

Lösningsförslag tenta 2012-08-24 (Med reservation för eventuella fel)

1. $X = 11010111_2$; $Y = 00101001_2$ (8 bitars ordlängd)

a) $[0, 2^n - 1] = [0, 2^8 - 1] = [0, 255]$ (1p)

b) $[-2^{n-1}, +2^{n-1} - 1] = [-2^{8-1}, +2^{8-1} - 1] = [-128, +127]$ (1p)

c) $S = X + Y$

$$\begin{array}{r} 876543210 \text{ bitnummer} \\ 111111110 \text{ carry} \\ 11010111 \text{ X} \\ +00101001 \text{ Y} \\ \hline 00000000 = S \end{array}$$
 (1p)

d) $\underline{N} = s_7 = \underline{0}$
 $\underline{Z} = \underline{1} (S = 0)$
 $\underline{V} = x_7 * y_7 * s_7' + x_7' * y_7' * s_7 = 1 * 0 * 0' + 1' * 0' * 0 = \underline{0}$
 $\underline{C} = c_8 = \underline{1}$ (1p)

e) $D = X + Y_{1k} + 1$

$$\begin{array}{r} 876543210 \text{ bitnummer} \\ 110101111 \text{ carry} \\ 11010111 \text{ X} \\ +11010110 \text{ Y}_{1k} \\ \hline 10101110 = D \end{array}$$
 (1p)

f) $\underline{N} = d_7 = \underline{1}$
 $\underline{Z} = \underline{0} (D \neq 0)$
 $\underline{V} = x_7 * y_{7ik} * d_7' + x_7' * y_{7ik}' * d_7 = 1 * 1 * 1' + 1' * 1' * 1 = \underline{0}$
 $\underline{C} = c_8' = 1' = \underline{0}$ (1p)

g) $\underline{X} = 11010111_2 = D7_{16} = 13 * 16 + 7 = 208 + 7 = \underline{215}$

$\underline{Y} = 00101001_2 = 29_{16} = 2 * 16 + 9 = \underline{41}$

$\underline{S} = 00000000_2 = \underline{0}$ Resultatet S är felaktigt eftersom C = 1.

$\underline{D} = 10101110_2 = AE_{16} = 10 * 16 + 14 = 160 + 14 = \underline{174}$ Resultatet D är korrekt eftersom C = 0. (1p)

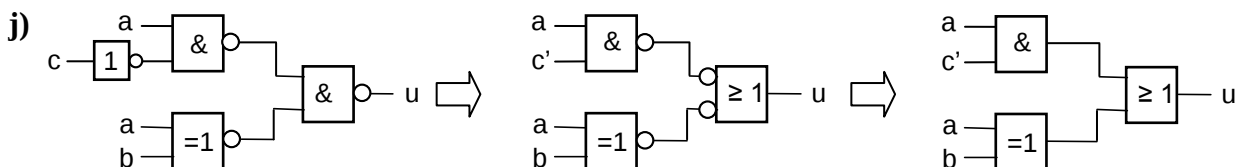
h) ($x_7 = 1$, neg) $X_{2k} = 2^8 - 215 = 256 - 215 = 41$ $\underline{X}$ motsvarar $\underline{-41}$

($y_7 = 0$, pos) $\underline{Y} = 00010111_2 = \underline{41}$

($s_7 = 0$, pos) $\underline{S} = \underline{0}$ Resultatet S är korrekt eftersom V = 0.

($d_7 = 1$, neg) $D_{2k} = 2^8 - 174 = 256 - 174 = 82$ $\underline{D}$ motsvarar $\underline{-82}$ Korrekt eftersom V = 0. (1p)

i) $N_{\min} = 1.000...0 * 2^{-126} = 16 * 2^{-4} * 2^{-126} = 16 * 2^{-130} = 16 * (2^{-10})^{13} \approx 16 * (10^{-3})^{13} = 16 * 10^{-39} = 1,6 * 10^{-38}$
 $N_{\max} = 1.111...1 * 2^{127} \approx 2 * 2^{127} = 2^{128} = 0,25 * 2^{130} = 0,25 * (2^{10})^{13} \approx 0,25 * (10^3)^{13} = 2,5 * 10^{38}$ (3p)



$u = ac' + (a \oplus b) = ac' + a'b + ab'$ som ger k-diagrammet:

		bc			
		00	01	11	10
a	0	0	0	1	1
	1	1	1	0	1

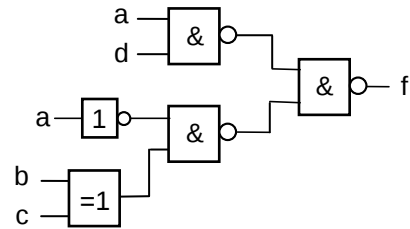
Ur diagrammet får man: $u = (a + b)(a' + b' + c')$

(3p)

2.

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

$$\begin{aligned} f &= ad + a'b'c + a'bc' = \\ &= ad + a'(b'c + bc') = \\ &= ad + a'(b \oplus c) \end{aligned}$$



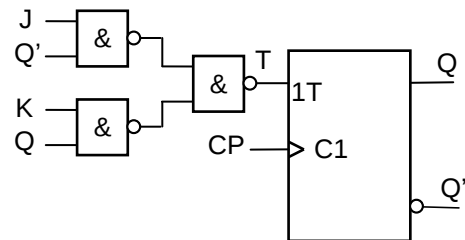
(4p)

3.

J	K	Q	Q ⁺	T
0	0	0	0	0
0	0	1	1	0
0	1	0	0	0
0	1	1	0	1
1	0	0	1	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	1

			T	Q
			0	1
00		0	0	0
01		0	1	1
11		1	1	1
10		1	0	0

$$T = JQ' + KQ$$



(5p)

4. $7A + 3B = 3(2A + B) + A$

CP	RTN	Styrsignaler (=1)
1	$2A \rightarrow R$	$OE_A, f_3, f_1, f_0, LD_R$
2	$B \rightarrow T$	OE_B, LD_T
3	$R + T \rightarrow R$	OE_R, f_3, f_1, LD_R
4	$2R \rightarrow R, R \rightarrow T$	$OE_R, f_3, f_1, f_0, LD_R, LD_T$
5	$R + T \rightarrow R$	OE_R, f_3, f_1, LD_R
6	$A \rightarrow T$	OE_A, LD_T
7	$R + T \rightarrow R$	OE_R, f_3, f_1, LD_R
8	$R \rightarrow A$	OE_R, LD_A

(5p)

5. a)

State	S-term	RTN-beskrivning	Styrsignaler (= 1)
Q_y	$Q_y \cdot I_{xx}$	$FF_{16} \rightarrow R$	f_3, f_2, f_1, f_0, LD_R
Q_{y+1}	$Q_{y+1} \cdot I_{xx}$	$R \rightarrow MA$	OE_R, LD_{MA}
Q_{y+2}	$Q_{y+2} \cdot I_{xx}$	$M \rightarrow PC, \text{Next Fetch}$	MR, LD_{PC}, NF

Q_y : Maxvärdet FF_{16} till R.

Q_{y+1} : FF_{16} från R till MA för att adressera högsta adressen i minnet där startadressen finns.

Q_{y+2} : Startadressen läses och placeras i PC.

Detta är RESET-sekvensen, som innebär att processorn startar programmet vars adress finns som data på adressen FF_{16} .

(2p)

b)

State	S-term	RTN-beskrivning	Aktiva styrsignaler (=1)
Q_5	$Q_5 \cdot I_{FC}$	$PC \rightarrow MA, PC+1 \rightarrow PC$	$OE_{PC}, LD_{MA}, IncPC$
Q_6	$Q_6 \cdot I_{FC}$	$M \rightarrow T$	MR, LD_T
Q_7	$Q_7 \cdot I_{FC}$	$PC \rightarrow MA, PC+1 \rightarrow PC$	$OE_{PC}, LD_{MA}, IncPC$
Q_8	$Q_8 \cdot I_{FC}$	$M \rightarrow MA$	MR, LD_{MA}
Q_9	$Q_9 \cdot I_{FC}$	$M \oplus T \rightarrow R, \text{Flags} \rightarrow CC$	$MR, f_2, f_1, f_0, LD_R, LD_{CC}$
Q_A	$Q_A \cdot I_{FC}$	$R \rightarrow M, \text{Next Fetch}$	OE_R, MW, NF

OPKOD
mask
Adr

(4p)

6.

- a) $Q_3: PC \rightarrow MA, PC + 1 \rightarrow PC$ Kopiera PC-värdet till MA. Öka PC.
 $Q_4: M \rightarrow IR$ Läs där "gamla" PC pekade (OP-kod) och placera den i IR. (2p)
- b) Stacken är placerad i minnet. Processorn kan skriva (push) på och läsa (pull) från stacken. Den fungerar enligt principen "Last In First Out", LIFO. Stackpekaren (SP) innehåller adressen till det översta värdet på stacken. Vid skrivning på stacken minskas först SP med ett för att skapa plats. Vid läsning ökas SP med ett efter läsningen. (2p)
- c) FLEX-datorn använder minnesorienterad I/O. Fördelen med det är att vanliga instruktioner som läser eller skriver i minnet kan användas för I/O samt att det inte behövs några extra styrsignaler för I/O. (2p)
- d) BLO utför hopp om $C = 1$. $3F_{16} = 63$
Hoppet utförs därför om: $63 + W \geq 2^8 = 256$ eller $W \geq 256 - 63 = 193$. $193 \leq W \leq 255$ (2p)
- e) BLT avser tal med tecken. För 8-bitars tal gäller då talintervallet $[-128, 127]$.
NEGA utför $-W$ om $W \neq -128$ (vilket motsvarar 128).
För $W \neq -128$ utförs hoppet om: $-W + 55_{16} < 0$ eller $-W + 85 < 0$,
dvs $W > 85$ eller $85 < W \leq +127$.
För $W = -128$ utförs hoppet om: $-128 + 85 < 0$. Dvs det utförs.
Hoppet utförs alltså för $85 < W \leq 128$ (4p)

f)

Adr	Data	~	Läge		
40	0F 14	4		LDAA #20	
42	11 50	4		LDX #TAB	
44	7C	5	LOOP	LDAB 1,X+	
45	14 FE	5		STAB \$FE	
47	44	4		DECA	
48	5C FA	5		BPL LOOP	$44 - 4A = FA$
4A	0C 0C	5		LDAB 12	
4C	1A 0F	6		ANDB #\$0F	
4E	5A 14	5		BRA NEXT	$64 - 50 = 14$
50			TAB		
64	00	3	NEXT	NOP	

(3p)

- g) $\underline{N} = 4+4+(5+5+4+5)*21+5+6+5+3 = 27+19*20 = 27+399 = \underline{426}$ klockpulser (3p)

7. (Först flödesplan)

(2p)

DIPSW	EQU	\$FD	
HEXDIS	EQU	\$FE	
LOOP	EQU	\$50	
	ORG	LOOP	
LOOP	LDAA	DIPSW	Läs indata (bitmönster)
	TFR	A,B	Spara en kopia av indata
TEST1	ANDA	##11010101	Behåll bitarna 7, 6, 4, 2 och 0, nolla övriga
	CMPA	##11000001	Bit 4 och 2 = 0 och bit 7, 6, och 0 = 1?
	BNE	TEST2	Nej, nästa test
	LDAA	#\$10	Ja
	STAA	HEXDIS	Visa värde på HEXDISPLAY
	BRA	LOOP	Nästa varv
TEST2	TFR	B,A	Återställ indata
	ANDA	##00010010	Behåll bitarna 4 och 1, nolla övriga
	CMPA	##00010010	Bit 4 och 1 = 1?
	BNE	TEST3	Nej, nästa test
	LDAA	#\$23	Ja
	STAA	HEXDIS	Visa värde på HEXDISPLAY
	BRA	LOOP	Nästa varv
TEST3	TFR	B,A	Återställ indata
	CMPA	##01001110	\$4E?
	BNE	NONE	Nej, ingen träff
	LDAA	#\$47	Ja
	STAA	HEXDIS	Visa värde på HEXDISPLAY
	BRA	LOOP	Nästa varv
NONE	LDAA	#\$EE	Ja
	STAA	HEXDIS	Visa värde på HEXDISPLAY
	BRA	LOOP	Nästa varv

(5p)