



## TENTAMEN

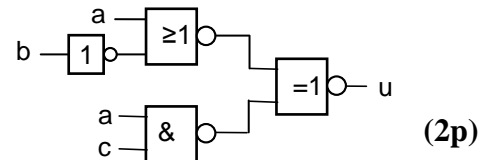
<b>KURSNAMN</b>	<b>Digital- och datorteknik</b>
<b>PROGRAM:</b>	<b>Data-, elektro- och mekatronikingenjör åk 1/ lp 1 och 2</b>
<b>KURSBETECKNING</b>	<b>LEU431</b>
<b>EXAMINATOR</b>	<b>Lars-Eric Arebrink</b>
<b>TID FÖR TENTAMEN</b>	<b>2014-08-21 kl 8.30 – 12.30</b>
<b>HJÄLPMEDEL</b>	<b>Av institutionen utgiven ”Instruktionslista för FLEX- eller FLIS-processorn” (INS1) Tabellverk eller miniräknare får ej användas.</b>
<b>ANSV LÄRARE:</b> <b>Besöker tentamen</b>	<b>Lars-Eric Arebrink, tel. 772 5718 vid flera tillfällen</b>
<b>ANSLAG AV RESULTAT</b>	<b>När rättningen är färdig anslås resultatet med anonyma koder och tid för granskning på kursens hemsida.</b>
<b>ÖVRIG INFORM.</b>	<b>Onödigt komplicerade lösningar kan ge poängavdrag. Svar på uppgifter skall normalt motiveras. <u>En svarsblankett finns sist i tentamenstesen.</u> För de uppgifter där svaret skall ges på svarsblanketten behöver inga lösningar redovisas. <u>Glöm inte lämna in svarsblanketten!</u> Den som är inskriven före HT 2013 kan använda FLEX- processorn istället för FLIS-processorn vid lösning av assembleruppgifter!</b>
<b>BETYGSGRÄNSER.</b>	<b>Tentamen omfattar totalt 60 poäng. Betyg 3: 24 poäng Betyg 4: 36 poäng Betyg 5: 48 poäng</b>
<b>SLUTBETYG</b>	<b>För slutbetyg 3, 4 eller 5 på kursen fordras betyg 3, 4 eller 5 på tentamen och godkända laborationer.</b>

1. I uppgift a-h nedan används 7-bitars tal  $X$ ,  $Y$ ,  $S$  och  $D$ . Aritmetiska operationer utförs på samma sätt och flaggor sätts på samma sätt som i ALU'n i bilaga 1. För tal med tecken används 2k-representation.  
 $X = 011\ 1001_2$  och  $Y = 100\ 0111_2$ .

- a) Vilket talområde måste  $X$ ,  $Y$ ,  $S$  och  $D$  tillhöra om de tolkas som tal utan tecken? (1p)
- b) Vilket talområde måste  $X$ ,  $Y$ ,  $S$  och  $D$  tillhöra om de tolkas som tal med tecken? (1p)
- c) Visa med penna och papper hur räkneoperationen  $S = X + Y$  utförs i en 7-bitars ALU. (1p)
- d) Vilka värden får flaggbitarna  $N$ ,  $Z$ ,  $V$  och  $C$  vid räkneoperationen i c)? (1p)
- e) Visa med penna och papper hur räkneoperationen  $D = X - Y$  utförs i en 7-bitars ALU. (1p)
- f) Vilka värden får flaggbitarna  $N$ ,  $Z$ ,  $V$  och  $C$  vid räkneoperationen i e)? (1p)
- g) Tolka bitmönstren  $X$ ,  $Y$ ,  $S$  och  $D$  som tal *utan* tecken och ange deras decimala motsvarighet. Avgör och motivera med hjälp av flaggorna om resultaten  $S$  och  $D$  är korrekta eller felaktiga. (1p)
- h) Tolka bitmönstren  $X$ ,  $Y$ ,  $S$  och  $D$  som tal *med* tecken och ange deras decimala motsvarighet. Avgör och motivera med hjälp av flaggorna om resultaten  $S$  och  $D$  är korrekta eller felaktiga. (1p)
- i) Flyttalsformatet "binary32" i standarden IEEE 754 har 1 teckenbit, 8 bitar karakteristika och 23 bitar "fraction". Visa approximativt det största värdet decimalt för talets absolutbelopp i detta format. (2p)

2.

- a) Markera det korrekta minimala SP-uttrycket för  $u$  i grindnätet till höger på den bifogade svarsblanketten!



- b) Ett karnaughdiagram för en boolesk funktion visas till höger. Markera ett uttryck på svarsblanketten som är lämpligt för realisering av den booleska funktionen  $f$ , då NOR-grindar med valfritt antal ingångar, XOR-grindar och NOT-grindar får användas.

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

Rutor med - representerar "dont care"-termer som får användas vid behov. Kretsrealiseringen behöver ej visas.

(4p)

3.

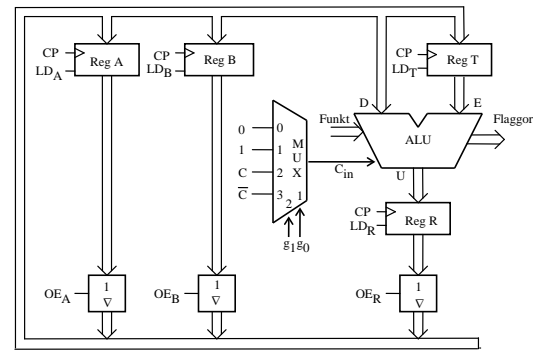
- a) Konstruera en T-vippa med hjälp av en SR-vippa och standardgrindar. (3p)
- b) Konstruera en autonom räknare med räknesekvansen  $q_2q_1q_0$ : 000, 001, 011, 111, 110, 100, 000, ... T-vippor, NAND-grindar med valfritt antal ingångar, XOR-grindar och NOT-grindar får användas. (6p)

4. I datavägen till höger innehåller register A(8) och B(8) från början värdena  $180_{10}$  resp.  $48_{10}$  på binär form. Därefter ges styr signaler och klockpulser enligt tabellen på svarsblanketten.

Kompletera tabellen på svarsblanketten med RTN-beskrivning samt hexadecimalt registerinnehåll för alla klockpulsintervall.

Vid laddning av ett register skall det nya innehållet ”synas” på raden efter laddningen i tabellen.

Flaggor sparar inte mellan klockpulserna i kopplingen ovan. Ett tidigare C-värde kan alltså inte användas i ett senare klockpulsintervall!



**ALU-funktioner: Se bilaga 1 på sidan 6 i tentamenstesen!**

**(4p)**

5. Figuren på sidan 45 i INS1 visar hur FLIS-datorn är uppbyggd. På sidorna 43 och 44 i INS1 visas hur ALU'ns funktion väljs med styr signalerna  $f_3 - f_0$  och  $C_{in}$ .

I tabellerna på svarsblanketten visas styr signalerna för två olika EXECUTE-sekvenser för samma maskininstruktion, en för FLIS- och en för FLEX-processorn.

- a) Kompletera den övre (FLIS) eller den undre (FLEX) tabellen med RTN-beskrivning! Förklara vilken assemblerinstruktion som beskrivs!

Som alternativ kan du alltså använda FLEX-datorn som visas i bilaga 3 med ALU'n i bilaga 1 och den undre tabellen istället för motsvarande för FLIS-datorn.

**(2p)**

- b) Instruktionen nedan "Test content of memory address Adr1 and branch to Adr2 if equal to zero" skall implementeras för FLIS(FLEX)-processorn med hjälp av styrenheten med fast logik. Operationskoden  $DF_{16}$  skall användas.

TBEQ Adr1,Adr2 RTN: U = M(Adr1), ALU(NZVC) → CC  
If Z = 1: PC + offset → PC

(U i RTN-beskrivningen ovan betecknar utdatavärdet från ALU'n.  
Med *offset* avses avståndet till Adr2.)

OPKOD
Adr1
offset

Kompletera tabellen på svarsblanketten med RTN-beskrivning och styr signaler för den efterfrågade EXECUTE-sekvensen.

**(6p)**

6. Besvara kortfattat följande frågor rörande FLIS- eller FLEX-processorn.

a) Det finns två principer för att ansluta inportar och utportar till processorns bussar, separatadresserad resp. minnesorienterad in- och utmatning. Vilken av dessa används i FLIS(FLEX)-processorn? Vilka är fördelarna med denna metod? **(2p)**

b) Före det villkorliga hoppet BPL i ett program utförs en subtraktion som påverkar flaggorna. Förklara vad som händer om subtraktionen före är  $50_{16} - D8_{16}$ . (8-bitars tal används.) **(2p)**

c) Översätt FLISP-subrutinen eller motsvarande FLEX-subrutin till höger till maskinkod på hexadecimal form och visa hur den placeras i minnet. Det skall framgå hur offset för branch-instruktionerna beräknas. **(3p)**

d) Subrutinen i c) anropas i FLISP-programmet (eller motsvarande FLEX-program) nedan.

```

ORG    $20
LDSP   #$FB
LDA    #$0A
JSR    TIME
STOP   BRA    STOP

```

Hur lång tid tar subrutinen att köra från och med JSR till och med RTS om processorn klockas med frekvensen 1MHz?

NUMB	EQU	-5
;		
	ORG	\$68
TIME	PSHA	
;		
LOOP1	PSHA	
;		
	LDA	#NUMB
LOOP2	INCA	
	NOP	
	BMI	LOOP2
	PULA	
;		
	DECA	
	NOP	
	BNE	LOOP1
;		
TIMEX	PULA	
	RTS	

(Använd motsvarande FLEX-instruktioner vid behov.)

**(3p)**

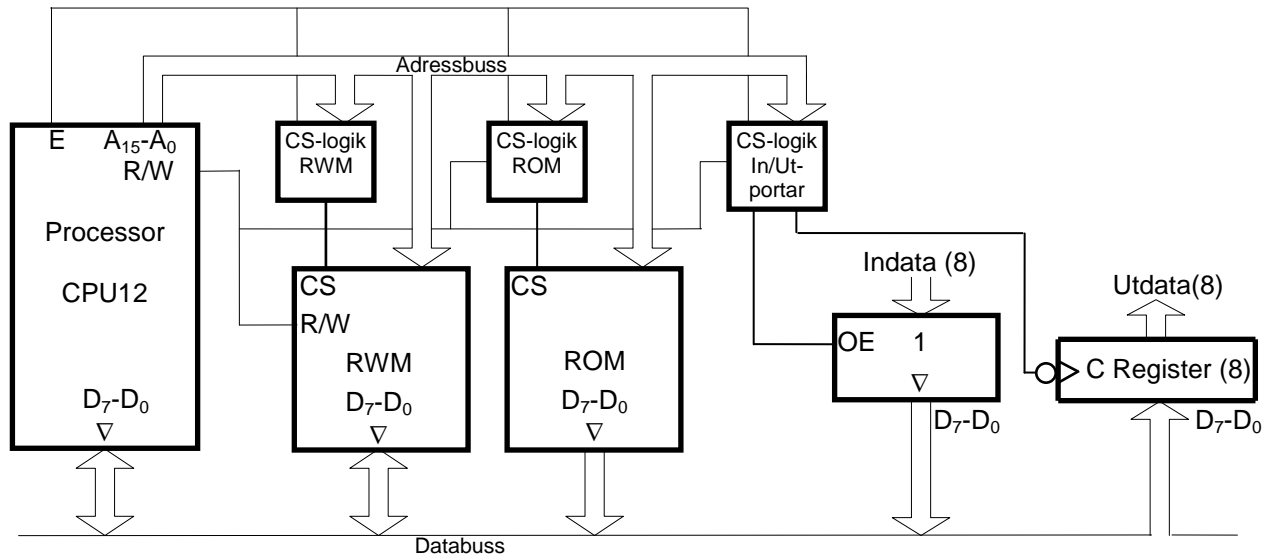
7. I en FLIS- eller FLEX-dators minne finns en tabell med 8-bitars tal lagrad. De lagrade värdena är tal utan tecken.

Skriv en subrutin TSUM i assemblerspråk för processorn som adderar samtliga tal i tabellen till en 16-bitars summa.

Vid anrop av subrutinen skall X-registret peka på tabellen och antalet dataord i tabellen skall finnas i A-registret. Vid återhopp skall summan, 16 bitar, finnas i X- och A-registret med mest signifikant del i X-registret.

Endast register X, A och CC får vara förändrade vid återhopp från subrutinen. För full poäng på uppgiften skall programmet vara "korrekt" radkommenterat. **(6p)**

## 8. Ett datorsystem visas nedan:



Figuren ovan visar principen för anslutning av externa minnesmoduler och externa in-/utportar till processorn CPU12.

En 64 kbyte ROM-modul skall i princip vara placerad från adress 0, men de första 2k adresserna i adressrummet skall inte aktivera någon minnesmodul eller port vid läsning eller skrivning.

En inport och en utport skall också anslutas. De skall ha samma adress och placeras direkt efter de första 2k adresserna i adressrummet. Det innebär att inporten skall prioriteras före ROM-modulen på denna adress.

En 8 kbyte RWM-modul skall dessutom placeras med start på adressen  $D000_{16}$  och prioriteras före ROM-modulen.

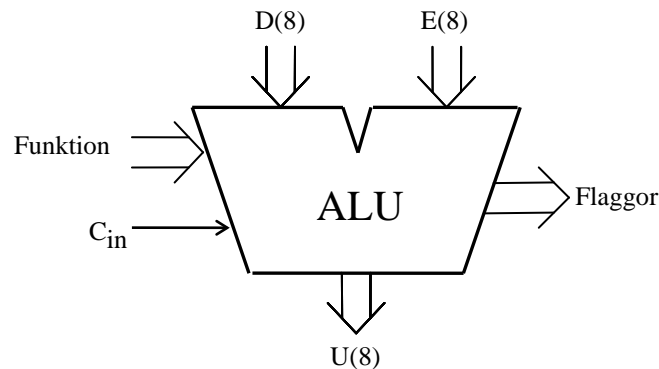
Rita CS-logiken för minnesmodulerna och portarna. Använd fullständig adressavkodning. Endast grundläggande logikgrindar med valfritt antal ingångar får användas. Visa inom vilka adressintervall de olika modulerna kan nås.

**Ledning:** Adressområdet för RWM-modulen är "lite problematiskt".

(7p)

## Bilaga 1

## ALU:ns funktion (FLEX-ALU'n)



ALU:ns **logik-** och **aritmetikoperationer** på indata **D** och **E** definieras av ingångarna **Funktion (F)** och **C<sub>in</sub>** enligt tabellen nedan. **F = (f<sub>3</sub>, f<sub>2</sub>, f<sub>1</sub>, f<sub>0</sub>)**.

I kolumnen Operation förklaras hur operationen utförs.

f <sub>3</sub> f <sub>2</sub> f <sub>1</sub> f <sub>0</sub>	U = f(D,E,C <sub>in</sub> )	
	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 <sub>1k</sub>
0 1 0 0	bitvis invertering	E <sub>1k</sub>
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 <sub>in</sub>	D + C <sub>in</sub>
1 0 0 1	D + FFH + C <sub>in</sub>	D - 1 + C <sub>in</sub>
1 0 1 0		D + E + C <sub>in</sub>
1 0 1 1	D + D + C <sub>in</sub>	2D + C <sub>in</sub>
1 1 0 0	D + E <sub>1k</sub> + C <sub>in</sub>	D - E - 1 + C <sub>in</sub>
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 lånesiffran (borrow) uppstår och C = 0 om lånesiffran 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<sub>1k</sub>** menas att samtliga bitar i **D** inverteras.

## Bilaga 2

### Assemblerspråket för FLEX- och FLIS-processorn.

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

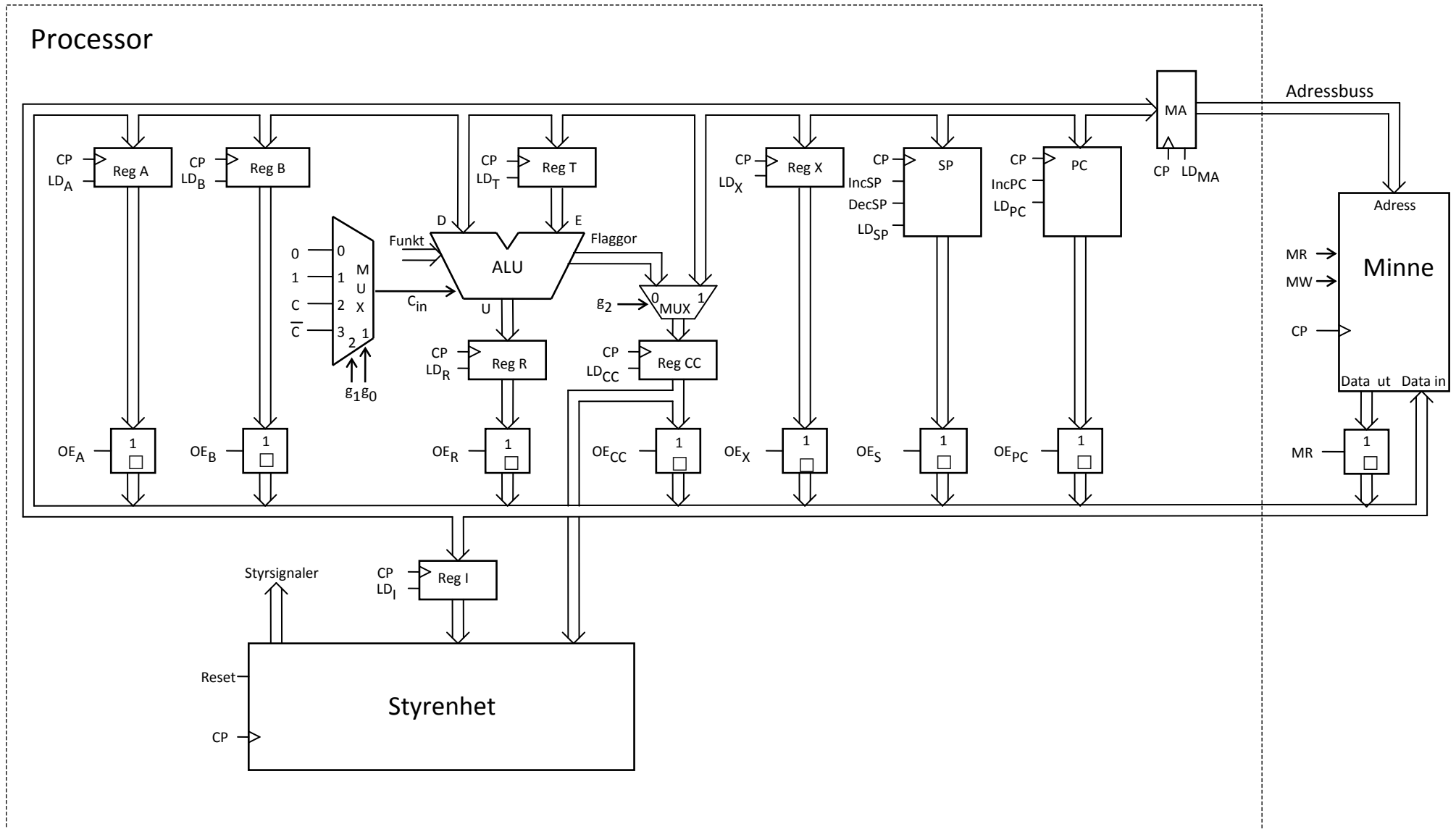
**Tabell 1**

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**

000	001	010	011	100	101	110	111	b <sub>6</sub> b <sub>5</sub> b <sub>4</sub> b <sub>3</sub> b <sub>2</sub> b <sub>1</sub> b <sub>0</sub>
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 (Observera att bilden visar FLEX-datorn)



Figur 1. Datorn FLEX.



# Svarsblankett (2014-08-21)

Anonym kod:

## Uppgift 2:

- a)
- |    |                       |    |                         |
|----|-----------------------|----|-------------------------|
| 1. | $u = (a' + b)(a + c)$ | 2. | $u = ab' + a'c'$        |
| 3. | $u = (a + b)(a' + c)$ | 4. | $u = (a + b')(a' + c')$ |
| 5. | $u = ab + a'c$        | 6. | $u = (a' + b')(a + c')$ |
| 7. | $u = a'b + ac$        | 8. | $u = a'b' + ac'$        |
- b)
- |    |                                      |    |   |
|----|--------------------------------------|----|---|
| 1. | $f = a'(b \oplus d)' + ab' + ad'$    | 2. | $f = a'(b \oplus d) + abd$                |
| 3. | $f = [a' + (b \oplus d)](a + b + d)$ | 4. | $f = a(b \oplus d)' + a'b'd'$             |
| 5. | $f = a(b \oplus d)'(a' + b' + d')$   | 6. | $f = [a + (b \oplus d)](a' + b)(a' + d)$  |
| 7. | $f = a(b \oplus d) + a'b + a'd$      | 8. | $f = [a' + (b \oplus d)](a + b')(a + d')$ |

## Uppgift 4: (Här används FLEX-ALU'n. Se bilaga 1!)

CP	Styrsignaler (=1)	RTN	A(8)	B(8)	T(8)	R(8)
1	$OE_B, LD_T$		B4	30	?	?
2	$OE_A, f_3, f_1, LD_R, LD_T$					
3	$OE_R, f_3, f_1, f_0, g_0, LD_R$					
4	$OE_R, f_3, f_1, f_0, LD_R$					
5	$OE_R, f_3, f_2, LD_R, LD_T$					
6	$OE_R, f_3, f_1, LD_R$					
7	$OE_R, LD_B$					
8	?					

## Uppgift 5:

### a) (FLISP)

State	S-term	RTN-beskrivning	Styrsignaler (=1)
Q <sub>4</sub>	Q <sub>4</sub> ·I <sub>xx</sub>		MR, LD <sub>TA</sub> , INC <sub>PC</sub>
Q <sub>5</sub>	Q <sub>5</sub> ·I <sub>xx</sub>		MR, g <sub>14</sub> , LD <sub>T</sub>
Q <sub>6</sub>	Q <sub>6</sub> ·I <sub>xx</sub>		OE <sub>A</sub> , f <sub>3</sub> , f <sub>2</sub> , g <sub>0</sub> , LD <sub>CC</sub> , NF

### (FLEX)

State	S-term	RTN-beskrivning	Styrsignaler (=1)
Q <sub>5</sub>	Q <sub>5</sub> ·I <sub>xx</sub>		OE <sub>PC</sub> , LD <sub>MA</sub> , Inc <sub>PC</sub>
Q <sub>6</sub>	Q <sub>6</sub> ·I <sub>xx</sub>		MR, LD <sub>MA</sub>
Q <sub>7</sub>	Q <sub>7</sub> ·I <sub>xx</sub>		MR, LD <sub>T</sub>
Q <sub>8</sub>	Q <sub>8</sub> ·I <sub>xx</sub>		OE <sub>A</sub> , f <sub>3</sub> , f <sub>2</sub> , g <sub>0</sub> , LD <sub>CC</sub> , NF

Instruktionen är:

Vänd!

