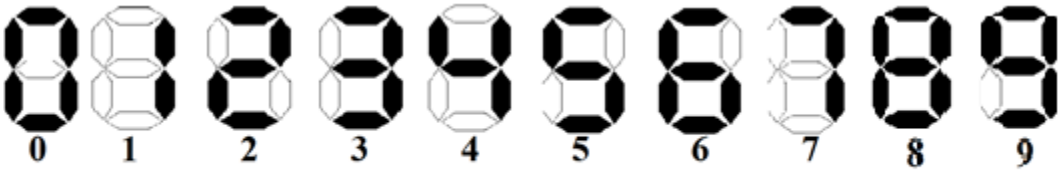
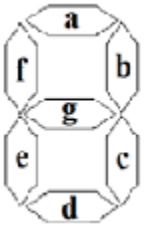
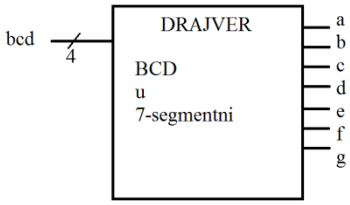
Projektovanje pomoću računara – TERMIN 7

Realizacija drajvera sedmosegmentnog displeja.
ZADATAK 1. Napisati entitet i arhitekturu kola za drajvovanje sedmosegmentnog displeja sa zajedničkom katodom. Na displeju treba ispisivati cifre od 0 od 9.







REŠENJE:
Potrebno je realizovati dekoder koji će u zavisnosti koju kombinaciju dovedemo na ulaz, na izlazu da aktivira određene segmente displeja kako bi ispisao cifru. Ovo je sličan zatadat kao iz ORT-a, samo što sada ovde opisujemo hardver, a kombinaciona mreža se sama generiše. Pošto je displej sa zajedničkom katodom, potrebno je da dovedemo +5V (logičku jedinicu) na anodu da bi zasvetleo neki segment.

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.std_logic_unsigned.all;

entity Drajver7Seg is
		port(
			bcd: in STD_LOGIC_VECTOR(3 downto 0);
			a: out STD_LOGIC;
			b: out STD_LOGIC;
			c: out STD_LOGIC;
			d: out STD_LOGIC;
			e: out STD_LOGIC;
			f: out STD_LOGIC;
			g: out STD_LOGIC
			);
end entity Drajver7Seg;

architecture Drajver_Arh of Drajver7Seg is
signal pa, pb, pc, pd, pe, pf, pg, oe: std_logic;
begin
	bcd2seg7: process(bcd)
	begin
```

```
        --inicijalizovanje vrednosti, ugašen displej
    pa<=’0’;  pb<=’0’; pc<=’0’; pd<=’0’; pe<=’0’; pf<=’0’; pg<=’0’;
    case bcd is
        when “0000” => pa<=’1’;  pb<=’1’; pc<=’1’; pd<=’1’; pe<=’1’; pf<=’1’;
        when “0001” => pb<=’1’; pc<=’1’;
        when “0010” => pa<=’1’;  pb<=’1’; pd<=’1’; pe<=’1’; pg<=’1’;
        when “0011” => pa<=’1’;  pb<=’1’; pc<=’1’; pd<=’1’;  pg<=’1’;
        when “0100” => pb<=’1’; pc<=’1’; pf<=’1’; pg<=’1’;
        when “0101” => pa<=’1’;  pc<=’1’; pd<=’1’; pf<=’1’; pg<=’1’;
        when “0110” => pa<=’1’;  pc<=’1’; pd<=’1’; pe<=’1’; pf<=’1’; pg<=’1’;
        when “0111” => pa<=’1’;  pb<=’1’; pc<=’1’;
        when “1000” => pa<=’1’;  pb<=’1’; pc<=’1’; pd<=’1’; pe<=’1’; pf<=’1’; pg<=’1’;
        when “1001” => pa<=’1’;  pb<=’1’; pc<=’1’; pd<=’1’; pf<=’1’; pg<=’1’;
        --ostali slucajevi nisu bitni pa dodeljujemo don’t care (X)
        when others => pa<=’X’;  pb<=’X’; pc<=’X’; pd<=’X’; pe<=’X’; pf<=’X’; pg<=’X’;
    end case;
end process bcd2seg7;
--potrebno je da sprecimo da izlaz dobije nedefinisano stanje za nelegalne kombinacije ulaza
oe<='1' when (bcd<10) else '0';  -- da bi se primenilo ispitivanje <10 poziva se paket unsigned.all
a<=pa when oe='1' else 'Z';  --Z je stanje visko impedanse
b<=pb when oe='1' else 'Z';
c<=pc when oe='1' else 'Z';
d<=pd when oe='1' else 'Z';
e<=pe when oe='1' else 'Z';
f<=pf when oe='1' else 'Z';
g<=pg when oe='1' else 'Z';
end entity;
```

DOMAĆI ZADATAK. Odraditi isti zadatak samo sa drugim displejom koji je sa zajedničkom anodom (+ je zajednički za sve segmente, pa da bi segment svetleo treba dovesti logičku nulu na katodu).

KORIŠĆENJE ZAJEDNIČKIH RESURSA

ZADATAK 1. Treba opisati kolo u kome rezultat zbira signala na ulazima *a* i *b* kontroliše stanje na izlazu tako da *je izlaz*=*c*, *d* ili *e*, ukoliko je *a*+*b*=1, 2 ili 3, respektivno.

Jedan od mogućih VHDL opisa je:

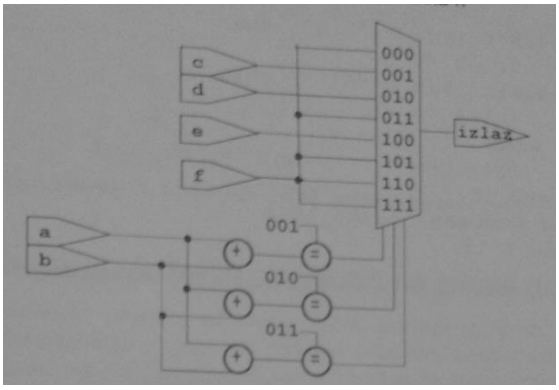
LOŠE REŠENJE:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.std_logic_unsigned.all;

entity Primer6_14 is
 port(
    a : in STD_LOGIC_VECTOR(2 downto 0);
    b : in STD_LOGIC_VECTOR(2 downto 0);
    c : in STD_LOGIC_VECTOR(2 downto 0);
    d : in STD_LOGIC_VECTOR(2 downto 0);
    e : in STD_LOGIC_VECTOR(2 downto 0);
    f : in STD_LOGIC_VECTOR(2 downto 0);
    izlaz : out STD_LOGIC_VECTOR(2 downto 0)
    );
end Primer6_14;

architecture Primer6_14 of Primer6_14 is
  begin
    demo6_10: process (a, b, c, d, e, f)
    begin
    if (a + b = "001") then
      izlaz <= c;
    elsif (a + b = "010") then
      izlaz <= d;
    elsif (a + b = "011") then
      izlaz <= e;
    else
      izlaz <= f;
    end if;
    end process  demo6_10;
end Primer6_14;
```

Gornji VHDL kod uz maksimalnu optimizaciju bi mogao da rezultira hradverom prikazanim na sledećoj slici:



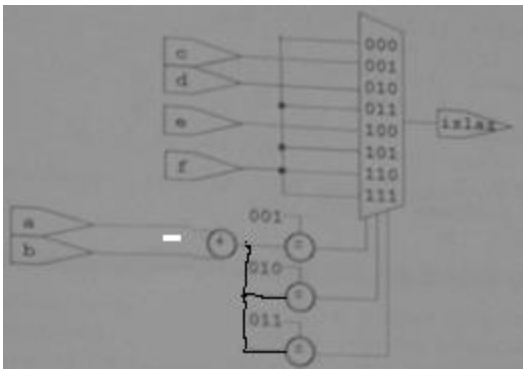
Isti zadatak možemo da odradimo na bolji način korišćenjem zajedničkih resursa:
DOBRO REŠENJE:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.std_logic_unsigned.all;

entity Primer6_15 is
  port(
    a : in STD_LOGIC_VECTOR(2 downto 0);
    b : in STD_LOGIC_VECTOR(2 downto 0);
    c : in STD_LOGIC_VECTOR(2 downto 0);
    d : in STD_LOGIC_VECTOR(2 downto 0);
    e : in STD_LOGIC_VECTOR(2 downto 0);
    f : in STD_LOGIC_VECTOR(2 downto 0);
    izlaz : out STD_LOGIC_VECTOR(2 downto 0)
     );
end Primer6_15;

architecture Primer6_15 of Primer6_15 is
 begin
   demo6_10a: process (a, b, c, d, e, f)
    variable sum: std_logic_vector (2 downto 0);
    begin
      sum := a + b;
      if (sum = "001") then
        izlaz <= c;
      elsif (sum = "010") then
        izlaz <= d;
      elsif (sum = "011") then
        izlaz <= e;
       else
        izlaz <= f;
      end if;
    end process  demo6_10a;
end Primer6_15;
```

Hardver iz gornjeg primera bi izgledao kao na narednoj slici:


c
d
e
f
000
001
010
011
100
101
110
111
ixlat
a
b
001
010
011