUTBETYG	**För slutbetyg 3, 4 eller 5 på kursen fordras betyg 3, 4 eller 5 på tentamen och godkända laborationer.**

1. I uppgift a-d nedan används 10-bitars tal. X = 10 1000 0001 och Y = 01 1100 0011.

a) Visa med penna och papper hur räkneoperationen R = X + Y utförs med en 10-bitars ALU. **(1p)**

b) Vilka värden får flaggbitarna N, Z, V och C vid räkneoperationen? **(1p)**

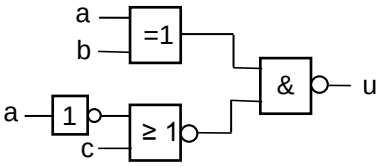
c) Tolka bitmönstren R, X och Y som tal *utan* tecken och ange deras decimala motsvarighet. Är resultatet R korrekt? Hur kan man avgöra detta med hjälp av flaggbitarna? **(1p)**

d) Tolka bitmönstren R, X och Y som tal *med* tecken och ange deras decimala motsvarighet. Är resultatet R korrekt? Hur kan man avgöra detta med hjälp av flaggbitarna? **(1p)**

e) Tolka två n-bitars bitmönster X och Y som tal *med* tecken och ge ett booleskt uttryck f(N,Z,V,C) som har värdet 1 om och endast om $X + Y \geq 0$. Flaggvärdena är de som gäller efter additionen. Det booleska uttrycket skall ge korrekt värde även när overflow inträffar. **(3p)**

f) Antag att N_{max} är det största positiva tal som kan representeras som ett 32-bitars flyttal enligt flyttalsstandarden IEEE 754-1985 (dvs 23 bitar av mantissan). Skriv, på hexadecimal form, det packade flyttal som motsvarar N_{max}. (Du behöver inte räkna ut vilket värde N_{max} har.) **(2p)**

g) Ge ett minimalt boolesk uttryck för u i grindnätet till höger.



(3p)

2. En boolesk funktion f(a,b,c,d) har karnaughdiagrammet till höger.

Realisera funktionen med så få grindar som möjligt. NOR-grindar med valfritt antal ingångar, XOR-grindar och NOT-grindar får användas. Endast insignalerna a, b, c och d finns tillgängliga.

ab \ cd	00	01	11	10
00	0	1	0	1
01	1	0	1	0
11	1	0	1	0
10	1	0	1	0

(6p)

3.

a) Ett synkront sekvensnät skall ha en insignal x och en utsignal u. Utsignalen u skall ges värdet "1" under ett bitintervall för varje insignalsekvens som består av en etta följd av en nolla och fyra ettor hos x.

Exempel: σ_X = ...1001110111101111000101111110111100...

σ_U = ...?0000000001000010000000100000100...

Utsignalen skall som i exemplet ges värdet "1" när den sista biten i en korrekt insignalsekvens anländer på x-ingången och behålla värdet "1" så länge x har kvar sitt värde under detta bitintervall.

Rita en tillståndsgraf för sekvensnätet. Hur många vippor skulle minst krävas vid realiseringen? **(3p+1p)**

b) Realisera en räknare med räknevillkoret x och räknesekvensen $Q = q_2q_1q_0$.

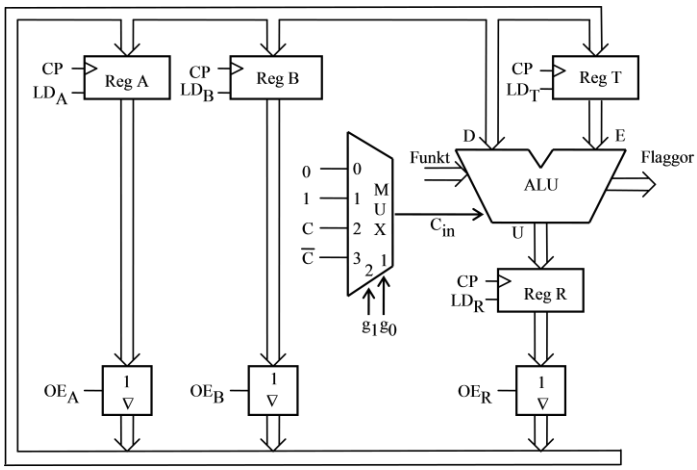
x = 0: $Q^+ = 100$

x = 1: 100, 011, 010, 001, 000, 111, 110, 101, 100,…

T-vippor, NAND-grindar med valfritt antal ingångar, XOR-grindar och NOT-grindar får användas. Det får förutsättas att räknaren startar i tillståndet $q_2q_1q_0 = 100$. **(6p)**

4. Ge RTN-beskrivning och styrsignaler för de tillstånd krävs för att utföra operationen enligt nedanstående RTN-beskrivning:

RTN-beskrivning: $6 \cdot A - 7 \cdot (B + 1) \rightarrow A$
(Aritmetisk multiplikation avses)



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

Förutsätt att register A och B från början innehåller de data som skall behandlas enligt uttrycket ovan och att innehållen i register R och T är okända. Register B får inte ändras. Bortse från risken för overflow. Använd så få tillstånd som möjligt. Samtliga funktioner ALU:n kan utföra framgår av bilaga 1.

(5p)

5. Figur 1 i bilaga 3 visar hur datorn FLEX är uppbyggd. Bilaga 1 visar hur ALU'ns funktion väljs med styrsignalerna f_3 - f_0 och C_{in}.

I tabellen nedan visas styrsignalerna för de olika tillstånden i EXECUTE-sekvensen för en av FLEX-processorns instruktioner.

State	S-term	RTN-beskrivning	Styrsignaler (=1)
Q_5	$Q_5 \cdot I_{xx}$		OE_{PC}, LD_{MA}.
Q_6	$Q_6 \cdot I_{xx}$		MR, LD_T.
Q_7	$Q_7 \cdot I_{xx}$		OE_X, f_3, f_1, LD_R.
Q_8	$Q_8 \cdot I_{xx}$		OE_R, LD_{PC}, NF.

a) Rita en tabell med "State"- och RTN-kolumner enligt ovan och fyll i RTN-beskrivningen. Förklara vilken assemblerinstruktion som beskrivs. **(2p)**

b) Instruktionen "Move byte" nedan skall implementeras för FLEX-processorn med hjälp av styrenheten med fast logik. Instruktionen består av två ord. Den läser dataordet på adressen Adr i minnet och skriver det i minnet på den adress som finns i X-registret. Efter skrivningen i minnet skall innehållet i X-registret ökas med 1. Register A, B, SP eller CC får ej påverkas. Operationskoden $F9_{16}$ skall användas.

MOVB Adr,1,X+ RTN: $M(Adr) \rightarrow M(X)$
$X+1 \rightarrow X$

Gör en tabell liknande tabellen ovan för den efterfrågade EXECUTE-sekvensen. **(4p)**

6. Besvara kortfattat följande frågor rörande FLEX-processorn.

a) Förklara vad som händer under processorns EXECUTE-fas. **(1p)**

b) I instruktionsuppsättningen finns de två villkorliga hoppen BMI och BLT. Om dessa instruktioner utförs direkt efter en aritmetisk operation så kommer de ibland att fungera på samma sätt och ibland på olika sätt beroende på den aritmetiska operationen. Vad är det hos den aritmetiska operationen som orsakar denna skillnad? **(2p)**

c) Vad händer om man låter FLEX-processorn utföra en instruktion där man har glömt att sätta styrsignalen NF = 1 i sista tillståndet i ”EXECUTE” om man använder styrenheten med fast logik? **(2p)**

d) För vilka värden W ($0 \leq W \leq 255$) utförs hoppet till adressen Hopp nedan?

```
LDAB   #$50
CMPB   #W
BGE    Hopp
```
(3p)

e) Översätt instruktionssekvensen till höger till maskinkod på hexadecimal form och visa hur den placeras i minnet. Det skall framgå hur offset för branch-instruktionerna beräknas. **(3p)**

f) Hur lång tid tar instruktionssekvensen i e) att köra om FLEX-processorn klockas med frekvensen 1MHz? **(3p)**

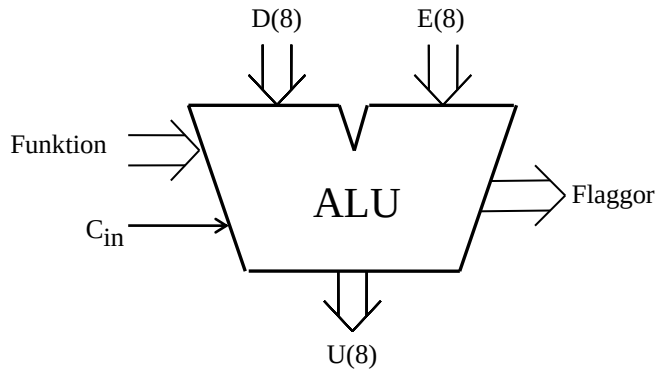
```
        ORG   $40
        LDAA  #-10
LOOP1   LDX   #-16
LOOP2   LEAX  4,X
        CPX   #0
        BNE   LOOP2
        INCA
        BMI   LOOP1
GOON    LDX   #$60
```

7. Skriv en subrutin TCHK i assemblerpråk för FLEX-processorn som undersöker en tabell med 8-bitars tal (med tecken) i minnet. Tabellen avslutas med ett dataord med värdet noll. Vid återhopp från subrutinen skall A-registret innehålla antalet negativa tal i tabellen och B-registret antalet tal med bit 6 = 1 och bit 3 = 0. Vid anrop av TCHK innehåller X-registret startadressen till tabellen.

För full poäng på uppgiften skall programmet vara korrekt radkommenterat. Endast register A, B och CC får vara förändrade vid återhopp från subrutinen. **(7p)**

Bilaga 1

ALU:ns funktion



ALU:ns **logik-** och **aritmetikoperationer** på indata **D** och **E** definieras av ingångarna **Funktion (F)** och $\mathbf{C_{in}}$ enligt tabellen nedan. **F** = **(f_3, f_2, f_1, f_0).**

I kolumnen Operation förklaras hur operationen utförs.

f_3 f_2 f_1 f_0	U = f(D,E,C_{in})	
	Operation	**Resultat**
0 0 0 0	bitvis nollställning	0
0 0 0 1		D
0 0 1 0		E
0 0 1 1	bitvis invertering	D_{1k}
0 1 0 0	bitvis invertering	E_{1k}
0 1 0 1	bitvis OR	D OR E
0 1 1 0	bitvis AND	D AND E
0 1 1 1	bitvis XOR	D XOR E
1 0 0 0	D + 0 + C_{in}	D + C_{in}
1 0 0 1	D + FFH + C_{in}	D – 1 + C_{in}
1 0 1 0		D + E + C_{in}
1 0 1 1	D + D + C_{in}	2D + C_{in}
1 1 0 0	D +E_{1k} + C_{in}	D – E – 1 + C_{in}
1 1 0 1	bitvis nollställning	0
1 1 1 0	bitvis nollställning	0
1 1 1 1	bitvis ettställning	FFH

Carryflaggan (C) innehåller minnessiffran ut (carry-out) från den mest signifikanta bitpositionen (längst till vänster) om en aritmetisk operation utförs av ALU:n.

Vid **subtraktion** gäller för denna ALU att
C = 1 om lånesiffra (borrow) uppstår och
C = 0 om lånesiffra inte uppstår.

Carryflaggans värde är 0 vid andra operationer än aritmetiska.

Overflowflaggan (V) visar om en aritmetisk operation ger "overflow" enligt reglerna för 2-komplementaritmetik.

V-flaggans värde är 0 vid andra operationer än aritmetiska.

Zeroflaggan (Z) visar om en ALU-operation ger värdet noll som resultat på U-utgången.

Signflaggan (N) är identisk med den mest signifikanta biten (teckenbiten) av utsignalen U från ALU:n.

Half-carryflaggan (H) är minnessiffran (carry) mellan de fyra minst signifikanta och de fyra mest signifikanta bitarna i ALU:n.

H-flaggans värde är 0 vid andra operationer än aritmetiska.

I tabellen ovan avser "**+**" och "**–**" **aritmetiska operationer**. Med t ex $\mathbf{D_{1k}}$ menas att samtliga bitar i **D** inverteras.

Bilaga 2

Assemblerspråket för FLEX-processorn.

Assemblerspråket använder sig av mnemoniska beteckningar liknande dem som processorkonstruktören MOTOROLA (FREESCALE) specificerat för maskininstruktioner för mikroprocessorerna 68XX och instruktioner till assemblatorn, s k pseudoinstruktioner eller assemblatordirektiv. Pseudoinstruktionerna listas i tabell 1.

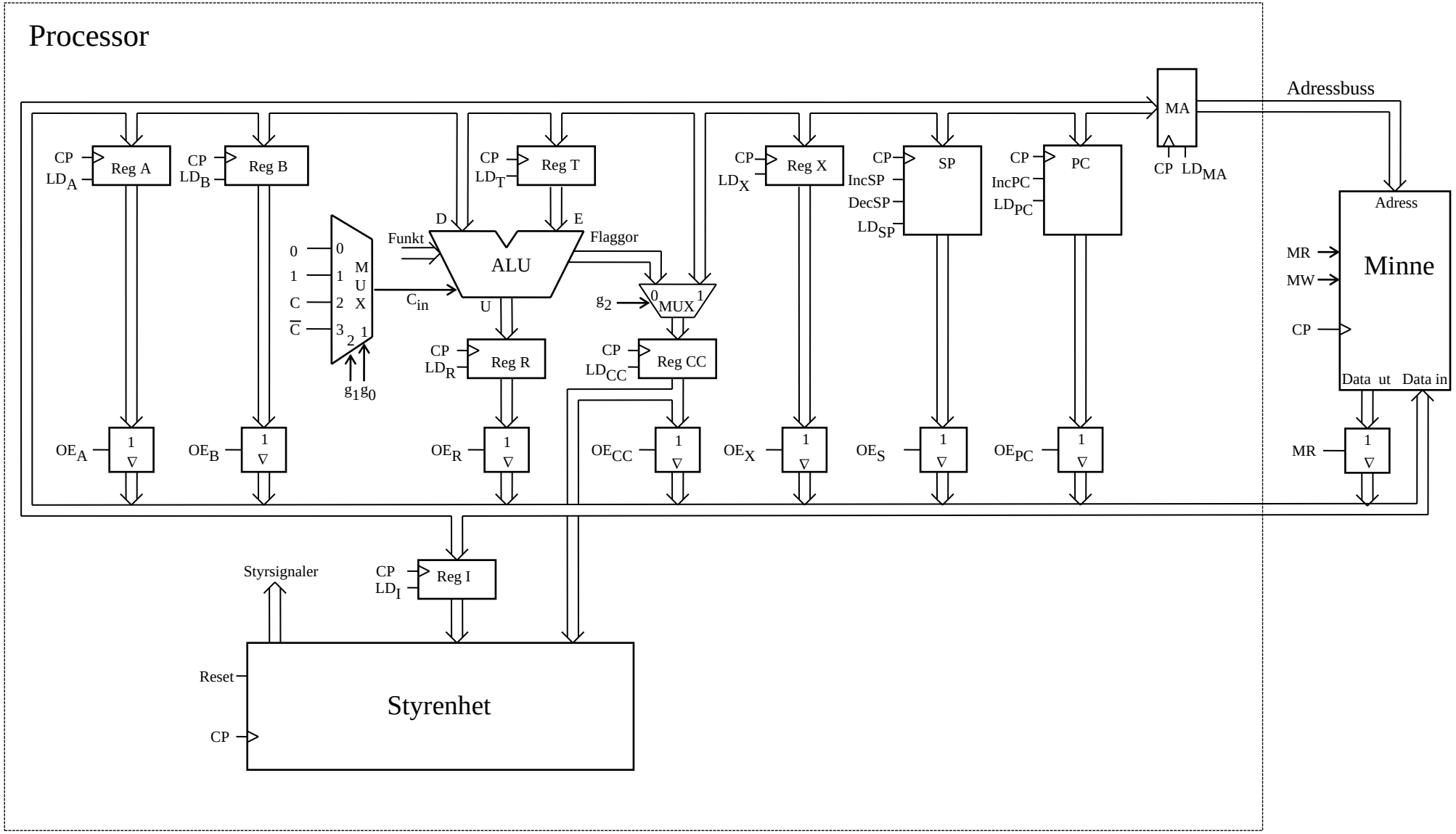
Tabell 1

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 adressen L. (RMB för Reseve Memory Bytes)
L EQU N	Ger symbolen L konstantvärdet N. (EQU för EQUates = beräknas till)
L FCB N1,N2	Avsätter en byte för varje argument i följd i minnet. Respektive byte ges konstantvärdet N1, N2 etc. Följden placeras med början på adressen L. (FCB för Form Constant Byte)
L FCS ”ABC”	Avsätter en byte för varje tecken i teckensträngen ”ABC” i följd i minnet. Respektive byte ges ASCII-värdet för A B C, etc. Följden placeras med början på adressen L. (FCS för Form Character String)

Tabell 2 7-bitars ASCII

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

Bilaga 3



Figur 1. Datorn FLEX.