



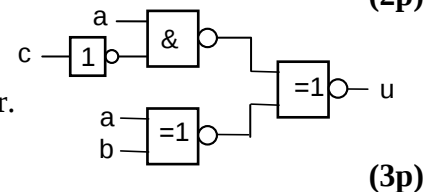
TENTAMEN

KURSNAMN	Digital- och datorteknik
PROGRAM:	Elektro Åk 1/ lp 4
KURSBETECKNING	EDA216
EXAMINATOR	Lars-Eric Arebrink
TID FÖR TENTAMEN	2011-08-18 kl 14.00 – 18.00
HJÄLPMEDEL	Av institutionen utgiven ”Instruktionslista för FLEX-processorn” (INS1) Tabellverk eller miniräknare får ej användas.
ANSV LÄRARE: Besöker tentamen	Rolf Snedsböl, tel. 772 1665 ca. kl 16
ANSLAG AV RESULTAT	När rättningen är färdig anslås resultatet med anonyma koder och tid för granskning på kursens hemsida. Lägg din anonyma kod på minnet. Den används när resultatet anslås!
ÖVRIG INFORM. BETYGSGRÄNSER. SLUTBETYG	Tentamen omfattar totalt 60 poäng. Onödigt komplicerade lösningar kan ge poängavdrag. Svar på uppgifter skall motiveras. Betyg 3: 24 poäng Betyg 4: 36 poäng Betyg 5: 48 poäng För slutbetyg 3, 4 eller 5 på kursen fordras betyg 3, 4 eller 5 på tentamen och godkända laborationer.

1. I uppgift a-h nedan används 8-bitars tal X, Y, S och D. $X = 00010111$ och $Y = 11101001$.

- Vilket talområde måste X, Y, S och D tillhöra om de tolkas som tal med tecken? (1p)
- Vilket talområde måste X, Y, S och D tillhöra om de tolkas som tal utan tecken? (1p)
- Visa med penna och papper hur räkneoperationen $S = X + Y$ utförs i en 8-bitars ALU. (1p)
- Vilka värden får flaggbitarna N, Z, V och C vid räkneoperationen i c)? (1p)
- Visa med penna och papper hur räkneoperationen $D = X - Y$ utförs i en 8-bitars ALU. (1p)
- Vilka värden får flaggbitarna N, Z, V och C vid räkneoperationen i e)? (1p)
- Tolka bitmönstren X, Y, S och D som tal *utan* tecken och ange deras decimala motsvarighet. Vilken eller vilka flaggbitar visar om resultatet är korrekt vid tal utan tecken? (1p)
- Tolka bitmönstren X, Y, S och D som tal *med* tecken och ange deras decimala motsvarighet. Vilken eller vilka flaggbitar anger om resultatet är korrekt vid tal med tecken? (1p)
- Antag att $N_{\min}$ är det minsta positiva tal, förutom 0, som kan representeras som ett 32-bitars flyttal med full upplösning enligt flyttalsstandarden IEEE 754-1985 (dvs 23 bitar av mantissan). Vilket decimalt värde har $N_{\min}$ approximativt. (2p)
- Hur många olika booleska funktioner av tre variabler finns det? (2p)

k) Ge ett "minimant" boolesk SP-uttryck för u i grindnätet till höger.



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.

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

3. Realisera en JK-vippa med hjälp av en SR-vippa och standardgrindar.

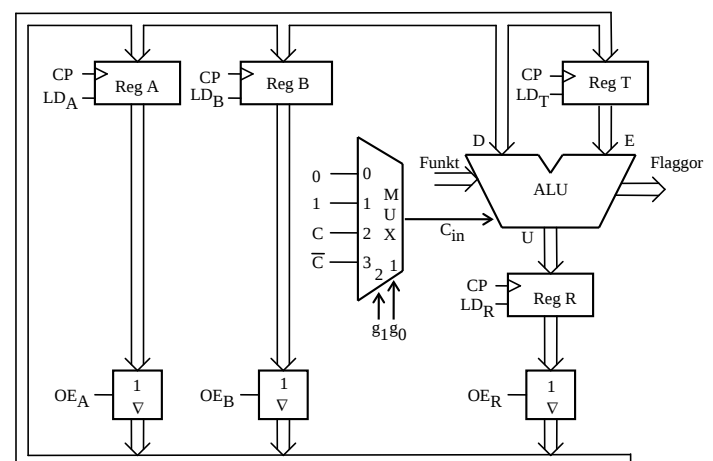
4. Ge RTN-beskrivning och styrsignaler för de tillstånd krävs för att utföra operationen enligt nedanstående RTN-beskrivning:

RTN-beskrivning: $5 \cdot (A + 1) + 3 \cdot (B - 1) \rightarrow A$
(Aritmetisk multiplikation avses)

Register B får inte ändras. Bortse från risken för overflow. Använd så få tillstånd som möjligt. Förutsatt att register A och B från början innehåller de data som skall behandlas enligt uttrycket ovan och att innehållena i register R och T är okända.

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

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 signalen C_{in} . OE_{PC}

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

State	S-term	RTN-beskrivning	Styrsignaler (= 1)
Q ₅	Q ₅ ·I _{xx}		OE _{PC} , LD _{MA} , IncPC
Q ₆	Q ₆ ·I _{xx}		MR, LD _{MA}
Q ₇	Q ₇ ·I _{xx}		MR, f ₁ , f ₀ , LD _R , LD _{CC}
Q ₈	Q ₈ ·I _{xx}		OE _R , MW, NF

NF i tabellens sista rad anger att nästa tillstånd (state) skall vara det första i FETCH-sekvensen.

- a) Ge RTN-beskrivningen för tillstånden Q₅-Q₈. (1p)
- b) Förklara vad instruktionen med EXECUTE-sekvensen ovan utför i varje klockcykel. Skriv instruktionen med assemblerspråk för FLEX-processorn. (2p)
- c) Med instruktionen nedan skall man kunna nollställa valfria bitar i ett minnesord. De bitpositioner som skall nollställas skall vara ettställda i instruktionens tredje ord, mask. Register A, B, X eller SP får inte påverkas av instruktionen som skall implementeras för FLEX-processorn med hjälp av styrenheten med fast logik.

BCLR n,X,#mask RTN: $M(X+n) \text{ AND } \text{mask}_{1k} \rightarrow M(X+n)$, Flags $\rightarrow$ CC

Samtliga funktioner ALU:n kan utföra framgår av bilaga 1. FLEX-datorn visas i bilaga 3. Gör en tabell liknande tabellen ovan för den efterfrågade EXECUTE-sekvensen. Använd operationskoden FB₁₆.

OPKOD
n
mask

(6p)

6. Besvara kortfattat följande frågor rörande FLEX-processorn.

- a) Förklara kortfattat vad som händer vid "Reset" av processorn. (1p)
- b) Det finns två principer för att ansluta inportar och utportar till processorns bussar, separatadresserad resp. minnesorienterad in- och utmatning. Vilken av dessa används i FLEX-processorn? Vilka är fördelarna med denna metod? (2p)

För vilka värden på dataordet W ($0 \leq W \leq 255$) utförs hoppen i c) och d)?

- c) LDAA #3F
BITA #W
BEQ Hopp (2p)
- d) LDAA #W
COMA
CMPA #55
BLT Hopp (4p)

6. forts.

- e) Översätt programavsnittet till höger till maskinkod. Det skall framgå hur "offset" för branchinstruktionerna beräknas.

(3p)

- f) Hur många klockpulser krävs för att köra hela programavsnittet i e).

(3p)

	ORG	\$30
	LDAA	#-10
ALOOP	LDX	#6
	LDAB	#5
XLOOP	DECB	
	LEAX	\$FE,X
	BNE	XLOOP
	INCA	
	BNE	ALOOP
	NOP	

7. En FLEX-processor skall användas i styrenheten för en maskin. Till inport FD_{16} är ett antal givare och switchar anslutna. En operatör skall styra maskinen via switcharna.

Huvudprogrammet för maskinstyrningen skall utformas som en evighetsslinga där inporten läses av en gång i början på varje varv. Programmet skall inledas med att stackpekaren först sätts till värdet FA_{16} . Sedan skall inporten läsas av och beroende på switcharnas och givarnas värden skall en av tre olika färdiga subrutiner anropas om motsvarande villkor i tabellen nedan är uppfyllt. Om inget av villkoren i tabellen är uppfyllt skall den färdiga subrutinen SUB4 anropas. Efter återhopp från subrutinerna skall ett nytt varv i slingan påbörjas.

Rita en flödesplan som beskriver programmet och skriv det i assemblerspråk för FLEX-processorn. Startadressen skall vara 10_{16} . Subrutinernas startadresser antas vara givna.

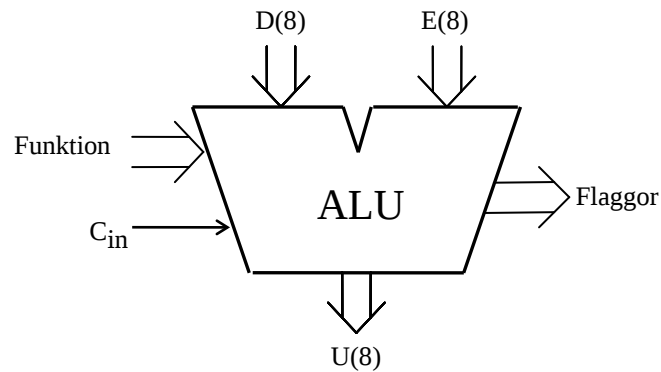
För full poäng på uppgiften skall programmet vara korrekt radkommenterat.

(2p+5p)

Villkor	Inport FD_{16}								Anropa subrutin
	b7	b6	b5	b4	b3	b2	b1	b0	
1	?	1	?	0	?	?	1	1	SUB1
2	?	?	0	1	?	?	1	?	SUB2
3	0	0	0	0	1	1	0	0	SUB3

Bilaga 1

ALU:ns funktion



ALU:ns **logik-** och **aritmetikoperationer** på indata **D** och **E** definieras av ingångarna **Funktion (F)** och **C_{in}** enligt tabellen nedan. **F = (f₃, f₂, f₁, f₀)**.

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 **D_{1k}** menas att samtliga bitar i **D** inverteras.

Bilaga 2

Assemblerspråket för FLEX-processorn.

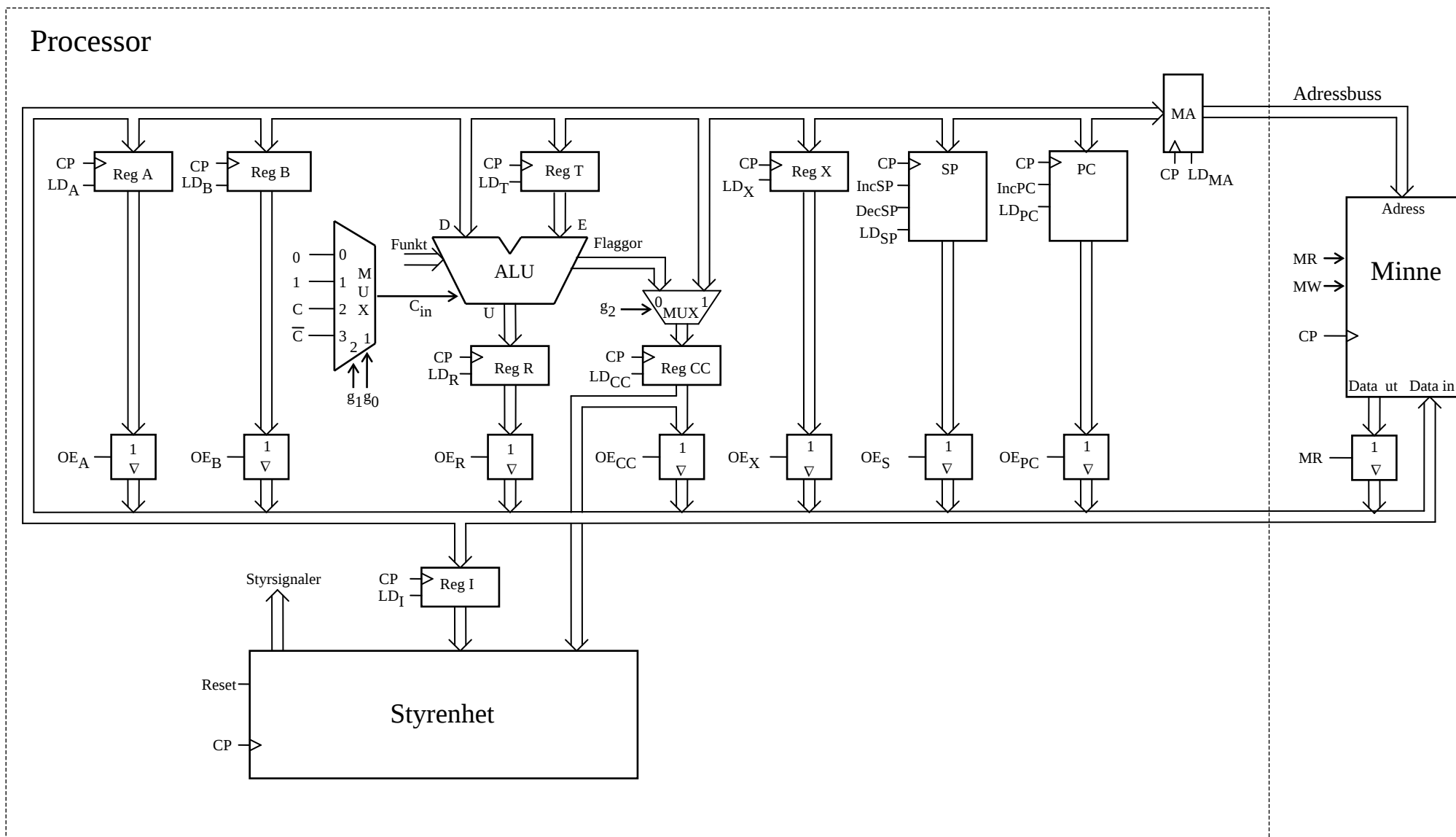
Assemblerspråket använder sig av mnemoniska beteckningar liknande dem som processorkonstruktören MOTOROLA specificerat för maskininstruktioner för mikroprocessorerna 68XX och instruktioner till assemblatorn, såsom pseudoinstruktioner eller assemblatordirektiv. Pseudoinstruktionerna listas i tabell 1.

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



Figur 1. Datorn FLEX.