

Lösningsförslag tenta 2014-01-16

(Version 3 med reservation för eventuella fel. Uppdaterad 140128.)

1. $X = 01\ 0110_2$; $Y = 01\ 1111_2$ (6 bitars ordlängd)

a) $[0, 2^n - 1] = [0, 2^6 - 1] = [0, 63]$ (1p)

b) $[-2^{n-1}, +2^{n-1} - 1] = [-2^{6-1}, +2^{6-1} - 1] = [-32, +31]$ (1p)

c) $S = X + Y$

6543210	bitnummer
0111100	carry
010110	X
+011111	Y
110101	$= S$

(1p)

d) $N = s_5 = \underline{1}$
 $Z = \underline{0} (S \neq 0)$
 $V = x_5 * y_5 * s_5' + x_5' * y_5' * s_5 = 0 * 0 * 1' + 0' * 0' * 1 = \underline{1}$
 $C = c_6 = \underline{0}$

NZVC = 1010 (1p)

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

6543210	bitnummer
0000001	carry
010110	X
+100000	Y _{1k}
110111	$= D$

(1p)

f) $N = d_5 = \underline{1}$
 $Z = \underline{0} (D \neq 0)$
 $V = x_5 * y_{5ik} * d_5' + x_5' * y_{5ik}' * d_5 = \underline{0 * 1 * 1'} + 0' * 1' * 1 = \underline{0}$
 $C = c_6' = 0' = \underline{1}$

NZVC = 1001 (1p)

g) $\underline{X} = 01\ 0110_2 = 16_{16} = 1 * 16 + 6 = \underline{22}$
 $\underline{Y} = 01\ 1111_2 = 1F_{16} = 1 * 16 + 15 = \underline{31}$
 $\underline{S} = 11\ 0101_2 = 35_{16} = 3 * 16 + 5 = \underline{53}$ Resultatet S är korrekt eftersom C = 0.
 $\underline{D} = 11\ 0111_2 = 37_{16} = 3 * 16 + 7 = \underline{55}$ Resultatet D är felaktigt eftersom C = 1. (1p)

h) ($x_5 = 0$, pos) $\underline{X} = \underline{22}$.
 ($y_5 = 0$, pos) $\underline{Y} = \underline{31}$
 ($s_5 = 1$, neg) $S_{2k} = 2^6 - 53 = 11$ $\underline{S}$ motsvarar $\underline{-11}$. Resultatet S är felaktigt eftersom V = 1.
 ($d_5 = 1$, neg) $D_{2k} = 2^6 - 55 = 9$ $\underline{D}$ motsvarar $\underline{-9}$. Resultatet D är korrekt eftersom V = 0. (1p)

i) Format: s/c/f.
 "+0": s = 0, c = 0 och f = 0. $N_{flyt} = 0/00000000/000...0 = \underline{0000\ 0000}_{16}$
 "+∞": s = 0, c = 255 och f = 0. $N_{flyt} = 0/11111111/000...0 = \underline{7F80\ 0000}_{16}$ (2p)

2.

a)

$$\begin{aligned} \underline{u} &= (a \oplus b)'(b + c) = \\ &= (a'b' + ab)(b + c) = \\ &= a'b'c + ab + abe = a'b'c + ab \quad (\text{Nr 6}) \end{aligned}$$

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

(2p)

b)

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

NOR-gindar ↔ "nollor"

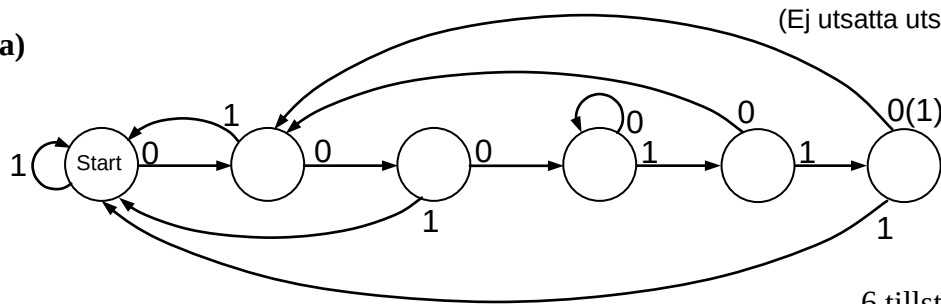
$\underline{f} = (a' + d)[a + (b \oplus c \oplus d)]$ (Nr 8)

[Nr 5 ger också korrekt funktion, men utgår från "ettor" och är därför inte lämpligt för NOR-gindar. (2p)]

(4p)

3.

a)



6 tillstånd ger minst 3 vippor
($2^2 < 6 \leq 2^3$)

(4p)

b)

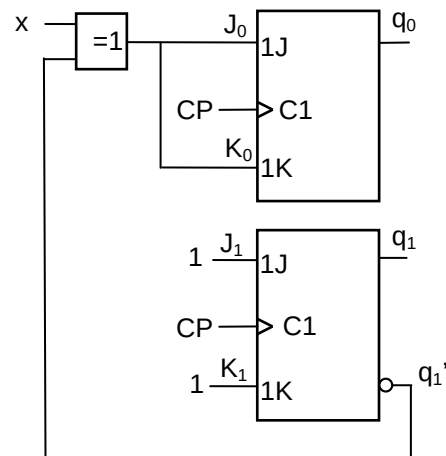
x	q ₁	q ₀	J ₁	K ₁	J ₀	K ₀	q ₁ ⁺	q ₀ ⁺
0	0	0	1	-	1	-	1	1
0	0	1	1	-	-	1	1	0
0	1	0	-	1	0	-	0	0
0	1	1	-	1	-	0	0	1
1	0	0	1	-	0	-	1	0
1	0	1	1	-	-	0	1	1
1	1	0	-	1	1	-	0	1
1	1	1	-	1	-	1	0	0

J ₀	q ₁ q ₀			
	00	01	11	10
x 0	1	-	-	0
x 1	0	-	-	1

K ₀	q ₁ q ₀			
	00	01	11	10
x 0	-	1	0	-
x 1	-	0	1	-

$$J_0 = K_0 = x'q_1' + xq_1$$

$$J_1 = K_1 = 1$$



(6p)

4.

CP	Styrsignaler (=1)	RTN	A(8)	B(8)	T(8)	R(8)
1	OE _A , LD _T	A → T	46	A5	?	?
2	OE _B , f ₃ , f ₁ , LD _R	B + T → R	46	A5	46	?
3	OE _R , f ₃ , f ₁ , f ₀ , LD _R	2R → R	46	A5	46	EB
4	OE _R , f ₃ , f ₁ , f ₀ , LD _R	2R → R	46	A5	46	D6
5	OE _R , f ₃ , f ₂ , LD _R	R - T - 1 → R	46	A5	46	AC
6	OE _R , LD _A	R → A	46	A5	46	65
7	?		65	A5	46	65

(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}	CC OR T → R	OE _{CC} , f ₂ , f ₁ , LD _R
Q ₆	Q ₆ ·I _{xx}	R → CC(b ₄ -b ₀), (New Fetch)	OE _R , g ₁₀ , g ₈ , g ₆ , g ₄ , g ₂ , LD _{CC} , 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}	CC OR T → R	OE _{CC} , f ₂ , f ₀ , LD _R
Q ₈	Q ₈ ·I _{xx}	R → CC, (Next Fetch)	OE _R , LD _{CC} , NF

Instruktionen är: ORCC #Data

(2p)

b)

FLISP:

State	S-term	RTN-beskrivning	Styrsignaler (=1)
Q ₄	Q ₄ ·I _{DF}	U = A, ALU(NZVC) → CC	OE _A , f ₂ , LD _{CC}
Q ₅	Q ₅ ·I _{DF} ·Z	If Z = 1: New Fetch	NF
Q ₅	Q ₅ ·I _{DF} ·Z'	else: M(PC) → T	MR, LD _T
Q ₅	Q ₅ ·I _{DF}	PC + 1 → PC	INC _{PC}
Q ₆	Q ₆ ·I _{DF}	PC + T → R	OE _{PC} , f ₃ , f ₁ , f ₀ , LD _R
Q ₇	Q ₇ ·I _{DF}	R → PC, 0 → T, (New Fetch)	OE _R , LD _{PC} , CLR _T , NF

OPKOD
Offset

FLEX:

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

(5p)

6.

- a) Båda talen har ordlängden 8 bitar och därmed talområdet $[-128_{10}, +127_{10}]$, eftersom hoppvillkoret för BMI avser tal med tecken.

Det villkorliga hoppet BMI Adr utförs om N-flaggan = 1, dvs. om subtraktionen ger ett resultat < 0 . Värde $7A_{16} = 122_{10}$ och motsvarar $+122$.

Hoppvillkoret blir: $122 - W < 0$; $122 < W$; $122 < W < 128$

Overflow inträffar dock om $122 - W \geq 128$ dvs. $122 - 128 \geq W$ eller $-6 \geq W$.

Overflow inträffar alltså i intervallet $-128 \leq W \leq -6$. Detta är utanför "hoppintervallet" ovan och innebär att hoppet utförs även här eftersom N-flaggan då får fel värde.

Hoppet utförs alltså om W tillhör intervallen $122 < W < 128$ eller $-128 \leq W \leq -6$.

Eftersom 2-komplementrepresentation används för negativa tal tar vi fram de verkliga värdena för dessa.

Det verkliga intervallet för de negativa talen blir: $256 - 128 \leq W \leq 256 - 6$ dvs. $128 \leq W \leq 250$.

De bägge intervallen kan då slås samman till: $122 < W \leq 250$

Hoppet utförs alltså om W tillhör intervallet: $122 < W \leq 250$

(4p)

b)

Adr	Data	~	Assemblerkod			
		(FLISP)		ORG	\$90	
90	10	3	DELAY	PSHA		
91	10	3		PSHA		
92	F0 F6	2	YLOOP	LDA	#CONST	
94	07	3	ILOOP	INCA		
95	00	2		NOP		
96	25 FC	4		BNE	ILOOP	94 - 98 = FC
98	47 00	4		INC	0,SP	
9A	25 F6	4		BNE	YLOOP	92 - 9C = F6
9C	BE 01	4		LEASP	1,SP	
9E	14	3		PULA		
9F	43	2		RTS		

(3p)

- c) BSR DELAY tar 5 klockpulser för FLISP och 7 för FLEX.

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

$$T_{\text{FLISP}} = 5 + 3 + 3 + [2 + (3 + 2 + 4) \cdot 10 + 4 + 4] \cdot 5 + 4 + 3 + 2 = 20 + [10 + 9 \cdot 10] \cdot 5 = \underline{520 \text{ st } (\mu\text{s})}$$

$$T_{\text{FLEX}} = 7 + 5 + 5 + [4 + (4 + 3 + 5) \cdot 10 + 5 + 5] \cdot 5 + 6 + 4 + 4 = 31 + [14 + 12 \cdot 10] \cdot 5 = \underline{701 \text{ st } (\mu\text{s})}$$

(3p)

7.

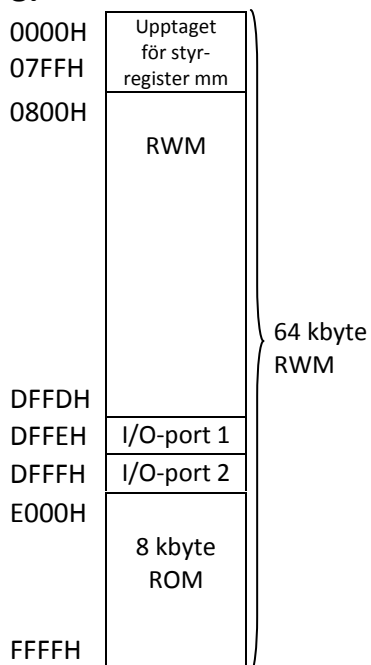
; Subrutin HEXCH

HEXCH	PSHX		Spara register på stack
	CLR	COUNT	Nollställ bokstavsräknare
HLOOP	LDA	,X+	Hämta data från textsträng
	BEQ	HEXIT	Strängslut
	ANDA	#\$7F	Nej, maska bort bit 7
	ORA	#\$20	Ettställ bit 5. (Stora bokstäver -> små)
	CMPA	#'a'	Testa om "a"
	BLO	HLOOP	Ej liten bokstav, fortsätt med nästa
	CMPA	#'f'	Testa om "f"
	BHI	HLOOP	Ej rätt liten bokstav, fortsätt med nästa
	INC	COUNT	Ja, öka bokstavsräknare
	BRA	HLOOP	Fortsätt med nästa
HEXIT	LDA	COUNT	Hämta resultat
	PULX		Återställ register
	RTS		

COUNT RMB 1 Plats för bokstavsräknare

(7p)

8.

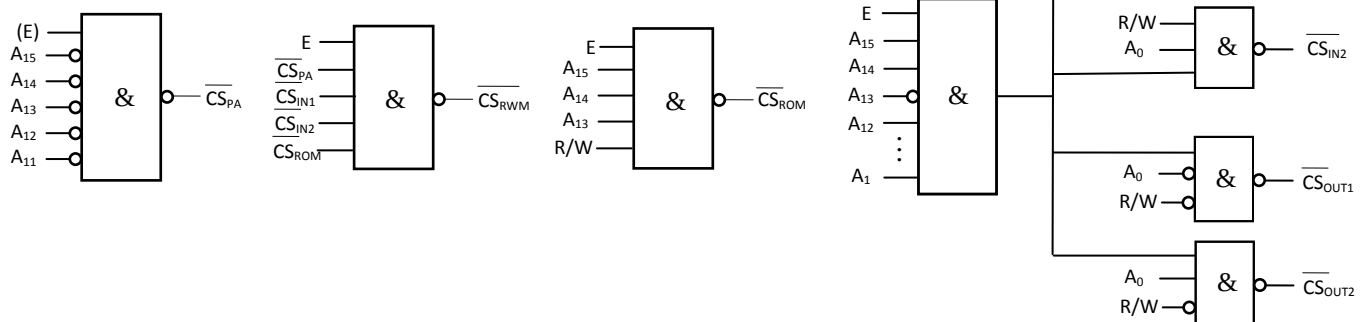


Passiv	A	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Start: 0000H =		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Slut: 07FFH =		0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
	CS	11 st															

ROM	A	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Start: E000H =		1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Slut: FFFFH =		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	CS	13 st															

I/O-port 2	A	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Adr: DFFFH =		1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
	CS																

I/O-port 1	A	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Adr: DFFE H =		1	1	0	1	1	1	1	1	1	1	1	1	1	1	0	
	CS																



(6p)