

Lösningsförslag tenta 2013-12-16

(Version 5 med reservation för eventuella fel. Uppdaterad 140417.)

1. $X = 1010\ 0101_2$; $Y = 0101\ 1011_2$ (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) $S = X + Y$

876543210	bitnummer
111111110	carry
10100101	X
+01011011	Y
00000000	= S

 (1p)

d) $N = s_7 = 0$
 $Z = 1$ ($S = 0$)
 $V = x_7 * y_7 * s_7' + x_7' * y_7' * s_7 = 1 * 0 * 0' + 1' * 0' * 0 = 0$
 $C = c_8 = 1$
 NZVC = 0101 (1p)

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

876543210	bitnummer
101001011	carry
10100101	X
+10100100	Y _{1k}
01001010	= D

 (1p)

f) $N = d_7 = 0$
 $Z = 0$ ($D \neq 0$)
 $V = x_7 * y_{7ik} * d_7' + x_7' * y_{7ik}' * d_7 = 1 * 1 * 0' + 1' * 1' * 0 = 1$
 $C = c_8' = 1' = 0$
 NZVC = 0010 (1p)

g) $X = 1010\ 0101_2 = A5_{16} = 10 * 16 + 5 = 165$

$Y = 0101\ 1011_2 = 5B_{16} = 5 * 16 + 11 = 91$

$S = 0000\ 0000_2 = 0$ Resultatet S är felaktigt eftersom $C = 1$.

$D = 0100\ 1010_2 = 4A_{16} = 4 * 16 + 10 = 74$ Resultatet D är korrekt eftersom $C = 0$. (1p)

h) ($x_7 = 1$, neg) $X_{2k} = 2^8 - 165 = 256 - 165 = 91$ X motsvarar -91

($y_7 = 0$, pos) $Y = 91$

($s_7 = 0$, pos) $S = 0000\ 0000_2 = 0$ Resultatet S är korrekt eftersom $V = 0$.

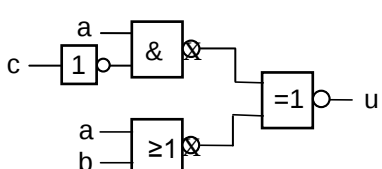
($d_7 = 0$, pos) $D = 74$ Fel eftersom $V = 1$. (1p)

i) $N = 125,125_{10} = 1111101.001 = 1.111101001 * 2^6$ Format: s/c/f

$s = 0$ (pos); $c = \exp + 2^{8-1} - 1 = 6 + 127 = 133 = 10000101_2$; $f = \text{mantissa} - 1$

$N_{\text{flyt}} = 0/100\ 0010\ 1/111\ 1010\ 0100\ 0000\ 0000\ 0000_2 = 42FA4000_{16}$ (2p)

2.

a)  $u = [(ac') \oplus (a+b)]' = (ac')(a+b) + (ac')'(a+b)' = ac' + abc' + (a'+c)a'b' = ac' + abc' + a'b' + a'b'e = ac' + a'b' = (a+b')(a'+c)$ (Nr 4)

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

(3p)

b)

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

NAND-gindar $\leftrightarrow$ "ettor"

$f = a'(b \oplus c \oplus d) + ad$ (Nr 5)

[Nr 8 ger också korrekt funktion, men utgår från "nollor" och är därför inte lämpligt för NAND-grindar. (2p)]

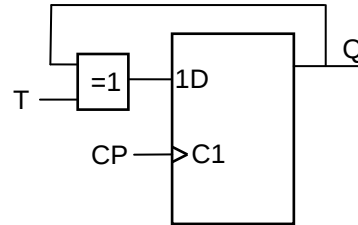
(4p)

3.

a)

T	Q	Q ⁺	D
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

			Q
D	0	1	
T	0	0	1
	1	1	0



$$D = T'Q + TQ' = T \oplus Q$$

(2p)

b)

För T-vippor gäller att $q^+ = q$ för $T = 0$ och $q^+ = q'$ för $T = 1$.

q ₂	q ₁	q ₀	T ₂	T ₁	T ₀	q ₂ ⁺	q ₁ ⁺	q ₀ ⁺
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	1	0	0	1	1	0
0	1	1	0	0	1	0	1	0
1	0	0	1	0	0	0	0	0
1	0	1	0	0	1	1	1	0
1	1	0	0	0	1	1	1	1
1	1	1	0	1	0	1	0	1

				q ₁ q ₀
T ₂	00	01	11	10
q ₂	0	0	0	1
	1	1	0	0

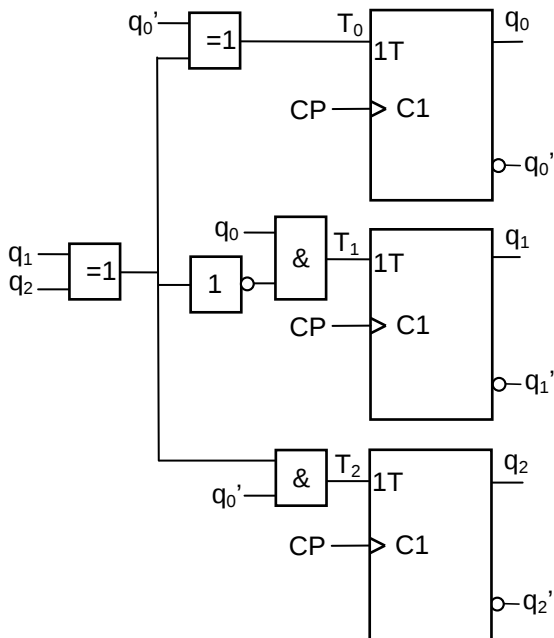
				q ₁ q ₀
T ₁	00	01	11	10
q ₂	0	0	1	0
	1	0	0	1

				q ₁ q ₀
T ₀	00	01	11	10
q ₂	0	1	0	1
	1	0	1	0

$$T_2 = q_2'q_1q_0' + q_2q_1'q_0' = (q_2 \oplus q_1)q_0'$$

$$T_1 = q_2'q_1'q_0 + q_2q_1q_0 = (q_2 \oplus q_1)'q_0$$

$$T_0 = (q_2 \oplus q_1 \oplus q_0)' = (q_2 \oplus q_1) \oplus q_0'$$



(6p)

4.

CP	Styrsignaler (=1)	RTN	A(8)	B(8)	T(8)	R(8)
1	OE _B , LD _T	B → T	C8	64	?	?
2	OE _A , f ₃ , f ₂ , LD _R	A - T - 1 → R	C8	64	64	?
3	OE _R , f ₃ , f ₁ , f ₀ , LD _R , LD _T	2R → R, R → T	C8	64	64	63
4	OE _R , f ₃ , f ₁ , LD _R	R + T → R	C8	64	63	C6
5	OE _R , LD _B	R → B	C8	64	63	29
6	?	?	C8	29	63	29

(4p)

5. a) FLISP:

State	Summaterm	RTN-beskrivning	Styrsignaler
Q ₄	Q ₄ · I _{xx}	M(PC) → T, PC+1 → PC	MR, LD _T , INC _{PC}
Q ₅	Q ₅ · I _{xx}	X + T → R	OE _X , f ₃ , f ₁ , f ₀ , LD _R
Q ₆	Q ₆ · I _{xx}	R → X, (New Fetch)	OE _R , LD _X , NF

FLEX:

State	Summaterm	RTN-beskrivning	Styrsignaler
Q ₅	Q ₅ · I _{xx}	PC → MA, PC+1 → PC	OE _{PC} , LD _{MA} , IncPC
Q ₆	Q ₆ · I _{xx}	M(MA) → T	MR, LD _T
Q ₇	Q ₇ · I _{xx}	X + T → R	OE _X , f ₃ , f ₁ , LD _R
Q ₈	Q ₈ · I _{xx}	R → X, (Next Fetch)	OE _R , LD _X , NF

Detta är: LEAX n,X

(2p)

b) FLISP:

State	S-term	RTN-beskrivning	Styrsignaler (=1)
Q ₄	Q ₄ · I _{DF}	M(PC) → TA, PC+1 → PC	MR, LD _{TA} , INC _{PC}
Q ₅	Q ₅ · I _{DF}	M(X) → T	MR, g ₁₃ , g ₁₂ , LD _T
Q ₆	Q ₆ · I _{DF}	M(TA) + T → R, 0 → T, ALU(NZVC) → CC	MR, g ₁₄ , f ₃ , f ₁ , f ₀ , LD _R , LD _{CC} , CLR _T
Q ₇	Q ₇ · I _{DF}	R → M(X)	OE _R , g ₁₃ , g ₁₂ , MW
Q ₈	Q ₈ · I _{DF}	X + 1 → R	OE _X , f ₃ , f ₀ , LD _R
Q ₉	Q ₉ · I _{DF}	R → X, (New Fetch)	OE _R , LD _X , NF

OPKOD
Adr

FLEX:

State	S-term	RTN-beskrivning	Styrsignaler (=1)
Q ₅	Q ₅ · I _{DF}	PC → MA, PC+1 → PC	OE _{PC} , LD _{MA} , IncPC
Q ₆	Q ₆ · I _{DF}	M(MA) → MA	MR, LD _{MA}
Q ₇	Q ₇ · I _{DF}	M(MA) → T	MR, LD _T
Q ₈	Q ₈ · I _{DF}	X → MA	OE _X , LD _{MA}
Q ₉	Q ₉ · I _{DF}	M(MA) + T → R, Flags → CC	MR, f ₃ , f ₁ , LD _R , LD _{CC}
Q _A	Q _A · I _{DF}	R → M(MA)	OE _R , MW
Q _B	Q _B · I _{DF}	X + 1 → R	OE _X , f ₃ , g ₀ , LD _R
Q _C	Q _C · I _{DF}	R → X, (Next Fetch)	OE _R , LD _X , NF

(4p)

6.

- a) Funktionen hos signalen NF är att ladda tillståndräknaren med värdet 3, som motsvarar första tillståndet i FETCH. Om sista tillståndet i EXECUTE inte aktiverar NF så fortsätter tillståndräknaren räkna tills den varvar efter 15 och börjar om med RESET följt av FETCH, dvs processorn startar om på det aktuella programmets startadress. **(2p)**

- b) Vi förutsätter att båda talen har ordlängden 8 bitar och därmed talområdet $[-128_{10}, +127_{10}]$, eftersom hoppvillkoret för BPL avser tal med tecken.

Det villkorliga hoppet BPL Adr utförs om N-flaggan = 0, dvs. om subtraktionen ger ett resultat ≥ 0 . Värdet $97_{16} = 151_{10}$ motsvarar det negativa talet $-(256 - 151)_{10} = -105_{10}$.

Hoppvillkoret blir $W - (-105_{10}) \geq 0$; $W \geq -105_{10}$; $-105_{10} \leq W \leq 127_{10}$

Om overflow inträffar får N-flaggan i detta fall värdet 1 vilket medför att hoppet då inte utförs.

Overflow inträffar om $W - (-105_{10}) \geq 128_{10}$ dvs. $W \geq 128_{10} - 105_{10} = 23_{10}$.

Hoppet görs därför om W tillhör intervallet: $-105_{10} \leq W \leq 22_{10}$.

Eftersom 2-komplementrepresentation används för negativa tal delar vi upp intervallet i en negativ och en positiv del, $-105_{10} \leq W \leq -1$ och $0 \leq W \leq 22_{10}$.

Det verkliga intervallet för den negativa delen blir:

$256_{10} - 105_{10} \leq W \leq 256_{10} - 1$ dvs. $151_{10} \leq W \leq 255_{10}$.

Hoppet utförs alltså om W tillhör något av intervallen: $0 \leq W \leq 22_{10}$ eller $151_{10} \leq W \leq 255_{10}$. **(3p)**

c)

Adr	Data	~	Assemblerkod			Offset
			Läge	Operation	Operand	
		4		LDA JSR	#10 DELAY	
			CONST	EQU ORG	5 \$80	
80	10	3	DELAY	PSHA		
81	10	3		PSHA		
82	F0 05	2	YLOOP	LDA	#CONST	
84	08	3	ILOOP	DECA		
85	25 FD	4		BNE	ILOOP	84 - 87 = FD
87	48 00	4		DEC	0,SP	
89	25 F7	4		BNE	YLOOP	82 - 8B = F7
8B	14	3		PULA		
8C	14	3		PULA		
8D	43	2		RTS		

(3p)

- d) JSR DELAY tar 4 klockpulser.

A-reg innehåller värdet 10 vid anropet, så yttre slingan (YLOOP) utförs 10 gånger.

$T = 4 + 3 + 3 + [2 + (3 + 4) \cdot 5 + 4 + 4] \cdot 10 + 3 + 3 + 2 = 18 + [10 + 7 \cdot 5] \cdot 10 = 468 \text{ st } (\mu\text{s})$

(3p)

7.

BOS	EQU	\$FA	Bottom of stack
INPORT	EQU	\$FB	DIPSWITCH
UTPORT	EQU	\$FC	Ljusdiodramp (LED)
DELAY	EQU	\$80	Färdig fördröjningsrutin (200 ms)
START	ORG	\$10	Huvudprogrammets startadress
	LDSP	#BOS	Initiera stackbotten
	CLR	UTPORT	Alla ljusdioder släckta
PLOOP	LDX	#PATTERN	Pekare till tabell för diodmönster
WAIT	TST	INPORT	Starta visning?
	BPL	WAIT	Nej, bit7 = 0
	LDA	,X+	Ja, bit7 = 1. Hämta bitmönster från tabell
	STA	UTPORT	Visa bitmönster
	JSR	DELAY	Vänta 200 ms
	CMPX	#PATTERN+8	Hela tabellen avverkad?
	BEQ	PLOOP	Ja, börja om från startläget
	BRA	WAIT	Nej, kolla om nästa bitmönster skall visas

;

;

Tabell med samtliga bitmönster

PATTERN	FCB	%10000001,%01000010,%00100100,%00011000
	FCB	%00011000,%00100100,%01000010,%10000001

(7p)

