EDA321 Digitalteknik-fortsättning D2 och E3

Tentamen *(EDA321-0205)*

Måndagen den 8 mars 2010, fm i V-salarna

Examinator
Arne Linde, tel. 772 1683

Tillåtna hjälpmedel

Inga hjälpmedel tillåtna. Detta innefattar även kalkylatorer och alla tabellverk.

Oläsliga eller svårtydda lösningar ger poängavdrag.

Allmänt:

Ett lösningsblad får endast innehålla redovisningsdelar som hör ihop med en uppgift (dock flera deluppgifter).

För full poäng på de uppgifter som omfattar konstruktioner krävs förutom korrekt funktion även en optimal (minimal) eller nära optimal lösning.
Fungerande men onödigt komplicerade lösningar ger varierande poängavdrag beroende på hur mycket lösningen avviker från den optimala.
För samtliga uppgifter gäller, att ofullständiga lösningar eller lösningar innehållande felaktigheter ger poängavdrag även om resultatet är korrekt.

Betygsättning
För godkänt betyg fordras minst 20 p av totalt 50 p.
20p ≤ **betyg 3** < 30p ≤ **betyg 4** < 40p ≤ **betyg 5**

För godkänt slutbetyg på hela kursen fordras godkänt betyg på tentamen och dessutom fordras godkänd laborationskurs.

Lösningar

Anslås senast onsdag 10 mars, kl 16.00 på kursens hemsida.

Granskning
Av rättning kan göras
tisdag 13/4 kl 11.45 - 12.30 rum E-4464 och onsdag 14/4 kl 11.45-12.00 rum E-4464.

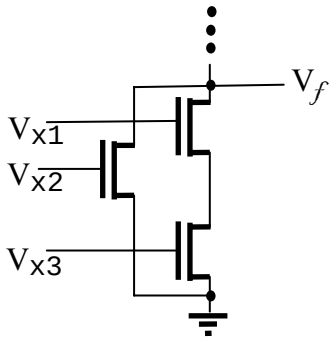
1. *Småfrågor (9p)*

a) Varför bör man inte ansluta en asynkron insignal till mer än en vippa? *(2p)*

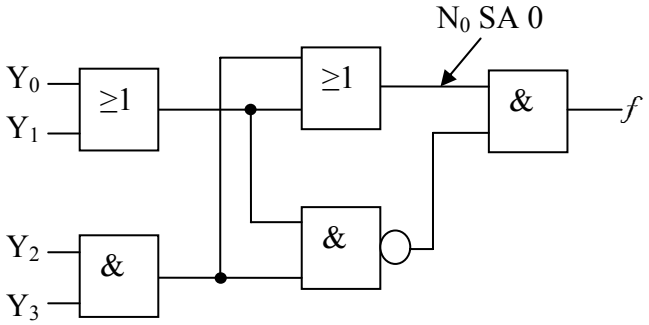
b) Betrakta VHDL- koden i rutan. Antag att X byter värde från 0 till 1. Vad händer på utgångarna U1-U3 och när sker detta i simulatorn. Svara med ett pulsdiagram med CLK, X, U1, U2 och U3. *(3p)*

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
ENTITY Uppg1c IS
PORT ( CLK, X : IN std_logic;
       U1, U2, U3: OUT  std_logic);
END Uppg1c;
ARCHITECTURE behav_U1c OF Uppg1c IS
SIGNAL vare:std_logic;
BEGIN
p1:PROCESS(Clk)
 BEGIN
  IF Clk'event AND Clk='1' THEN
    vare<=X;
    U1 <= vare;
  END IF;
  U2 <= vare;
 END PROCESS p1;
 U3 <= vare;
END behav_U1c;
```

c) Figuren nedan visar halva transistornätet för en CMOS krets. Rita den andra halvan som innehåller PMOS transistorerna. *(2p)*



d) Ta fram en testvektorer för N_0 låst till 0 utgående från nätet till höger. *(2p)*



2. *Minimering av Booleska funktioner. (8p)*

Betrakta funktionen nedan.

$f(x_3x_2x_1x_0) = m\,(2, 5, 7, 8, 12, 14) + d(3, 10, 13)$

a) Ta fram samtliga primimplikanter till funktionen ovan. *(3p)*

b) Ta fram en minimal disjunktiv form till funktionen. *(2p)*

c) Ta fram en minimal hasardfri implementering av funktionen med så få termer som möjligt. *(3p)*

3. *VHDL (10p)*
Betrakta VHDL- koden i rutan här intill och svara på följande frågor:

a) Vilken typ av tillståndsmaskin beskrivs? *(1p)*

b) Rita tillståndsgraf för tillstånds-maskinen. *(2p)*
För full poäng skall samtliga tillståndsövergångar finnas med.

c) Fyll i pulsdiagrammet som finns i bilaga A. *(2p)*

Utgå från VHDL-koden här intill. De fördröjningar som finns ska klart framgå.
(Glöm inte att bifoga Bilaga A när du lämnar in tesen!)

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
ENTITY Uppgift3a IS
PORT ( CLK, Reset   : IN    std_logic;
       X    : IN    std_logic_vector (1 DOWNTO 0);
       U    : OUT   std_logic_vector (1 DOWNTO 0));
END Uppgift3a;
ARCHITECTURE behav_Uppgift3a OF Uppgift3a IS
TYPE state_type IS (S0,S1,S2,S3);
SIGNAL state:state_type;
BEGIN
p1:PROCESS(Clk, Reset)
BEGIN
 IF Reset = '1' THEN state<=S0;
 ELSIF Clk'event AND Clk='1' THEN
  CASE state IS
    WHEN S0=> IF X="00" THEN state<=S1; U<="10";
                  ELSIF X="11" THEN state<=S2; U<="01";
                  ELSE U<="00";
                  END IF;
    WHEN S1=> IF X="11" THEN state<=S2; U<="01";
                  ELSIF X="00" THEN state<=S3; U<="10";
                  ELSE U<="11";
                  END IF;
    WHEN S2=> IF X="11" THEN state<=S1; U<="11";
                  ELSIF X="00" THEN state<=S3; U<="00";
                  ELSE U<="01";
                  END IF;
   WHEN S3=> IF X="11" THEN state<=S0; U<="10";
                  ELSIF X="00" THEN state<=S1; U<="10";
                  ELSE U<="01";
                  END IF;
    END CASE;
  END IF;
END PROCESS p1;
END behav_Uppgift3a;
```

d) Skriv i VHDL en uppräknare för 2 BCD-siffror.
Räknaren skall kunna nollställas av reset.
Räknaren skall gå till ett förutbestämt värde och därefter börja om från noll.

Räknevillkoret styrs av ingången en (count enable), som är aktiv hög.
Reset-ingången är aktiv hög och synkron.

Entiteten skall se ut så här

```
ENTITY bcdxx IS
   PORT(clk,reset,en:IN STD_LOGIC;
   maxc:IN STD_LOGIC_VECTOR(7 DOWNTO 0);
   ceo:OUT STD_LOGIC;
   count:OUT STD_LOGIC_VECTOR(7 DOWNTO 0));
END bcdxx;
```

Räknaren skall ge ut en puls på ceo av längden ett klockpulsintervall för var n+1:e klockpuls där n anges av maxc. n ligger i intervallet 2 – 99. *(5p)*

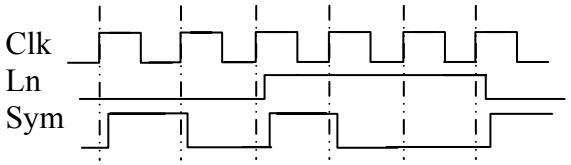
4. *Synkrona sekvensnät, oberoende deluppgifter. (12p)*
Vid implementering av en applikation används kodningen av symbolerna enligt tabell.
Din uppgift är att ta fram ett synkront sekvensnät som detekterar slutet på kordordet.

Symbol	Kodord
α	00
β	01
γ	110
δ	111
ε	10

Koden är prefixfri, vilket innebär, att inget kordord utgör början till något längre kodord. Tack vare denna egenskap är det möjligt att ur en lång kontinuerlig följd av seriellt överförda kodord urskilja de olika kodorden utan att behöva tillgripa kodordsskiljande tecken.

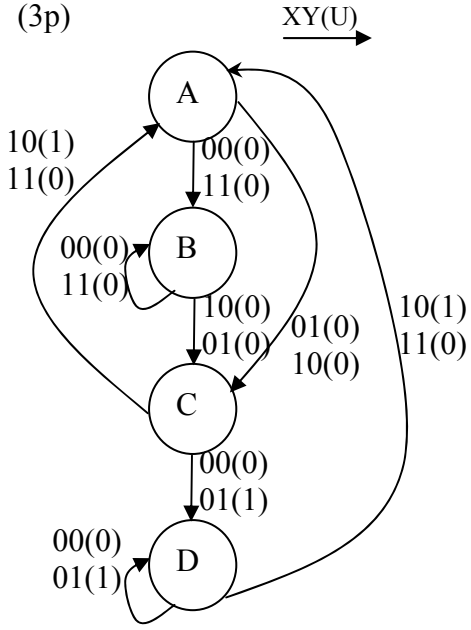
Kodorden anländer seriellt och är synkroniserade med systemklockan.
Utsignalen Sym som markerar att symbolen är slut skall finnas samtidigt som den sista biten i symbolen anländer.

Ex : In **0 0 0 1 1 1 0** 1 1 1 1 0
Sym **0 1 0 1 0 0 1** 0 0 1 0 1



a) Ta fram en tillståndsgraf för uppgiften ovan. Kravet är att utsignalen som markerar symbolslut är aktiv vid flanken som samplar kodordet. (3p)

b) Minimera tillståndsgrafen från a) (3p)



Lös 2 av uppgifterna c-e.

c) Utgå från sekvensnätet här intill och ta fram en ”snål” kodning baserat på tumregler. (3 av 6p)

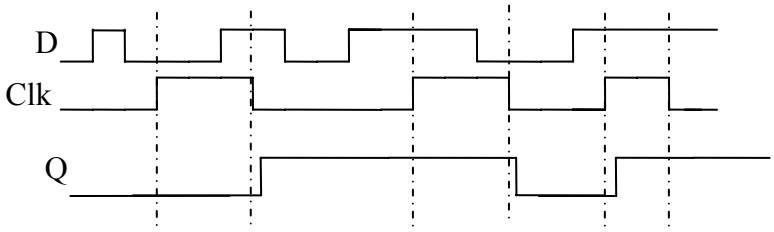
d) Ta fram en kodnig för sekvensnätet här intill enligt principen för ”One-hot” kodning. (3 av 6p)
Samt ta fram de booleska uttrycken för nästa tillstånd.

e) Implementera nätet i VHDL-kod. (3 av 6p)

OBS lös **ENDAST** 2 av uppgifterna c-e.
(Löses flera sker poängavdrag från maxpoängen, 6p)

5. *Asynkrona sekvensnät, oberoende deluppgifter (11p)*
En D-vippa som klockas på både positiv och negativ flank skall konstrueras.
• Två insignaler (D,Clk) och en utsignal Q.
• Insignalerna ändrar aldrig värde samtidigt.
• Utsignalen Q = 1 då D=1 och Clk ändras (på flanken, både positiv och negativ).
• Utsignalen Q = 0 då D=0 och Clk ändras (på flanken, både positiv och negativ).

Exempel:



a) Rita upp tillståndsgrafen för det asynkrona sekvensnätet. Nätet skall vara av Mealy-typ och behöver inte vara optimerat. (3p)

b) Beskriv vad en primitiv flödestabell är. (2p)

c) Minimera flödestabellen nedan. (4p)

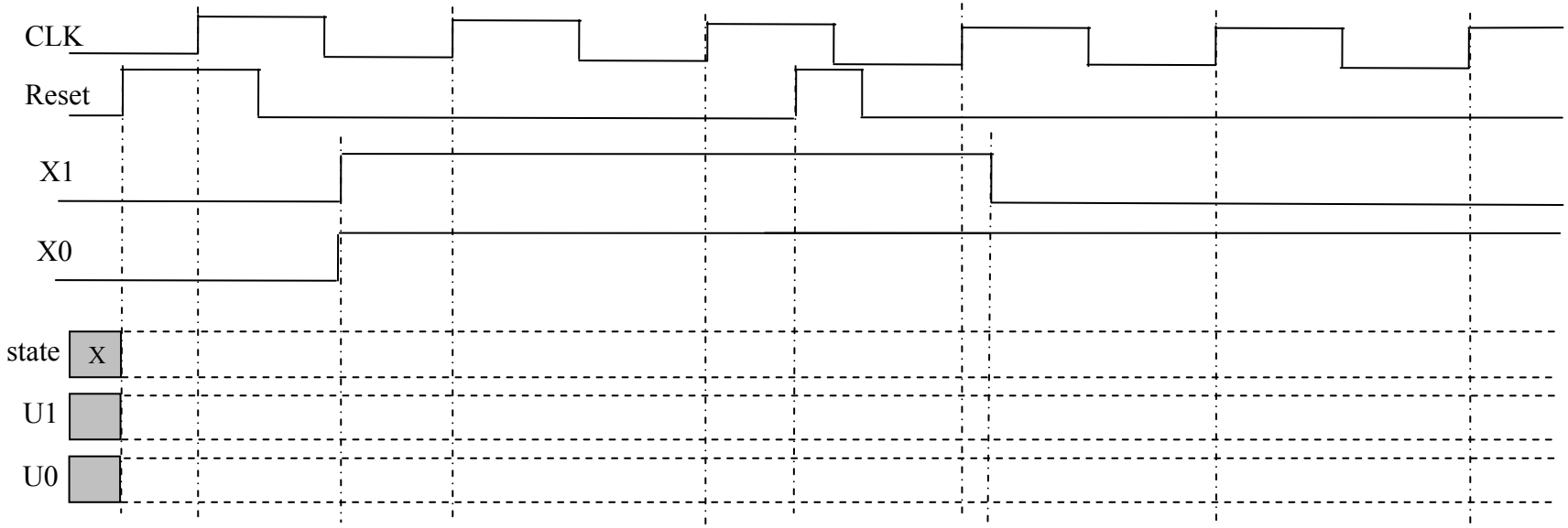
δ(λ)	00	01	11	10
A	**A(00)**	*B(00)*	-	*C(0-)*
B	*A(00)*	**B(00)**	*D(10)*	**B(11)**
C	*A(0-)*	**C(11)**	*D(1-)*	**C(01)**
D	-	*C(1-)*	**D(10)**	*B(1-)*

d) Utgå från flödestabellen nedan och ta fram en kapplöpningsfri kodning för tillstånden. Om flödestabellen behöver modifieras så skall det klart framgå. (3p)

δ(λ)	00	01	11	10
A	**A(11)**	**A(00)**	*C(--)*	*C(-1)*
B	*A(-1)*	**B(01)**	*C(-1)*	-
C	*D(--)*	*B(-1)*	**C(11)**	**C(01)**
D	**D(10)**	*A(-0)*	-	*C(--)*

Kod	Poäng	Sida

Bilaga A : till uppgift 3.

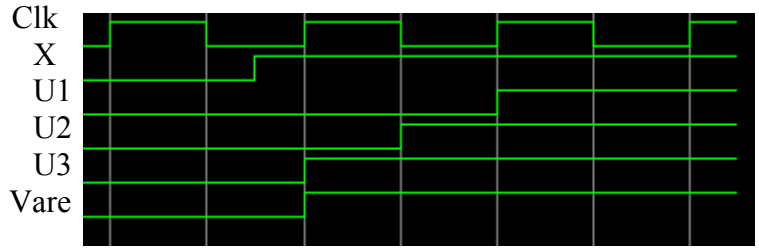


Riv här!

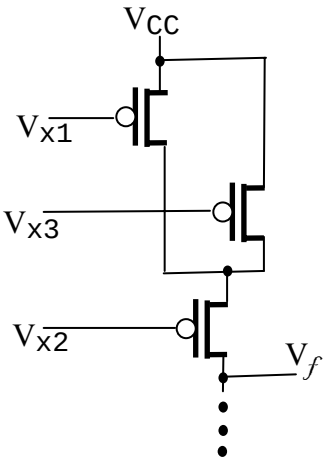
Lösningar:

1 a) Eftersom man inte kan garantera setupp och hålltider så kan vipporna få olika värden och i värsta fall hamna i metastabilt tillstånd.

b)

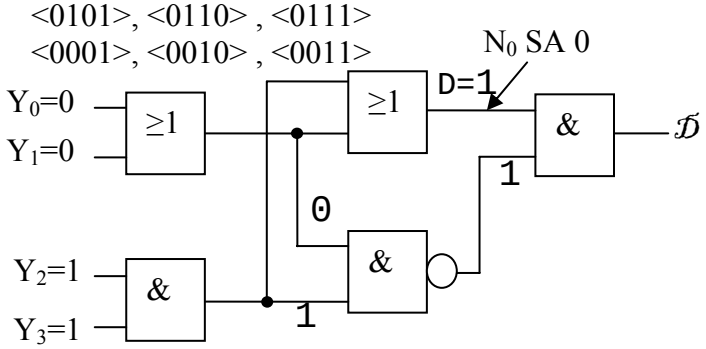


c) Se figur →



d) Testvektorer

<$y_3y_2y_1y_0$> → <1100> ,
<1010>, <1001>, <1011>,
<0101>, <0110> , <0111>
<0001>, <0010> , <0011>



2 a) $f(x_3x_2x_1x_0) = m(2_1, 5_2, 7_3, 8_1, 12_2, 14_3) + d(3_2, 10_2, 13_3)$

2	0010V	2, 3	001-	8,10,12,14 1--0
8	1000V	2,10	-010	
		8,10	10-0V	
3	0011V	8,12	1-00V	
5	0101V			
10	1010V	3, 7	0-11	
12	1100V	5, 7	01-1	
		5,13	-101	
7	0111V	12,13	110-	
13	1101V	10,14	1-10V	
14	1110V	12,14	11-0V	

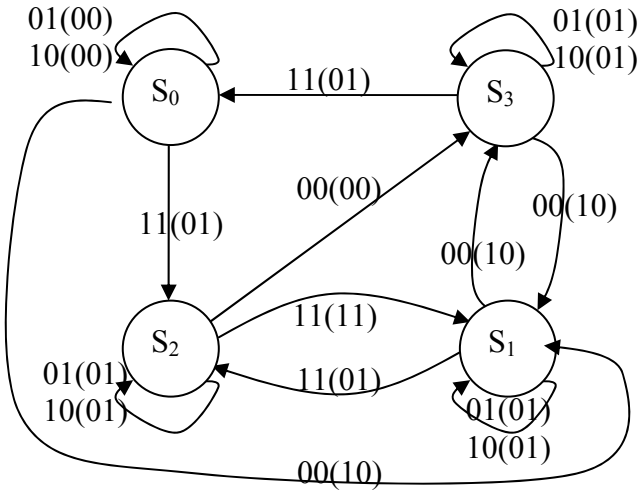
Primimplikatorer: $x_3'x_2'x_1$, $x_2'x_1x_0'$, $x_3'\,x_1\,x_0$, $x_3'x_2\,x_0$, $x_2x_1'x_0$, $x_3x_2x_1'$ och x_3x_0'

b) $f = x_3x_0' + x_3'x_2\,x_0 + x_3'x_2'x_1$

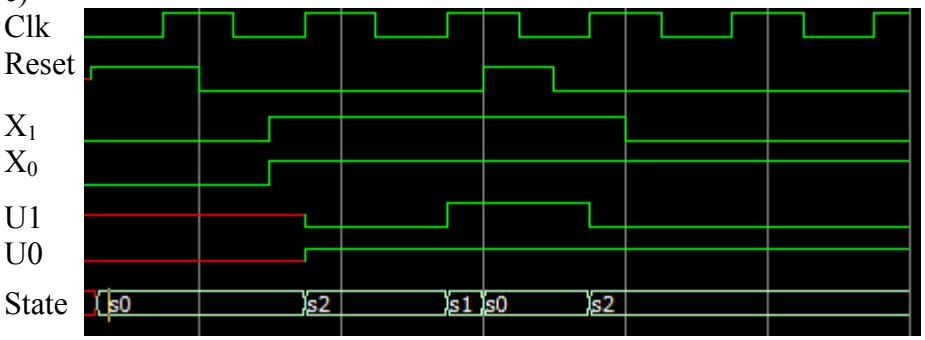
c) resultatet i b) är hasardfri eftersom hasarder endast uppstår mot ”don’t care”.

3 a) Synkron Mealy

b)



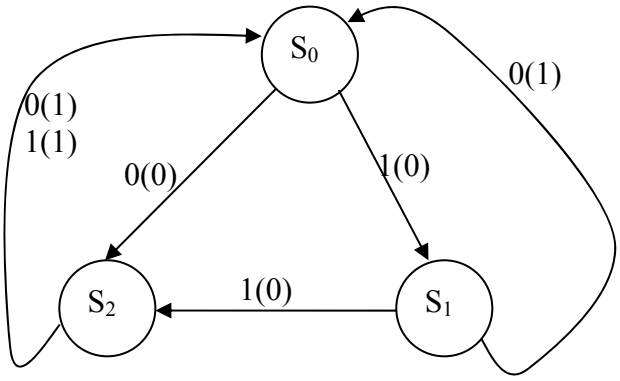
c)



d)

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_unsigned.all;
ENTITY Uppgift3d IS
PORT ( CLK, Reset, en   : IN    std_logic;
       maxc     : IN    std_logic_vector (7 DOWNTO 0);
       ceo: OUT std_logic;
       count     : OUT   std_logic_vector (7 DOWNTO 0));
END Uppgift3d;
```

```
ARCHITECTURE behav_Uppgift3d OF Uppgift3d IS
SIGNAL IntCnt:std_logic_vector (7 DOWNTO 0);
BEGIN
count <= IntCnt;
p1:PROCESS(Clk)
BEGIN
  IF Clk'event AND Clk='1' THEN
   ceo<='0';
   IF Reset = '1' THEN IntCnt<= "00000000";
   ELSIF en='1' THEN
     IF IntCnt = maxc then
       IntCnt<= "00000000";
       ceo<='1';
      ELSE
        IntCnt(3 DOWNTO 0)<=IntCnt(3 DOWNTO 0)+1;
        IF IntCnt(3 DOWNTO 0)= "1001" THEN
          IntCnt(3 DOWNTO 0)<="0000";
          IntCnt(7 DOWNTO 4)<=IntCnt(3 DOWNTO 0)+1;
          IF IntCnt(7 DOWNTO 4)= "1001" THEN
            IntCnt(7 DOWNTO 4)<="0000";
          END IF;
        END IF;
      END IF;
   END IF;
  END IF;
END PROCESS p1;
END behav_Uppgift3d;
```



4 a,b) se figur

c) Regel 1: A:2(C-D), B:2(A-B), C:2(A-B), D:2(C-D)
→ 4(A-B), 4(C-D)
Regel 2: A:(B-C), B:(B-C), C:(A-D)→2:(B-C), :(A-D)
Regel 2: Utsignal 0 ger inget, 1: 10(1):(C-D), 01(1):(C-D)
→ 2(C-D)
Skall uppfylla (A-B), (C-D) sedan (B-C) och i sista hand (A-D).
A B → A:00, B:01, D:10 och C:11, alla villkor uppfyllda.
D C

d)Kodar A:0001, B:0010, C:0100 och D:1000 där kodordet är $Q_3Q_2Q_1Q_0$.

$Q_0^+ = X(Q_3+Q_2)$

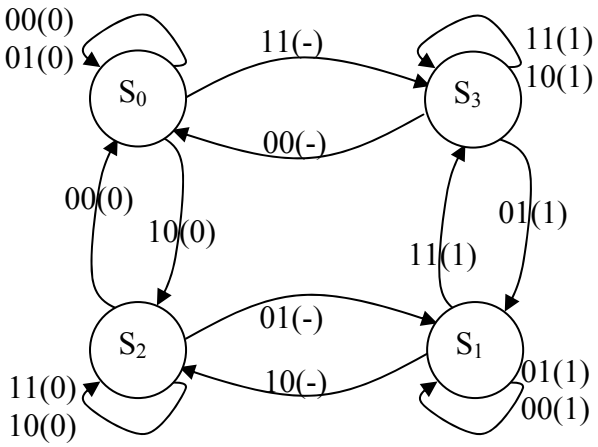
$Q_1^+ = (Q_1+Q_0)(XY+X'Y')$

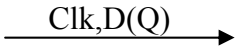
$Q_2^+ = (Q_1+Q_0)(X'Y+XY')$

$Q_2^+ = X'(Q_3+Q_2)$

e) -

5 a)

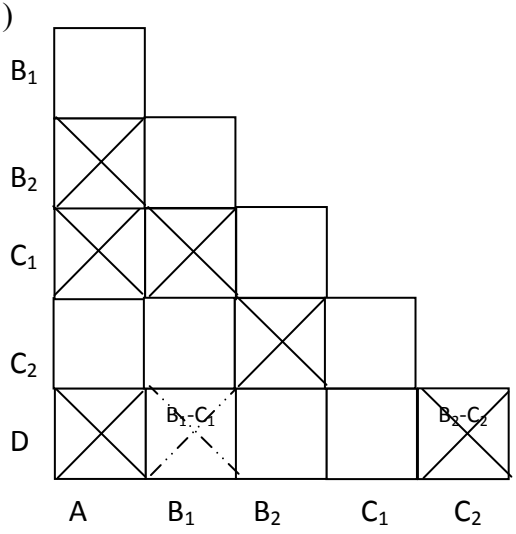




5.b)Vi börjar att göra om till primitiv flödestabell.

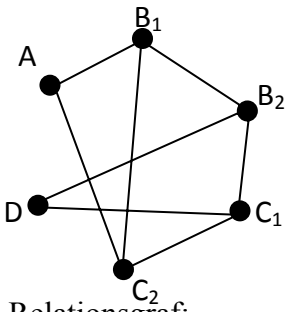
δ(λ)	00	01	11	10
A	**A(00)**	*B(00)*	-	*C(0-)*
B	*A(00)*	**B(00)**	*D(10)*	**B(11)**
C	*A(0-)*	**C(11)**	*D(1-)*	**C(01)**
D	-	*C(1-)*	**D(10)**	*B(1-)*

δ(λ)	00	01	11	10
A	**A(00)**	*B_1(00)*	-	*C_2(0-)*
B_1	*A(00)*	**B_1(00)**	*D(-0)*	-
B_2	*A(--)*	-	*D(1-)*	**B_2(11)**
C_1	*A(--)*	**C_1(11)**	*D(1-)*	-
C_2	*A(0-)*	-	*D(--)*	**C_2(01)**
D	-	*C_1(1-)*	**D(10)**	*B_2(1-)*



AαB α:{A, B_1, C_2} och β:{ B_2, C_1, D}

δ(λ)	00	01	11	10
α	**α(00)**	**α(00)**	*β(-0)*	**α(01)**
β	*α(--)*	**β(11)**	**β (10)**	**β (11)**



Relationsgraf:

5. c)

δ(λ)	00	01	11	10
A	**A(11)**	**A(00)**	*C(--)*	*C(-1)*
B	*A(-1)*	**B(01)**	*C(-1)*	-
C	*D(--)*	*B(-1)*	**C(11)**	**C(01)**
D	**D(10)**	*A(-0)*	-	*C(--)*

Fungerar dåligt!
Modifierar flödestabellen för att bli av med diagonalen.

δ(λ)	00	01	11	10
A	**A(11)**	**A(00)**	*B(0-)*	*D(11)*
B	*A(-1)*	**B(01)**	*C(-1)*	-
C	*D(--)*	*B(-1)*	**C(11)**	**C(01)**
D	**D(10)**	*A(-0)*	-	*C(--)*

Kodar sedan A:00, B:01, C:11 och D:10.