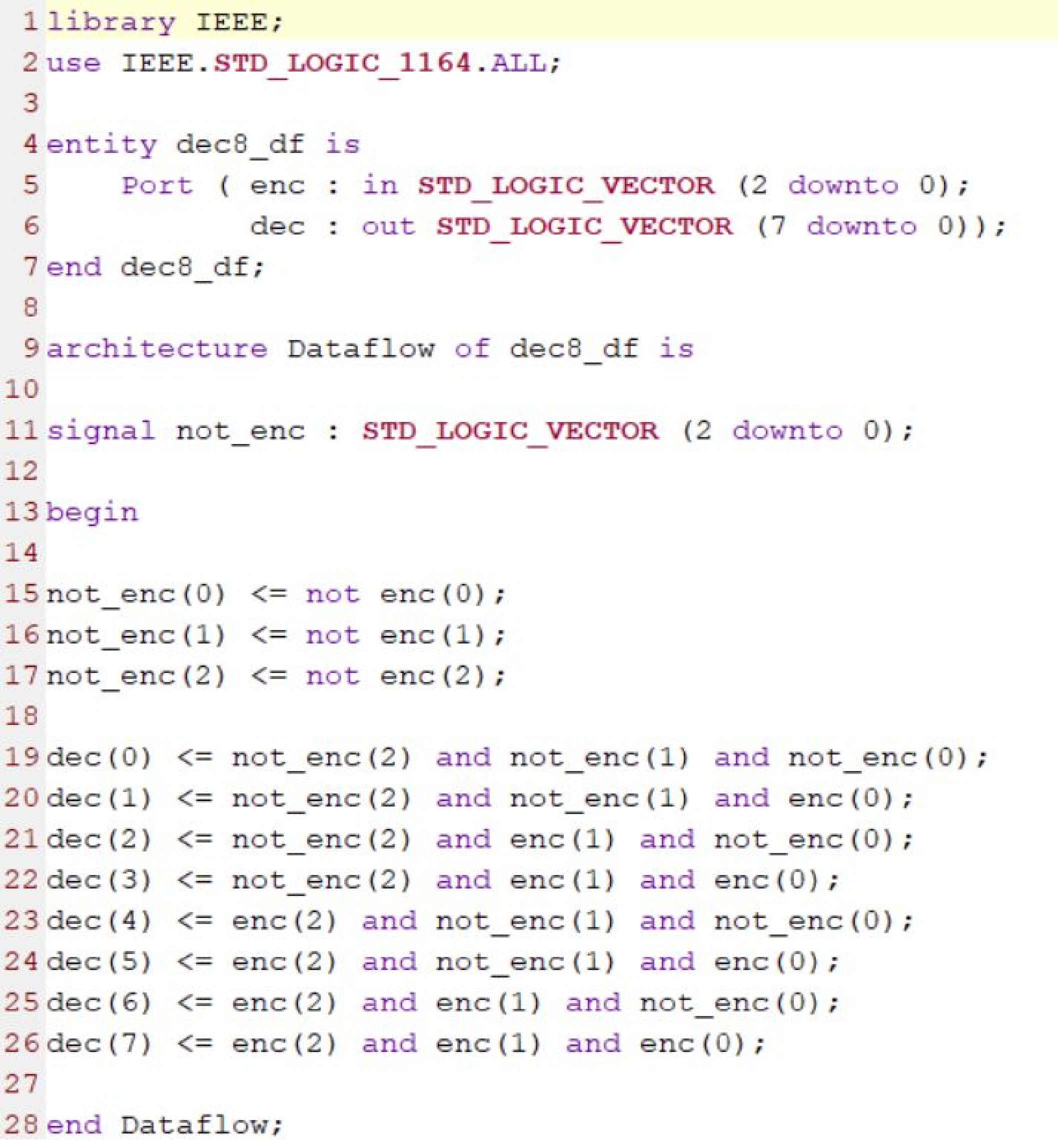
Αναφορά 1ης Εργαστηριακής Άσκησης

Ομάδα Β 10

Νικόλαος Τζιρής 0313425
Σταυρογιαννης Νεκταριος-Χρηστος 03113555
Κατσαρος Περικλης 03113302

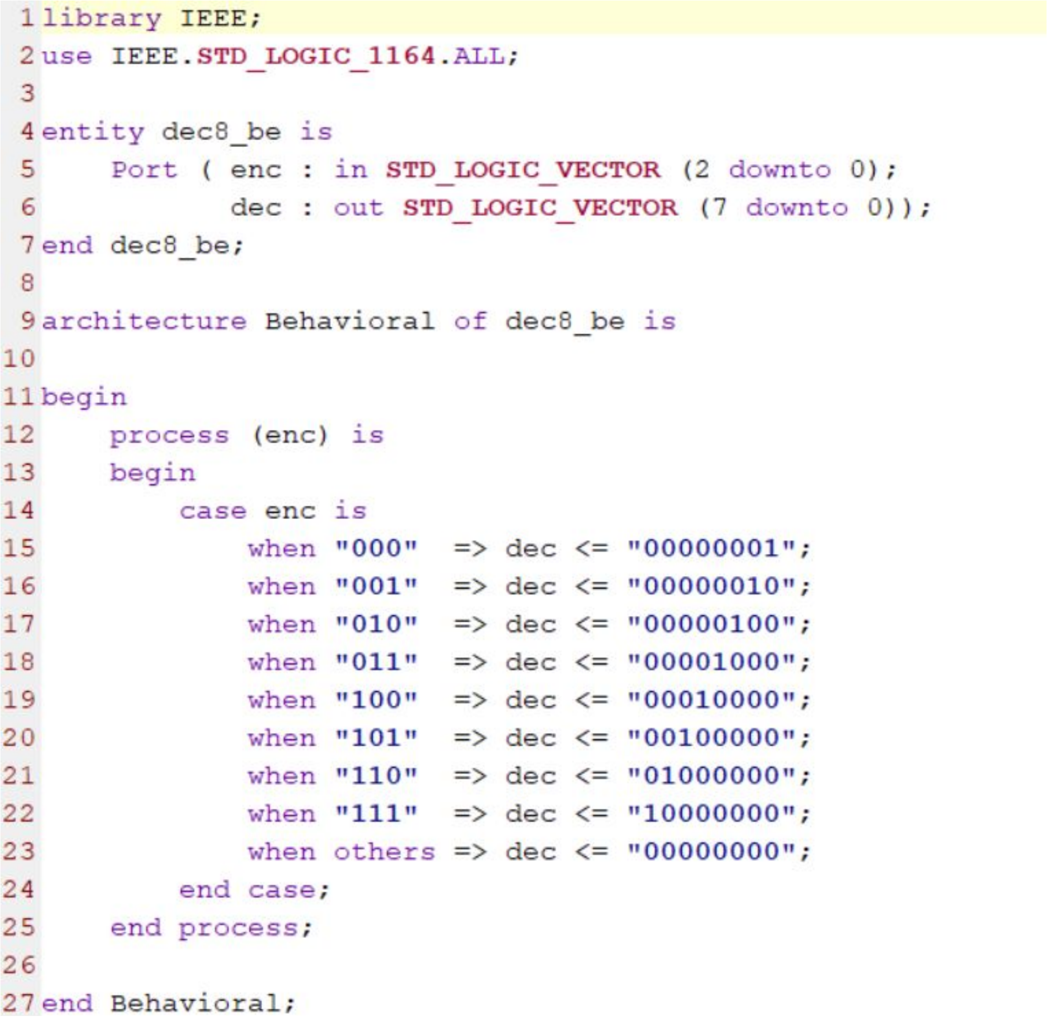
A.2

Υλοποίηση με αρχιτεκτονική **dataflow**:



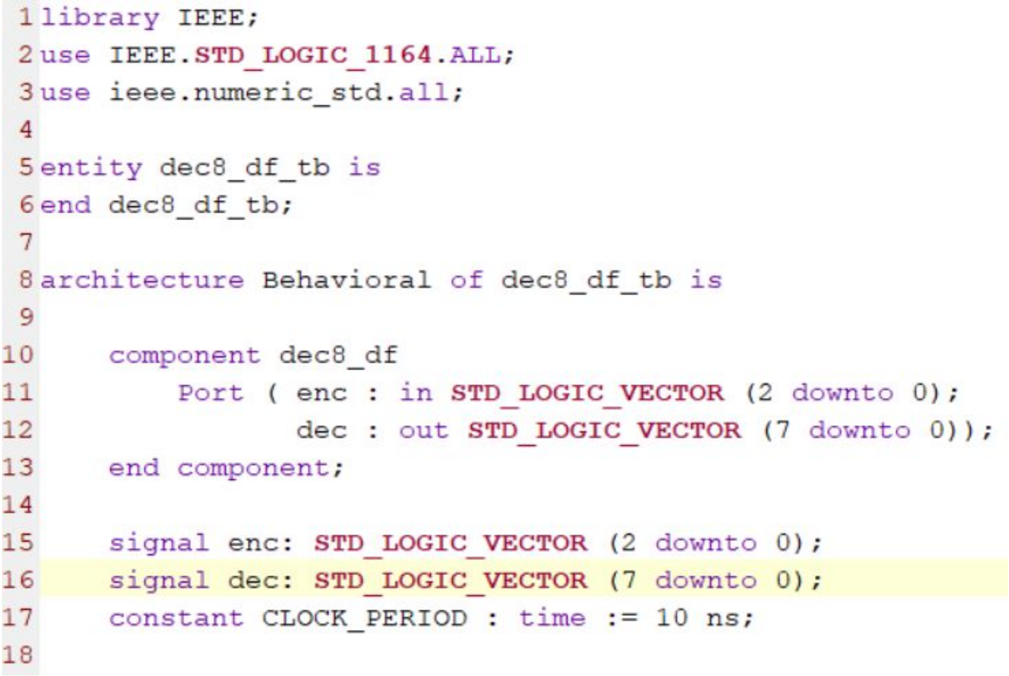
```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity dec8_df is
    Port ( enc : in STD_LOGIC_VECTOR (2 downto 0);
           dec : out STD_LOGIC_VECTOR (7 downto 0));
end dec8_df;

architecture Dataflow of dec8_df is

signal not_enc : STD_LOGIC_VECTOR (2 downto 0);

begin

not_enc(0) <= not enc(0);
not_enc(1) <= not enc(1);
not_enc(2) <= not enc(2);

dec(0) <= not_enc(2) and not_enc(1) and not_enc(0);
dec(1) <= not_enc(2) and not_enc(1) and enc(0);
dec(2) <= not_enc(2) and enc(1) and not_enc(0);
dec(3) <= not_enc(2) and enc(1) and enc(0);
dec(4) <= enc(2) and not_enc(1) and not_enc(0);
dec(5) <= enc(2) and not_enc(1) and enc(0);
dec(6) <= enc(2) and enc(1) and not_enc(0);
dec(7) <= enc(2) and enc(1) and enc(0);

end Dataflow;
```

Υλοποίηση με αρχιτεκτονική **behavioral**:

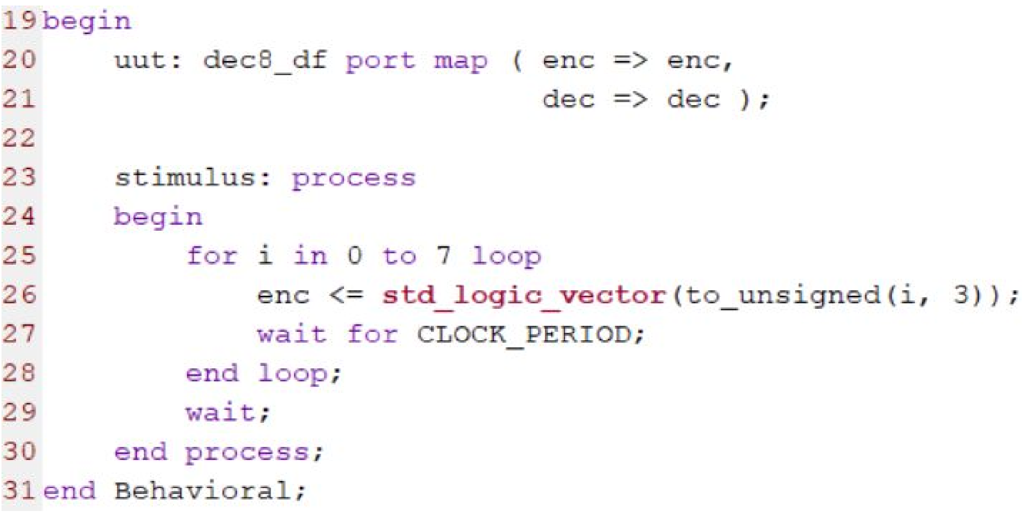


```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity dec8_be is
    Port ( enc : in STD_LOGIC_VECTOR (2 downto 0);
           dec : out STD_LOGIC_VECTOR (7 downto 0));
end dec8_be;

architecture Behavioral of dec8_be is

begin
    process (enc) is
    begin
        case enc is
            when "000"  => dec <= "00000001";
            when "001"  => dec <= "00000010";
            when "010"  => dec <= "00000100";
            when "011"  => dec <= "00001000";
            when "100"  => dec <= "00010000";
            when "101"  => dec <= "00100000";
            when "110"  => dec <= "01000000";
            when "111"  => dec <= "10000000";
            when others => dec <= "00000000";
        end case;
    end process;

end Behavioral;
```

Υλοποίηση testbench για την αρχιτεκτονική **dataflow**:

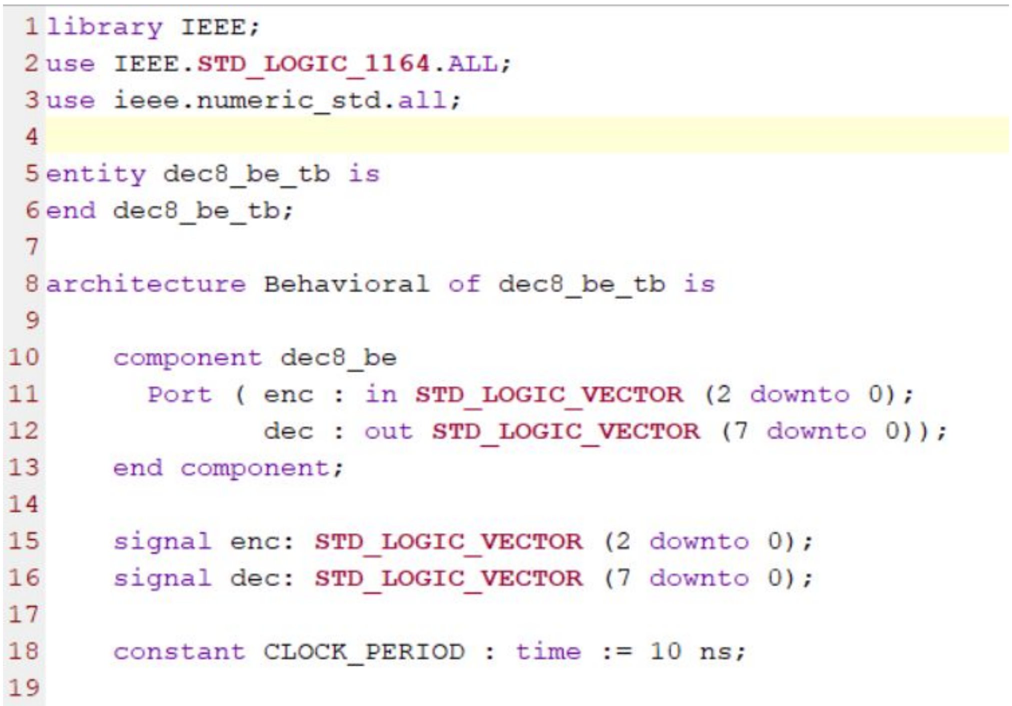


```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.numeric_std.all;

entity dec8_df_tb is
end dec8_df_tb;

architecture Behavioral of dec8_df_tb is

    component dec8_df
        Port ( enc : in STD_LOGIC_VECTOR (2 downto 0);
               dec : out STD_LOGIC_VECTOR (7 downto 0));
    end component;

    signal enc: STD_LOGIC_VECTOR (2 downto 0);
    signal dec: STD_LOGIC_VECTOR (7 downto 0);
    constant CLOCK_PERIOD : time := 10 ns;

```

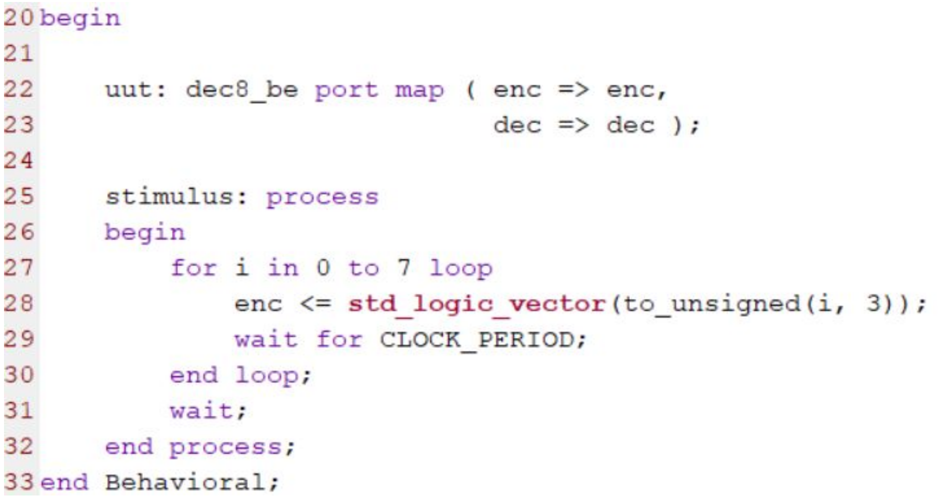


```
begin
    uut: dec8_df port map ( enc => enc,
                            dec => dec );

    stimulus: process
    begin
        for i in 0 to 7 loop
            enc <= std_logic_vector(to_unsigned(i, 3));
            wait for CLOCK_PERIOD;
        end loop;
        wait;
    end process;
end Behavioral;
```

Υλοποίηση testbench για **behavioral**:

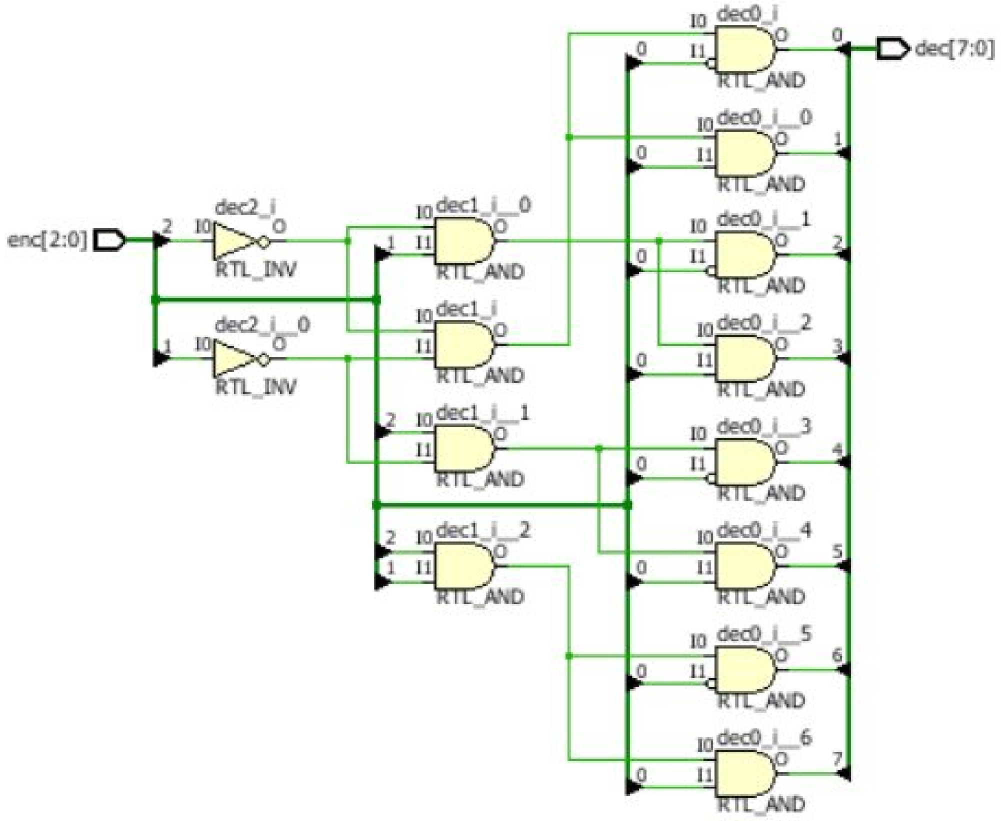


```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.numeric_std.all;

entity dec8_be_tb is
end dec8_be_tb;

architecture Behavioral of dec8_be_tb is

    component dec8_be
      Port ( enc : in STD_LOGIC_VECTOR (2 downto 0);
             dec : out STD_LOGIC_VECTOR (7 downto 0));
    end component;

    signal enc: STD_LOGIC_VECTOR (2 downto 0);
    signal dec: STD_LOGIC_VECTOR (7 downto 0);

    constant CLOCK_PERIOD : time := 10 ns;

```

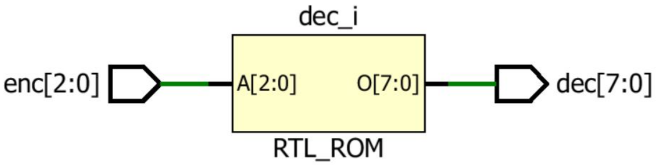


```vhdl
begin

    uut: dec8_be port map ( enc => enc,
                            dec => dec );

    stimulus: process
    begin
        for i in 0 to 7 loop
            enc <= std_logic_vector(to_unsigned(i, 3));
            wait for CLOCK_PERIOD;
        end loop;
        wait;
    end process;
end Behavioral;
```

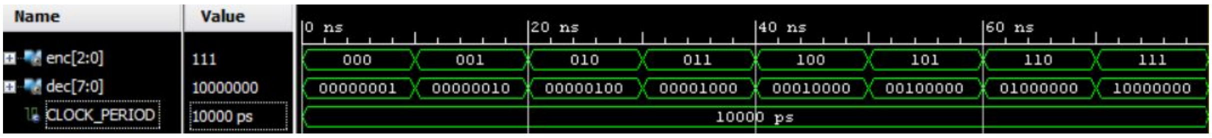
RTL σχήμα για την αρχιτεκτονική **dataflow**:



RTL σχήμα για την αρχιτεκτονική **behavioral**:



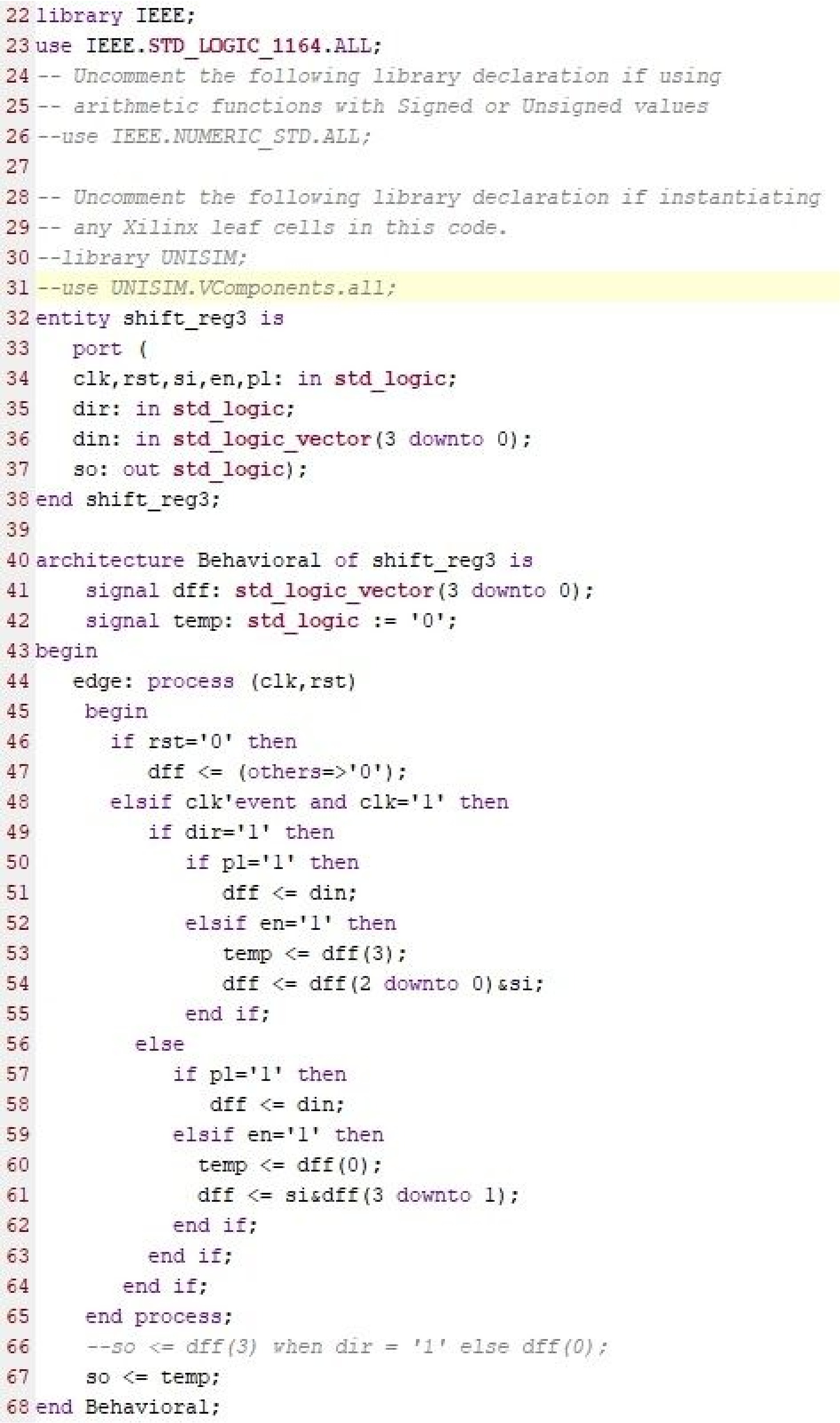
Η προσομοίωση και για τις δύο αρχιτεκτονικές δίνει τα ίδια αποτελέσματα. Αυτό είναι αναμενόμενο γιατί και οι δύο περιγράφουν ένα δυαδικό αποκωδικοποιητή 3 σε 8. Παρακάτω το αποτέλεσμα της προσομοίωσης:



```
Name          Value      |0 ns                  |20 ns                 |40 ns                 |60 ns
enc[2:0]      111        000        001        010        011        100        101        110        111
dec[7:0]      10000000   00000001   00000010   00000100   00001000   00010000   00100000   01000000   10000000
CLOCK_PERIOD  10000 ps                                    10000 ps
```

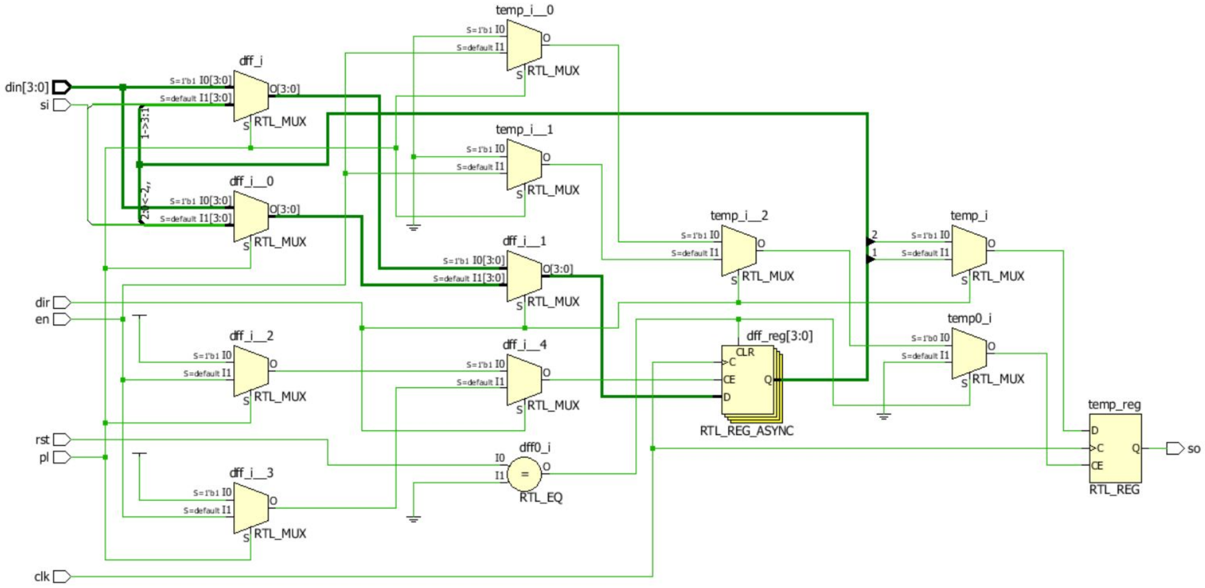
B2

Υλοποίηση:



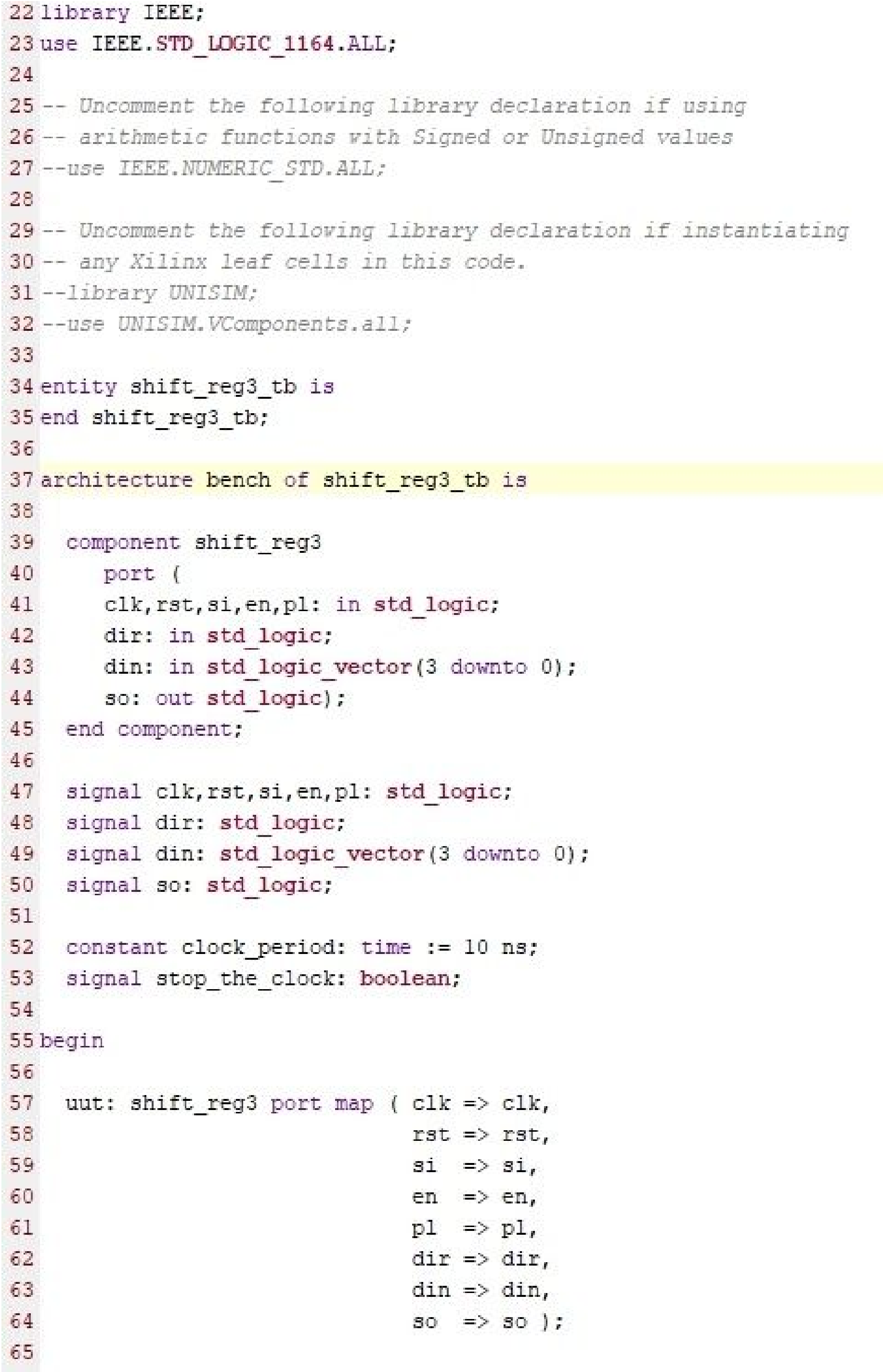
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;
entity shift_reg3 is
   port (
   clk,rst,si,en,pl: in std_logic;
   dir: in std_logic;
   din: in std_logic_vector(3 downto 0);
   so: out std_logic);
end shift_reg3;

architecture Behavioral of shift_reg3 is
    signal dff: std_logic_vector(3 downto 0);
    signal temp: std_logic := '0';
begin
   edge: process (clk,rst)
    begin
      if rst='0' then
         dff <= (others=>'0');
      elsif clk'event and clk='1' then
         if dir='1' then
            if pl='1' then
               dff <= din;
            elsif en='1' then
               temp <= dff(3);
               dff <= dff(2 downto 0)&si;
            end if;
        else
           if pl='1' then
              dff <= din;
           elsif en='1' then
             temp <= dff(0);
             dff <= si&dff(3 downto 1);
           end if;
         end if;
       end if;
    end process;
    --so <= dff(3) when dir = '1' else dff(0);
    so <= temp;
end Behavioral;
```

Το κύκλωμα που προκύπτει είναι:

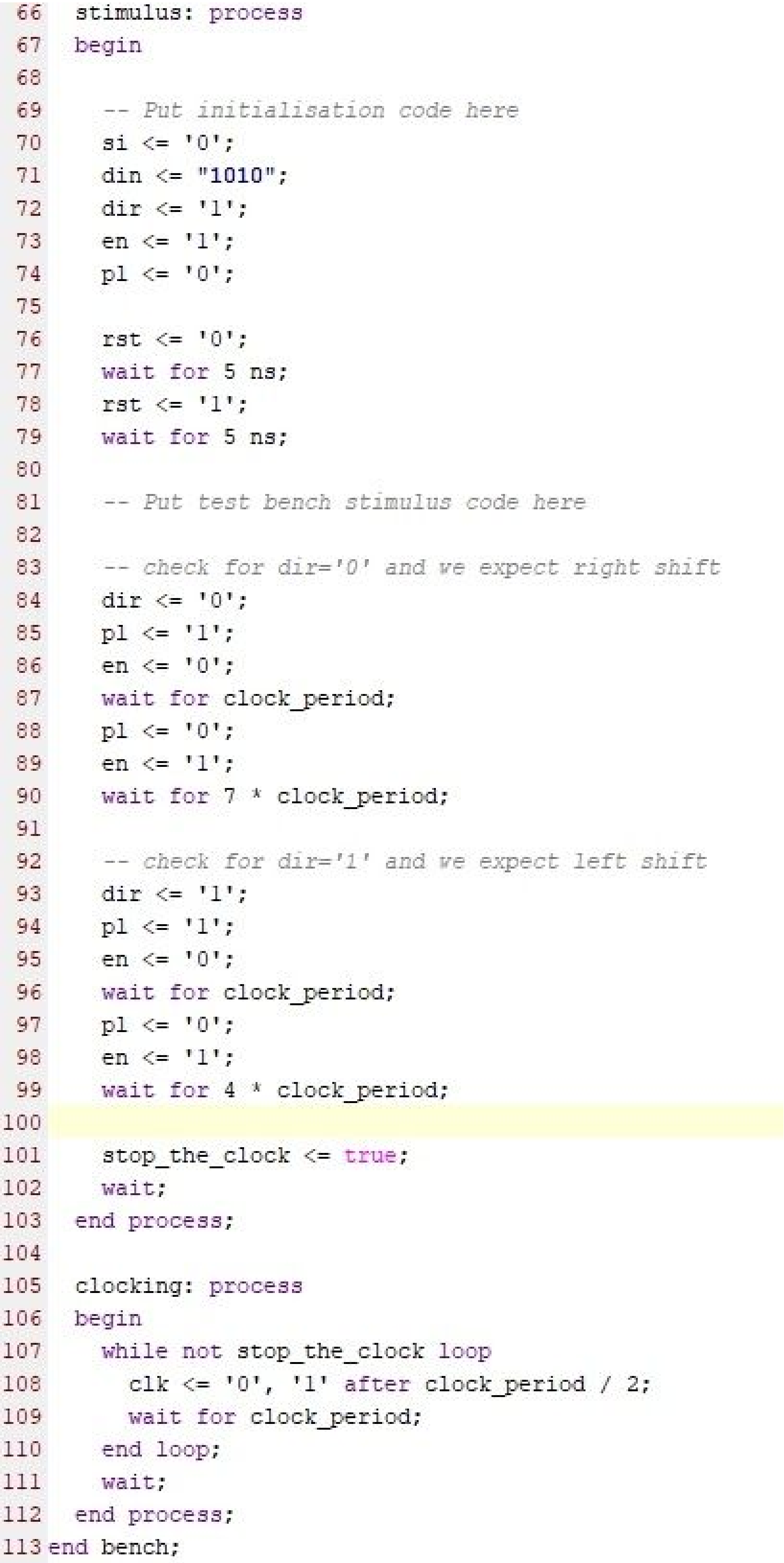


οι διαφορές με το σχήμα 2.8 της εκφώνησης είναι ότι έχουμε περισσότερους multiplexers. Αυτό είναι αναμενόμενο καθώς έχουμε υλοποιήσει και την δεξιά ολίσθηση.

Υλοποίηση testbench:

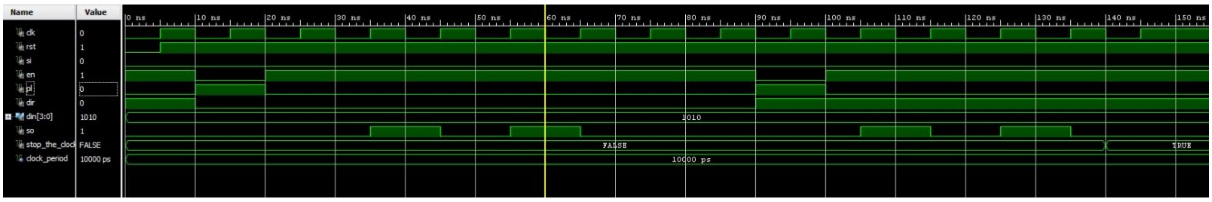


```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity shift_reg3_tb is
end shift_reg3_tb;

architecture bench of shift_reg3_tb is

  component shift_reg3
     port (
     clk,rst,si,en,pl: in std_logic;
     dir: in std_logic;
     din: in std_logic_vector(3 downto 0);
     so: out std_logic);
  end component;

  signal clk,rst,si,en,pl: std_logic;
  signal dir: std_logic;
  signal din: std_logic_vector(3 downto 0);
  signal so: std_logic;

  constant clock_period: time := 10 ns;
  signal stop_the_clock: boolean;

begin

  uut: shift_reg3 port map ( clk => clk,
                             rst => rst,
                             si  => si,
                             en  => en,
                             pl  => pl,
                             dir => dir,
                             din => din,
                             so  => so );

```



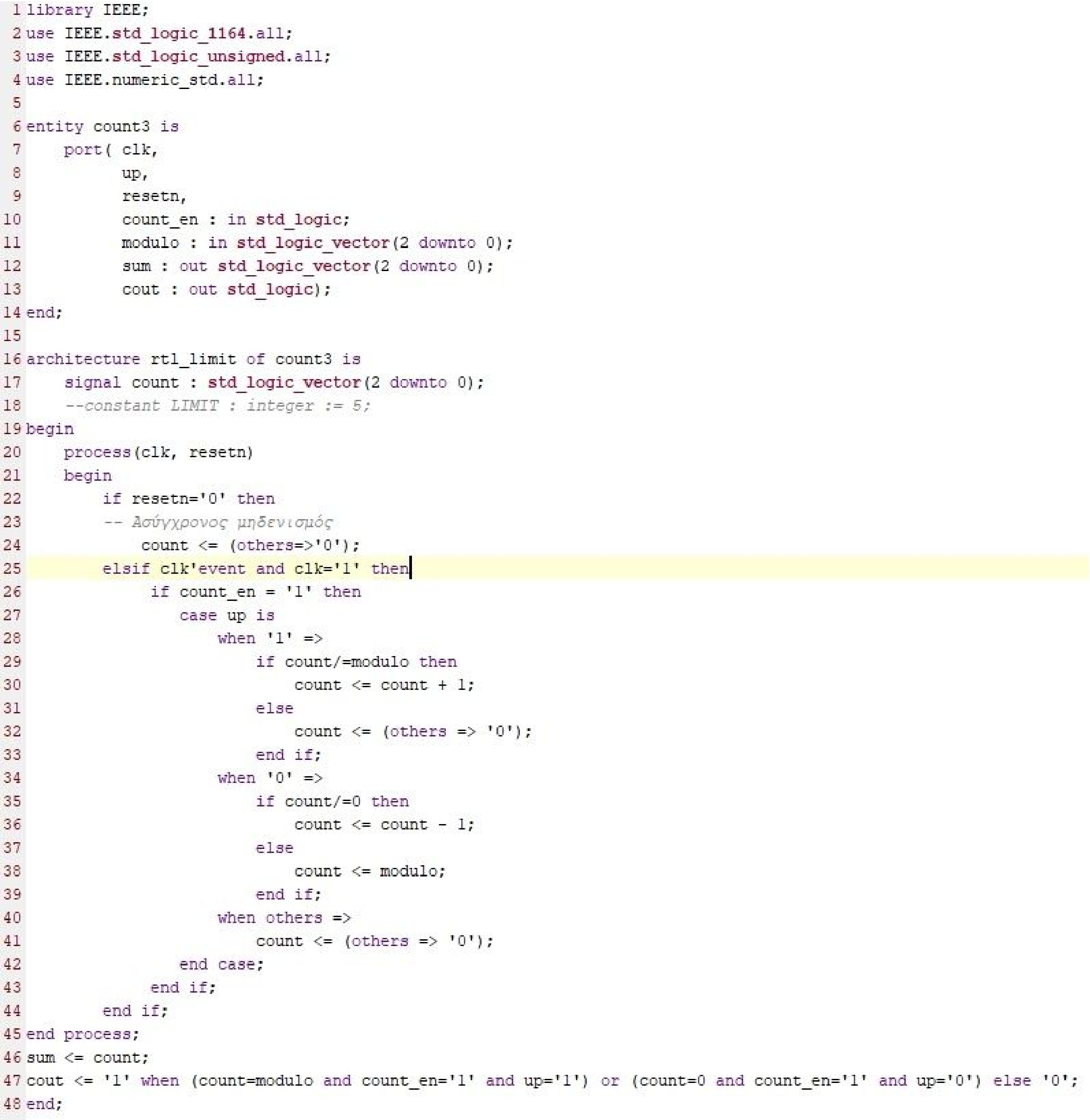
```
  stimulus: process
  begin

    -- Put initialisation code here
    si <= '0';
    din <= "1010";
    dir <= '1';
    en <= '1';
    pl <= '0';

    rst <= '0';
    wait for 5 ns;
    rst <= '1';
    wait for 5 ns;

    -- Put test bench stimulus code here

    -- check for dir='0' and we expect right shift
    dir <= '0';
    pl <= '1';
    en <= '0';
    wait for clock_period;
    pl <= '0';
    en <= '1';
    wait for 7 * clock_period;

    -- check for dir='1' and we expect left shift
    dir <= '1';
    pl <= '1';
    en <= '0';
    wait for clock_period;
    pl <= '0';
    en <= '1';
    wait for 4 * clock_period;

    stop_the_clock <= true;
    wait;
  end process;

  clocking: process
  begin
    while not stop_the_clock loop
      clk <= '0', '1' after clock_period / 2;
      wait for clock_period;
    end loop;
    wait;
  end process;
end bench;
```

Προσομοίωση:



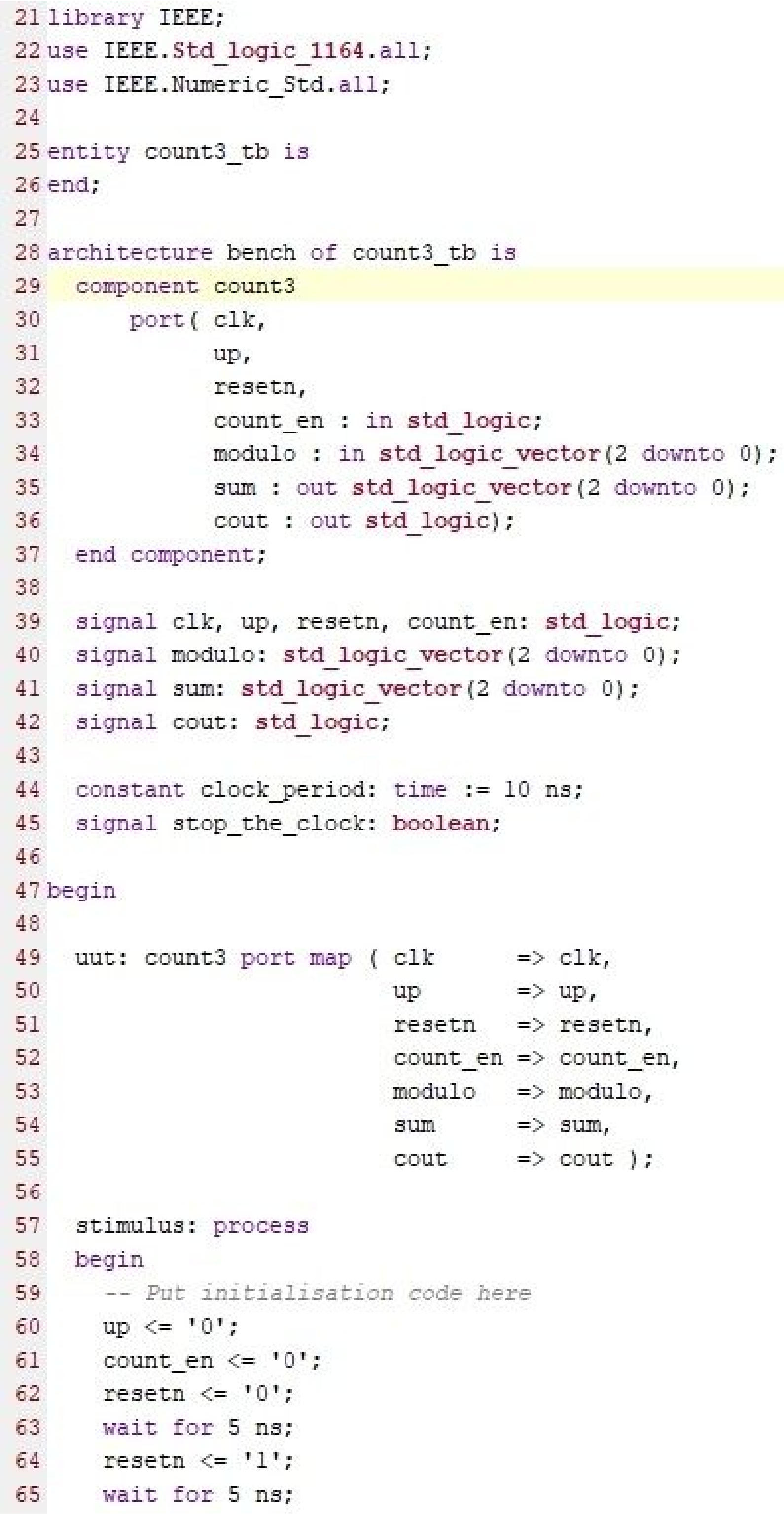
B3

Υλοποίηση:

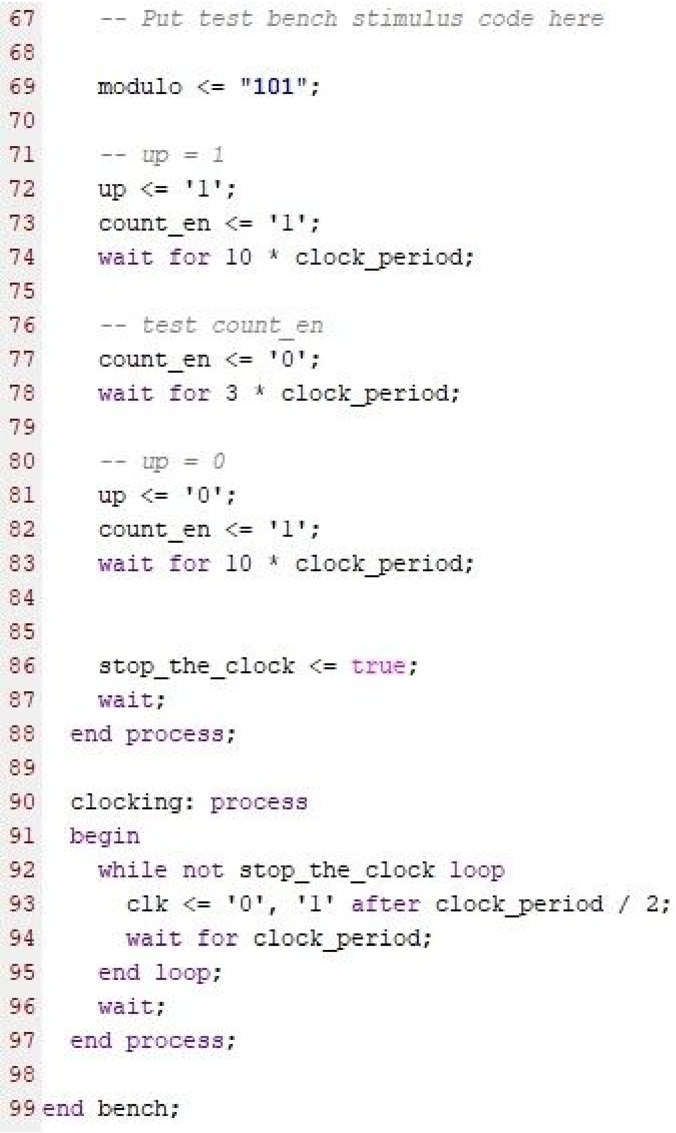


```vhdl
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;
use IEEE.numeric_std.all;

entity count3 is
    port( clk,
          up,
          resetn,
          count_en : in std_logic;
          modulo : in std_logic_vector(2 downto 0);
          sum : out std_logic_vector(2 downto 0);
          cout : out std_logic);
end;

architecture rtl_limit of count3 is
    signal count : std_logic_vector(2 downto 0);
    --constant LIMIT : integer := 5;
begin
    process(clk, resetn)
    begin
        if resetn='0' then
        -- Ασύγχρονος μηδενισμός
            count <= (others=>'0');
        elsif clk'event and clk='1' then
             if count_en = '1' then
                case up is
                    when '1' =>
                        if count/=modulo then
                            count <= count + 1;
                        else
                            count <= (others => '0');
                        end if;
                    when '0' =>
                        if count/=0 then
                            count <= count - 1;
                        else
                            count <= modulo;
                        end if;
                    when others =>
                        count <= (others => '0');
                end case;
             end if;
        end if;
end process;
sum <= count;
cout <= '1' when (count=modulo and count_en='1' and up='1') or (count=0 and count_en='1' and up='0') else '0';
end;
```

Υλοποίηση testbench:

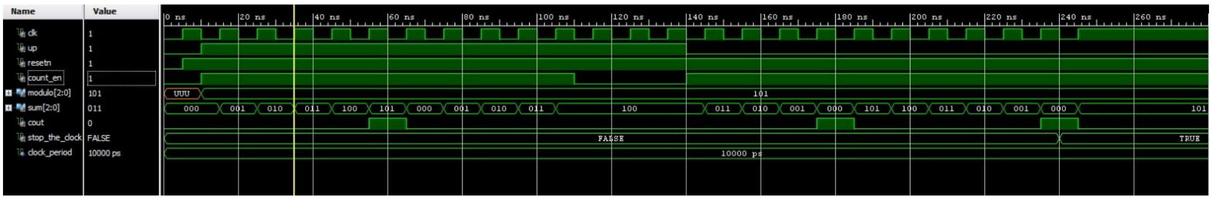


```
library IEEE;
use IEEE.Std_logic_1164.all;
use IEEE.Numeric_Std.all;

entity count3_tb is
end;

architecture bench of count3_tb is
  component count3
      port( clk,
            up,
            resetn,
            count_en : in std_logic;
            modulo : in std_logic_vector(2 downto 0);
            sum : out std_logic_vector(2 downto 0);
            cout : out std_logic);
  end component;

  signal clk, up, resetn, count_en: std_logic;
  signal modulo: std_logic_vector(2 downto 0);
  signal sum: std_logic_vector(2 downto 0);
  signal cout: std_logic;

  constant clock_period: time := 10 ns;
  signal stop_the_clock: boolean;

begin

  uut: count3 port map ( clk      => clk,
                         up       => up,
                         resetn   => resetn,
                         count_en => count_en,
                         modulo   => modulo,
                         sum      => sum,
                         cout     => cout );

  stimulus: process
  begin
    -- Put initialisation code here
    up <= '0';
    count_en <= '0';
    resetn <= '0';
    wait for 5 ns;
    resetn <= '1';
    wait for 5 ns;
```



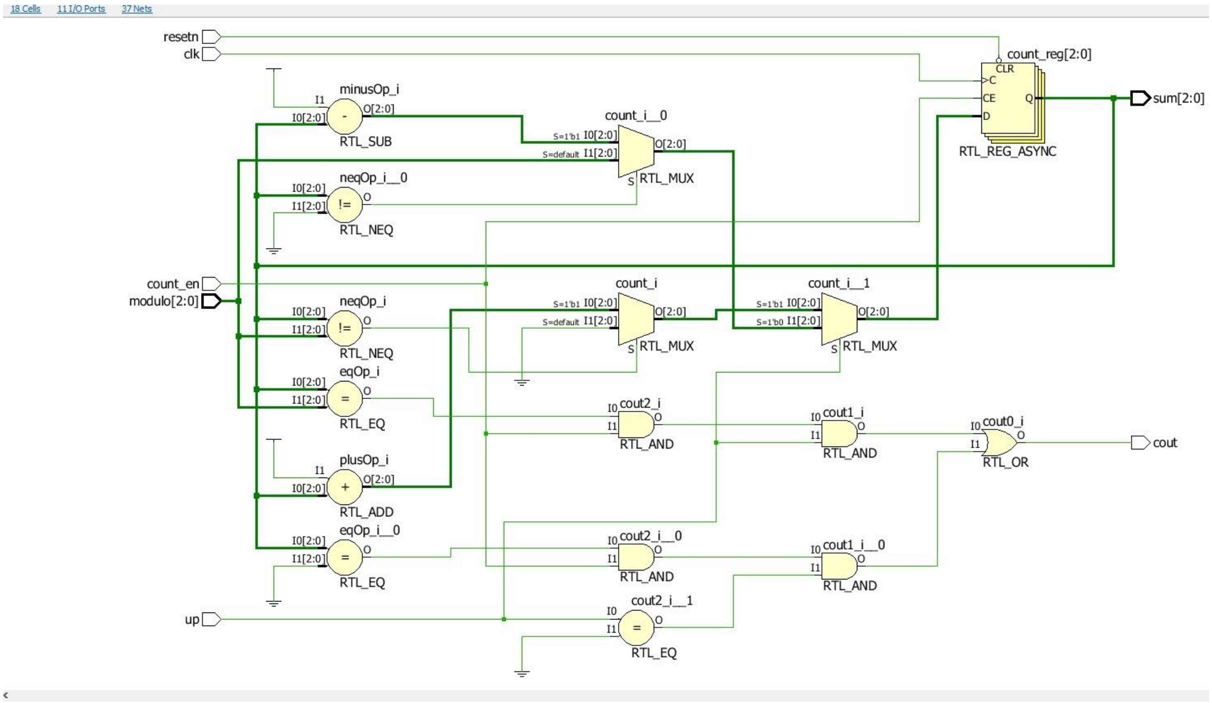
```vhdl
    -- Put test bench stimulus code here

    modulo <= "101";

    -- up = 1
    up <= '1';
    count_en <= '1';
    wait for 10 * clock_period;

    -- test count_en
    count_en <= '0';
    wait for 3 * clock_period;

    -- up = 0
    up <= '0';
    count_en <= '1';
    wait for 10 * clock_period;


    stop_the_clock <= true;
    wait;
  end process;

  clocking: process
  begin
    while not stop_the_clock loop
      clk <= '0', '1' after clock_period / 2;
      wait for clock_period;
    end loop;
    wait;
  end process;

end bench;
```

Προσομοίωση:



Name	Value
clk	1
up	1
resetn	1
count_en	1
modulo[2:0]	101
sum[2:0]	011
cout	0
stop_the_clock	FALSE
clock_period	10000 ps

Το κύκλωμα που μας δίνει ο synthesizer είναι το παρακάτω:



Παρατηρούμε οτι θα μπορούσε να σχεδιαστεί απλούστερα με το χέρι με τη χρήση 3 flip flops και ίσως μερικών mux για τη λογική up/down.