Lunds Universitet
LTH
Ingenjörshögskolan
IDA IEA
Helsingborg

Tentamensskrivning 9 juni 2015

EDI 610 **Digitala system** **15 poäng, varav tentamen 4,5 p**

Kursansvarig: Bernt-Arne Jönsson **Skrivtid 08.00-13.00**

Inga hjälpmedel
Obs! Räknare ej tillåten.

Skrivningen omfattar uppgifterna 1-8
Bilaga till skrivningen: Syntaxhjälp till VHDL

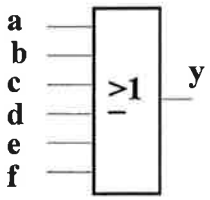
Maximalt antal poäng: 60 poäng

Krav för godkänt: 30 p

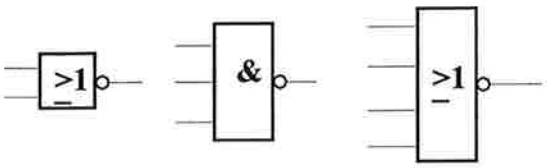
Ordentliga motiveringar skall lämnas.
Alla lösa blad skall vara samlade i omslagsarket.
Inlämnade uppgifter skall vara försedda med uppgiftens nummer.
Lösningarna skrivs in i nummerordning.
Skriv namn på varje ark
Omslagsarket skall vara fullständigt ifyllt med inskrivningsår, namn och personnummer
Kryssa för lösta uppgifter och ange antalet inlämnade blad.

1. Realisera en 6-ingångars ELLER – grind med en 2-ingångars NOR-grind, en 4-ingångars NOR-grind samt en 3-ingångars NAND-grind. Motivera med logiska uttryck din lösning.

Grind som skall realiseras:



Tillgängliga grindar:



(8p)

2. x=(x3, x2, x1, x0) är decimala siffror 0-9 i BCD-kod. Realisera en i en kombinationskrets funktionen z=3*x, där z=(z4, z3, z2, z1, z0) är heltal i vanlig binärkod. Exempel: om x= 0111, dvs. 7, så skall z= 10101 (dvs. . 21). Realisera z-funktionen i ett minimalt nät!(8p)

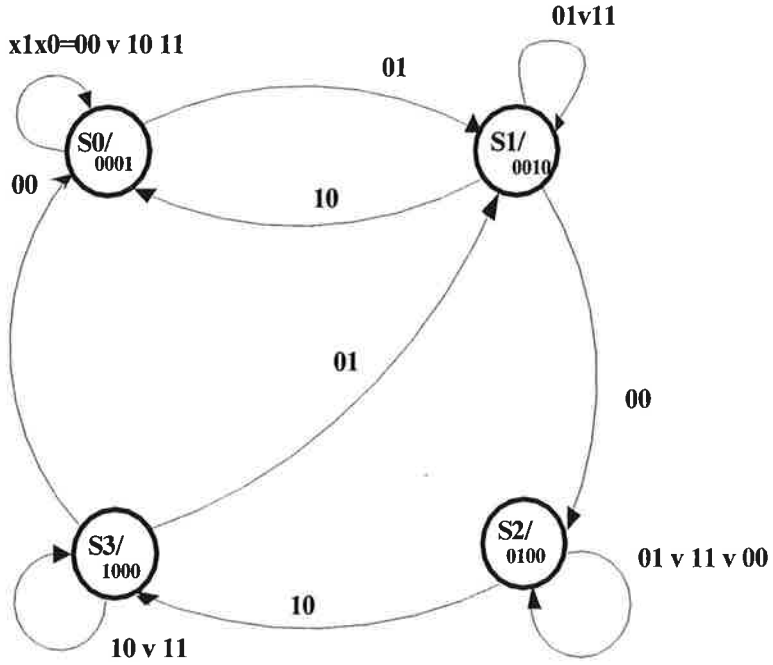
3. a) omvandla 89_{10} till basen 2.(1p)
b) omvandla 89_{10} till basen 16.(1p)
c) omvandla 89_{10} till basen 8.(1p)
d) omvandla 89_{10} till basen 4. (anm. Ovanlig bas) (1p).
e) negativa tal representeras ofta med 2- komplement. Skriv 2-komplementrepresentationen av 89. (där 89 är ett 8-bitars tal).
e) utför operationen 80-89 i binärkod, där negativa tal har tvåkomplementrepresentation. Talen är 8-bitarstal. Visa exakt hur du genomfört dina räkningar. Svara sen i decimalkod.(1p)
f) Antag att du har en processor som arbetar med 32-bitars heltal. Antag vidare att talen är med tecken (signed i programspråket C). Vilket är det största talet?(1p)
g) Det minsta talet?(1p)
OBS! Du kan gärna svara i uppgifterna f och g med tal på formen 2^N plus/minus något tal.(8p)

4. Betrakta nedanstående sanningstabell.

X4	X3	X2	X1	X0	f
0	0	0	0	0	1
0	0	0	0	1	1
0	0	0	1	0	1
0	0	0	1	1	1
0	1	0	0	0	1
0	1	0	1	0	1
1	0	0	0	1	1
1	0	0	1	1	1

I övriga kombinationer är f=0.
X4-X0 är insignaler till ett kombinatoriskt nät och f är utsignalen.
a) Tag fram det minimala sp-nätet (dvs. ett nät som avslutas med en eller- grind)
b) Tag fram det minimala ps-nätet (dvs. ett nät som avslutas med en och- grind).
Obs! valfri metod att ta fram näten, men du ska givetvis redovisa hur du gjort.
Rita näten.(7p)

5.

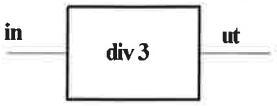


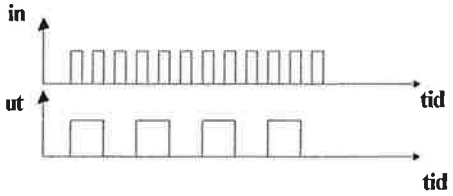
Realisera ovanstående sekvensnät med d-vippor. Utsignalen u är One –hot (se figuren ovan).Insignalen är x_1x_0.(7p)

6. Realisera sekvensnätet i uppg 4 i VHDL. Komplettera med styrsignalerna Enable och Reset, båda aktivt höga. Dvs. då Enable=1 så har vi funktionen från uppg 4, och om Enable=0 , så är tillståndet oförändrat. Om Reset=1 så blir nästa tillstånd S0. Enable dominerar över Reset, så om Enable=0, så har Reset ingen verkan.
Anm: en liknande fil finns i slutet på tentamen.(7p)

7.

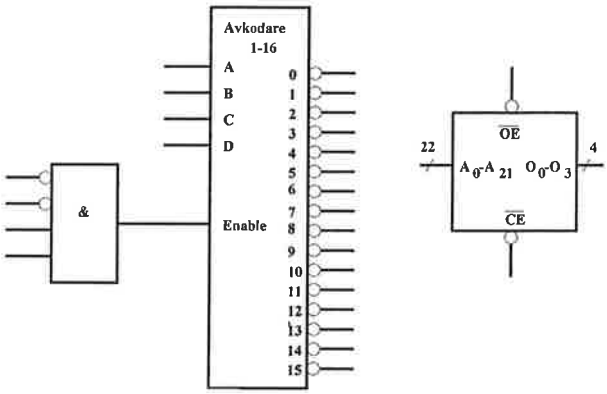
Ovan ser du ett asynkront nät som delar frekvensen med 3. Konstruera nätet med





standardgrindar. (7p)

8.



I ett minnessystem ingår bl.a ett läsminne. Minnet skall byggas upp med minneskapslar (4Mx4), se symbolen ovan. Minnessystemet adresseras med 28 adressbitar A0- A27, sålunda omfattande adressområdet 0000000 – FFFFFFF. Läsminnet skall ha en kapacitet om 20 Mord á 8 bitar och vara placerat med lägsta adressen C000000. Chipenable skall generas av avkodaren., till vilket skall anslutas adressbitarna A22-A27. Avkodarens Enable är aktivt hög. Rita blockschema för läsminnet och använd symbolerna ovan! Ange på avkodarens samtliga utgångar inom vilket adressområde den ger chipenable(8p)

```
1   -- enkel modula 6 räknare med upp/nedfunktion
2   library IEEE;
3   use IEEE.std_logic_1164.all;
4
5   entity mod6 is
6     port(upp, clock:in std_logic;
7         q: out std_logic_vector(2 downto 0));
8   end entity mod6;
9
10  architecture beteende of mod6 is
11  type state_type is (s0,s1,s2,s3 ,s4, s5);
12  signal present_state, next_state:state_type;
13  begin
14     process(present_state, upp)
15     begin
16         case present_state is
17              when s0 =>  if upp='1' then
18                          next_state<=s1;
19                     else
20                          next_state<=s5;
21                     end if;
22           when s1  => if upp='1' then
23                          next_state<=s2;
24                     else
25                          next_state<=s0;
26                     end if;
27           when s2  => if upp='1' then
28                          next_state<=s3;
29                     else
30                          next_state<=s1;
31                     end if;
32           when s3  => if upp='1' then
33                          next_state<=s4;
34                     else
35                          next_state<=s2;
36                     end if;
37           when s4  => if upp='1'  then
38                          next_state<=s5;
39                     else
40                          next_state<=s3;
41                     end if;
42           when s5  => if upp='1' then
43                          next_state<=s0;
44                     else
45                          next_state<=s4;
46                     end if;
47        end case;
48     end process;
49
50     process(present_state)
51     begin
52        case (present_state) is
53           when s0  => q<="000";
54           when s1  => q<="001";
55           when s2  => q<="010";
56           when s3  => q<="011";
```

```
57              when s4  => q<="100";
58              when s5  => q<="101";
59           end case;
60        end process;
61
62        process(clock)
63        begin
64           if rising_edge(clock) then
65              present_state<=next_state;
66           end if;
67        end process;
68     end architecture beteende;
69
70
```