آزمایشگاه مدار منطقی

نویسنده:

علیرضا طباطبائیان(9723052)

Amirkabir University of Technology
(Tehran Polytechnic)

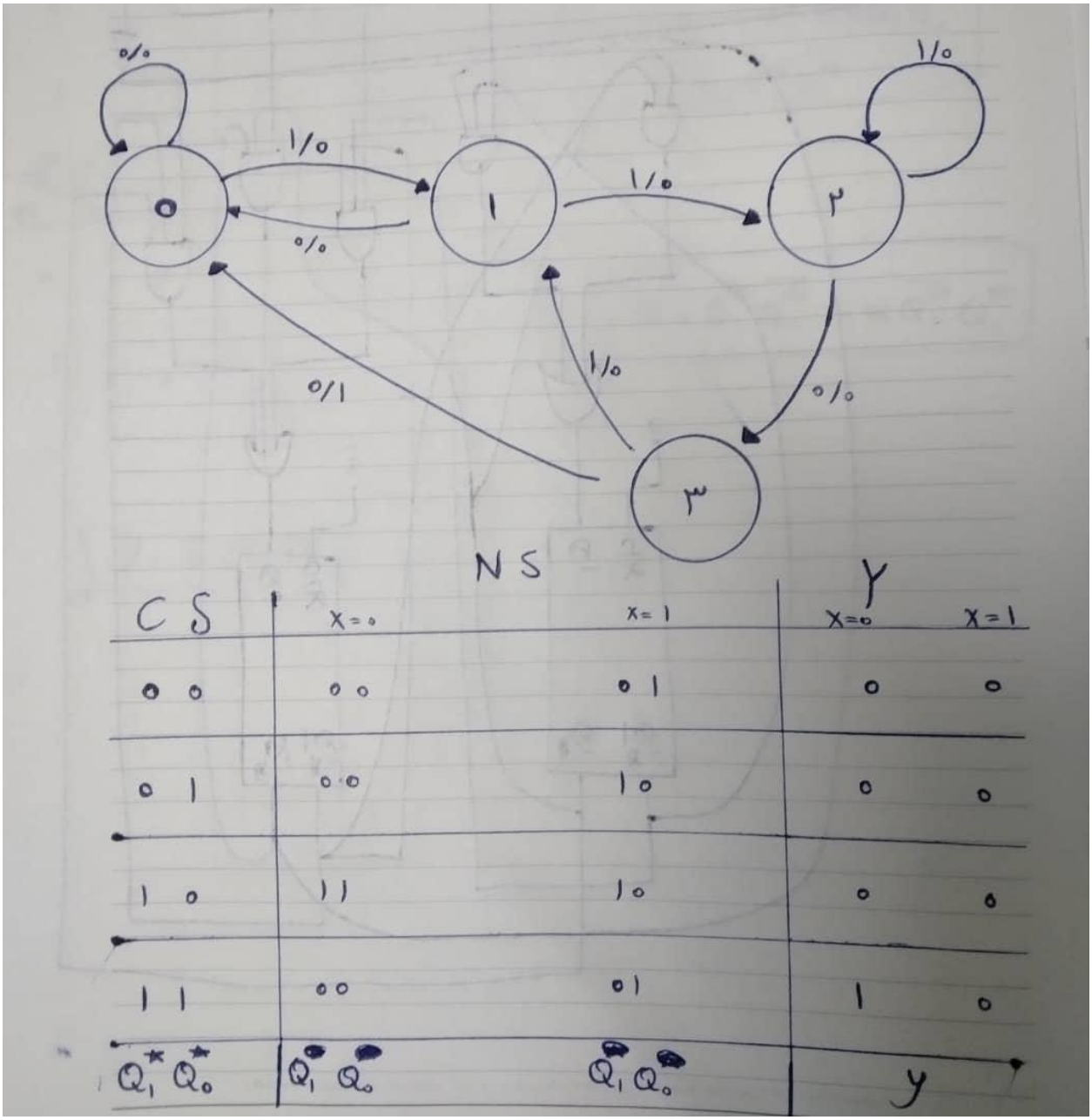
استاد آزمایشگاه:

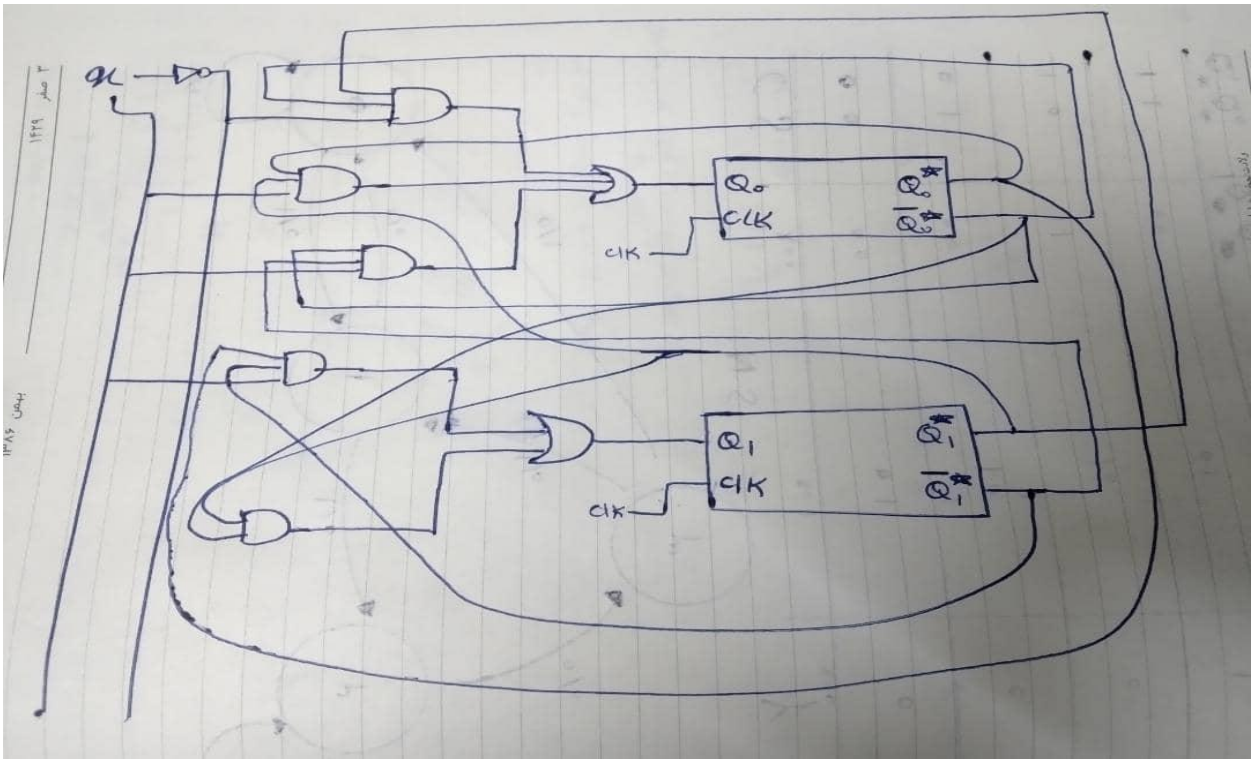
دکتر علیرضا طاهری نوید

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

سوال 1:

طراحی بر روی کاغذ آورده شده است و کد ها نیز آورده شده است(فقط درباره قسمت امتیازی باهاتون صحبت کردم و شما هم کدم را تایید کردید ولی متوجه نشدیم چرا نتیجه دلخواه را در خروجی نمیده)





Tue. 12 Feb. 2008

$x \backslash Q_1^* Q_0^*$	00	01	11	10
0	0	0	0	1
1	1	0	1	0

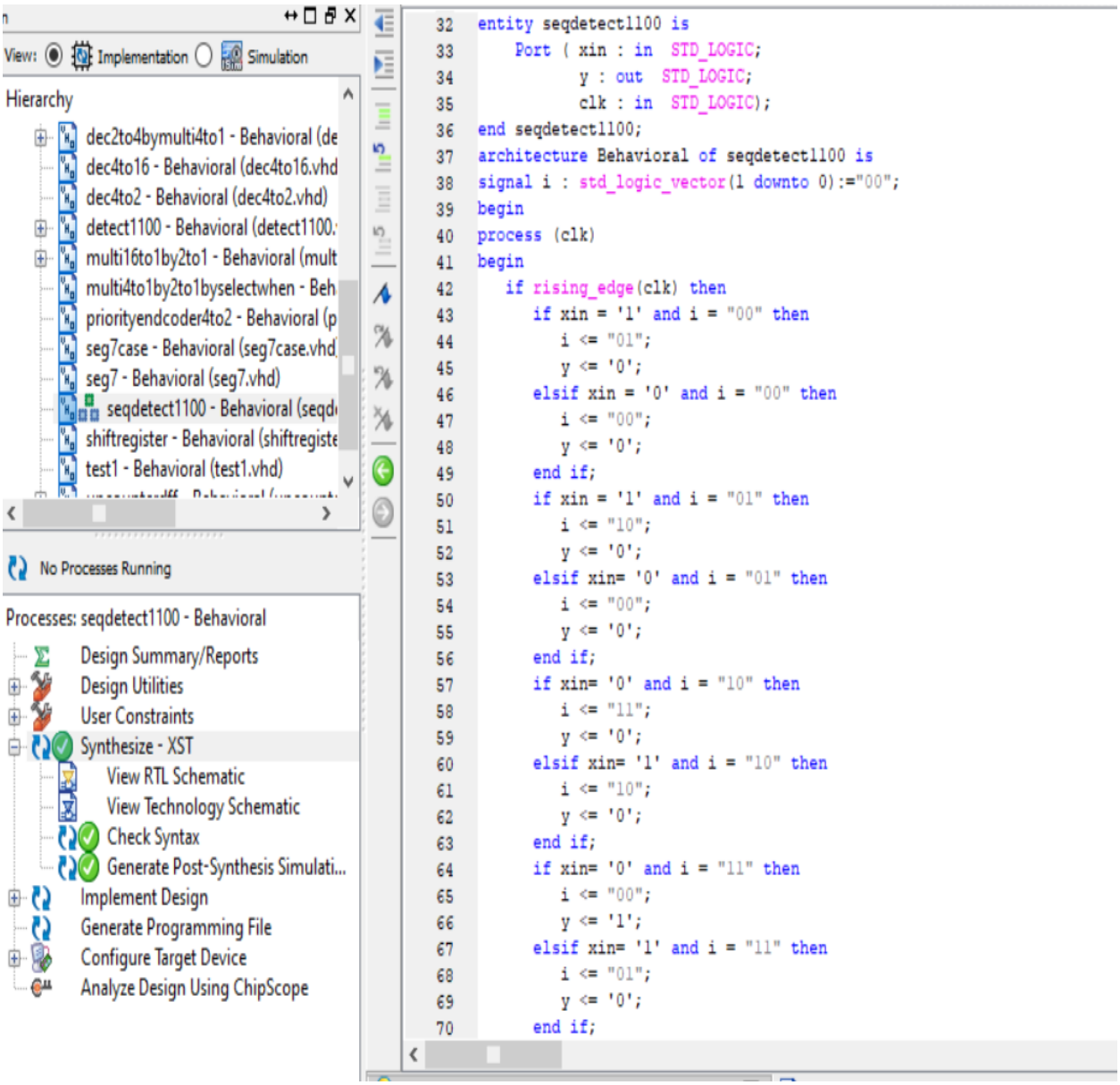
$$Q_0 = x\,\overline{Q_1^*}\,\overline{Q_0^*} + x\,Q_1^*\,Q_0^* + \overline{x}\,Q_1^*\,\overline{Q_0^*}$$

$x \backslash Q_1^* Q_0^*$	00	01	11	10
0	0	0	0	1
1	0	1	0	1

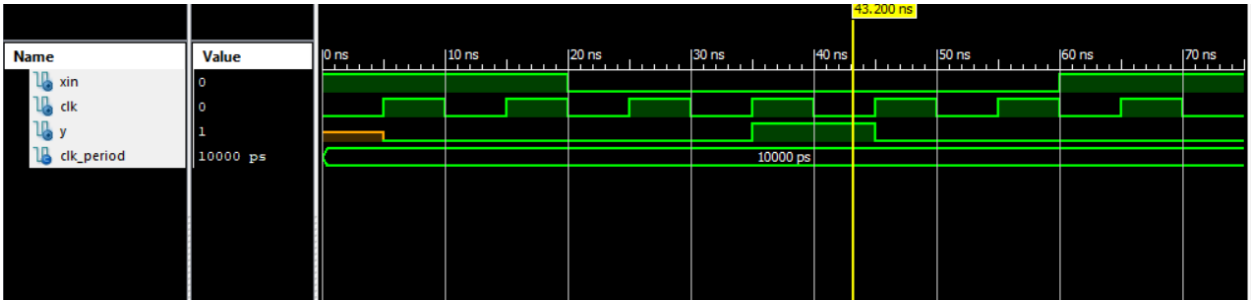
$$Q_1 = Q_1^*\,\overline{Q_0^*} + x\,\overline{Q_1^*}\,Q_0^*$$

$$y = \overline{x}\,Q_1^*\,Q_0^*$$

برای کد، سیگنالی به اسم i را تعریف میکنیم که بیانگر وضعیت سیستم باشد و به نوعی همان شمارنده ما میباشد:

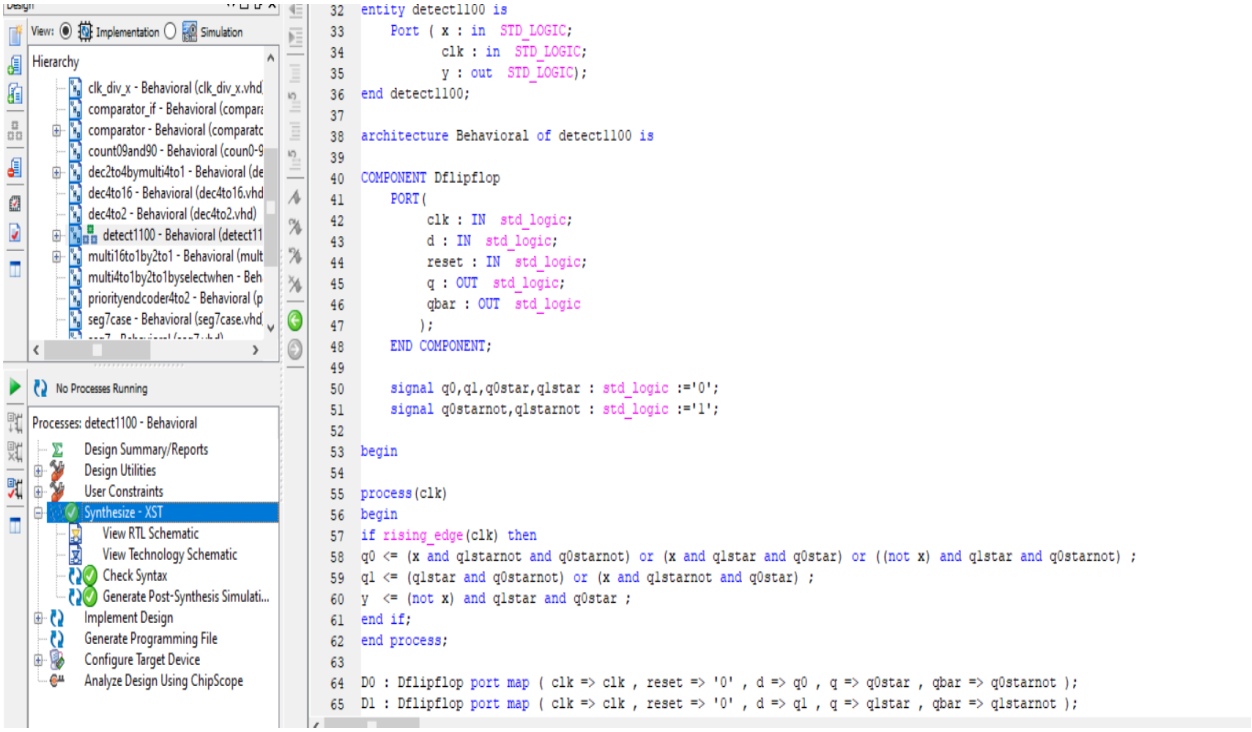


```
        clk <= '1';
        wait for clk_period/2;
    end process;


    -- Stimulus process
    stim_proc: process
    begin
        xin <= '1';
        wait for clk_period;
        xin <= '1';
        wait for clk_period;
        xin <= '0';
        wait for clk_period;
        xin <= '0';
        wait for clk_period;
        xin <= '0';
        wait for clk_period;
        xin <= '0';
        wait for clk_period;
        xin <= '1';
        wait for clk_period;
        xin <= '1';
        wait for clk_period;
        xin <= '0';
        wait for clk_period;
        xin <= '1';
        wait for clk_period;

        wait;
    end process;

END;
```



قسمت امتیازی سوال یک:

ابتدا دو فلیپ فلاپ همانند جلسات گذشته ایجاد کردیم و اینجا از کامپونت آنها استفاده میکنیم:




```
32  entity detect1100 is
33      Port ( x : in  STD_LOGIC;
34             clk : in  STD_LOGIC;
35             y : out  STD_LOGIC);
36  end detect1100;
37
38  architecture Behavioral of detect1100 is
39
40  COMPONENT Dflipflop
41      PORT(
42          clk : IN  std_logic;
43          d : IN  std_logic;
44          reset : IN  std_logic;
45          q : OUT  std_logic;
46          qbar : OUT  std_logic
47         );
48      END COMPONENT;
49
50      signal q0,q1,q0star,q1star : std_logic :='0';
51      signal q0starnot,q1starnot : std_logic :='1';
52
53  begin
54
55  process(clk)
56  begin
57  if rising_edge(clk) then
58  q0 <= (x and q1starnot and q0starnot) or (x and q1star and q0star) or ((not x) and q1star and q0starnot) ;
59  q1 <= (q1star and q0starnot) or (x and q1starnot and q0star) ;
60  y  <= (not x) and q1star and q0star ;
61  end if;
62  end process;
63
64  D0 : Dflipflop port map ( clk => clk , reset => '0' , d => q0 , q => q0star , qbar => q0starnot );
65  D1 : Dflipflop port map ( clk => clk , reset => '0' , d => q1 , q => q1star , qbar => q1starnot );
```

```
   -- Stimulus process
   stim_proc: process
   begin
      x <= '0';
      wait for clk_period;
      x <= '0';
      wait for clk_period;
      x <= '1';
      wait for clk_period;
      x <= '1';
      wait for clk_period;
      x <= '0';
      wait for clk_period;
      x <= '0';
      wait for clk_period;
      x <= '0';
      wait for clk_period;
      x <= '0';
      wait for clk_period;
      x <= '1';
      wait for clk_period;
      x <= '1';
      wait for clk_period;
      x <= '0';
      wait for clk_period;
      x <= '0';
      wait for clk_period;


      wait;
   end process;

END;
```

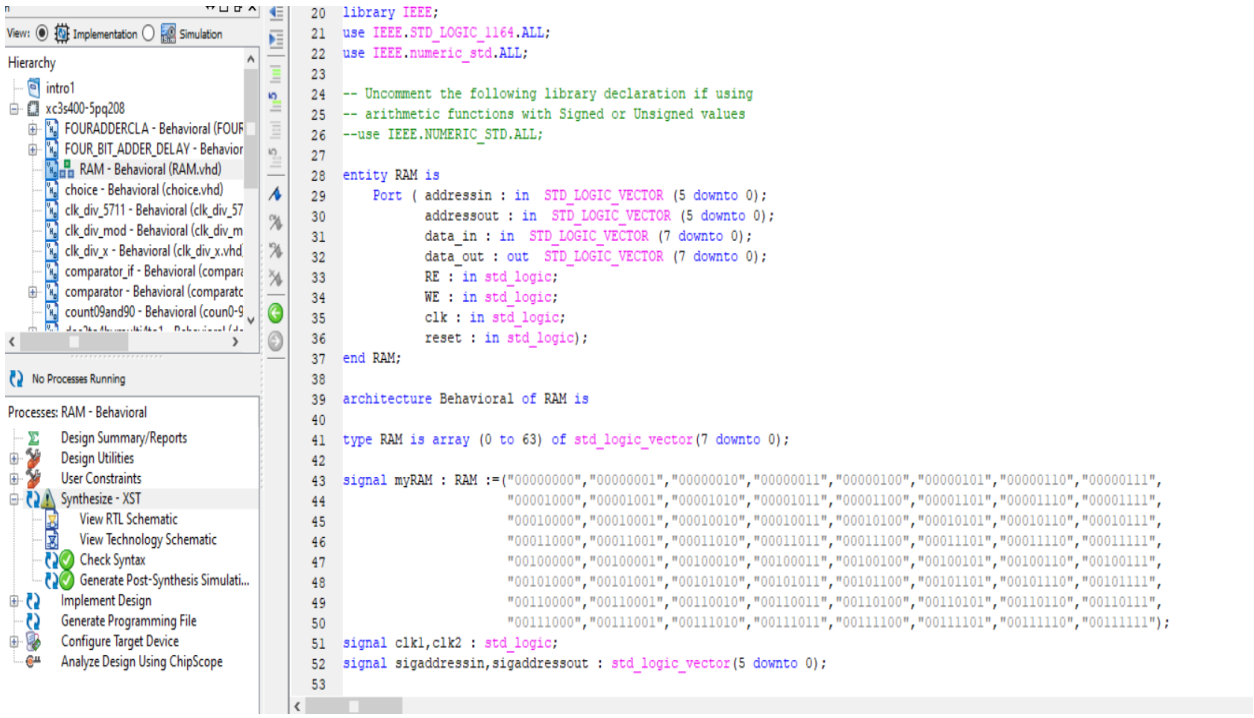
سوال 3:

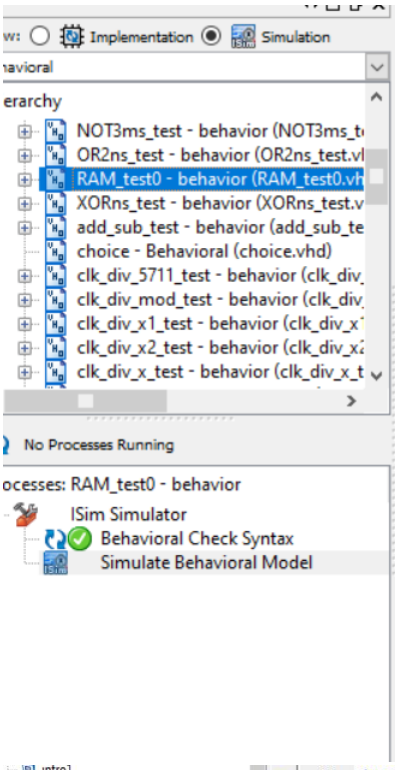
ابتدا یک تایپ جدید تعریف کرده و سپس توسط سیگنال ، آرایه مربوط به آن را ایجاد و مقدار دهی میکنیم.

سپس 2 کلاک جداگانه تعریف میکنیم یعنی یک کلاک را به دو clk1 **و** clk2 **وصل میکنیم تا 2 کلاک بدست آید.**

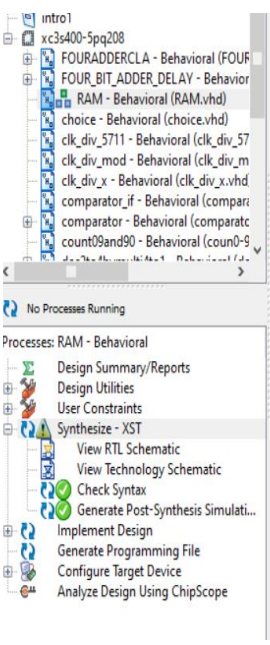
سپس در پراسس، ریست آسنکرون را تعریف میکنیم .

و سپس دستورات خواندن و نوشتن را وارد میکنیم که این دو فرآیند به طور موازی میتوانند انجام شوند.






```
97    begin
98
99       reset <= '0';
100      RE <= '1';
101      WE <= '0';
102      addressout <= "010101";
103      addressin <= "000000";
104      data_in <= "11111111";
105      wait for 10 ns;
106      reset <= '0';
107      RE <= '0';
108      WE <= '1';
109      addressout <= "000000";
110      addressin <= "000000";
111      data_in <= "11111111";
112      wait for 10 ns;
113      reset <= '0';
114      RE <= '1';
115      WE <= '1';
116      addressout <= "000000";
117      addressin <= "000000";
118      data_in <= "10101010";
119      wait for 20 ns;
120      reset <= '1';
121      wait for 10 ns;
122      reset <= '0';
123      RE <= '1';
124      WE <= '0';
125      addressout <= "000000";
126      addressin <= "000000";
127      data_in <= "11111111";
128      wait for 10 ns;
```




```
51  signal clk1,clk2 : std_logic;
52  signal sigaddressin,sigaddressout : std_logic_vector(5 downto 0);
53
54  begin
55
56  clk1 <= clk;
57  clk2 <= clk;
58
59  process (clk1,clk2,reset)
60  begin
61     if reset = '1' then
62        sigaddressin  <= "000000";
63        sigaddressout <= "000000";
64        myRAM <= ( others => "00000000" );
65     elsif true then
66        sigaddressin <= addressin;
67        sigaddressout <= addressout;
68        if rising_edge(clk1) and RE = '1' then
69           data_out <= myRAM(to_integer(unsigned(sigaddressout)));
70        end if;
71        if rising_edge(clk2) and WE = '1' then
72           myRAM(to_integer(unsigned(sigaddressin))) <= data_in;
73        end if;
74     end if;
75  end process;
76
77
78  end Behavioral;
79
80
```


35.000 ns
Name
Value
0 ns
10 ns
20 ns
30 ns
40 ns
50 ns
60 ns
70 ns
addressin[5:0]
000000
000000
addressout[5:0]
000000
010101
000000
data_in[7:0]
10101010
11111111
10101010
11111111
re
1
we
1
clk
1
reset
0
data_out[7:0]
10101010
UUUU...
00010101
11111111
10101010
00000000
clk_period
10000 ps
10000 ps