

# Projekat iz predmeta Projektovnje pomoću računara - Hijerarhijsko projektovanje

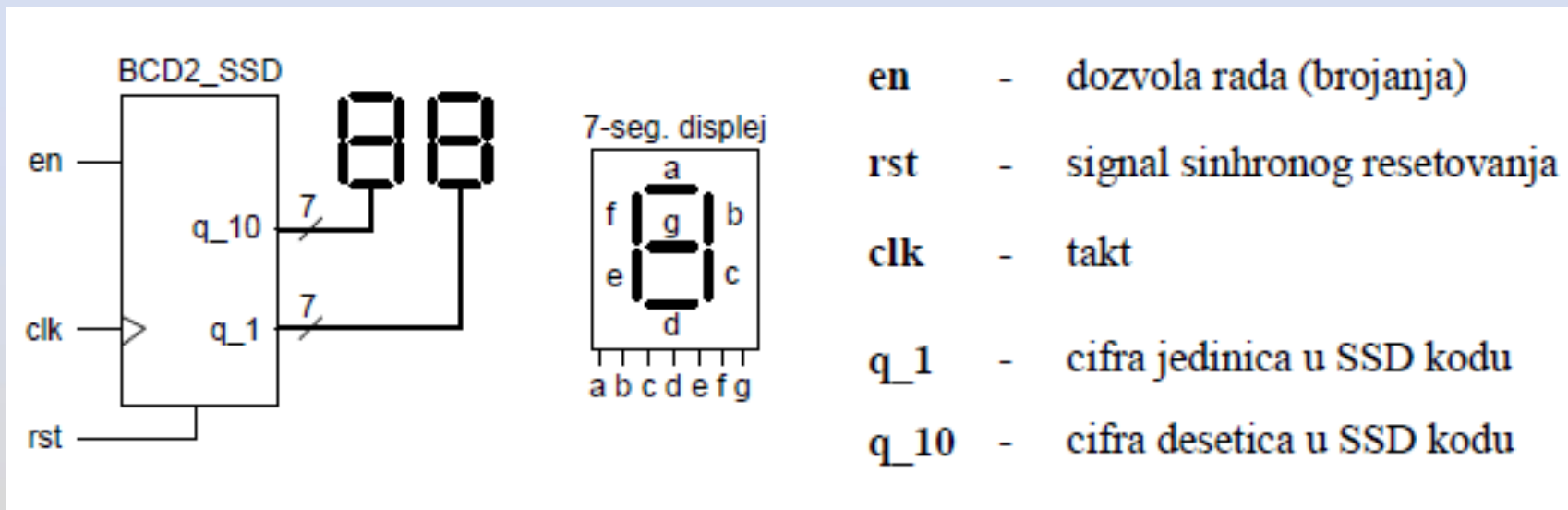
Profesor: dr Miloš Stojanović

Studenti: Mančić Milorad REr 48/13  
Petrović Aleksandar REr 36/13

# Projekovanje pomoću računara

## laboratorijska vežba 7

- Kreirati strukturni VHDL opis dvocifarskog BCD brojača sa SSD izlazom, (SSD izlaz – izlaz za pobudu sedmosegmentnih displeja).





# Izbor integrisanog kola

Biramo konkretno kolo za koje će biti realizovan projekat!

Ostalo ostaviti kako jeste.

New Project Wizard

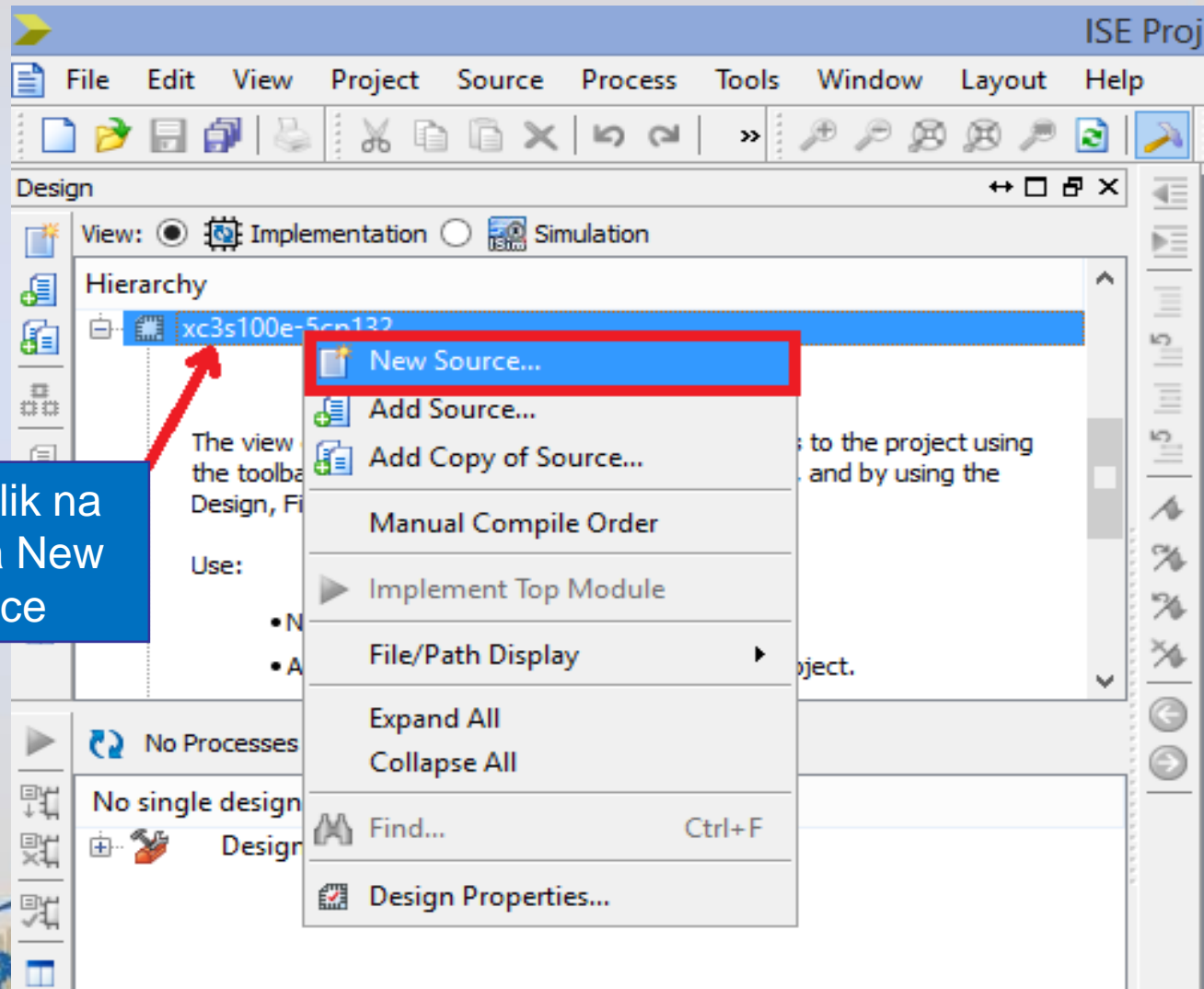
Project Settings  
Specify device and project properties.

Select the device and design flow for the project

Property Name	Value
Evaluation Development Board	None Specified
Product Category	All
Family	Spartan3E
Device	XC3S100E
Package	CP132
Speed	-5
Top-Level Source Type	HDL
Synthesis Tool	XST (VHDL/Verilog)
Simulator	ISim (VHDL/Verilog)
Preferred Language	VHDL
Property Specification in Project File	Store all values
Manual Compile Order	<input type="checkbox"/>
VHDL Source Analysis Standard	VHDL-93
Enable Message Filtering	<input type="checkbox"/>

More Info      < Back      Next >      Cancel

# Kreiranje paketa



Desni klik na kolo, pa New Source

# Kreiranje paketa

Biramo VHDL  
Package

New Source Wizard

← Select Source Type  
Select source type, file name and its location.

- IP (CORE Generator & Architecture Wizard)
- Schematic
- User Document
- Verilog Module
- Verilog Test Fixture
- VHDL Module
- VHDL Library
- VHDL Package**
- VHDL Test Bench
- Embedded Processor

Naziv paketa

File name: BCD2ssdPackage

Location: D:\BCD2ssd

Add to project

More Info Next > Cancel

# Kreiranje paketa

Informacije o  
paketu

New Source Wizard

Summary

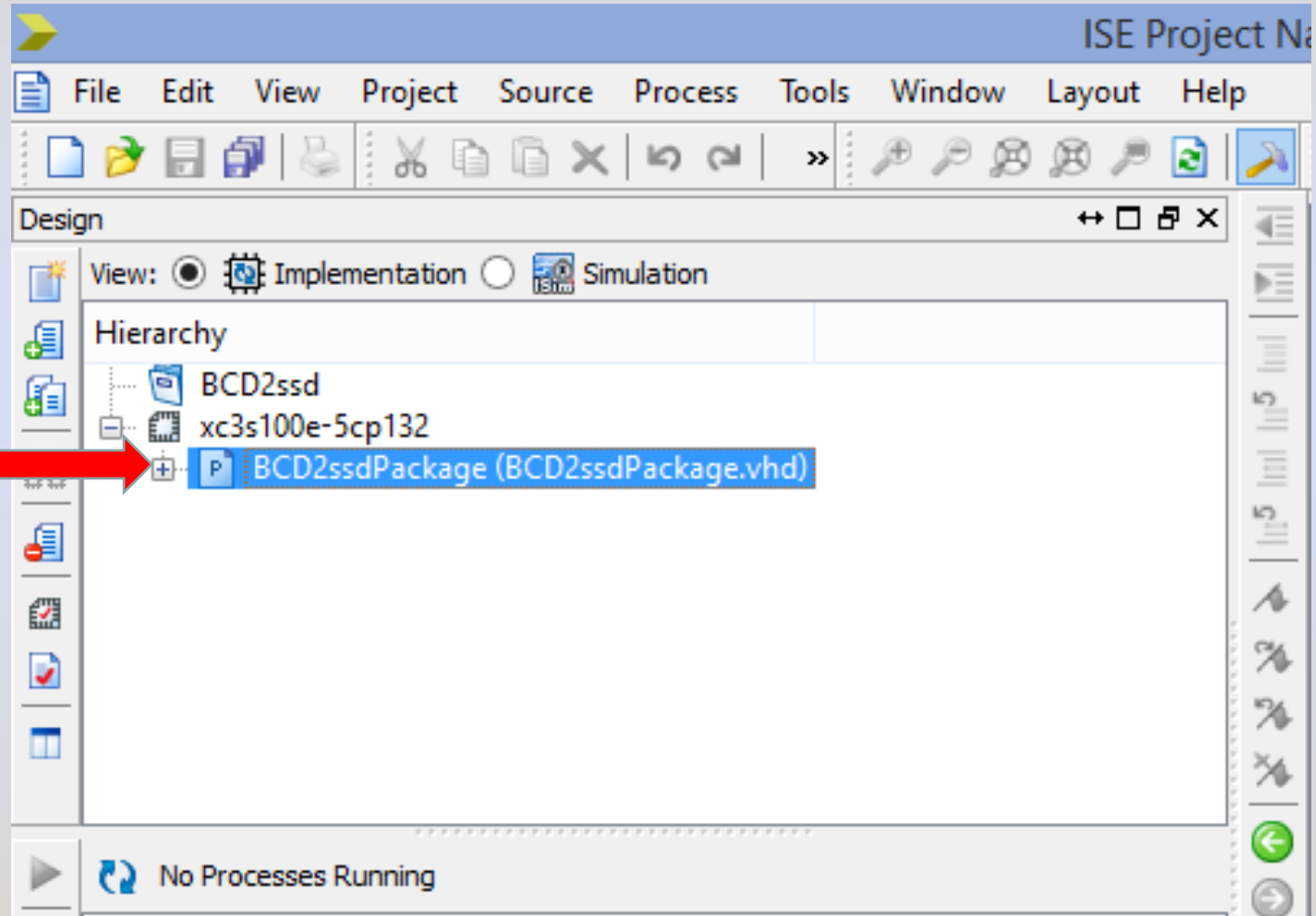
Project Navigator will create a new skeleton source with the following specifications.

Add to Project: Yes  
Source Directory: D:\BCD2ssd  
Source Type: VHDL Package  
Source Name: BCD2ssdPackage.vhd

More Info < Back Finish Cancel

Klik na finish za  
zavrsetak kreiranja  
paketa.

# Realizacija paketa



Dvo-klik na  
kreirani paket



# Realizacija paketa

Ime paketa

Deklaracija  
komponenti

Kod paketa

Kraj paketa

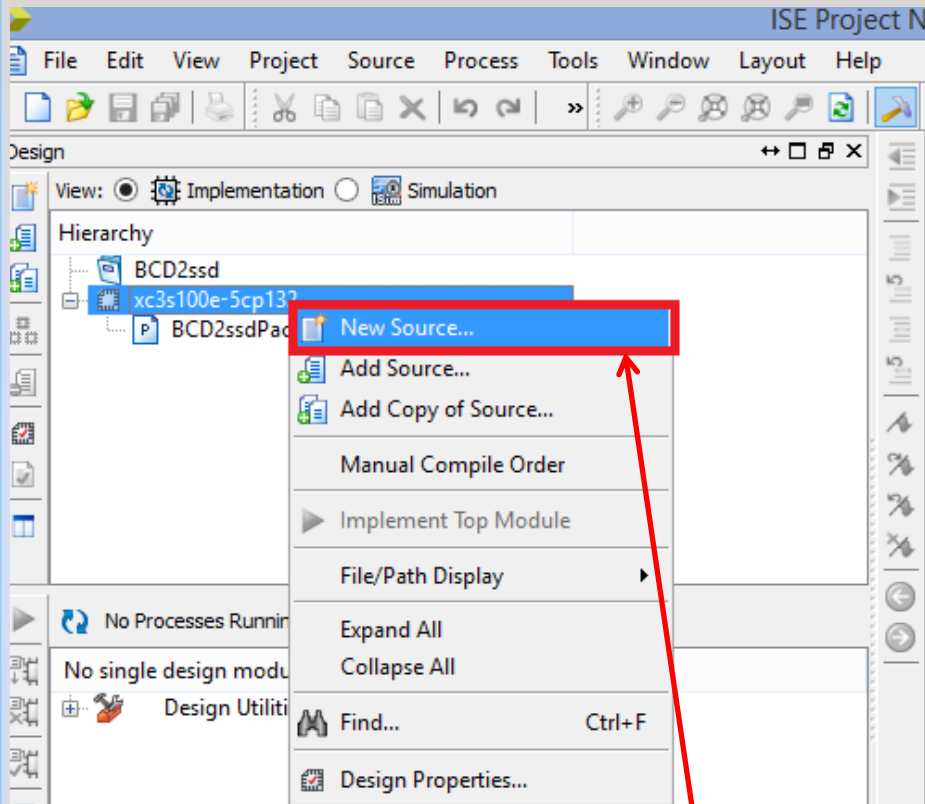
```
1  LIBRARY IEEE;
2  USE IEEE.STD_LOGIC_1164.all;
3
4  PACKAGE BCD2ssdPackage IS
5
6  COMPONENT BCD_count IS
7      PORT (
8          en : IN STD_LOGIC;
9          cout : OUT STD_LOGIC;
10         rst : IN STD_LOGIC;
11         clk : IN STD_LOGIC;
12         q : OUT STD_LOGIC_VECTOR(3 DOWNTO 0)
13     );
14 END COMPONENT;
15 COMPONENT BCD_to_SSD IS
16     PORT (
17         BCD : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
18         SSD : OUT STD_LOGIC_VECTOR(6 DOWNTO 0));
19 END COMPONENT;
20 COMPONENT BCD_SSD IS
21     PORT(en : IN STD_LOGIC;
22         cout : OUT STD_LOGIC;
23         rst : IN STD_LOGIC;
24         clk : IN STD_LOGIC;
25         ssd : OUT STD_LOGIC_VECTOR(6 DOWNTO 0));
26 END COMPONENT;
27
28 END BCD2ssdPackage;
```

# Karakteristike paketa

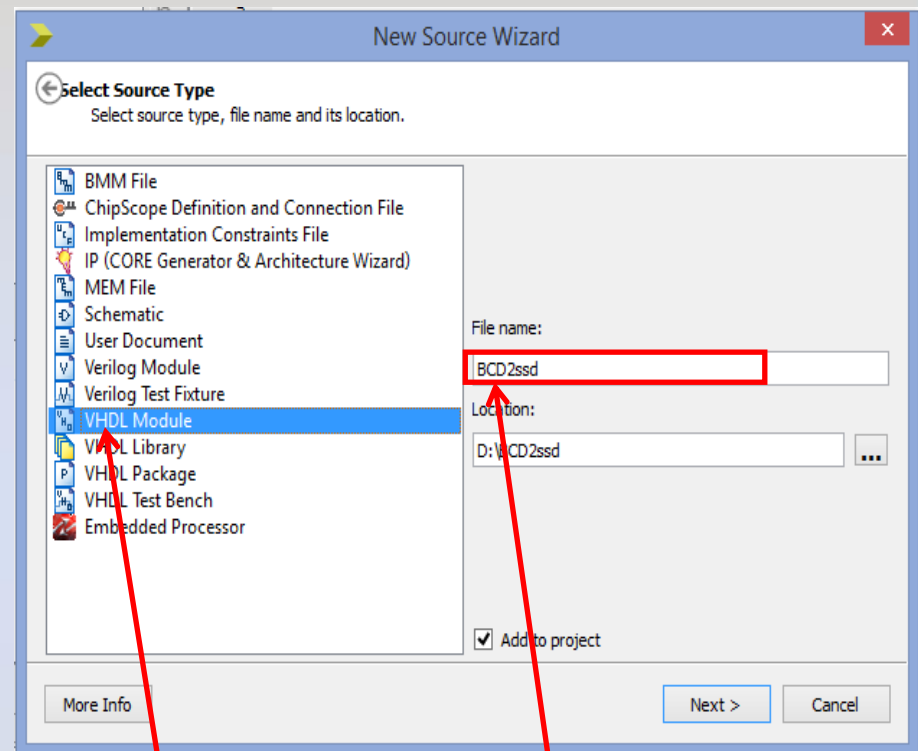
- U paketu se vrši deklaracija komponenti koje će se koristiti u projektu. U našem slučaju komponente: Brojač(BCD\_count), konvertor(BCD\_to\_SSD), sedmosegmentni displej za jednu cifru(BCD\_SSD).
- Kad se uvede paket, nadalje u projektu nije potrebno deklarirati komponente već samo uključiti paket u projekat, naredbom:
- **USE BCD2ssdPackage.ALL;**



# Uvođenje vršnog VHDL modula



1.  
Odaberemo New  
Source

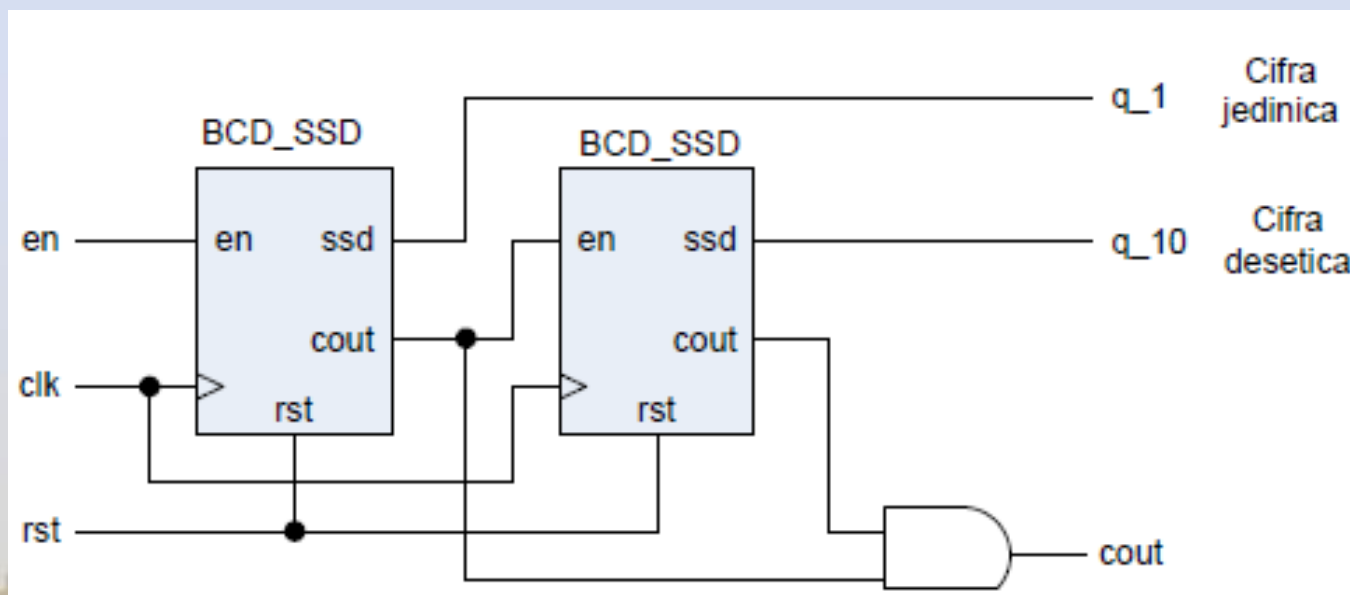


2.  
Zatim, VHDL  
Module

3.  
Nazovemo  
Modul BCD2ssd

# Modul *BCD2ssd*

- Ovo je vršni VHDL modul. Koristi komponentu *BCD\_SSD* koja se dva puta instancira. AND kolo se realizuje konkurentnom naredbom dodele. U istoj arhitekturi je dozvoljeno kombinovati različite stilove opisivanja, strukturni, konkurentni, sekvencijalni.



# Kod za vršni modul

Uključivanje paketa,  
USE *work.BCD2ssdPackage.ALL*;

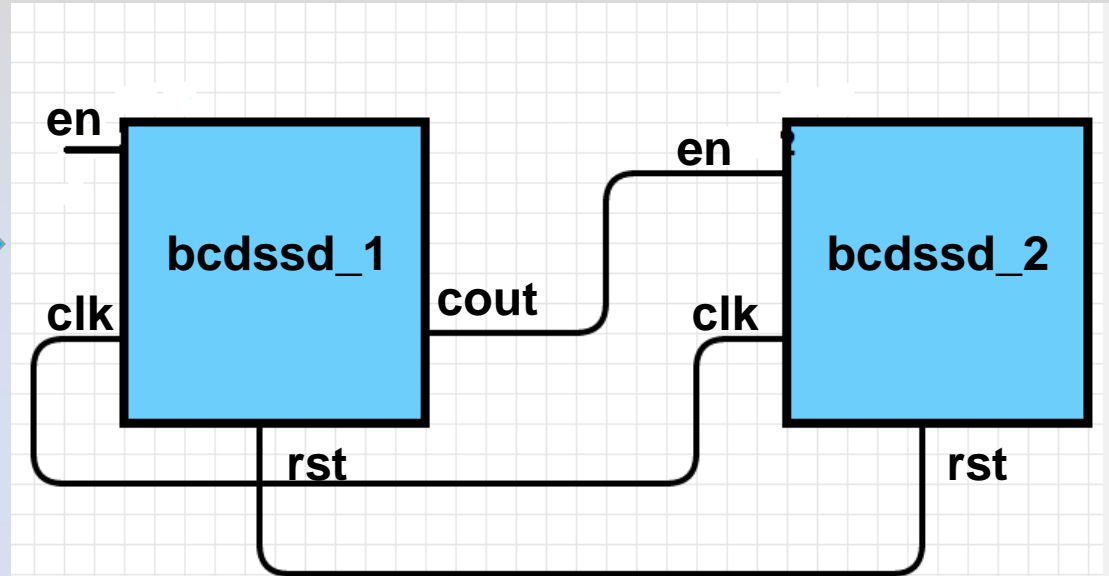
Entitet

Arhitektura,  
u arhitekturi se  
vrši kreiranje i  
povezivanje  
komponenata.

```
1  LIBRARY IEEE;
2  USE IEEE.STD_LOGIC_1164.ALL;
3
4  USE work.BCD2ssdPackage.ALL;
5
6  ENTITY BCD2ssd IS
7      Port ( en : in  STD_LOGIC;
8            rst : in  STD_LOGIC;
9            clk : in  STD_LOGIC;
10           cout : out STD_LOGIC;
11           q_1  : out STD_LOGIC_VECTOR (6 downto 0);
12           q_10 : out STD_LOGIC_VECTOR (6 downto 0));
13  end BCD2ssd;
14
15  ARCHITECTURE BCD2ssd of BCD2ssd is
16      SIGNAL cout_1, cout_10 : STD_LOGIC;
17      BEGIN
18          bcdssd_1: BCD_SSD
19              PORT MAP(en => en,
20                    cout => cout_1,
21                    rst => rst,
22                    clk => clk,
23                    ssd => q_1);
24          bcdssd_2: BCD_SSD
25              PORT MAP(en => cout_1,
26                    cout => cout_10,
27                    rst => rst,
28                    clk => clk,
29                    ssd => q_10);
30          cout <= cout_1 AND cout_10;
31  END BCD2ssd;
```

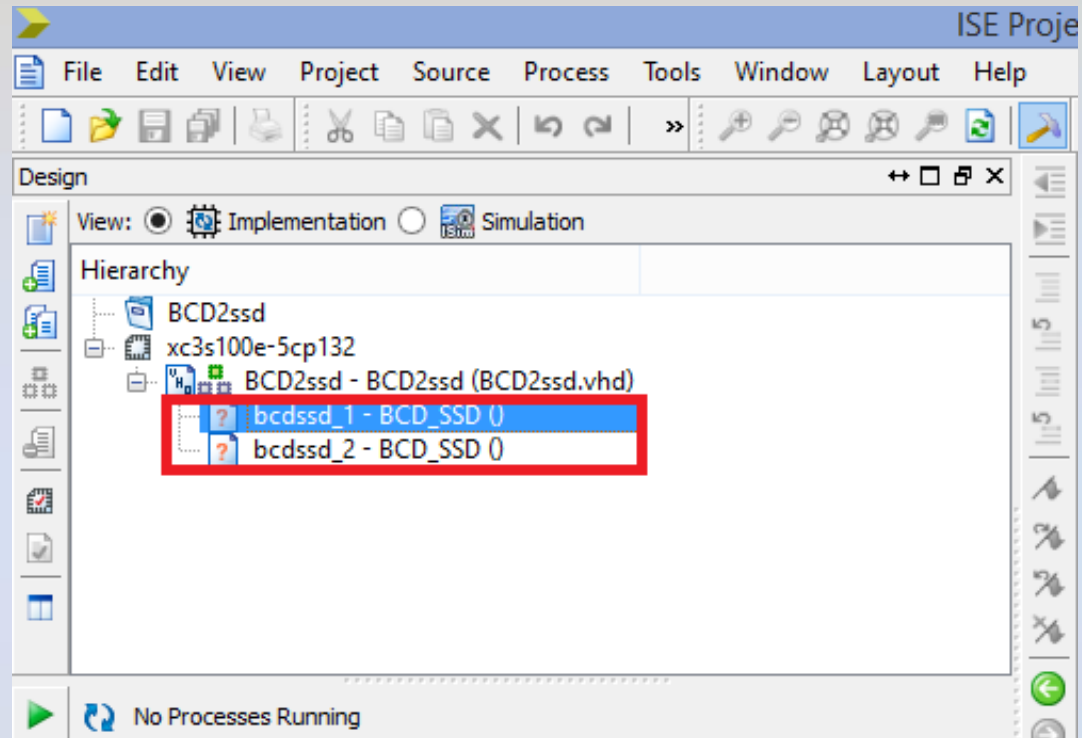
# Kreiranje komponenti i mapiranje portova

```
bcdssd_1: BCD_SSD
  PORT MAP(en => en,
           cout => cout_1,
           rst => rst,
           clk => clk,
           ssd => q_1);
bcdssd_2: BCD_SSD
  PORT MAP(en => cout_1,
           cout => cout_10,
           rst => rst,
           clk => clk,
           ssd => q_10);
cout <= cout_1 AND cout_10;
```



# Kreiranje podmodula

Posle deklaracije povezivanja komponenti u **glavnom modulu** javljaju se dva nova **podmodula**, *bcdssd\_1* i *bcdssd\_2*.



**Svaka izmena u jednom podmodulu automatski će se prenositi i na drugi.**



# Opis podmodula - BCD\_SSD brojač

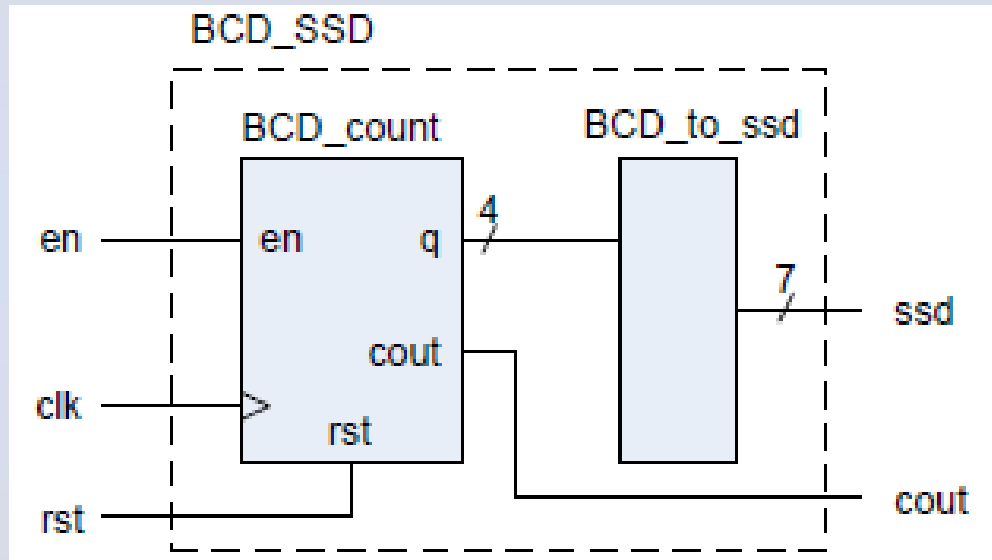
- Ovo je prvi od dva stukturna koda. Kreira jednocifarski BCD\_SSD brojač. Komponente BCD\_count i BCD\_to\_SSD smo deklarirali u paketu, u kodu ih instanciramo i povezujemo.
- Prvo, koristi se nominalno povezivanje, a ne poziciono. Nominalno je očiglednije i preporučuje se. (Poziciono koriste alati za automatsko generisanje VHDL koda).
- Drugo, portovi komponenti ne mogu direktno međusobno da se povežu, već su neophodni interni signali. Npr. Izlaz q komponente BCD\_count se povezuje sa internim signalom bcd\_i, a onda se ovaj signal povezuje na ulazni port bcd komponente BCD\_to\_SSD.





# Opis podmodula

- Šema jednog podmodula i njegove komponente koje su međusobno povezane.



# Opis podmodula

ISE Project Navigator (P.58f) - D:\BCD2ssd\BCD2ssd.xise - [BCD2ssd.vhd]

File Edit View Project Source Process Tools Window Layout Help

Design

View: Implementation Simulation

Hierarchy

- BCD2ssd
  - xc3s100e-5cp132
    - BCD2ssd - BCD2ssd (BCD2ssd.vhd)**
    - bcdssd\_1 - BCD\_SSD ()
    - bcdssd\_2 - BCD\_SSD ()

Processes: BCD2ssd - BCD2ssd

- Design Summary/Reports
- Design Utilities
- User Constraints
- Synthesize - XST
- Implement Design
- Generate Programming File
- Configure Target Device
- Analyze Design Using ChipScope

Start Design Files Libraries

Console

New Source Wizard

Select Source Type  
Select source type, file name and its location.

- BMM File
- ChipScope Definition and Connection File
- Implementation Constraints File
- IP (CORE Generator & Architecture Wizard)
- MEM File
- Schematic
- User Document
- Verilog Module
- Verilog Test Fixture
- VHDL Module**
- VHDL Library
- VHDL Package
- VHDL Test Bench
- Embedded Processor

File name: BCD\_SSD

Location: D:\BCD2ssd

Add to project

More Info Next > Cancel

1. Desni klik, pa **New Source**

# Opis podmodula

Naziv entiteta i  
arhikteture

Define Module  
Specify ports for module.

Entity name: BCD\_SSD  
Architecture name: BCD\_SSD

Port Name	Direction	Bus	MSB	LSB
en	in	<input type="checkbox"/>		
cout	out	<input type="checkbox"/>		
rst	in	<input type="checkbox"/>		
clk	in	<input type="checkbox"/>		
ssd	out	<input checked="" type="checkbox"/>	6	0
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		

More Info      < Back      Next >      Cancel

Nakon **next** biramo **finish**.

# Opis podmodula

The image shows the Xilinx ISE IDE interface. On the left, the Design Hierarchy window displays a tree structure for a project named 'xc3s100e-5cp132'. A red box labeled '1.' highlights the sub-module 'bcdssd\_1 - BCD\_SSD - BCD\_SSD (BCD\_SSD.vhd)'. A red arrow labeled '2.' points from this box to the ISE Text Editor window. The text editor shows the VHDL code for 'BCD\_SSD.vhd'. A red box labeled '3.' highlights the 'ENTITY BCD\_SSD IS' and 'PORT' section. Another red box labeled '4.' highlights the 'END BCD\_SSD;' line at the bottom of the code. The code includes library declarations, package declarations, and the entity/architecture definitions.

```
1 LIBRARY IEEE;
2 USE IEEE.STD_LOGIC_1164.ALL;
3 USE work.BCD2ssdPackage.ALL;
4
5 ENTITY BCD_SSD IS
6     PORT (
7         en : IN STD_LOGIC;
8         cout : OUT STD_LOGIC;
9         rst : IN STD_LOGIC;
10        clk : IN STD_LOGIC;
11        ssd : OUT STD_LOGIC_VECTOR(6 DOWNTO 0)
12    );
13 END BCD_SSD;
14
15 ARCHITECTURE BCD_SSD OF BCD_SSD IS
16
17     -- interni signali
18
19     SIGNAL bcd_i : STD_LOGIC_VECTOR(3 DOWNTO 0);
20 BEGIN
21     bcd_cnt: BCD_count
22     PORT MAP(en => en,
23             rst => rst,
24             q => bcd_i,
25             cout => cout,
26             clk => clk);
27
28     bcd_conv: BCD_to_SSD
29     PORT MAP(BCD => bcd_i,
30             SSD => ssd);
31 END BCD_SSD;
```

4.

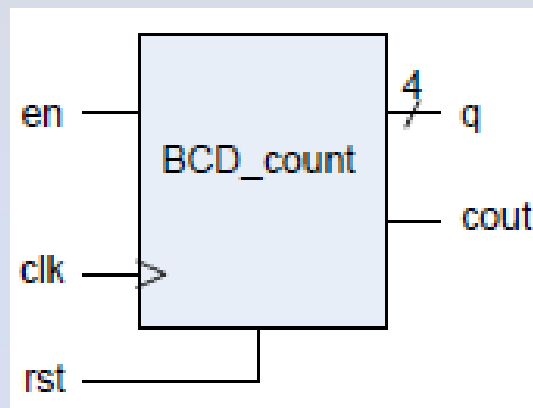
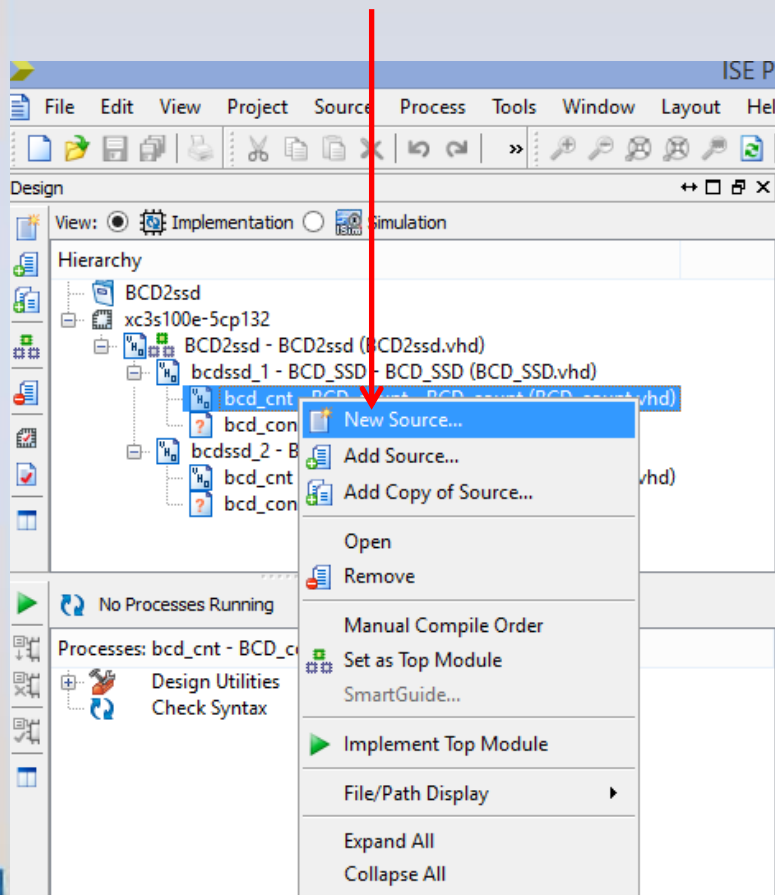
# Objašnjenje podmodula

1. U arhitekturi podmodula (**bcdssd\_1** i **bcdssd\_2**) kreiraju se novi moduli *bcd\_cnt* (brojač) i *bcd\_conv* (konvertor).
2. Uključivanje korisničkog paketa (*work.BCD2ssdPackage*)
3. Entitet naseg podmodula
4. Arhitektura podmodula



# bcd\_cnt

Desni klik na **bcd\_cnt**, zatim New Source.



# bcd\_cnt

New Source Wizard

Select Source Type  
Select source type, file name and its location.

- BMM File
- ChipScope Definition and Connection File
- Implementation Constraints File
- IP (CORE Generator & Architecture Wizard)
- MEM File
- Schematic
- User Document
- Verilog Module
- Verilog Test Fixture
- VHDL Module**
- VHDL Library
- VHDL Package
- VHDL Test Bench
- Embedded Processor

File name: BCD\_count

Location: D:\BCD2ssd

Add to project

More Info Next > Cancel

1. 2. 3. 6.

New Source Wizard

Define Module  
Specify ports for module.

Entity name: BCD\_count

Architecture name: BCD\_count

Port Name	Direction	Bus	MSB	LSB
en	in	<input type="checkbox"/>		
cout	out	<input type="checkbox"/>		
rst	in	<input type="checkbox"/>		
clk	in	<input type="checkbox"/>		
q	out	<input checked="" type="checkbox"/>	3	0
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		

More Info < Back Next > Cancel

4. 5. 7.

# bcd\_cnt

Paket **NUMERIC\_STD** je standardni **IEEE** paket za aritmetiku u VHDL-u.

Entitet

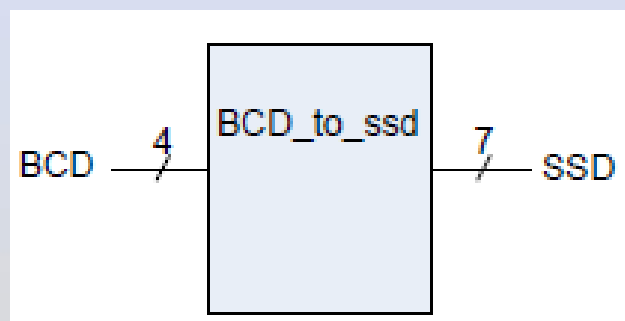
Arhitektura

```
1  LIBRARY IEEE;
2  USE IEEE.STD_LOGIC_1164.ALL;
3  USE IEEE.NUMERIC_STD.ALL;
4
5  ENTITY BCD_count IS
6      PORT(en : IN STD_LOGIC;
7            cout : OUT STD_LOGIC;
8            rst : IN STD_LOGIC;
9            clk : IN STD_LOGIC;
10           q : OUT STD_LOGIC_VECTOR(3 DOWNTO 0));
11 END BCD_count;
12
13 ARCHITECTURE BCD_count OF BCD_count IS
14     SIGNAL r_reg, r_next : UNSIGNED(3 DOWNTO 0);
15     SIGNAL ci : STD_LOGIC;
16 BEGIN
17     PROCESS(clk, rst)
18     BEGIN
19         IF(rst = '1') THEN
20             r_reg <= (OTHERS => '0');
21         ELSIF(clk'EVENT AND clk = '1') THEN
22             IF(en = '1') THEN
23                 r_reg <= r_next;
24             END IF;
25         END IF;
26     END PROCESS;
27     ci <= '1' WHEN r_reg = 9 ELSE
28         '0';
29     r_next <= (OTHERS => '0') WHEN ci = '1' ELSE
30         r_reg + 1;
31     q <= STD_LOGIC_VECTOR(r_reg);
32     cout <= ci;
33 END BCD_count;
```



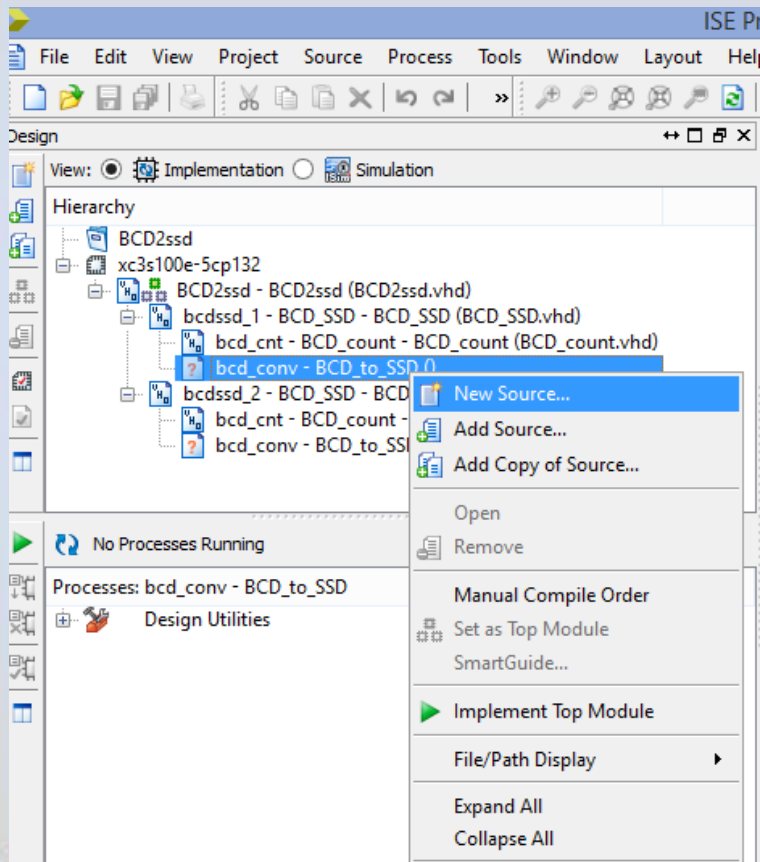
# Konvertor (bcd\_conv)

- **bcd\_conv** je kombinaciono kolo koje vrši konverziju binarno kodiranih decimalnih cifra u **7-bitni** kod za pobudu 7-segmentnog displeja. U **VHDL** opisu koji sledi, ova konverzija je ostvarena posredstvom konkurentne naredbe **select**. Naravno, umesto konkurentne naredbe **select**, mogli smo da koristimo **proces** i naredbu **case**.



# bcd\_conv

Postupak za kreiranje *arhitekture* i *entiteta* **bcd\_conv**.



# bcd\_conv

New Source Wizard

## Select Source Type

Select source type, file name and its location.

- BMM File
- ChipScope Definition and Connection File
- Implementation Constraints File
- IP (CORE Generator & Architecture Wizard)
- MEM File
- Schematic
- User Document
- Verilog Module
- Verilog Test Fixture
- VHDL Module**
- VHDL Library
- VHDL Package
- VHDL Test Bench
- Embedded Processor

File name:

BCD\_to\_SSD

Location:

D:\BCD2ssd

Add to project

More Info

Next >

New Source Wizard

## Define Module

Specify ports for module.

Entity name BCD\_to\_SSD

Architecture name BCD\_to\_SSD

Port Name	Direction	Bus	MSB	LSB
BCD	in	<input checked="" type="checkbox"/>	3	0
SSD	out	<input checked="" type="checkbox"/>	6	0
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		
	in	<input type="checkbox"/>		

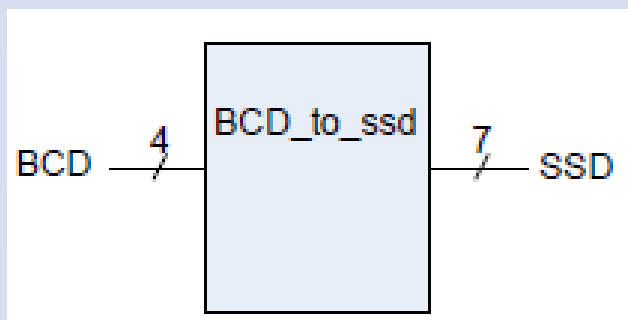
More Info

< Back

Next >

Cancel

# bcd\_conv



```
1  --Konvertor 4 -> 7 bita.
2  LIBRARY IEEE;
3  USE IEEE.STD_LOGIC_1164.ALL;
4
5  ENTITY BCD_to_SSD IS
6      PORT(BCD : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
7           SSD : OUT STD_LOGIC_VECTOR(6 DOWNTO 0));
8  END BCD_to_SSD;
9
10 ARCHITECTURE BCD_to_SSD OF BCD_to_SSD IS
11 BEGIN
12 WITH BCD SELECT
13 SSD <= "1111110" WHEN "0000",
14        "0110000" WHEN "0001",
15        "1101101" WHEN "0010",
16        "1111001" WHEN "0011",
17        "0110011" WHEN "0100",
18        "1011011" WHEN "0101",
19        "1011111" WHEN "0110",
20        "1110000" WHEN "0111",
21        "1111111" WHEN "1000",
22        "1111011" WHEN OTHERS;
23 END BCD_to_SSD;
24
```

# Sinteza

Prvo selektovati  
vršni modul  
BCD2ssd

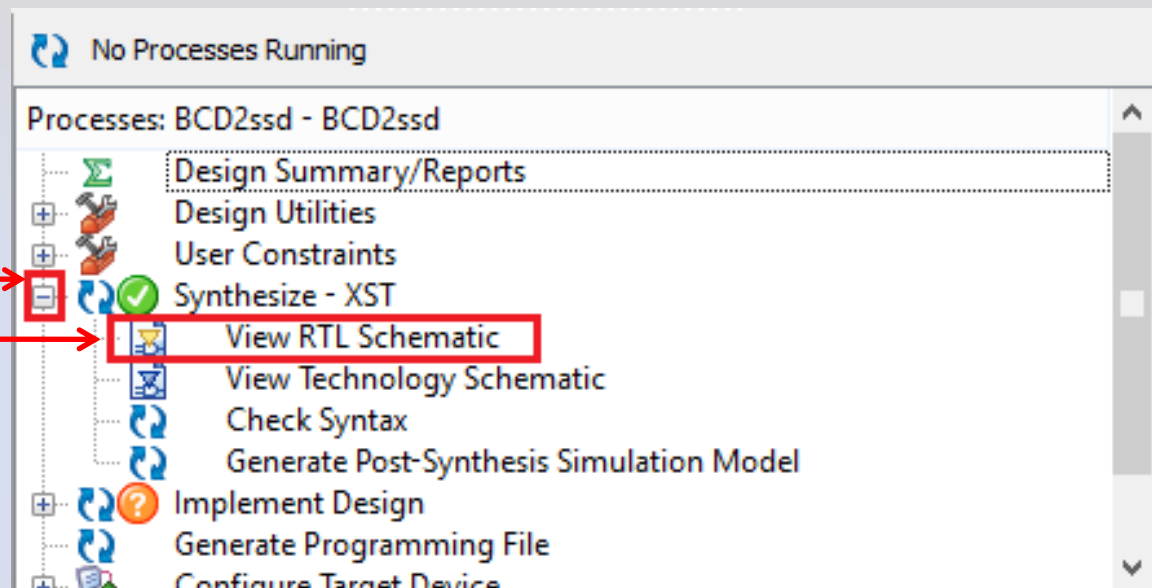
Zatim obaviti  
sintezu dvo-klikom  
na *Synthesize -  
XST*

Poruka o uspešno  
obavljenoj sintezi

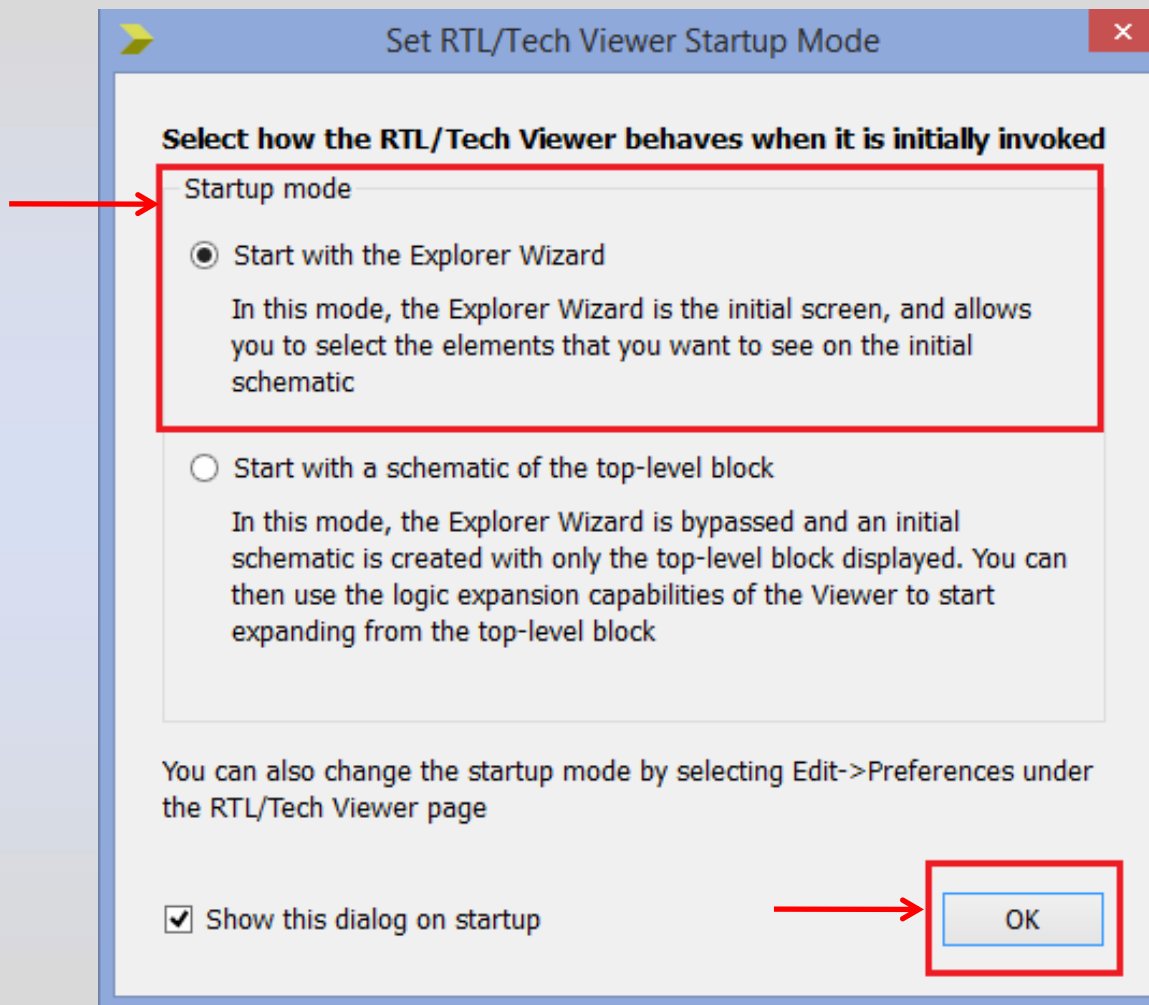
The screenshot displays the Xilinx ISE software interface. The 'Hierarchy' window shows the project structure with the top-level module 'BCD2ssd - BCD2ssd (BCD2ssd.vhd)' selected. The 'Processes' window shows the 'Synthesize - XST' process highlighted. The 'Console' window at the bottom displays the message: 'Process "Synthesize - XST" completed successfully'.

# View RTL Schematic

Klik na plus kod sinteze, zatim **dupli klik** na View RTL Schematic



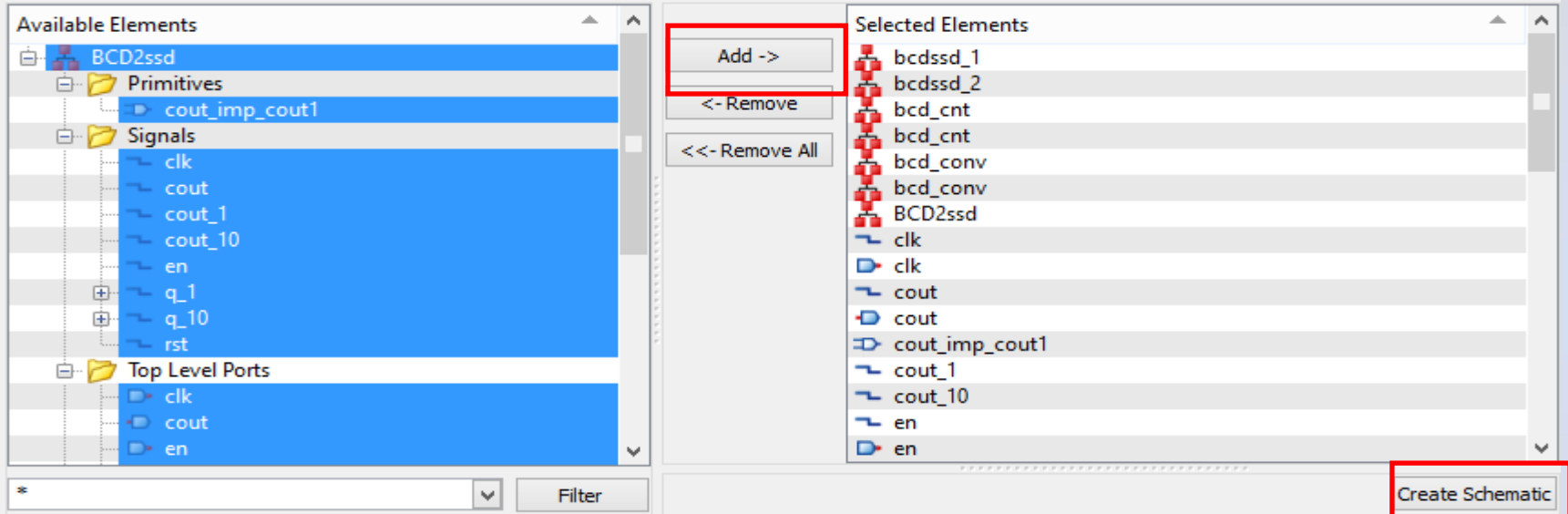
# View RTL Schematic



# View RTL Schematic

## Create RTL Schematic

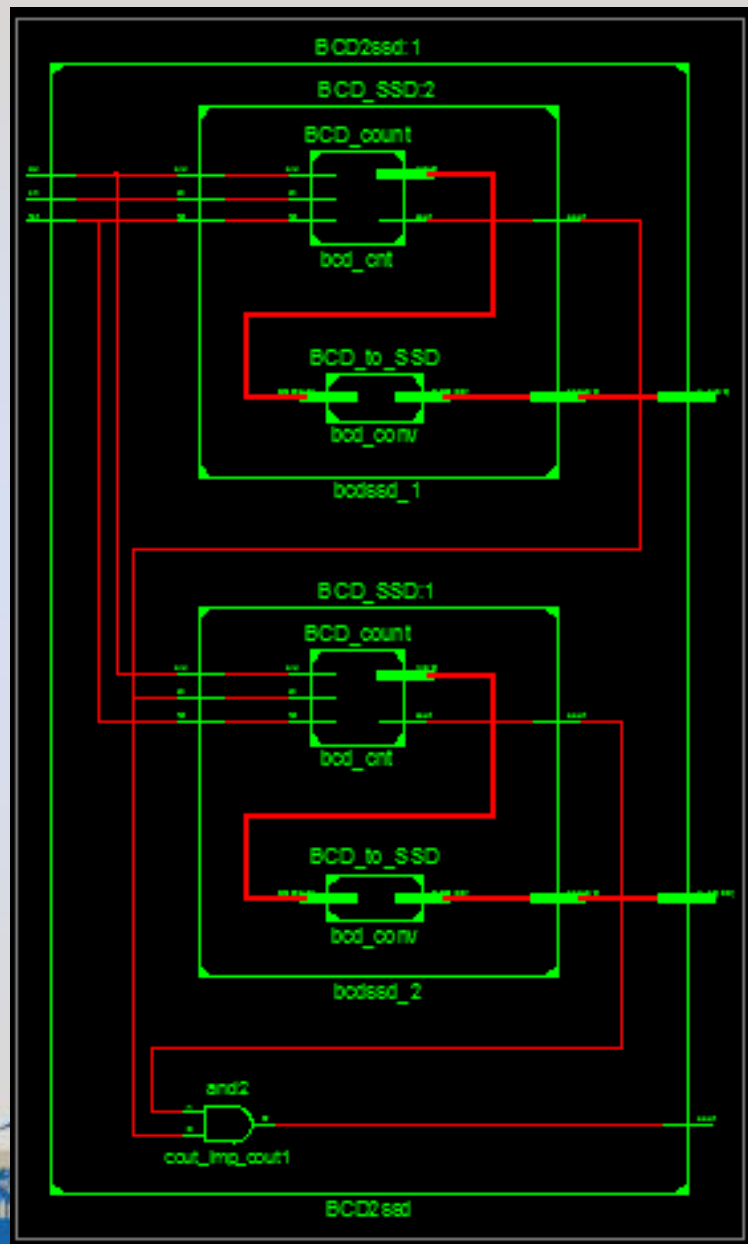
- 1) Select items you want on the schematic from the "Available Elements" list and move them to the "Selected Elements" list
  - Use the Filter control to filter the "Available Elements" list by name
- 2) Press the "Create Schematic" button to generate a schematic view using the items in the "Selected Elements" list



Proširimo foldere na + (**Primitives, Signals, Top Level Ports, bcdssd\_1, bcdssd\_2**), a zatim ih selektujemo sve. Onda biraemo **Add**, a zatim **Create Schematic**.



# RTL Schematic



# Realizacija projekta na FPGA

- Predhodni kod u arhitekturi modula koji je prikazan za uspešnu funkcionalnu realizaciju FPGA sedmosegmentnog displeja nije dovoljan.
- Da bi tačno radilo odbrojavanje jedinica i desetica na displeju, kao i aktiviranje određenih cifara u određeno vreme potrebno je izvršiti određene dopune/izmene.
- Na sledećim slajdovima prikazane su dopune u kodu sa predhodnih slajdova iz istog projekta.



# Realizacija projekta na FPGA

Prva dopuna (izmena) koda je u glavnom modulu našeg projekta, **BCD2ssd**.

```
1 LIBRARY IEEE;
2 USE IEEE.STD_LOGIC_1164.ALL;
3 USE work.BCD2SSD_PCK.ALL;
4
5 ENTITY BCD2_SSD IS
6     Port (en : IN STD_LOGIC;
7           rst : IN STD_LOGIC;
8           clk : IN STD_LOGIC;
9           cout : OUT STD_LOGIC;
10          1. q : OUT STD_LOGIC_VECTOR(6 DOWNTO 0);
11            dot : OUT std_logic;
12            anods: OUT std_logic_vector(3 DOWNTO 0));
13 END BCD2_SSD;
14
15 --Glavna arhitektura (Ne dirati)
16 ARCHITECTURE BCD2_SSD OF BCD2_SSD IS
17
18     3. SIGNAL cout_1, cout_10, sel : STD_LOGIC;
19     SIGNAL q_1_TMP : STD_LOGIC_VECTOR(6 DOWNTO 0);
20     SIGNAL q_10_TMP : STD_LOGIC_VECTOR(6 DOWNTO 0);
21
22     SIGNAL counter : integer range 0 to 124999 := 0;
23
24 BEGIN
25
26     dot<='1';
27
28     PROCESS (clk)
29     BEGIN
30     IF rising_edge(clk) THEN
31         IF (counter = 124999) THEN
32             sel <= NOT(sel);
33             counter <= 0;
34         ELSE
35             counter <= counter + 1;
36         END IF;

```

```
37     END IF;
38     END PROCESS;
39
40     anods <= "1011" WHEN sel='0'
41             ELSE "0111";
42
43     q <= q_1_TMP WHEN sel='0'
44         ELSE q_10_TMP;
45
46     bcdssd_1: BCD_SSD
47         PORT MAP(en => en,
48                 cout => cout_1,
49                 rst => rst,
50                 clk => clk,
51                 ssd => q_1_TMP);
52     bcdssd_2: BCD_SSD
53         PORT MAP(en => cout_1,
54                 cout => cout_10,
55                 rst => rst,
56                 clk => clk,
57                 ssd => q_10_TMP);
58
59     cout <= cout_1 AND cout_10;
60
61 END BCD2_SSD;
```

3.

2.

5.

4.

# Realizacija projekta na FPGA

1. Deklaracija novih portova u entitetu: ***q***, ***dot*** i ***anods***.
2. Pored signala `cout_1` i `cout_10`, uvodimo **sel** (*selekcioni signal*). Zadatak ovog signala je da aktivira određenu cifru (jedinicu ili desetice).
3. Uvođenje novih signala ***q\_1\_TMP*** i ***q\_2\_TMP***. Uvođenje signala ***counter***.
4. U liniji **26.** dodajemo naredbu: `dot <= '1'`; Ove naredba konstantno drži tačku ugašenu tokom rada naseg brojača na displeju.



# Realizacija projekta na FPGA

4. U naredbama od 28. do 38. vrši se skaliranja takta, sa 50MHz na 200Hz:

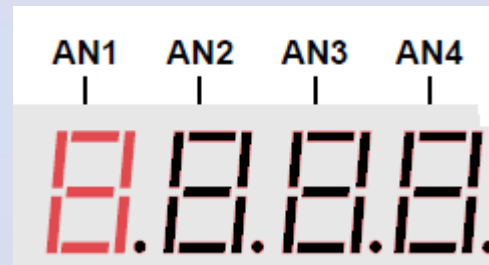
```
28 PROCESS (clk)
29 BEGIN
30 IF rising_edge(clk) THEN
31     IF (counter = 124999) THEN
32         sel <= NOT(sel);
33         counter <= 0;
34     ELSE
35         counter <= counter + 1;
36     END IF;
37 END IF;
38 END PROCESS;
```

# Realizacija projekta na FPGA

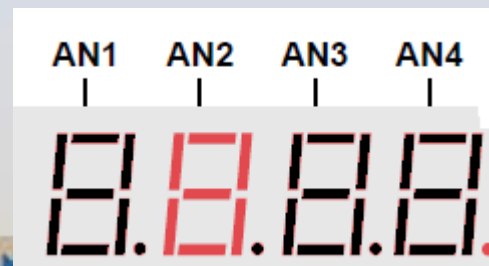
4. U naredbama od 40. do 44. vrši se **selekcija** odabira cifre koja će biti prikazana na displeju (**jedinica** ili **desetica**) – vrši se **vremensko multipleksiranje** sa frekvencom od 200Hz).

```
40 anods <= "1011" WHEN sel='0'  
41     ELSE "0111";  
42  
43 q <= q_1_TMP WHEN sel='0'  
44     ELSE q_10_TMP;
```

Selekcija desetice za anods <= "0111"



Selekcija jedinica za anods <= "1011"



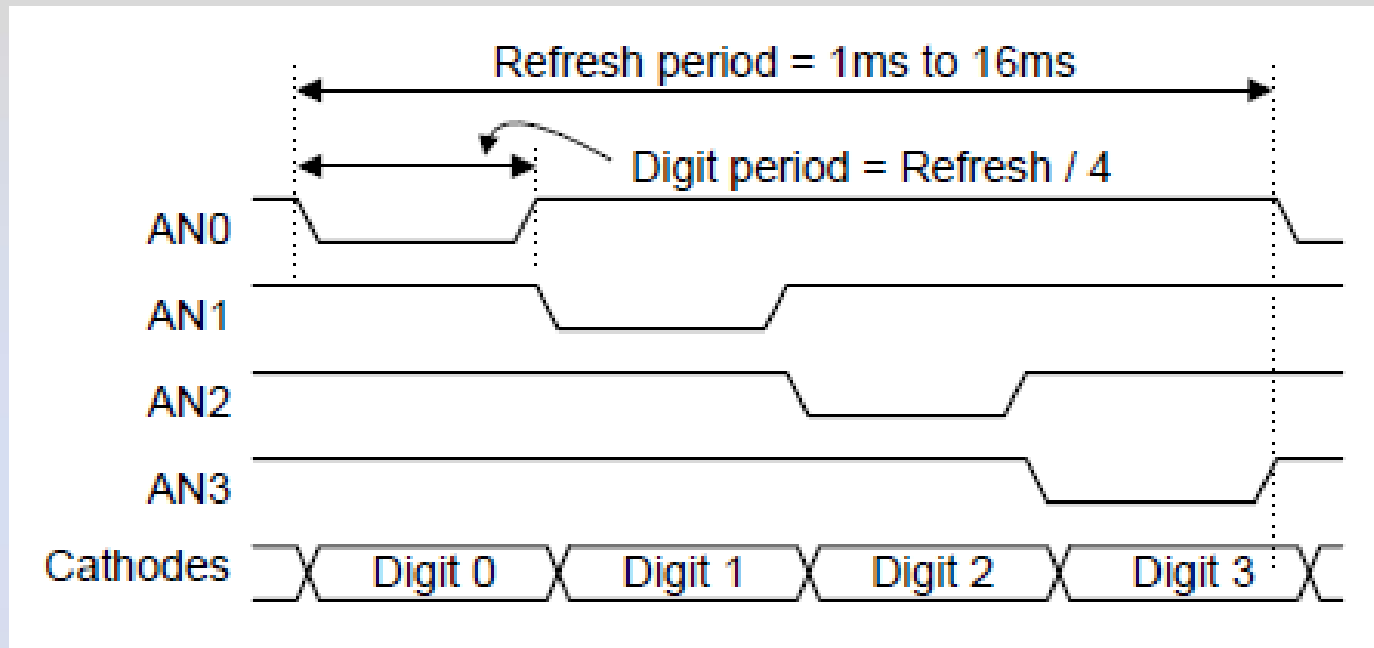
# Pitanje

- Kako bi se realizovalo vremensko multipleksiranje korišćenjem **PROCESS-a** i naredbe **IF** ?

```
40  anods <= "1011" WHEN sel='0'  
41      ELSE "0111";  
42  
43  q <= q_1_TMP WHEN sel='0'  
44      ELSE q_10_TMP;
```



# Realizacija projekta na FPGA



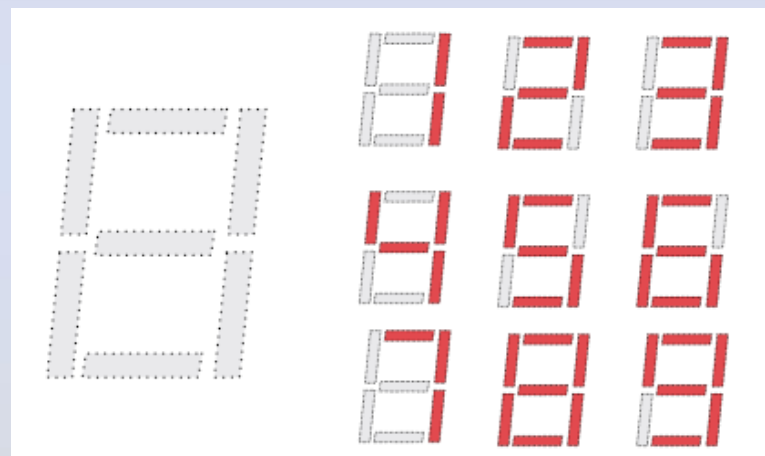
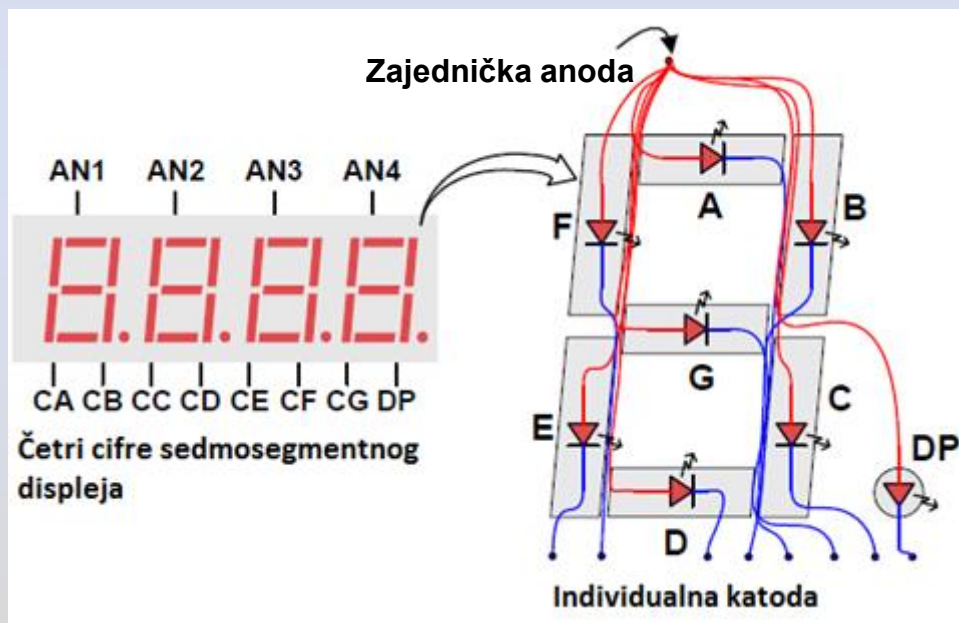
Primer vremenskog multipleksiranja sedmosegmentnog displeja sa 4 cifre.



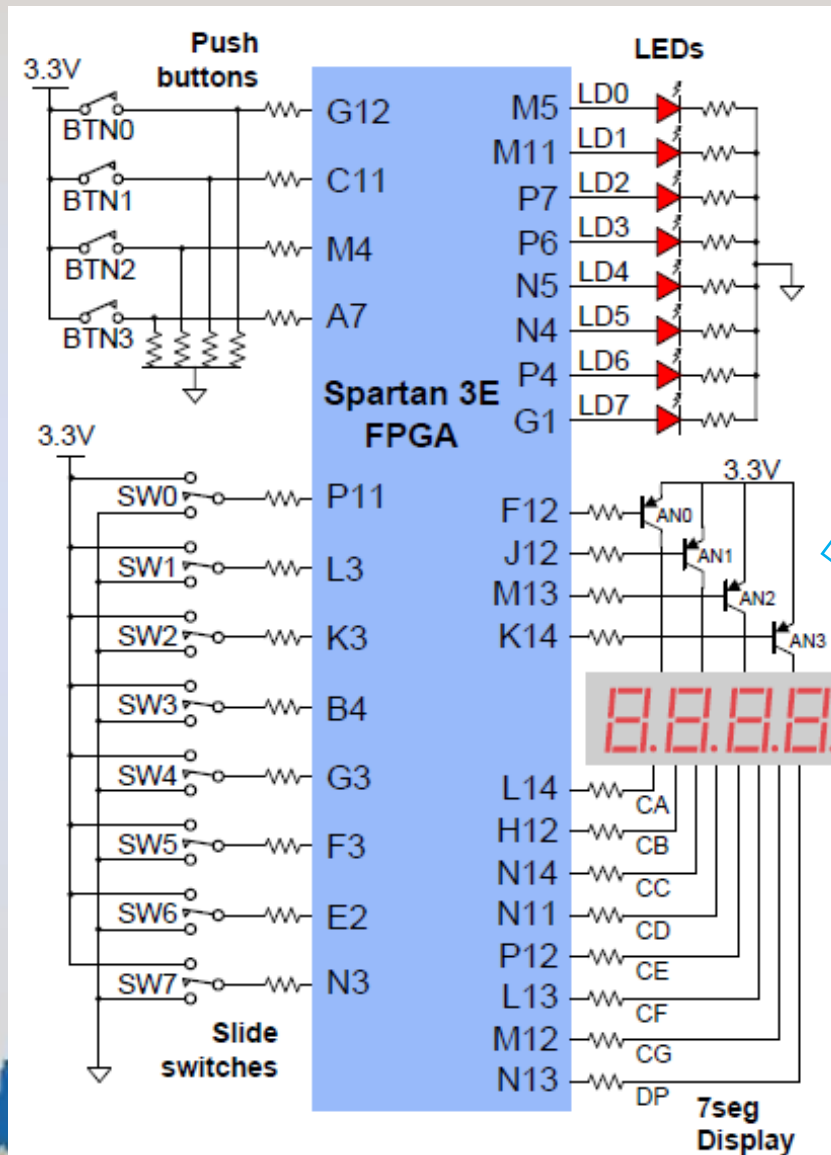
# Realizacija projekta na FPGA

U zavisnosti od selekcionog signala aktiviraće se odgovarajuća anoda, a brojač će aktivirati različite kombinacije cifara na izabranoj anodi.

Razne kombinacije prikazane na displeju



# Povezivanje sa FPGA



Pinovi za izbor anode tj. pozicije cifre.

Pinovi za određenu kombinaciju cifre.

# Realizacija projekta na FPGA

5. Promena imena signala kao u deklarativnom delu, **q\_1\_TMP** i **q\_10\_TMP**.

```
bcdssd_1: BCD_SSD
  PORT MAP(en => en,
           cout => cout_1,
           rst => rst,
           clk => clk,
           ssd => q_1_TMP);

bcdssd_2: BCD_SSD
  PORT MAP(en => cout_1,
           cout => cout_10,
           rst => rst,
           clk => clk,
           ssd => q_10_TMP);
```

# Realizacija projekta na FPGA

- Slede izmene koda u brojaču **bcd\_cnt** prvog podmodula (*bcdssd\_1*).
- **Ne zaboravimo da se svaka izmena u jednom podmodulu automatski prenosi i na drugi!**



# Realizacija projekta na FPGA

Uvođenje signala **prescaler** koji je neophodan kako bi naš brojač brojao na **1 sekundu** a ne na **50MHz**.

```
1  LIBRARY IEEE;
2  USE IEEE.STD_LOGIC_1164.ALL;
3  USE IEEE.NUMERIC_STD.ALL;
4
5  ENTITY BCD_count IS
6      PORT(en : IN STD_LOGIC;
7            cout :OUT STD_LOGIC;
8            rst : IN STD_LOGIC;
9            clk : IN STD_LOGIC;
10             q : OUT STD_LOGIC_VECTOR(3 DOWNTO 0));
11  END BCD_count;
12
13  ARCHITECTURE BCD_count OF BCD count IS
14      SIGNAL prescaler : unsigned (25 downto 0);
15      SIGNAL r_reg, r_next : UNSIGNED(3 DOWNTO 0);
16      SIGNAL ci : STD_LOGIC;
17
18  BEGIN
19      PROCESS(clk,rst)
20      BEGIN
21          IF(rst = '1') THEN
22              r_reg <= (OTHERS => '0');
23          ELSIF(clk'EVENT AND clk = '1') THEN
24              IF(en = '1') THEN
25                  IF prescaler <= "10111110101111000010000000" THEN
26                      prescaler <= prescaler + 1;
27                  ELSE
28                      prescaler <=(others => '0');
29                      r_reg <= r_next;
30                  END IF;
31              END IF;
32          END IF;
33      END PROCESS;
34
35      ci <= '1' WHEN r_reg = 9 ELSE
36          '0';
37      r_next <= (OTHERS => '0') WHEN ci = '1' ELSE
38          r_reg + 1;
39      q <= STD_LOGIC_VECTOR(r_reg);
40      cout <= ci;
41  END BCD_count;
```

# Realizacija projekta na FPGA

Invertovali smo 1 u 0, i 0 u 1  
zato što će displej da aktivira  
diode na 0, a deaktivira na 1.

```
1  --Konvertor 4 -> 7 bita.
2  LIBRARY IEEE;
3  USE IEEE.STD_LOGIC_1164.ALL;
4
5  ENTITY BCD_to_SSD IS
6      PORT (BCD : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
7            SSD : OUT STD_LOGIC_VECTOR(6 DOWNTO 0));
8  END BCD_to_SSD;
9
10 ARCHITECTURE BCD_to_SSD OF BCD_to_SSD IS
11 BEGIN
12 WITH BCD SELECT
13 SSD <= "0000001" WHEN "0000",
14        "1001111" WHEN "0001",
15        "0010010" WHEN "0010",
16        "0000110" WHEN "0011",
17        "1001100" WHEN "0100",
18        "0100100" WHEN "0101",
19        "0100000" WHEN "0110",
20        "0001111" WHEN "0111",
21        "0000000" WHEN "1000",
22        "0000100" WHEN OTHERS;
23 END BCD to SSD;
```

# Realizacija projekta na FPGA

Posle izmena u kodu neophodno je pokrenuti sintezu radi provere ispravnosti našeg koda (*provera sintakse...*)

1. Klik na **BCD2\_SSD**
2. Dvo-klik na **Synthesize – XST** ili desni klik pa **Re-run**
3. Uspešno obavljena sinteza!

The screenshot shows the 'Design' window with the 'Implementation' view selected. The 'Hierarchy' panel shows the project structure, including the 'bcd2ssd' component and its sub-components. A red arrow labeled '1.' points to the 'BCD2\_SSD - BCD2\_SSD (BCD2\_SSD\_P.vhd)' component. The 'Processes' panel shows a list of processes, with 'Synthesize - XST' highlighted. A red arrow labeled '2.' points to this process. The 'Console' panel at the bottom shows the message 'Process "Synthesize - XST" completed successfully', which is highlighted with a red box. A red arrow labeled '3.' points to this message.

# Realizacija projekta na FPGA

**Samom izmenom u kodu, dodavali smo nove signale i naredbe. Svaka izmena i dopuna u kodu utiču na izgled RTL šeme.**

1. Dvo-klik na View RTL Schematic
2. Ok

The screenshot shows the Xilinx IDE interface. The top window displays the RTL Schematic for a design named 'BCD2ssd'. The code in the background is:

```
1 --Konvertor 4 -> 7 bita.  
2 LIBRARY IEEE;  
3 USE IEEE.STD_LOGIC_1164.ALL;
```

The 'Processes' pane on the left shows the design flow. A red arrow labeled '1.' points to the 'View RTL Schematic' icon in the 'Synthesize - XST' group.

The 'Set RTL/Tech Viewer Startup Mode' dialog box is open, showing the following options:

- Start with the Explorer Wizard  
In this mode, the Explorer Wizard is the initial screen, and allows you to select the elements that you want to see on the initial schematic
- Start with a schematic of the top-level block  
In this mode, the Explorer Wizard is bypassed and an initial schematic is created with only the top-level block displayed. You can then use the logic expansion capabilities of the Viewer to start expanding from the top-level block

At the bottom of the dialog, there is a checkbox labeled 'Show this dialog on startup' which is checked. A red arrow labeled '2.' points to the 'OK' button.



# Realizacija projekta na FPGA

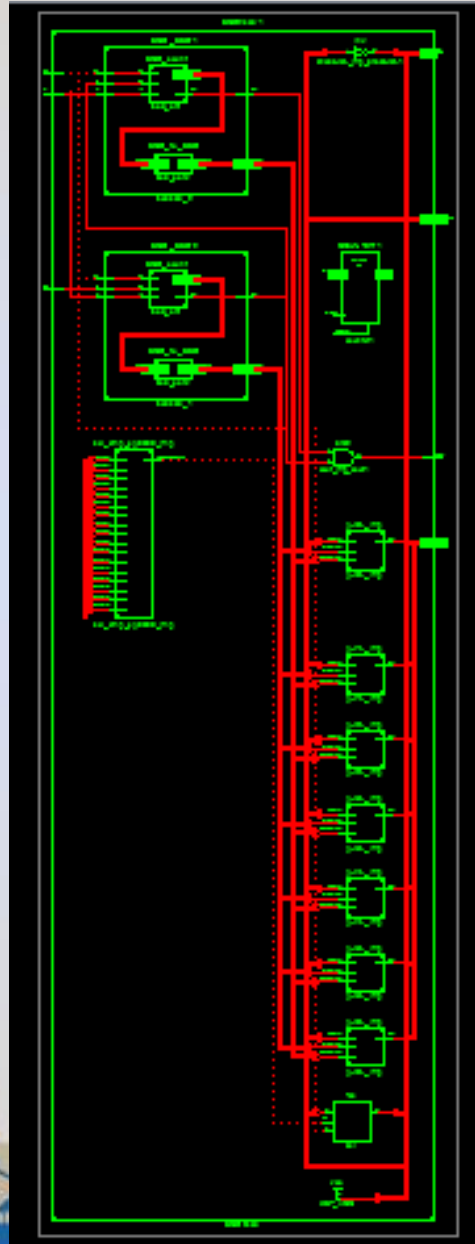
## Create RTL Schematic

- 1) Select items you want on the schematic from the "Available Elements" list and move them to the "Selected Elements" list
  - Use the Filter control to filter the "Available Elements" list by name
- 2) Press the "Create Schematic" button to generate a schematic view using the items in the "Selected Elements" list

The screenshot shows the 'Create RTL Schematic' dialog. On the left, the 'Available Elements' list is organized into folders: 'Primitives' (containing 'anods<3>\_imp\_anods<3>1', 'counter1', 'cout\_imp\_cout1', 'sel', 'XST\_VCC') and 'Signals' (containing 'anods', 'clk', 'counter', 'cout', 'cout\_1', 'cout\_10', 'en', 'q'). A red arrow labeled '1.' points from the 'Add ->' button to the 'Available Elements' list. On the right, the 'Selected Elements' list contains: 'anods(0)', 'anods(1)', 'anods(2)', 'anods(3)', 'anods(3)', 'anods(3)', 'anods<3>\_imp\_anods<3>1', 'bcdssd\_1', 'bcdssd\_2', 'bcd\_cnt', 'bcd\_cnt', 'bcd\_conv', 'bcd\_conv', 'BCD2ssd', and 'clk'. A red arrow labeled '2.' points from the 'Create Schematic' button at the bottom right to the 'Selected Elements' list.

Proširimo foldere na + (**Primitives, Signals, Top Level Ports, bcdssd\_1, bcdssd\_2**) i zatim ih selektujemo sve. Onda idemo na **Add**, a zatim na **Create Schematic**.

# RTL Schematic



# Implementacija

1. Selektujemo glavni modul
2. Dvo-klik na **Implement Design**
3. Uspešna implementacija!

The screenshot displays the Xilinx ISE Design Suite interface during the implementation phase. The 'Design' window is set to 'Implementation' view. The 'Hierarchy' pane shows the project structure with 'BCD2ssd' selected. The 'Processes' pane shows the 'Implement Design' step is active and highlighted. The 'Console' pane at the bottom shows the message: 'Process "Generate Post-Place & Route Static Timing" completed successfully'.

1. →

2. →

3. →

```
1  --Konvertov  
2  LIBRARY I  
3  USE IEEE.  
4  
5  ENTITY BCD  
6  PORT (BCD  
7  SS  
8  END BCD_t  
9  
10 ARCHITECT  
11 BEGIN  
12 WITH BCD  
13 SSD <= "0  
14 "1  
15 "0  
16 "0  
17 "1  
18 "0  
19 "0  
20 "0  
21 "0  
22 "0  
23 END BCD_t
```

Design Summary (Imple

Console

Total time: 2 secs

Process "Generate Post-Place & Route Static Timing" completed successfully

# Ograničenja – User Constraints File (UCF)

1. Desni klik preko oznake kola pa **New Source**

2. Biramo **Implementation Constraints File**

3. Dodelimo mu ime BCD2ssd.

4. Next -> Finish

1. Desni klik preko oznake kola pa **New Source**
2. Biramo **Implementation Constraints File**
3. Dodelimo mu ime BCD2ssd.
4. Next -> Finish

# Kod ucf-a

The screenshot shows the ISE Project Navigator interface. The title bar reads "ISE Project Navigator (P.58f) - D:\BCD2ssd\BCD2ssd". The menu bar includes File, Edit, View, Project, Source, Process, Tools, Window, Layout, and Help. The Design window is active, showing a Hierarchy tree on the left with the following structure:

- BCD2ssd
  - xc3s100e-5cp132
    - BCD2ssd - BCD2ssd (BCD2ssd.vhd)
      - bcdssd\_1 - BCD\_SSD - BCD\_SSD (BCD\_SSD.vhd)
      - bcdssd\_2 - BCD\_SSD - BCD\_SSD (BCD\_SSD.vhd)
      - BCD2ssd.ucf

```
1 NET "en" LOC = "N3";  
2 NET "clk" LOC = "B8";  
3 NET "rst" LOC = "A7";  
4 NET "cout" LOC = "G1";  
5  
6 NET "dot" LOC = "N13";  
7  
8 NET "q<6>" LOC = "L14";  
9 NET "q<5>" LOC = "H12";  
10 NET "q<4>" LOC = "N14";  
11 NET "q<3>" LOC = "N11";  
12 NET "q<2>" LOC = "P12";  
13 NET "q<1>" LOC = "L13";  
14 NET "q<0>" LOC = "M12";  
15  
16 NET "anods<0>" LOC = F12;  
17 NET "anods<1>" LOC = J12;  
18 NET "anods<2>" LOC = M13;  
19 NET "anods<3>" LOC = K14;
```

# Povezivanje portova modula koji projektujemo i pinova FPGA kola

Iz korisničkog uputstva za razvojnu ploču.

