Lösningsförslag tenta 2012-12-21
(Version 4 med reservation för eventuella fel.)

1. X = 1101 1001_2; Y = 0101 0011_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
110100110   carry
 11011001   X
+01010011   Y
 00101100 = S
```
(1p)

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

NZVC = 0001 **(1p)**

e)
```
D = X+Y1k+1
876543210     bitnummer
111110011   1 carry
 11011001   X
+10101100   Y1k
 10000110 = D
```
(1p)

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

NZVC = 1000 **(1p)**

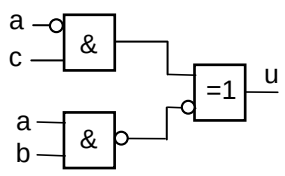
g) $\underline{X} = 1101\ 1001_2 = D9_{16} = 13*16 + 9 = \underline{217}$
$\underline{Y} = 0101\ 0011_2 = 53_{16} = 5*16 + 3 = \underline{83}$
$\underline{S} = 0010\ 1100_2 = 2C_{16} = 2*16 + 12 = \underline{44}$ Resultatet S är felaktigt eftersom C = 1.
$\underline{D} = 1000\ 0110_2 = 86_{16} = 8*16 + 6 = \underline{134}$ Resultatet D är korrekt eftersom C = 0. **(1p)**

h) (x_7 = 1, neg) $X_{2k} = 2^8 - 217 = 256 - 217 = 39$ $\underline{X}$ motsvarar $\underline{-39}$
(y_7 = 0, pos) $\underline{Y} = 0101\ 0011_2 = \underline{83}$
(s_7 = 0, pos) $\underline{S} = 0010\ 1100_2 = \underline{44}$ Resultatet S är korrekt eftersom V = 0.
(d_7 = 1, neg) $D_{2k} = 2^8 - 134 = 256 - 134 = \underline{122}$ $\underline{D}$ motsvarar $\underline{-122}$. Korrekt eftersom V = 0. **(1p)**

i) 4 decimala sifferpositioner krävs inklusive teckensiffra: A = 0196; B = 0320, B_{9k} = 9679
Princip: `D = A+B9k+1`
```
 3210     siffernummer
 0111   1 carry
 0196   A
+9679   B9k
 9876 = D (Resultatet är negativt då d3=9.)
```
För att kontrollera resultatet kan man 10-komplementera det för att se beloppet.
$D_{10k} = D_{9k} + 1 = 0123 + 1 = 0124$; D motsvarar alltså –124 **(2p)**

j) $N_{flyt} = 424B8000_{16}$ = 0/100 0010 0/100 1011 1000 0000 0000 0000_2
s c f
s = 0 (+)
c = 1000 0100_2 = 128 + 4 = 127 + 5; exp = 127 + 5 – 127 = 5
m = 1.f = 1.100 1011 1000 0000 0000 0000

$\underline{N_2}$ = +1.100 1011 1000 0000 0000 0000*2^5 = (32 + 16 + 2 + ,5 + ,25 + ,125) = $\underline{50{,}875}$ **(2p)**

k)


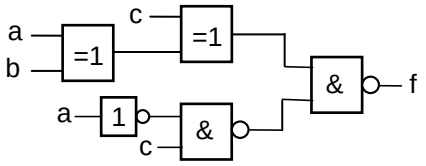
$\underline{u} = (a'c)'ab + a'c(ab)' =$
$= (a+c')ab + a'c(a'+ b') =$
$= ab + \cancel{abc}' + a'c + \cancel{a'b'c} =$
$= ab + a'c$ **(3p)**

2. $\underline{f} = a'c + a'b'c' + a'bc + abc' + ab'c = a'c + a'(b'c' + bc) + a(bc' + b'c) =$

$= a'c + a'(b \oplus c)' + a(b \oplus c) = a'c + [a \oplus (b \oplus c)]' = a'c + (a \oplus b \oplus c)'$

alt: $b'c + (a \oplus b \oplus c)'$

$a'b' + (a \oplus b \oplus c)'$



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

(4p)

3.

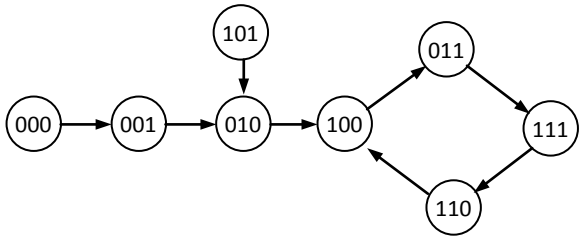
a)

T	Q^+
0	Q
1	Q'

(1p)

b) $J_2 = q_1,\ K_2 = q_1';\quad J_1 = q_2 + q_0,\ K_1 = q_0';\quad J_0 = q_1',\ K_0 = q_2 + q_1'$

q_2	q_1	q_0	J_2	K_2	J_1	K_1	J_0	K_0	q_2^+	q_1^+	q_0^+
0	0	0	0	1	0	1	1	1	0	0	1
0	0	1	0	1	1	0	1	1	0	1	0
0	1	0	1	0	0	1	0	0	1	0	0
0	1	1	1	0	1	0	0	0	1	1	1
1	0	0	0	1	1	1	1	1	0	1	1
1	0	1	0	1	1	0	1	1	0	1	0
1	1	0	1	0	1	1	0	1	1	0	0
1	1	1	1	0	1	0	0	1	1	1	0



(5p)

4. $7A - 5B = 2A + 5(A - B)$

CP	RTN	Styrsignaler (=1)
1	B → T	OE_B, LD_T
2	A - T → R	OE_A, f_3, f_2, g_0, LD_R
3	2R → R, R → T	OE_R, f_3, f_1, f_0, LD_R, LD_T
4	2R → R	OE_R, f_3, f_1, f_0, LD_R
5	R + T → R	OE_R, f_3, f_1, LD_R
6	A → T	OE_A, LD_T
7	R + T → R	OE_R, f_3, f_1, LD_R
8	R + T → R	OE_R, f_3, f_1, LD_R
9	R → A	OE_R, LD_A

(4p)

5. a)

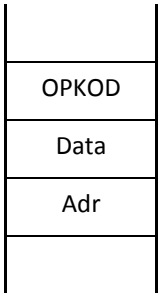
State	S-term	RTN-beskrivning	Styrsignaler (=1)
Q_5	$Q_5 \cdot I_{xx}$	PC → MA, PC+1 → PC	OE_{PC}, LD_{MA}, IncPC
Q_6	$Q_6 \cdot I_{xx}$	M → T	MR, LD_T
Q_7	$Q_7 \cdot I_{xx}$	A + T → R, Flags → CC	OE_A, f_3, f_1, LD_R, LD_{CC}
Q_8	$Q_8 \cdot I_{xx}$	R → A, (Next Fetch)	OE_R, LD_A, NF

Från början pekar PC på minnesordet efter OP-koden.
Q_5: Minnet adresseras med innehållet i PC, som också ökas med ett.
Q_6: Dataordet efter OP-koden laddas i T-reg.
Q_7: A och minnesordet efter OP-koden adderas och flaggorna laddas.
Q_8: Summan till A.

Detta är instruktionen <u>ADDA #Data</u> **(3p)**

b)

State nr	S-term	RTN-beskrivning	Styrsignaler (=1)
Q_5	$Q_5 \cdot I_{FE}$	PC → MA, PC+1 → PC	OE_{PC}, LD_{MA}, IncPC
Q_6	$Q_6 \cdot I_{FE}$	M → T	MR, LD_R
Q_7	$Q_7 \cdot I_{FE}$	PC → MA, PC+1 → PC	OE_{PC}, LD_{MA}, IncPC
Q_8	$Q_8 \cdot I_{FE}$	M → MA	MR, LD_{MA}
Q_9	$Q_9 \cdot I_{FE}$	M + T → R, Flags → CC	MR, f_3, f_1, LD_R, LD_{CC}
Q_{10}	$Q_{10} \cdot I_{FE}$	R → M, (Next Fetch)	OE_R, MW, NF



(4p)

6.

a) Under RESET-fasen bildar ALU:n adressen FF_{16} som laddas i R-registret och sedan flyttas till MA-registret. Därefter läser processorn innehållet på adressen FF_{16}, som skall vara startadressen till det program man vill starta, lägger den i PC och övergår till FETCH-fasen. **(2p)**

b) Programräknaren PC innehåller adressen till nästa instruktion eller del av instruktion. PC håller alltså reda på var programmet finns i minnet. **(2p)**

c) BHI (>) avser tal utan tecken. För 8-bitars tal gäller då talintervallet [0, 255]. $80 = 128
Hoppvillkoret blir: $W - 128 > 0$ eller $W > 128$, dvs. $\underline{128 < W \leq 255}$. **(2p)**

d) BLE (≤) avser tal med tecken. För 8-bitars tal gäller då talintervallet [-128, 127]. $40 = 64
Hoppvillkoret blir $W - 64 \leq 0$ eller $W \leq 64$, dvs. $-128 \leq W \leq 64$.

Eftersom negativa värden ersätts med 2-komplementet av motsvarande positiva värde delar vi upp intervallet i en positiv och en negativ del:

$\underline{0 \leq W \leq 64}$ och $-128 \leq W \leq -1$.

Verkligt värde för den negativa delen blir då: $256 - 128 \leq W \leq 256 - 1$ eller $\underline{128 \leq W \leq 255}$ **(3p)**

e)

Adr	Data (Hex)	~	Läge	
-			XVAL	EQU -10
-				ORG $80
80	6C	5	TIME	PSHA
81	6F	5		PSHX
82	11 F6	4	LOOP	LDX #XVAL
84	E1	4	LOOPX	INX
85	00	3		NOP
86	5E FC	5		BNE LOOPX 84–88=FC
88	44	4		DECA
89	5D 02	5		BEQ TEXIT 8D–8B=02
8B	5A F5	5		BRA LOOP 82–8D=F5
8D	73	4	TEXIT	PULX
8E	70	4		PULA
8F	6A	4		RTS

(3p)

f) $\underline{T} = 7+5+5+(4+(4+3+5)*10+4+5+5)*5-5+4+4+4 = 24+(18+120)*5 = 24+690 = \underline{714 \text{ klockpulser } (\mu s)}$ **(3p)**

g)

Adr	Data (Hex)			
1F	-	ORG	$20	
20	0B 41	LDAA	$41	Hämta PLB
22	28 43	ADDA	$43	Addera med QLB
24	13 45	STAA	$45	Lagra RLB
26	0B 40	LDAA	$40	Hämta PHB
28	2C 42	ADCA	$42	Addera med QHB och carry
2A	13 44	STAA	$44	Lagra RHB
2C	-			

(3p)

7.

```
START   EQU     $30             Programstart
BOS     EQU     $F0             Bottom of stack
DIPSW   EQU     $FD             Inport för DIPSWITCH
LED     EQU     $FE             Utport för LED-display

        ORG     START
        LDS     #BOS            Init stack
        LDX     #BITTAB         Pekare till bitmönstertabell
        CLRB                    Räknare för tabellindex

RLOOP   LDAA    DIPSW           Läs switchar
        ANDA    #%10000001      Maska fram bit 0 och 7

        CMPA    #%00000001      Mönster för minskning av index
        BNE     NODEC           Ingen minskning, kolla ökning
        DECB                    Minska variabel modulo 8
        BRA     RUN             Högerflyt

NODEC   CMPA    #%10000000      Mönster för ökning av index
        BNE     RUN             Ingen ökning. Inget flyt
        INCB                    Öka variabel modulo 8. Vänsterflyt

RUN     ANDB    #%00000111      Maska bort bit 3-7(Räkna modulo 8)
        LDAA    B,X             Hämta värde från bitmönstertabell
        STAA    LED             Visa ljus

        JSR     DELAY           Vänta

        BRA     RLOOP
```

(6p)