آزمایشگاه مدار منطقی

نویسنده:

علیرضا طباطبائیان(9723052)

Amirkabir University of Technology
(Tehran Polytechnic)

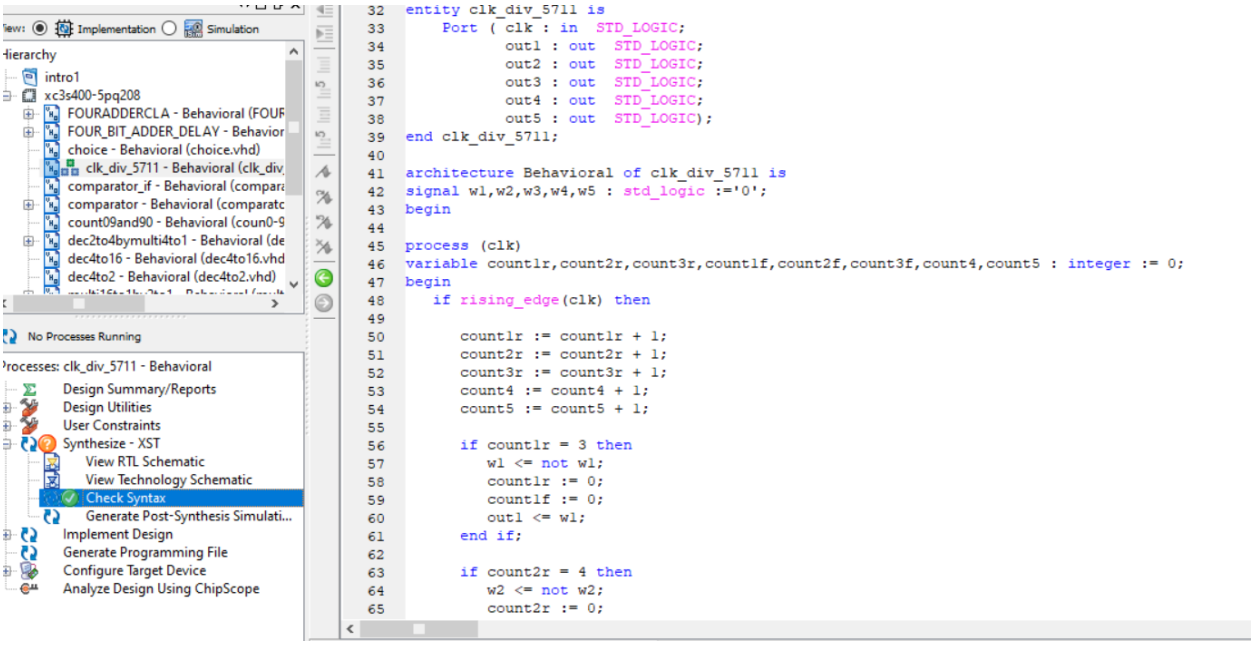
استاد آزمایشگاه:
دکتر علیرضا
طاهری نوید

THE PRESENT IS THEIRS, THE FUTURE, FOR
WHICH I REALLY WORK, IS MINE.

تمرین ۱:

برای تقسیم کلاک بصورت نرم افزاری باید یک شمارنده حساس به لبه بالا ساخت که هر n کلاک یکبار مقدار ۰ و ۱ را به هم تبدیل کند(توسط گیت not)

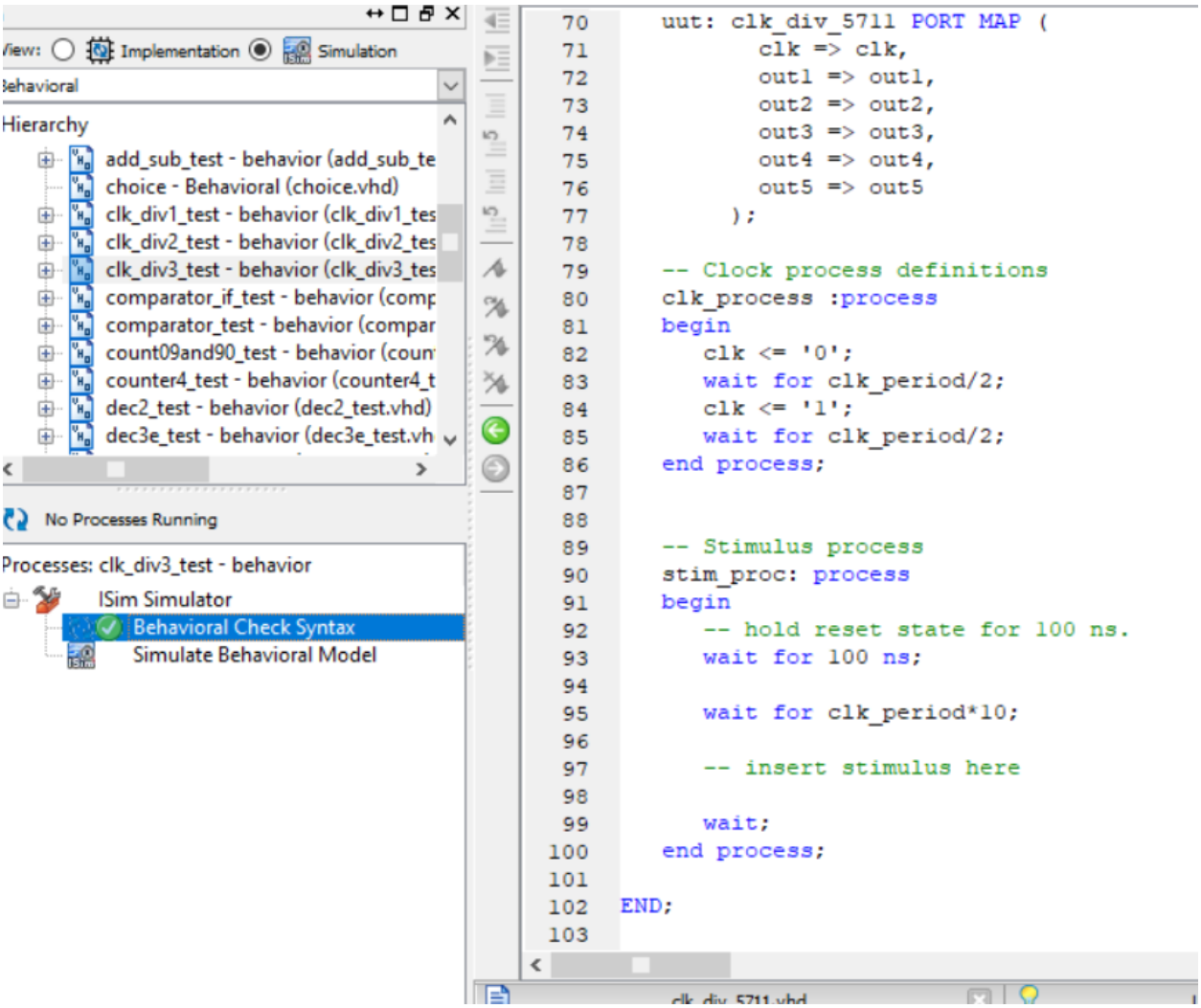
این روش برای تقسیم به اعداد زوج به راحتی قابل استفاده است اما برای اعداد فرد باید از یک ترفند استفاده کرد که این ترفند نیازمند استفاده از لبه پایین رونده کلاک نیز میباشد.



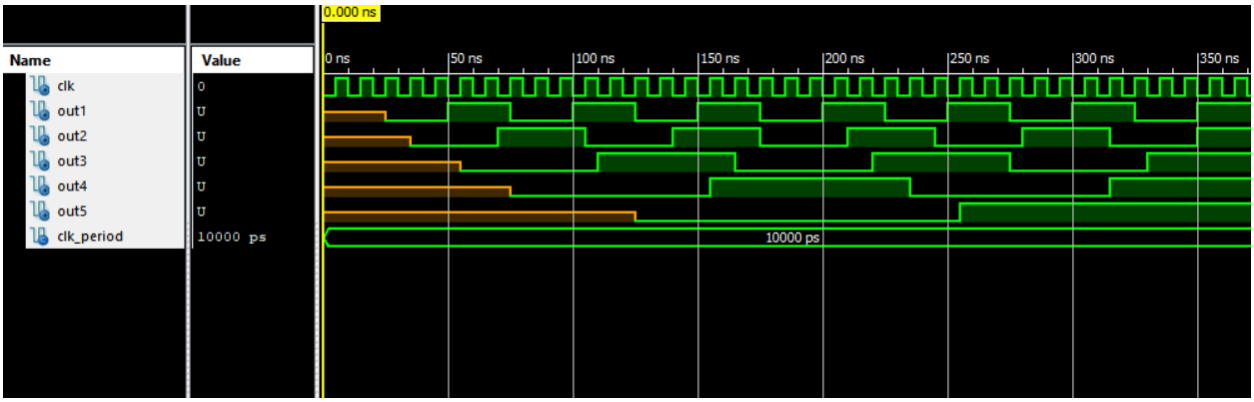
```
63          if count2r = 4 then
64             w2 <= not w2;
65             count2r := 0;
66             count2f := 0;
67             out2 <= w2;
68          end if;
69
70          if count3r = 6 then
71             w3 <= not w3;
72             count3r := 0;
73             count3f := 0;
74             out3 <= w3;
75          end if;
76
77          if count4 = 8 then
78             w4 <= not w4;
79             count4 := 0;
80             out4 <= w4;
81          end if;
82
83          if count5 = 13 then
84             w5 <= not w5;
85             count5 := 0;
86             out5 <= w5;
87          end if;
88
89       elsif falling_edge(clk) then
90
91          count1f := count1f + 1;
92          count2f := count2f + 1;
93          count3f := count3f + 1;
94
95          if count1f = 3 then
96             w1 <= not w1;
```

```
   elsif falling_edge(clk) then

      count1f := count1f + 1;
      count2f := count2f + 1;
      count3f := count3f + 1;

      if count1f = 3 then
         w1 <= not w1;
         count1f := 0;
         count1r := 0;
         out1 <= w1;
      end if;

      if count2f = 4 then
         w2 <= not w2;
         count2f := 0;
         count2r := 0;
         out2 <= w2;
      end if;

      if count3f = 6 then
         w3 <= not w3;
         count3f := 0;
         count3r := 0;
         out3 <= w3;
      end if;

   end if;
end process;

end Behavioral;
```

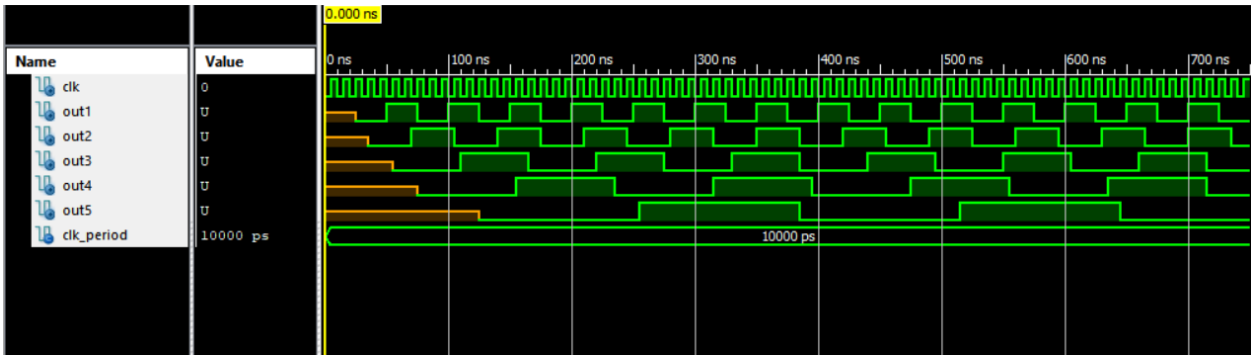
تست بنچ(که البته چیز خاصی در آن نیاز نبود بنویسیم):


View: Implementation Simulation
Behavioral
Hierarchy
add_sub_test - behavior (add_sub_te
choice - Behavioral (choice.vhd)
clk_div1_test - behavior (clk_div1_tes
clk_div2_test - behavior (clk_div2_tes
clk_div3_test - behavior (clk_div3_tes
comparator_if_test - behavior (comp
comparator_test - behavior (compar
count09and90_test - behavior (coun
counter4_test - behavior (counter4_t
dec2_test - behavior (dec2_test.vhd)
dec3e_test - behavior (dec3e_test.vh
No Processes Running
Processes: clk_div3_test - behavior
ISim Simulator
Behavioral Check Syntax
Simulate Behavioral Model
70 uut: clk_div_5711 PORT MAP (
71 clk => clk,
72 out1 => out1,
73 out2 => out2,
74 out3 => out3,
75 out4 => out4,
76 out5 => out5
77);
78
79 -- Clock process definitions
80 clk_process :process
81 begin
82 clk <= '0';
83 wait for clk_period/2;
84 clk <= '1';
85 wait for clk_period/2;
86 end process;
87
88
89 -- Stimulus process
90 stim_proc: process
91 begin
92 -- hold reset state for 100 ns.
93 wait for 100 ns;
94
95 wait for clk_period*10;
96
97 -- insert stimulus here
98
99 wait;
100 end process;
101
102 END;
103
clk_div_5711.vhd

نتایج:

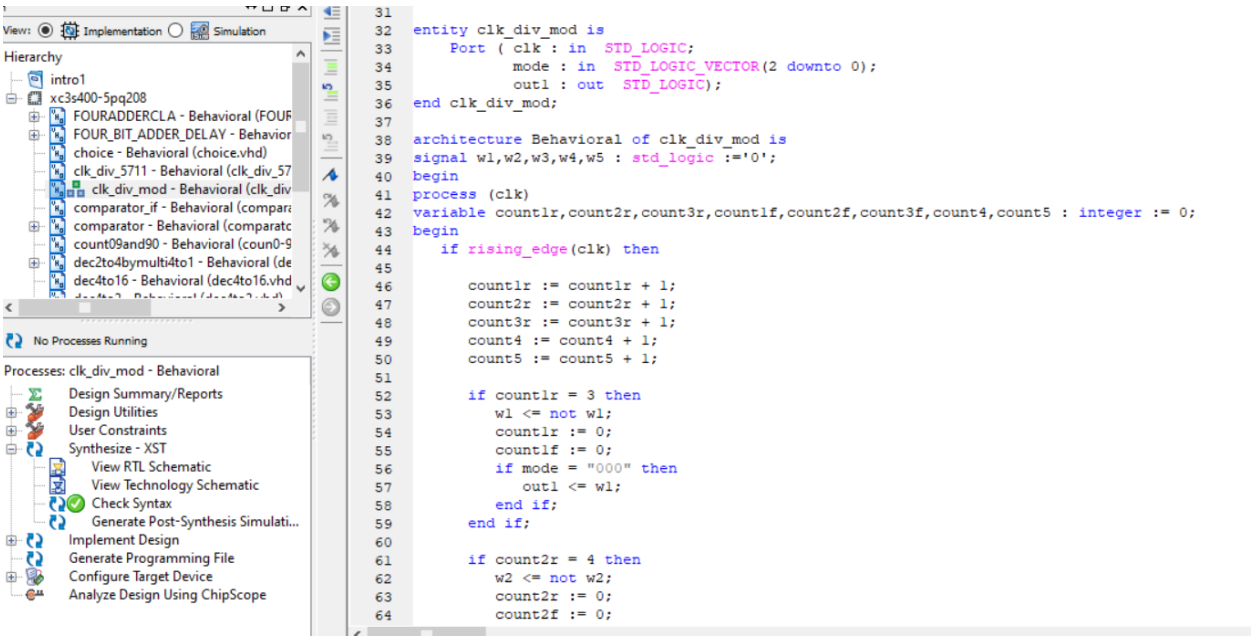


با کمی زوم بک:



تمرین ۲:

برای حالت mode **دار از همان کد قبلی استفاده میکنیم با این تفاوت که فقط یک خروجی داریم و قبل از هر اتصال** sig **به خروجی یک** if **قرار میدهیم.**



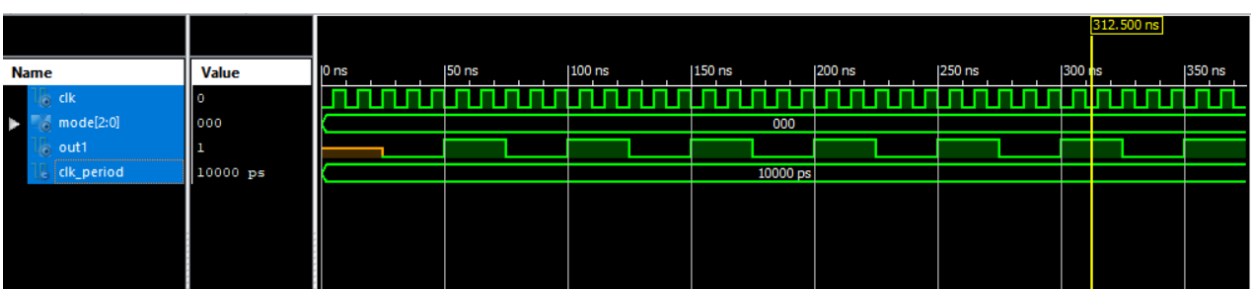
```
        if count2r = 4 then
           w2 <= not w2;
           count2r := 0;
           count2f := 0;
           if mode = "001" then
              out1 <= w2;
           end if;
        end if;

        if count3r = 6 then
           w3 <= not w3;
           count3r := 0;
           count3f := 0;
           if mode = "010" then
              out1 <= w3;
           end if;
        end if;

        if count4 = 8 then
           w4 <= not w4;
           count4 := 0;
           if mode = "011" then
              out1 <= w4;
           end if;
        end if;

        if count5 = 13 then
           w5 <= not w5;
           count5 := 0;
           if mode = "100" then
              out1 <= w5;
           end if;
        end if;
```

```
95      elsif falling_edge(clk) then
96
97         count1f := count1f + 1;
98         count2f := count2f + 1;
99         count3f := count3f + 1;
100
101        if count1f = 3 then
102           w1 <= not w1;
103           count1f := 0;
104           count1r := 0;
105           if mode = "000" then
106              out1 <= w1;
107           end if;
108        end if;
109
110        if count2f = 4 then
111           w2 <= not w2;
112           count2f := 0;
113           count2r := 0;
114           if mode = "001" then
115              out1 <= w2;
116           end if;
117        end if;
118
119        if count3f = 6 then
120           w3 <= not w3;
121           count3f := 0;
122           count3r := 0;
123           if mode = "010" then
124              out1 <= w3;
125           end if;
126        end if;
127
128     end if;
```

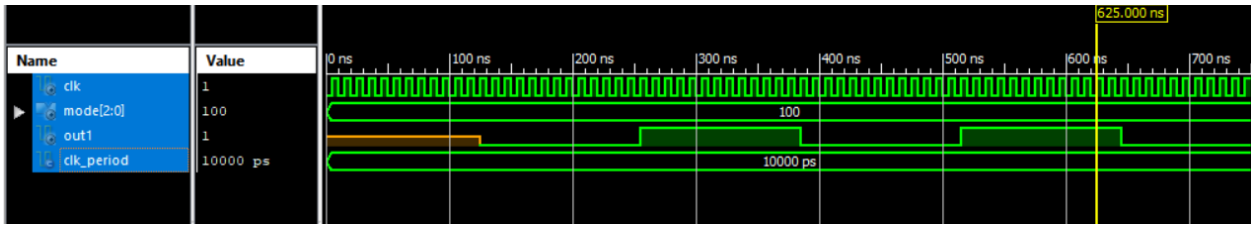
Language Templates

برای تست بنچ ۲ حالت مختلف mode **برابر ۰ و ۴ که به ترتیب کلاک را تقسیم به ۵ و ۲۶ میکنند امتحان میکنیم:**

```
61  BEGIN
62
63     -- Instantiate the Unit Under Test (U
64     uut: clk_div_mod PORT MAP (
65            clk => clk,
66            mode => mode,
67            out1 => out1
68          );
69
70     -- Clock process definitions
71     clk_process :process
72     begin
73        clk <= '0';
74        wait for clk_period/2;
75        clk <= '1';
76        wait for clk_period/2;
77     end process;
78
79
80     -- Stimulus process
81     stim_proc: process
82     begin
83        mode <= "000";
84        wait for 100 ns;
85
86        wait for clk_period*10;
87
88        -- insert stimulus here
89
90        wait;
91     end process;
92
93  END;
```

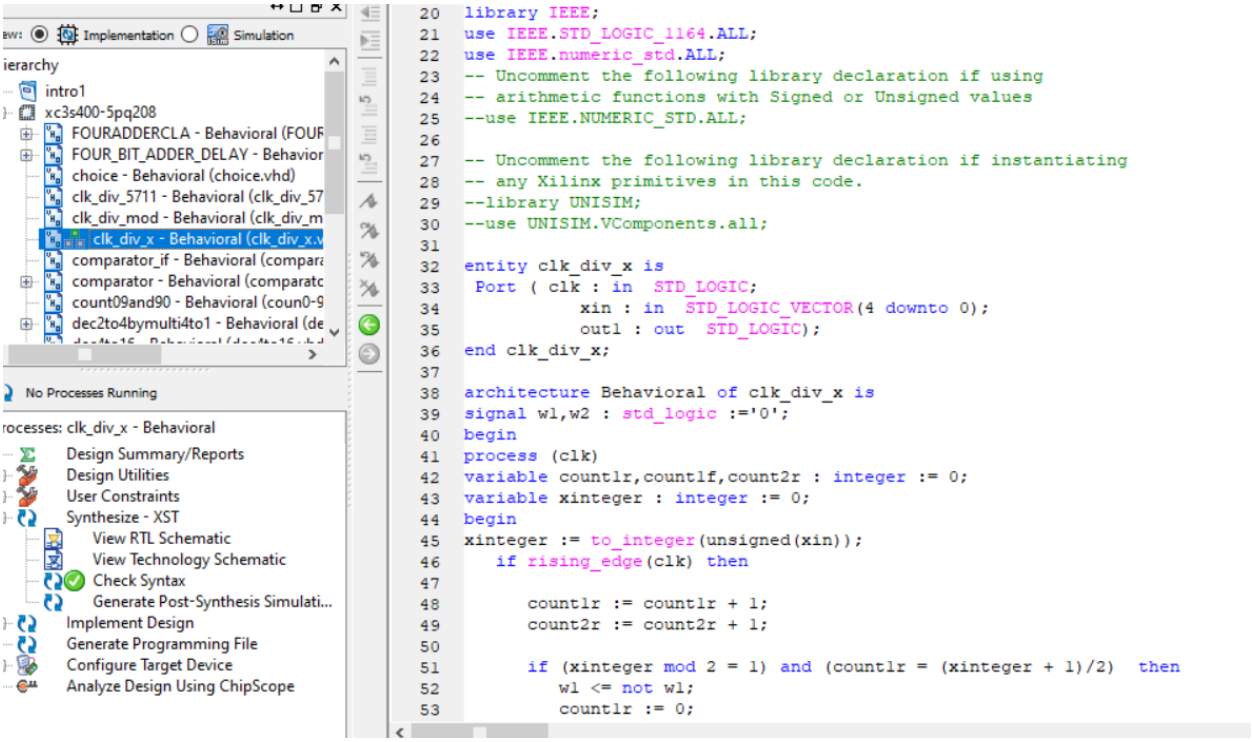


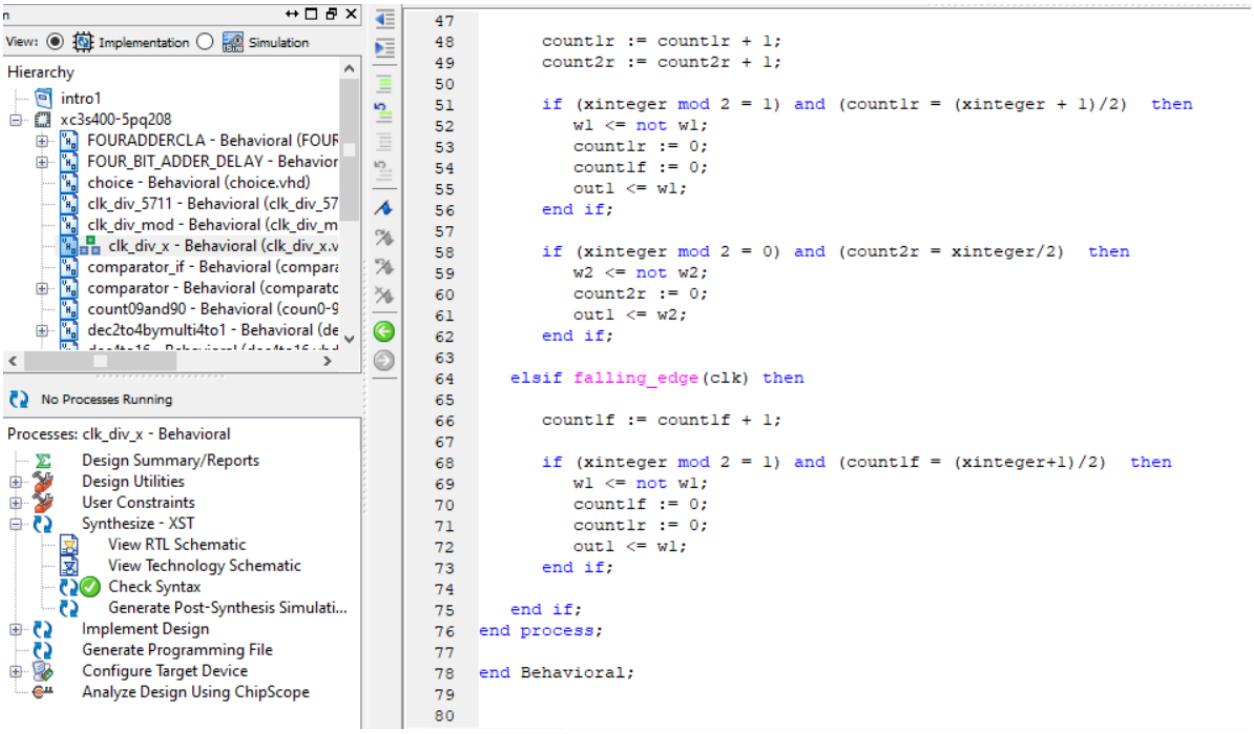
```
BEGIN

   -- Instantiate the Unit Under Tes
   uut: clk_div_mod PORT MAP (
          clk => clk,
          mode => mode,
          out1 => out1
        );

   -- Clock process definitions
   clk_process :process
   begin
      clk <= '0';
      wait for clk_period/2;
      clk <= '1';
      wait for clk_period/2;
   end process;


   -- Stimulus process
   stim_proc: process
   begin
      mode <= "100";
      wait for 100 ns;

      wait for clk_period*10;

      -- insert stimulus here

      wait;
   end process;

END;
```

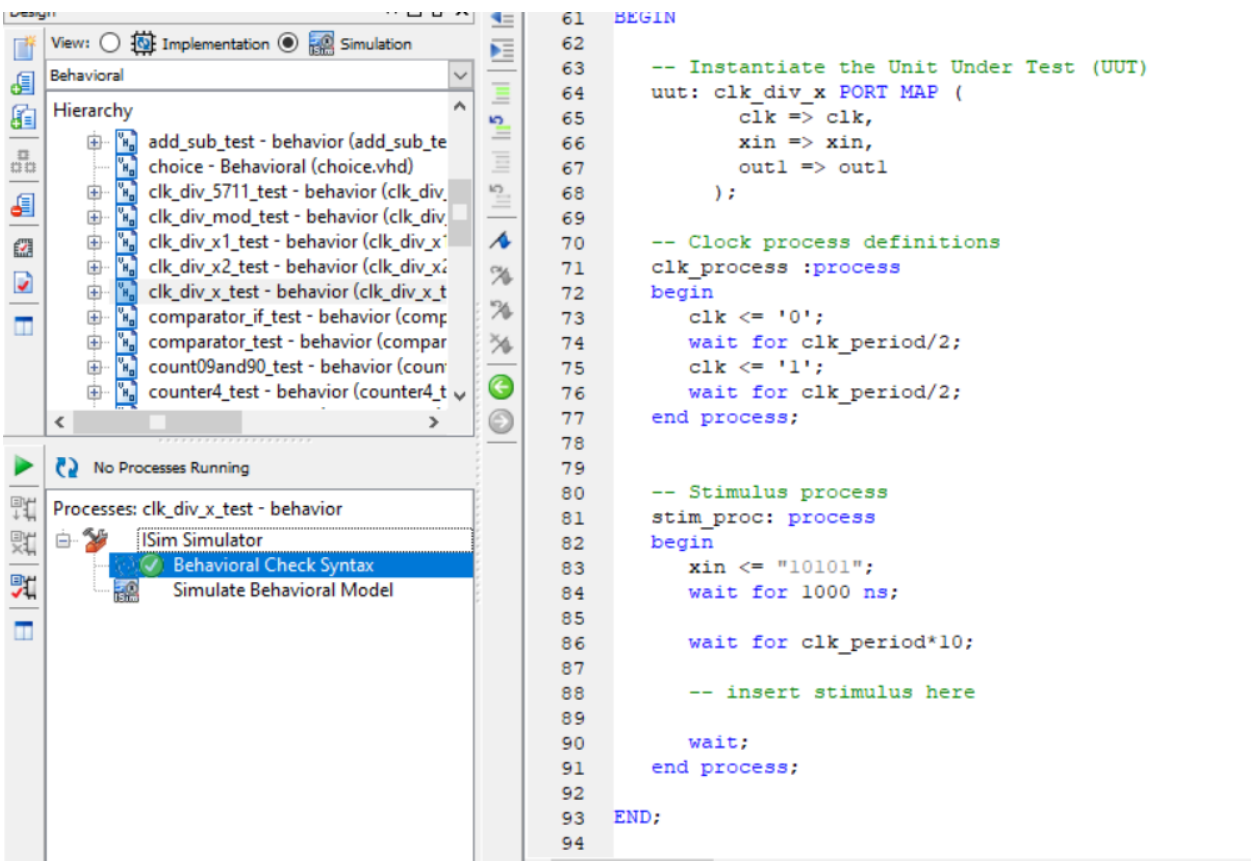


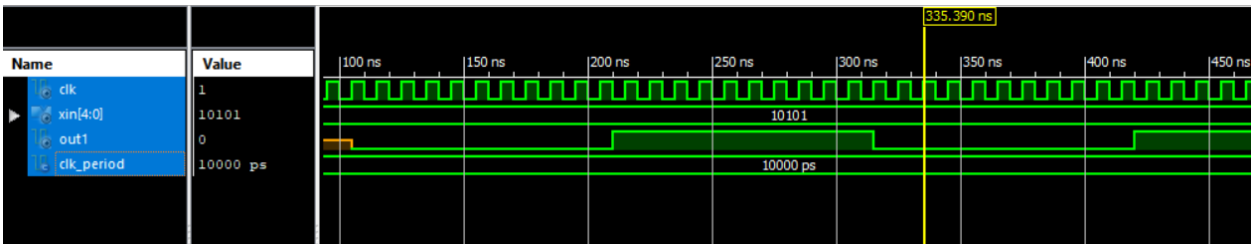
امتیازی:

عدد xin **را در ورودی دریافت کرده(میتوان بیت های بیشتری به آن اختصاص داد اما در اینجا ۵ بیت تعریف شده اما هر مقدار بیتی میتوان به آن اختصاص داد)و سپس توسط تابع** to_integer **آن را به** integer **تبدیل میکنیم و سپس زوج و فرد بودن آن را توسط تابع** mod **تشخیص میدهیم و سپس طبق قاعده بخش های قبلی عمل میکنیم.**

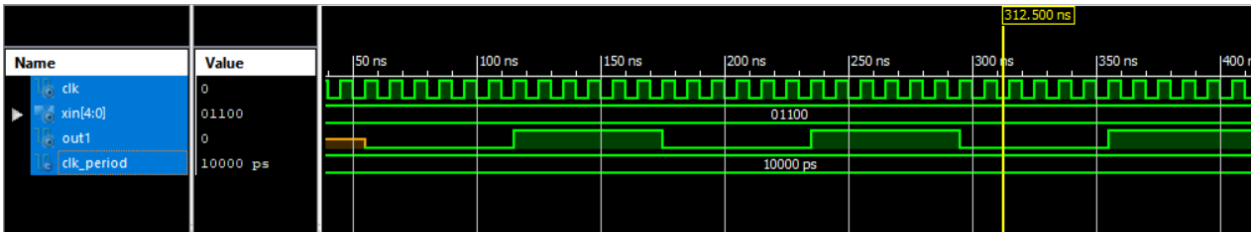



View: Implementation Simulation
Hierarchy
intro1
xc3s400-5pq208
FOURADDERCLA - Behavioral (FOUR
FOUR_BIT_ADDER_DELAY - Behavior
choice - Behavioral (choice.vhd)
clk_div_5711 - Behavioral (clk_div_57
clk_div_mod - Behavioral (clk_div_m
clk_div_x - Behavioral (clk_div_x.v
comparator_if - Behavioral (compara
comparator - Behavioral (comparatc
count09and90 - Behavioral (coun0-9
dec2to4bymulti4to1 - Behavioral (de
No Processes Running
Processes: clk_div_x - Behavioral
Design Summary/Reports
Design Utilities
User Constraints
Synthesize - XST
View RTL Schematic
View Technology Schematic
Check Syntax
Generate Post-Synthesis Simulati...
Implement Design
Generate Programming File
Configure Target Device
Analyze Design Using ChipScope
47
48 countlr := countlr + 1;
49 count2r := count2r + 1;
50
51 if (xinteger mod 2 = 1) and (countlr = (xinteger + 1)/2) then
52 wl <= not wl;
53 countlr := 0;
54 countlf := 0;
55 outl <= wl;
56 end if;
57
58 if (xinteger mod 2 = 0) and (count2r = xinteger/2) then
59 w2 <= not w2;
60 count2r := 0;
61 outl <= w2;
62 end if;
63
64 elsif falling_edge(clk) then
65
66 countlf := countlf + 1;
67
68 if (xinteger mod 2 = 1) and (countlf = (xinteger+1)/2) then
69 wl <= not wl;
70 countlf := 0;
71 countlr := 0;
72 outl <= wl;
73 end if;
74
75 end if;
76 end process;
77
78 end Behavioral;
79
80


View: Implementation Simulation
Behavioral
Hierarchy
add_sub_test - behavior (add_sub_te
choice - Behavioral (choice.vhd)
clk_div_5711_test - behavior (clk_div_
clk_div_mod_test - behavior (clk_div_
clk_div_x1_test - behavior (clk_div_x
clk_div_x2_test - behavior (clk_div_x
clk_div_x_test - behavior (clk_div_x_t
comparator_if_test - behavior (comp
comparator_test - behavior (compar
count09and90_test - behavior (coun
counter4_test - behavior (counter4_t
No Processes Running
Processes: clk_div_x_test - behavior
ISim Simulator
Behavioral Check Syntax
Simulate Behavioral Model
61 BEGIN
62
63 -- Instantiate the Unit Under Test (UUT)
64 uut: clk_div_x PORT MAP (
65 clk => clk,
66 xin => xin,
67 outl => outl
68);
69
70 -- Clock process definitions
71 clk_process :process
72 begin
73 clk <= '0';
74 wait for clk_period/2;
75 clk <= '1';
76 wait for clk_period/2;
77 end process;
78
79
80 -- Stimulus process
81 stim_proc: process
82 begin
83 xin <= "10101";
84 wait for 1000 ns;
85
86 wait for clk_period*10;
87
88 -- insert stimulus here
89
90 wait;
91 end process;
92
93 END;
94




```
BEGIN

   -- Instantiate the Unit Under Test (UUT)
   uut: clk_div_x PORT MAP (
          clk => clk,
          xin => xin,
          out1 => out1
        );

   -- Clock process definitions
   clk_process :process
   begin
      clk <= '0';
      wait for clk_period/2;
      clk <= '1';
      wait for clk_period/2;
   end process;


   -- Stimulus process
   stim_proc: process
   begin
      xin <= "01100";
      wait for 1000 ns;

      wait for clk_period*10;

      -- insert stimulus here

      wait;
   end process;

END;
```


Name
Value
clk
0
xin[4:0]
01100
out1
0
clk_period
10000 ps
312.500 ns
50 ns
100 ns
150 ns
200 ns
250 ns
300 ns
350 ns
01100
10000 ps