VHDL, SPR, II kolokvijum, priprema,

Napomena: Na kolokvijumu dolazi 1 zadatak (40%) i 1 praktican projekat (60%). Radi se u grupama prema utvrdjenom rasporedu od I kolokvijuma.

A. ZADACI

Priloziti VHDL Kod i Simulacija za sledeće zadatke?

1. 4-bit Unsigned Up Counter with Asynchronous Clear

The following table shows pin definitions for a 4-bit unsigned up counter with asynchronous clear.

IO Pins	Description
C	Positive-Edge Clock
CLR	Asynchronous Clear (active High)
Q[3:0]	Data Output

```
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity counter is
  port(C, CLR : in  std_logic;
       Q : out std_logic_vector(3 downto 0));
end counter;
architecture archi of counter is
  signal tmp: std_logic_vector(3 downto 0);
  begin
      process (C, CLR)
        begin
          if (CLR='1') then
            tmp <= "0000";
          elsif (C'event and C='1') then
            tmp <= tmp + 1;
          end if;
      end process;
      Q <= tmp;
end archi;
```

2. 4-bit Unsigned Up/Down counter with Asynchronous Clear

The following table shows pin definitions for a 4-bit unsigned up/down counter with asynchronous clear.

IO Pins	Description
C	Positive-Edge Clock
CLR	Asynchronous Clear (active High)
UP_DOWN	up/down count mode selector
Q[3:0]	Data Output

VHDL Code

Following is the VHDL code for a 4-bit unsigned up/down counter with asynchronous clear.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity counter is
  port(C, CLR, UP_DOWN : in std_logic;
       Q : out std_logic_vector(3 downto 0));
end counter;
architecture archi of counter is
  signal tmp: std_logic_vector(3 downto 0);
  begin
    process (C, CLR)
      begin
        if (CLR='1') then
          tmp <= "0000";
        elsif (C'event and C='1') then
          if (UP_DOWN='1') then
            tmp <= tmp + 1;
          else
            tmp <= tmp - 1;
          end if;
        end if;
    end process;
    Q <= tmp;
end archi;
```

3. 4-bit Unsigned Up Counter with Asynchronous Load from Primary Input

The following table shows pin definitions for a 4-bit unsigned up counter with asynchronous load from primary input.

IO Pins	Description

C	Positive-Edge Clock
ALOAD	Asynchronous Load (active High)
D[3:0]	Data Input
Q[3:0]	Data Output

VHDL Code

Following is the VHDL code for a 4-bit unsigned up counter with asynchronous load from primary input.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity counter is
  port(C, ALOAD : in  std_logic;
       D : in std_logic_vector(3 downto 0);
       Q : out std_logic_vector(3 downto 0));
end counter;
architecture archi of counter is
  signal tmp: std_logic_vector(3 downto 0);
  begin
    process (C, ALOAD, D)
      begin
        if (ALOAD='1') then
          tmp <= D;
        elsif (C'event and C='1') then
          tmp <= tmp + 1;
        end if;

    end process;
    Q <= tmp;
end archi;
```

4. 8-bit Shift-Left Register with Positive-Edge Clock, Asynchronous Clear, Serial In, and Serial Out

Note Because this example includes an asynchronous clear, XST will not infer SRL16.

The following table shows pin definitions for an 8-bit shift-left register with a positive-edge clock, asynchronous clear, serial in, and serial out.

IO Pins	Description
C	Positive-Edge Clock
SI	Serial In
CLR	Asynchronous Clear (active High)
SO	Serial Output

VHDL Code

Following is the VHDL code for an 8-bit shift-left register with a positive-edge clock, asynchronous clear, serial in, and serial out.

```
library ieee;
use ieee.std_logic_1164.all;

entity shift is
  port(C, SI, CLR : in std_logic;
       SO : out std_logic);
end shift;
architecture archi of shift is
  signal tmp: std_logic_vector(7 downto 0);
  begin
    process (C, CLR)
      begin
        if (CLR='1') then
          tmp <= (others => '0');
        elsif (C'event and C='1') then
          tmp <= tmp(6 downto 0) & SI;
        end if;
    end process;
    SO <= tmp(7);
end archi;
```

5. 8-bit Shift-Left/Shift-Right Register with Positive-Edge Clock, Serial In, and Parallel Out

Note For this example XST will not infer SRL16.

The following table shows pin definitions for an 8-bit shift-left/shift-right register with a positive-edge clock, serial in, and serial out.

IO Pins	Description
C	Positive-Edge Clock
SI	Serial In
LEFT_RIGHT	Left/right shift mode selector
PO[7:0]	Parallel Output

VHDL Code

Following is the VHDL code for an 8-bit shift-left/shift-right register with a positive-edge clock, serial in, and serial out.

```
library ieee;
use ieee.std_logic_1164.all;

entity shift is
port(C, SI, LEFT_RIGHT : in std_logic;
```

```
      PO : out std_logic_vector(7 downto 0));
end shift;
architecture archi of shift is
  signal tmp: std_logic_vector(7 downto 0);
  begin
    process (C)
      begin
        if (C'event and C='1') then
          if (LEFT_RIGHT='0') then
            tmp <= tmp(6 downto 0) & SI;
          else
            tmp <= SI & tmp(7 downto 1);
          end if;
        end if;
    end process;
    PO <= tmp;
end archi;
```

6. Logical shifter

The following table shows pin descriptions for a logical shifter.

IO pins	Description
D[7:0]	Data Input
SEL	shift distance selector
SO[7:0]	Data Output

VHDL

Following is the VHDL code for a logical shifter.

```
library ieee;
 use ieee.std_logic_1164.all;
 use ieee.numeric_std.all;

 entity lshift is
  port(DI : in unsigned(7 downto 0);
       SEL : in unsigned(1 downto 0);
        SO : out unsigned(7 downto 0));
 end lshift;
 architecture archi of lshift is
   begin
     with SEL select
       SO <= DI when "00",
             DI sll 1 when "01",
             DI sll 2 when "10",
             DI sll 3 when others;
 end archi;
```

7. **RING JONSON COUNTER**

library IEEE;

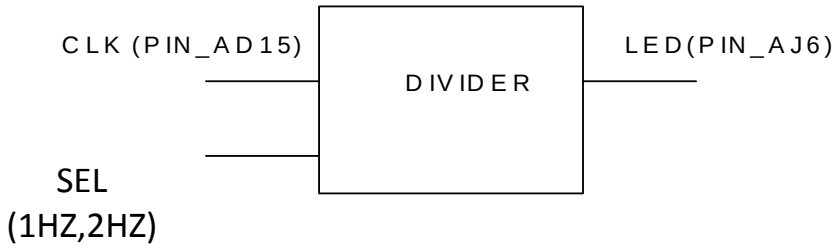
```
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity ring_counter is
port (
      DAT_O : out unsigned(3 downto 0);
      RST_I : in std_logic;
      CLK_I : in std_logic
      );
end ring_counter;

architecture Behavioral of ring_counter is

signal temp : unsigned(3 downto 0):=(others => '0');

begin

DAT_O <= temp;

process(CLK_I)
begin
   if( rising_edge(CLK_I) ) then
      if (RST_I = '1') then
         temp <= (0=> '1', others => '0');
      else
         temp(1) <= temp(0);
         temp(2) <= temp(1);
         temp(3) <= temp(2);
         temp(0) <= temp(3);
      end if;
   end if;
end process;

end Behavioral;

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity johnson_counter is
port (
      DAT_O : out unsigned(3 downto 0);
      RST_I : in std_logic;
      CLK_I : in std_logic
      );
end johnson_counter;

architecture Behavioral of johnson_counter is

signal temp : unsigned(3 downto 0):=(others => '0');
```

```
begin

DAT_O <= temp;

process(CLK_I)
begin
   if( rising_edge(CLK_I) ) then
      if (RST_I = '1') then
         temp <= (others => '0');
      else
         temp(1) <= temp(0);
         temp(2) <= temp(1);
         temp(3) <= temp(2);
         temp(0) <= not temp(3);
      end if;
   end if;
end process;

end Behavioral;
```

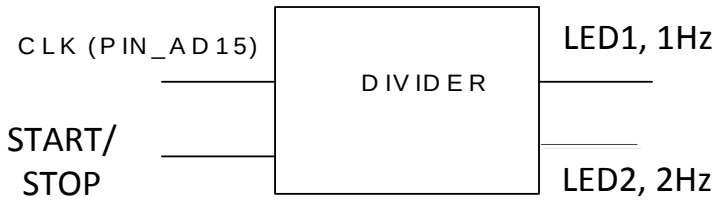
B. Praktični problemi

Demonstrirati rad kola na DE2-70 ploči?

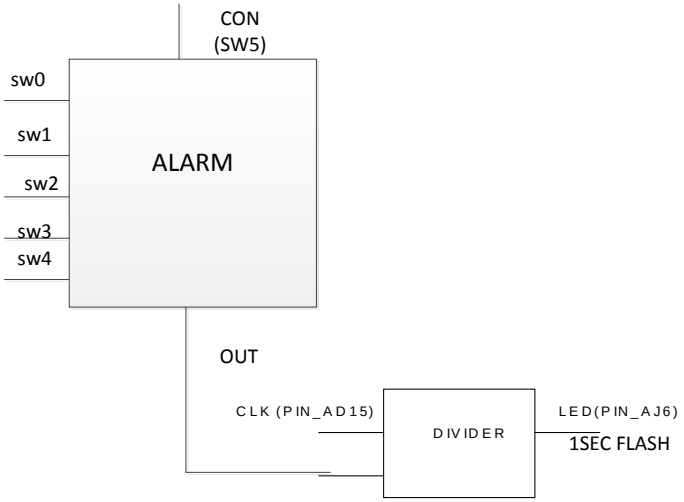
1. Projektovati kolo treptućeg svijetla koje se pogoni klokom frekvencije 1Hz ili 2Hz na pinu LED. Ulaz je CLK 50MHz, a izlaz LEDR[0]. Frekvencija treperenja se bira parametrom SEL (SW1) a. Upotrebljava se DE2-70 ploča.



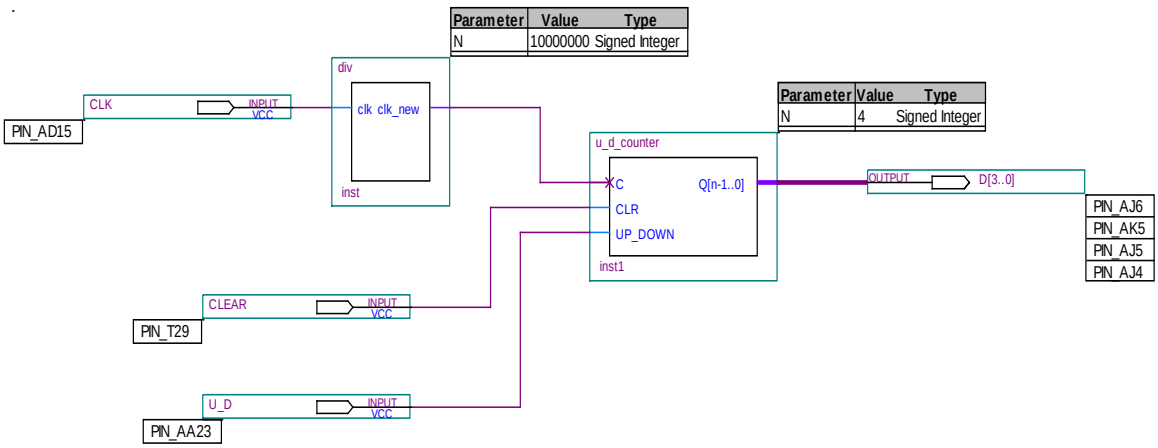
2. Projektovati kolo treptućeg svijetla koje se pogoni klokom frekvencije CLK 50MHz, a na izlazima LED1 i LED2 daje signal od 1Hz i 2Hz. START/STOP pogoni ili stopira rad kola (SW1). Upotrebljava se DE2-70 ploča. Izabrati led diode sa ploce po zelji.



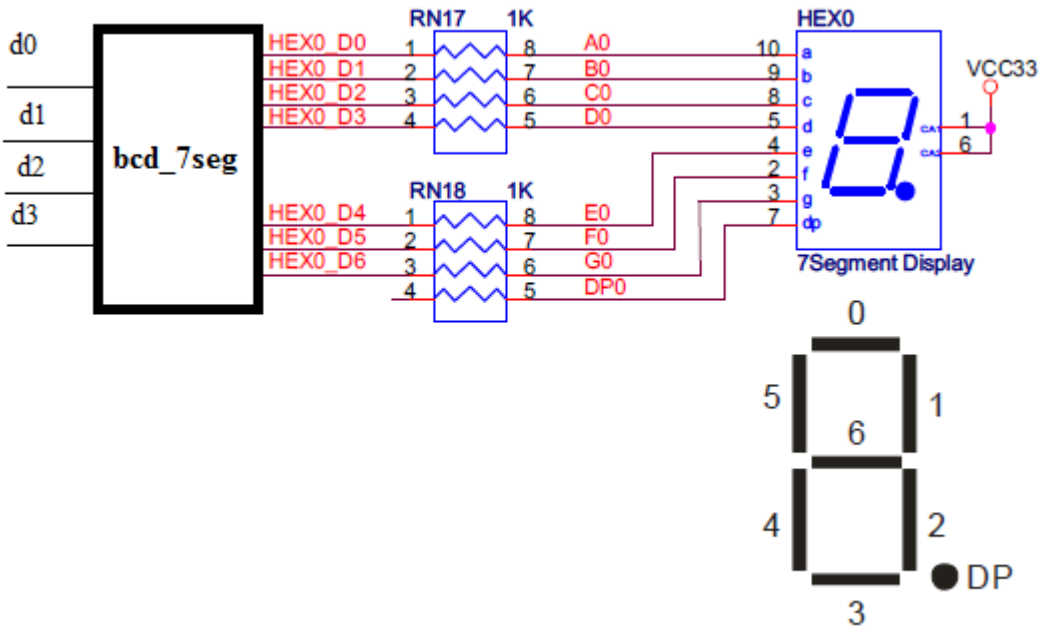
3. Projektovati kolo sifrarnika. Ulazi su priključeni na SW0 (PIN_AA23), SW1 (PIN_AB26), SW2 (PIN_AB25), SW3 (PIN_AC27) prekidače koji simuliraju dovodjenje 0 ili 1. SW4(PIN_AC26) je prekidač koji dozvoljava rad Sistema. SW4 =1, sifrarnik radi ili obrnuto. Postoje 2 izlaza OUT1(PIN_AJ6) i OUT2(PIN_AK5) koji pokazuju da li je sifra u redu ili nije, a vezani su na odgovarajuće LED diode.
4. Projektovati kolo alarma za automobil. Alarm je aktivan sa CON (SW5). SW0-SW4 simuliraju otvorena vrata. Izlaz OUT sa logickom 1 omogucava rad flash kola koje predstavlja divider sa izlaznom frekvencijom od 1Hz. Ako su bilo koja od vrata otvorena izlazna LED dioda “blinkuje” ili obrnuto.



5. Projektovati kolo brojača koji se pogoni klokom frekvencije 1Hz, a moze da broji na gore ili na dolje, sto zavisi od stanja ulaza U_D. Brojac ima i asinhroni clear ulaz, CLEAR. Izlazi brojaca, D[3..0] su vezani na LED diode na DE2-70 ploči.
   U_D: SW0
   CLEAR: KEY0
   CLK: 50MHz clock
   D[0]: LEDR0, D[0]: LEDR1, D[0]: LEDR2, D[0]: LEDR3,



6. Projektovati kolo bcd_7seg dekodera. d0, d1, d2, d3 ulazi su priključeni na SW0, SW1, SW2, SW3 prekidače koji simuliraju dovodjenje 0 ili 1. a, b,c,d, e,f,g izlazi su prilključeni na odgovarajuće segmente displeja. Raspored pinova čipa na koje su vezani odgovarajući ulazi i izlazi



	a	Output	PIN_AE8
	b	Output	PIN_AF9
	c	Output	PIN_AH9
	d	Output	PIN_AD10
	d0	Input	PIN_AA23
	d1	Input	PIN_AB26
	d2	Input	PIN_AB25
	d3	Input	PIN_AC27
	e	Output	PIN_AF10
	f	Output	PIN_AD11
	g	Output	PIN_AD12