# Lösningsförslag tenta 2014-01-16

**(Version 4 med reservation för eventuella fel. Uppdaterad 140417.)**

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

NZVC = 1010 **(1p)**

**e)**
```
D = X+Y1k+1
6543210   bitnummer
0000001   carry
 010110   X
+100000   Y1k
 110111 = D
```
**(1p)**

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

NZVC = 1001 **(1p)**

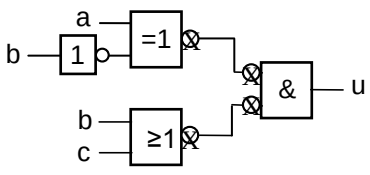
**g)** $\underline{X} = 01\ 0110_2 = 16_{16} = 1*16 + 6 = \underline{22}$
$\underline{Y} = 01\ 1111_2 = 1F_{16} = 1*16 + 15 = \underline{31}$
$\underline{S} = 11\ 0101_2 = 35_{16} = 3*16 + 5 = \underline{53}$ Resultatet S är korrekt eftersom C = 0.
$\underline{D} = 11\ 0111_2 = 37_{16} = 3*16 + 7 = \underline{55}$ 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. $\underline{N_{flyt}} = 0/00000000/000…0 = \underline{0000\ 0000}_{16}$
"+∞": s = 0, c = 255 och f = 0. $\underline{N_{flyt}} = 0/11111111/000…0 = \underline{7F80\ 0000}_{16}$ **(2p)**

**2.**

**a)**




$\underline{u} = (a⊕b)'(b+c) =$
$= (a'b'+ ab)(b+c) =$
$= a'b'c+ab+ \cancel{abc} = a'b'c+ab$ (Nr 6)

| a \ bc | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |

(**2p**)

**b)**

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

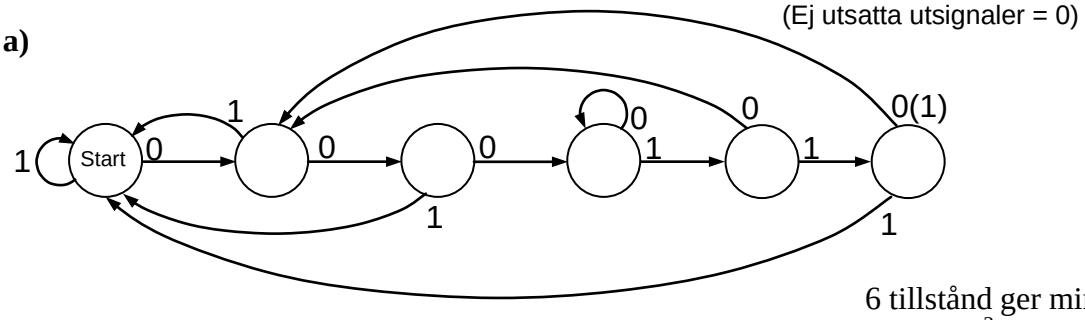
NOR-gindar ↔ "nollor"
$\underline{f} = \underline{(a'+d)[a+(b⊕c⊕d)]}$ (Nr 8)

[Nr 5 ger också korrekt funktion, men utgår från "ettor" och är därför inte lämpligt för NOR-grindar. **(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 |

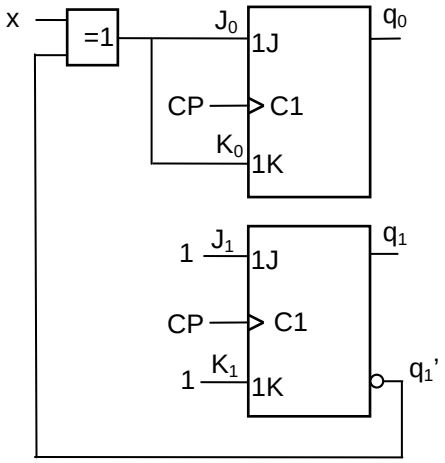
$J_0$

| x \ $q_1q_0$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 1 | - | - | 0 |
| 1 | 0 | - | - | 1 |

$K_0$

| x \ $q_1q_0$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | - | 1 | 0 | - |
| 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 |
|---|---|---|---|
| $Q_4$ | $Q_4 \cdot I_{xx}$ | M(PC) → T, PC+1 → PC | MR, $LD_T$, $INC_{PC}$ |
| $Q_5$ | $Q_5 \cdot I_{xx}$ | CC OR T → R | $OE_{CC}$, $f_2$, $f_1$, $LD_R$ |
| $Q_6$ | $Q_6 \cdot I_{xx}$ | 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 |
|---|---|---|---|
| $Q_5$ | $Q_5 \cdot I_{xx}$ | PC → MA, PC+1 → PC | $OE_{PC}$, $LD_{MA}$, IncPC |
| $Q_6$ | $Q_6 \cdot I_{xx}$ | M(MA) → T | MR, $LD_T$ |
| $Q_7$ | $Q_7 \cdot I_{xx}$ | CC OR T → R | $OE_{CC}$, $f_2$, $f_0$, $LD_R$ |
| $Q_8$ | $Q_8 \cdot I_{xx}$ | R → CC, (Next Fetch) | $OE_R$, $LD_{CC}$, NF |

Instruktionen är: ORCC #Data **(2p)**

**b)** FLISP:

| State | S-term | RTN-beskrivning | Styrsignaler (=1) |
|---|---|---|---|
| $Q_4$ | $Q_4 \cdot I_{DF}$ | U = A, ALU(NZVC) → CC | $OE_A$, $f_2$, $LD_{CC}$ |
| $Q_5$ | $Q_5 \cdot I_{DF} \cdot Z$ | If Z = 1: New Fetch | NF |
| $Q_5$ | $Q_5 \cdot I_{DF} \cdot Z'$ | else: M(PC) → T | MR, $LD_T$ |
| $Q_5$ | $Q_5 \cdot I_{DF}$ | PC + 1 → PC | $INC_{PC}$ |
| $Q_6$ | $Q_6 \cdot I_{DF}$ | PC + T → R | $OE_{PC}$, $f_3$, $f_1$, $f_0$, $LD_R$ |
| $Q_7$ | $Q_7 \cdot I_{DF}$ | R → PC, 0 → T, (New Fetch) | $OE_R$, $LD_{PC}$, $CLR_T$, NF |


FLEX:

| State | S-term | RTN-beskrivning | Styrsignaler (=1) |
|---|---|---|---|
| $Q_5$ | $Q_5 \cdot I_{DF}$ | U = A, Flags → CC | $OE_A$, $f_0$, $LD_{CC}$ |
| $Q_6$ | $Q_6 \cdot I_{DF}$ | PC → MA, PC + 1 → PC | $OE_{PC}$, $LD_{MA}$, $Inc_{PC}$ |
| $Q_6$ | $Q_6 \cdot I_{DF} \cdot Z$ | If Z = 1: (Next Fetch) | NF |
| $Q_7$ | $Q_7 \cdot I_{DF}$ | M(MA) → T | MR, $LD_T$ |
| $Q_8$ | $Q_8 \cdot I_{DF}$ | PC + T → R | $OE_{PC}$, $f_3$, $f_1$, $LD_R$ |
| $Q_9$ | $Q_C \cdot I_{DF}$ | 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. Värdet $7A_{16} = 122_{10}$ 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: $\underline{122 < W \leq 250}$ **(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.
$\underline{T_{FLISP}} = 5+3+3+[2+(3+2+4)\cdot 10+4+4]\cdot 5+4+3+2 = 20+[10+9\cdot 10\ ]\cdot 5 = \underline{520 \text{ st } (\mu s)}$

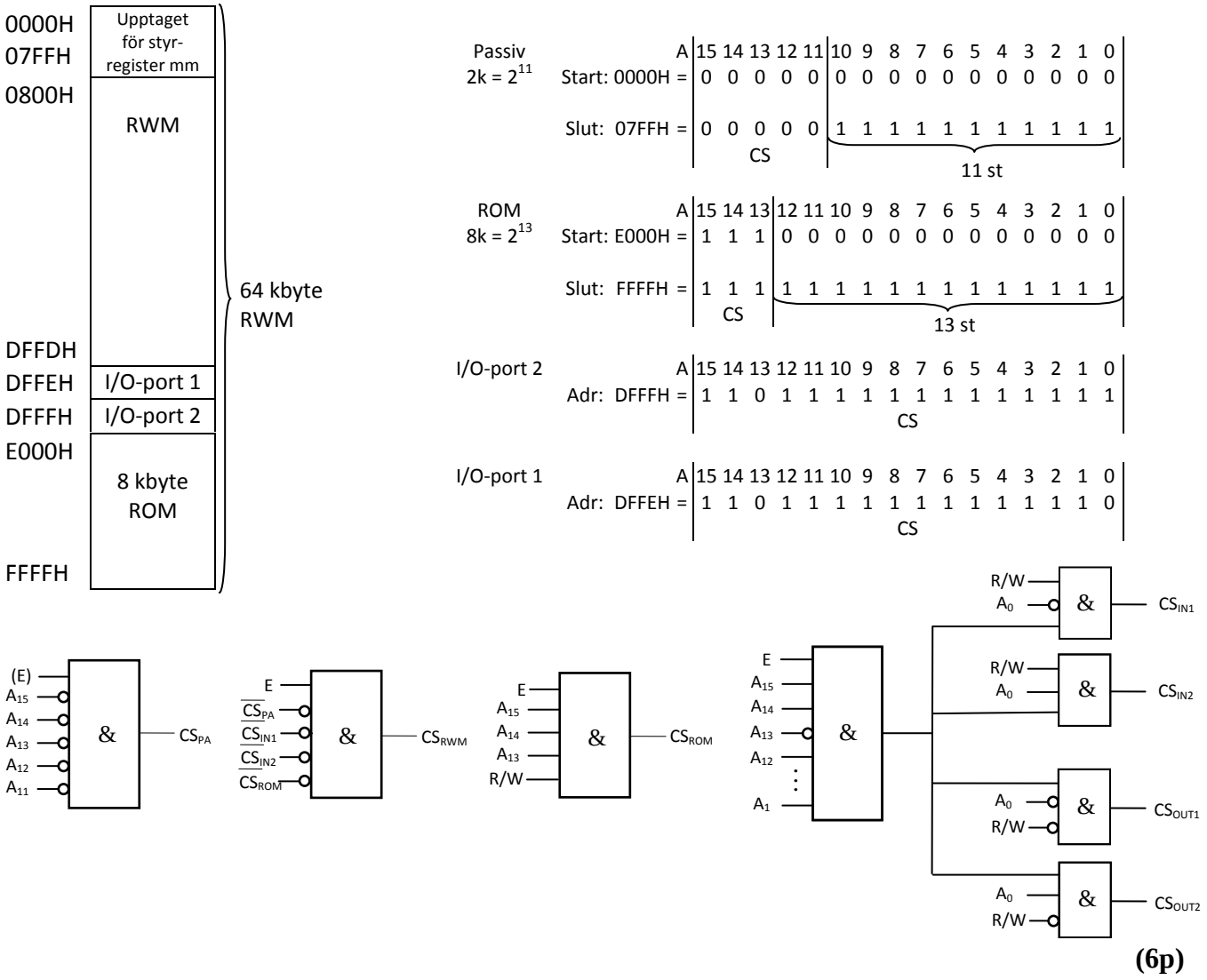
$\underline{T_{FLEX}} = 7+5+5+[4+(4+3+5)\cdot 10+5+5]\cdot 5+6+4+4 = 31+[14+12\cdot 10\ ]\cdot 5 = \underline{701 \text{ st } (\mu s)}$ **(3p)**

**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.**




**(6p)**