

آزمایشگاه مدار منطقی



Amirkabir University of Technology
(Tehran Polytechnic)



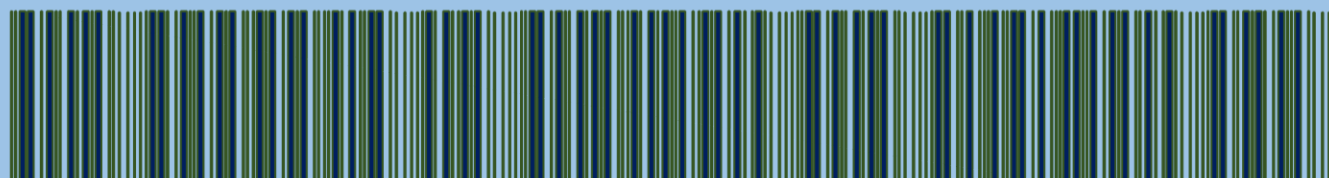
نویسنده:

علیرضا طباطبائی (9723052)

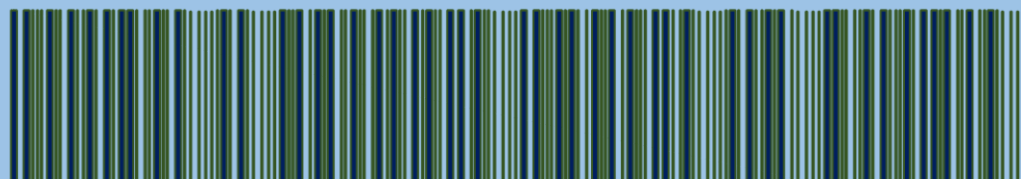
استاد آزمایشگاه:

دکتر علیرضا

ظاهری نوید



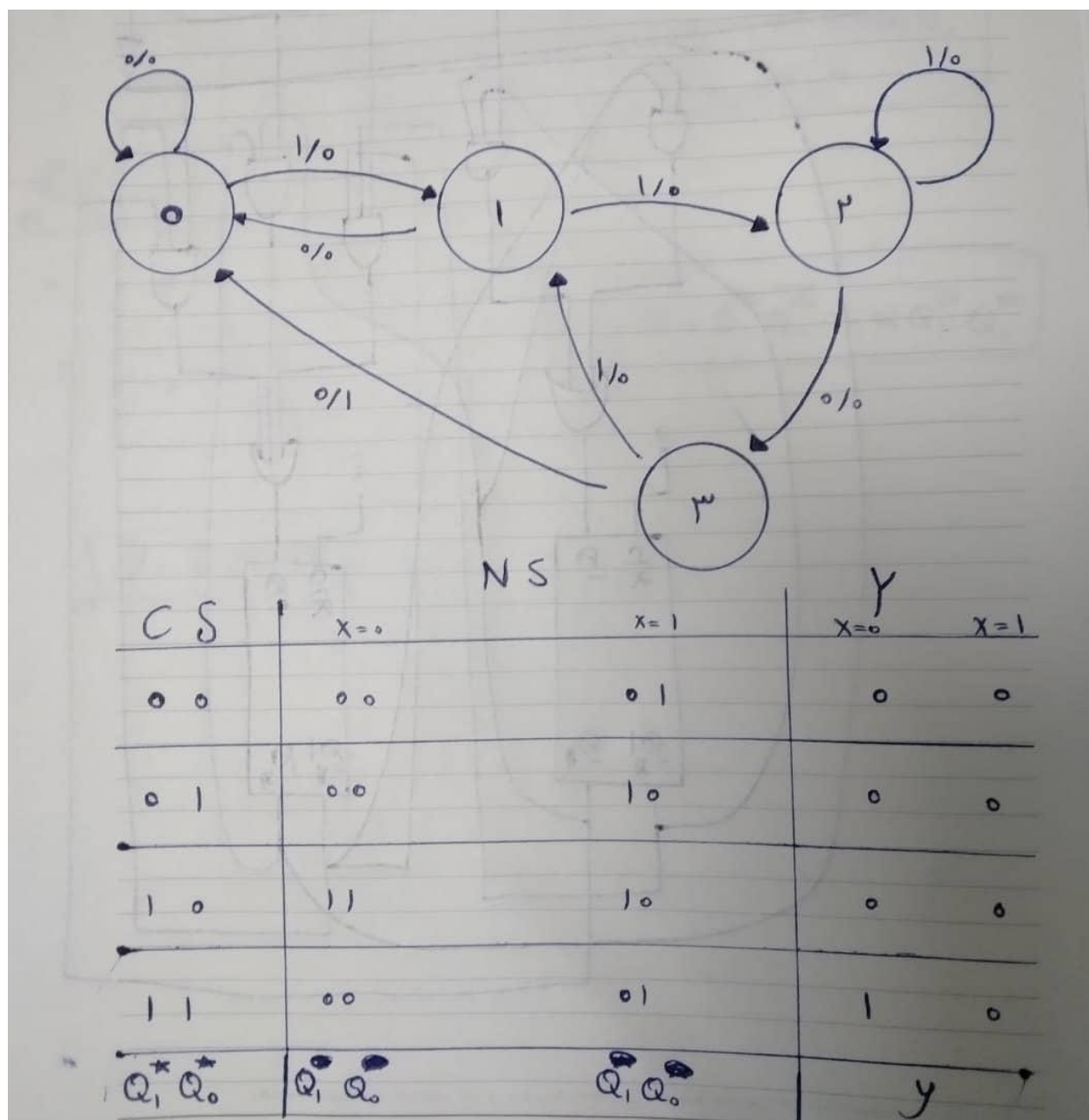
THE PRESENT IS THEIRS, THE FUTURE, FOR

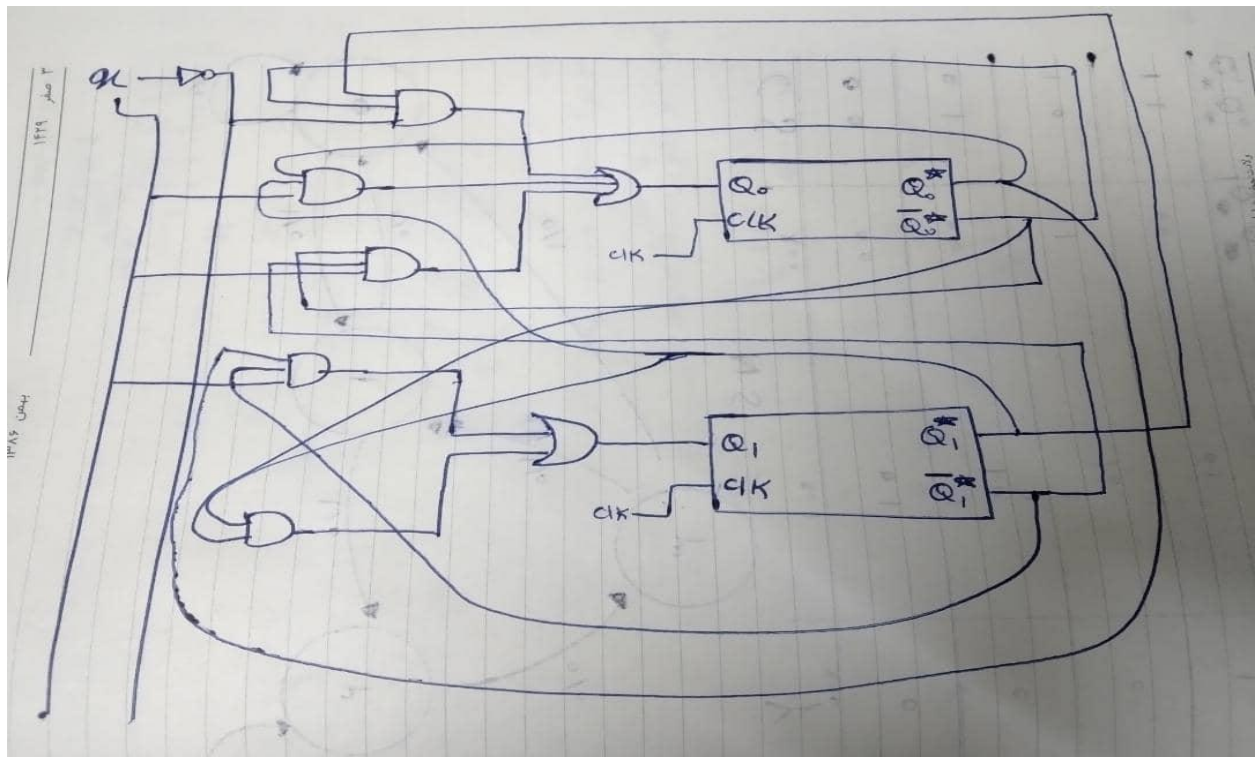


WHICH I REALLY WORK, IS MINE.

سوال 1:

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





Tue, 12 Feb. 2008

$Q_1^* Q_0^*$	00	01	11	10
0	0	0	0	1
1	1	0	1	0

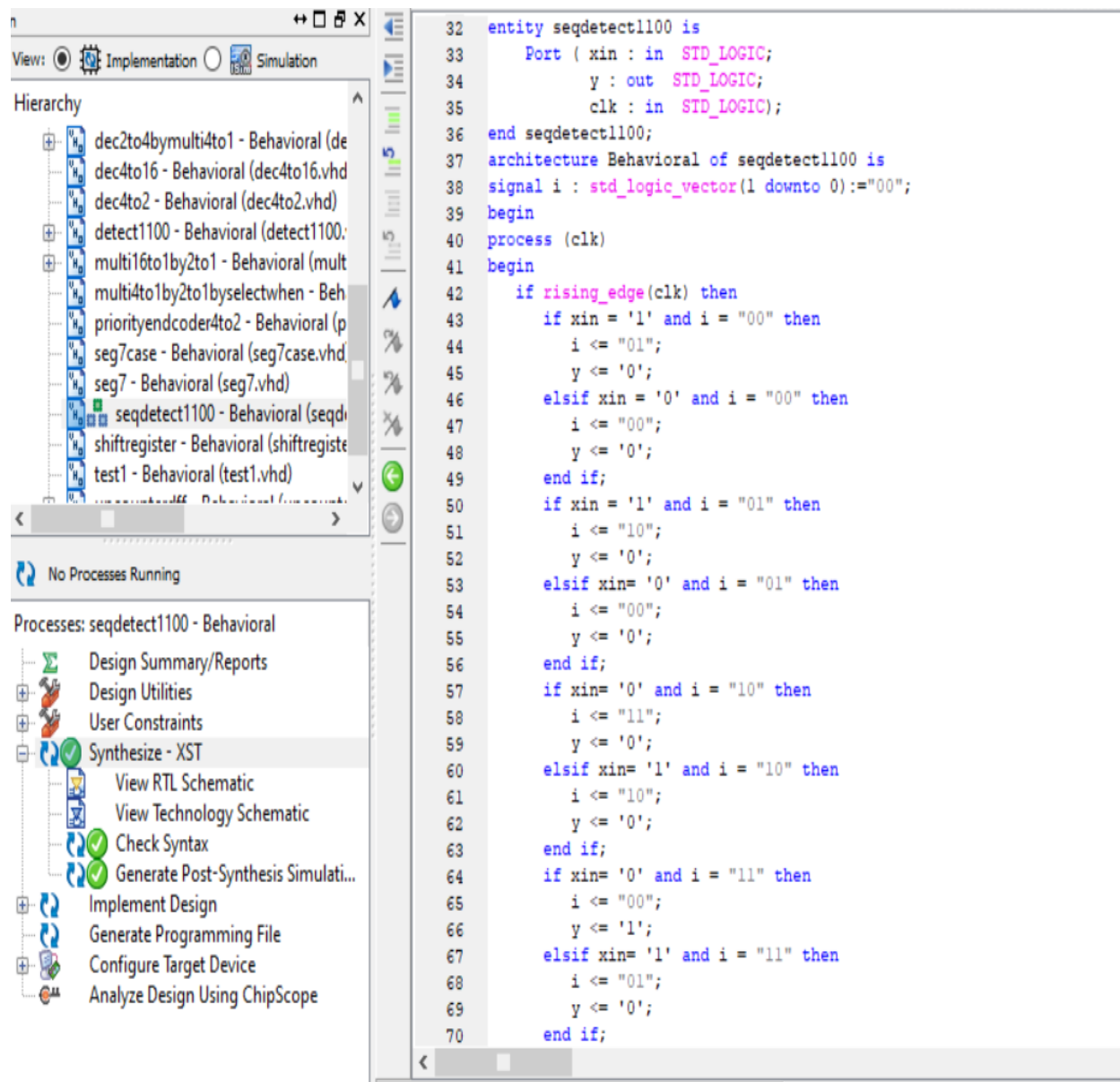
$$Q_0 = \overline{Q_1^*} \overline{Q_0^*} + Q_1^* \overline{Q_0^*} + \overline{Q_1^*} Q_0^*$$

$Q_1^* Q_0^*$	00	01	11	10
0	0	0	0	1
1	0	1	0	1

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

$$Y = \overline{Q_1^*} Q_1^* Q_0^*$$

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



The screenshot shows the Xilinx ISE IDE interface. On the left, the 'Hierarchy' pane lists the project components, including 'seqdetect1100 - Behavioral (seqdetect1100.vhd)'. Below it, the 'Processes' pane shows the synthesis steps, with 'Generate Post-Synthesis Simulation' highlighted. The main window displays the VHDL code for 'seqdetect1100'.

```

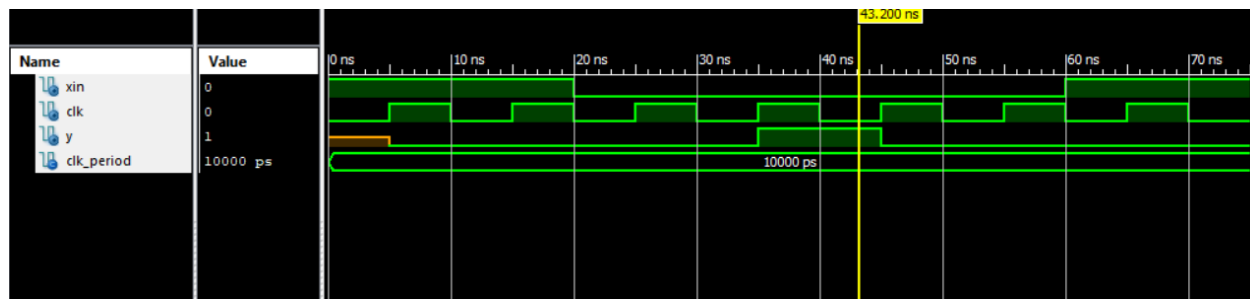
32 entity seqdetect1100 is
33     Port ( xin : in  STD_LOGIC;
34           y : out  STD_LOGIC;
35           clk : in  STD_LOGIC);
36 end seqdetect1100;
37 architecture Behavioral of seqdetect1100 is
38     signal i : std_logic_vector(1 downto 0):="00";
39 begin
40     process (clk)
41     begin
42         if rising_edge(clk) then
43             if xin = '1' and i = "00" then
44                 i <= "01";
45                 y <= '0';
46             elsif xin = '0' and i = "00" then
47                 i <= "00";
48                 y <= '0';
49             end if;
50             if xin = '1' and i = "01" then
51                 i <= "10";
52                 y <= '0';
53             elsif xin= '0' and i = "01" then
54                 i <= "00";
55                 y <= '0';
56             end if;
57             if xin= '0' and i = "10" then
58                 i <= "11";
59                 y <= '0';
60             elsif xin= '1' and i = "10" then
61                 i <= "10";
62                 y <= '0';
63             end if;
64             if xin= '0' and i = "11" then
65                 i <= "00";
66                 y <= '1';
67             elsif xin= '1' and i = "11" then
68                 i <= "01";
69                 y <= '0';
70             end if;

```

```

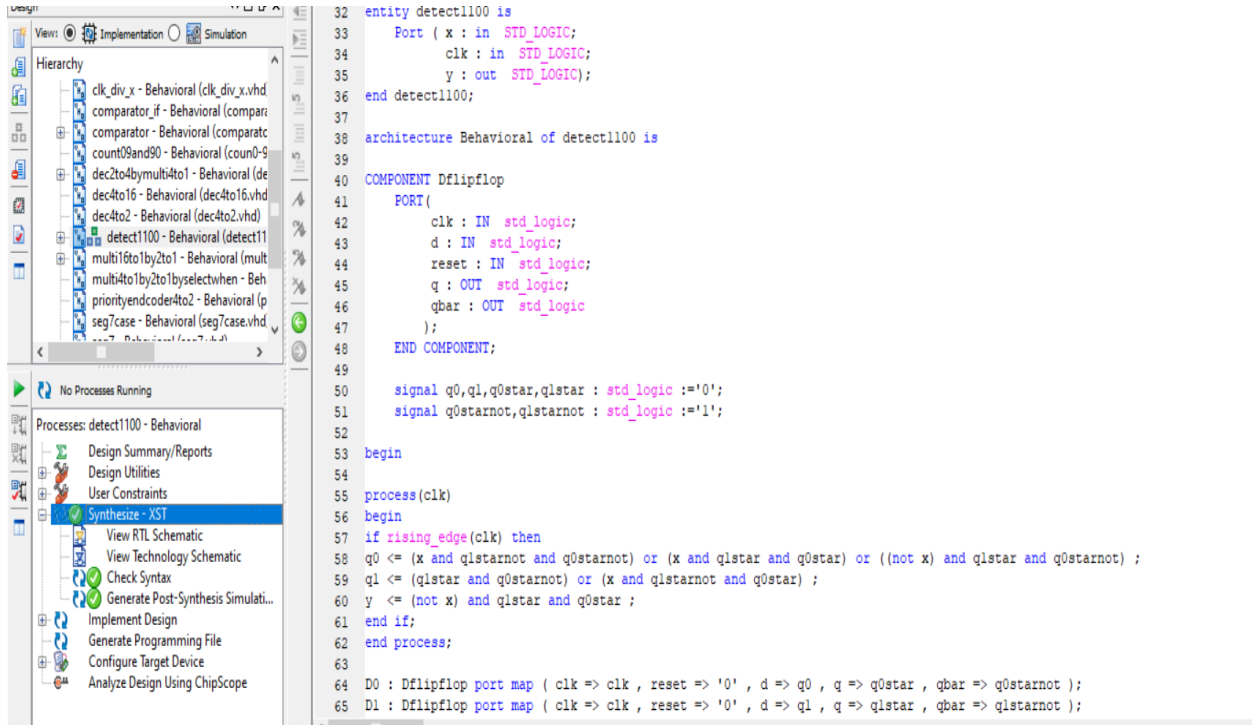
75         clk <= 1;
76         wait for clk_period/2;
77     end process;
78
79
80     -- Stimulus process
81     stim_proc: process
82     begin
83         xin <= '1';
84         wait for clk_period;
85         xin <= '1';
86         wait for clk_period;
87         xin <= '0';
88         wait for clk_period;
89         xin <= '0';
90         wait for clk_period;
91         xin <= '0';
92         wait for clk_period;
93         xin <= '0';
94         wait for clk_period;
95         xin <= '1';
96         wait for clk_period;
97         xin <= '1';
98         wait for clk_period;
99         xin <= '0';
100        wait for clk_period;
101        xin <= '1';
102        wait for clk_period;
103
104        wait;
105    end process;
106
107 END;
108

```



قسمت امتیازی سوال یک:

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




```
-- 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;
```

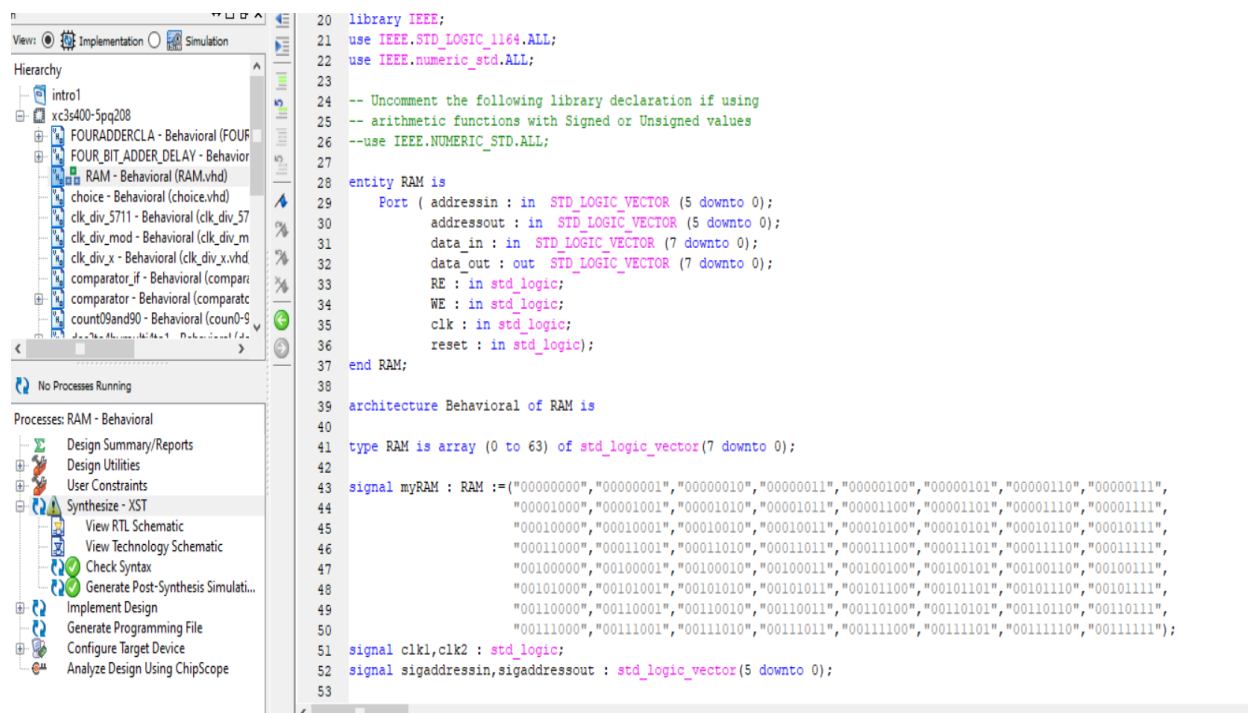
سوال 3:

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 پراسسی، ریست آسنکرون را تعریف میکنیم .

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



```
20 library IEEE;
21 use IEEE.STD_LOGIC_1164.ALL;
22 use IEEE.numeric_std.ALL;
23
24 -- Uncomment the following library declaration if using
25 -- arithmetic functions with Signed or Unsigned values
26 --use IEEE.NUMERIC_STD.ALL;
27
28 entity RAM is
29     Port ( addressin : in  STD_LOGIC_VECTOR (5 downto 0);
30           addressout : in  STD_LOGIC_VECTOR (5 downto 0);
31           data_in : in  STD_LOGIC_VECTOR (7 downto 0);
32           data_out : out STD_LOGIC_VECTOR (7 downto 0);
33           RE : in std_logic;
34           WE : in std_logic;
35           clk : in std_logic;
36           reset : in std_logic);
37 end RAM;
38
39 architecture Behavioral of RAM is
40
41     type RAM is array (0 to 63) of std_logic_vector(7 downto 0);
42
43     signal myRAM : RAM :=("00000000","00000001","00000010","00000011","00000100","00000101","00000110","00000111",
44                           "00001000","00001001","00001010","00001011","00001100","00001101","00001110","00001111",
45                           "00010000","00010001","00010010","00010011","00010100","00010101","00010110","00010111",
46                           "00011000","00011001","00011010","00011011","00011100","00011101","00011110","00011111",
47                           "00100000","00100001","00100010","00100011","00100100","00100101","00100110","00100111",
48                           "00101000","00101001","00101010","00101011","00101100","00101101","00101110","00101111",
49                           "00110000","00110001","00110010","00110011","00110100","00110101","00110110","00110111",
50                           "00111000","00111001","00111010","00111011","00111100","00111101","00111110","00111111");
51
52     signal clk1,clk2 : std_logic;
53     signal sigaddressin,sigaddressout : std_logic_vector(5 downto 0);
```


W: ☐ Implementation ☒ Simulation

hierarchy

- NOT3ms_test - behavior (NOT3ms_test.v)
- OR2ns_test - behavior (OR2ns_test.v)
- RAM_test0 - behavior (RAM_test0.v)**
- XORns_test - behavior (XORns_test.v)
- add_sub_test - behavior (add_sub_test.v)
- choice - Behavioral (choice.vhd)
- clk_div_5711_test - behavior (clk_div_5711_test.v)
- clk_div_mod_test - behavior (clk_div_mod_test.v)
- clk_div_x1_test - behavior (clk_div_x1_test.v)
- clk_div_x2_test - behavior (clk_div_x2_test.v)
- clk_div_x_test - behavior (clk_div_x_test.v)

No Processes Running

Processes: RAM_test0 - behavior

- ISim Simulator
- Behavioral Check Syntax
- Simulate Behavioral Model

```

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;

```

intro1

- xc3s400-5pq208
- FOURADDERCLA - Behavioral (FOURADDERCLA.v)
- FOUR_BIT_ADDER_DELAY - Behavioral (FOUR_BIT_ADDER_DELAY.v)
- RAM - Behavioral (RAM.vhd)
- choice - Behavioral (choice.vhd)
- clk_div_5711 - Behavioral (clk_div_5711.v)
- clk_div_mod - Behavioral (clk_div_mod.v)
- clk_div_x - Behavioral (clk_div_x.vhd)
- comparator_if - Behavioral (comparator_if.v)
- comparator - Behavioral (comparator.v)
- count09and90 - Behavioral (count09and90.v)

No Processes Running

Processes: RAM - Behavioral

- Design Summary/Reports
- Design Utilities
- User Constraints
- Synthesize - XST
- View RTL Schematic
- View Technology Schematic
- Check Syntax
- Generate Post-Synthesis Simulation
- Implement Design
- Generate Programming File
- Configure Target Device
- Analyze Design Using ChipScope

```

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 => "000000000" );
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

```

