به نام خدا

تمرین سوم

محمدمهدی آقاجانی

استاد : دکتر صاحب الزمانی

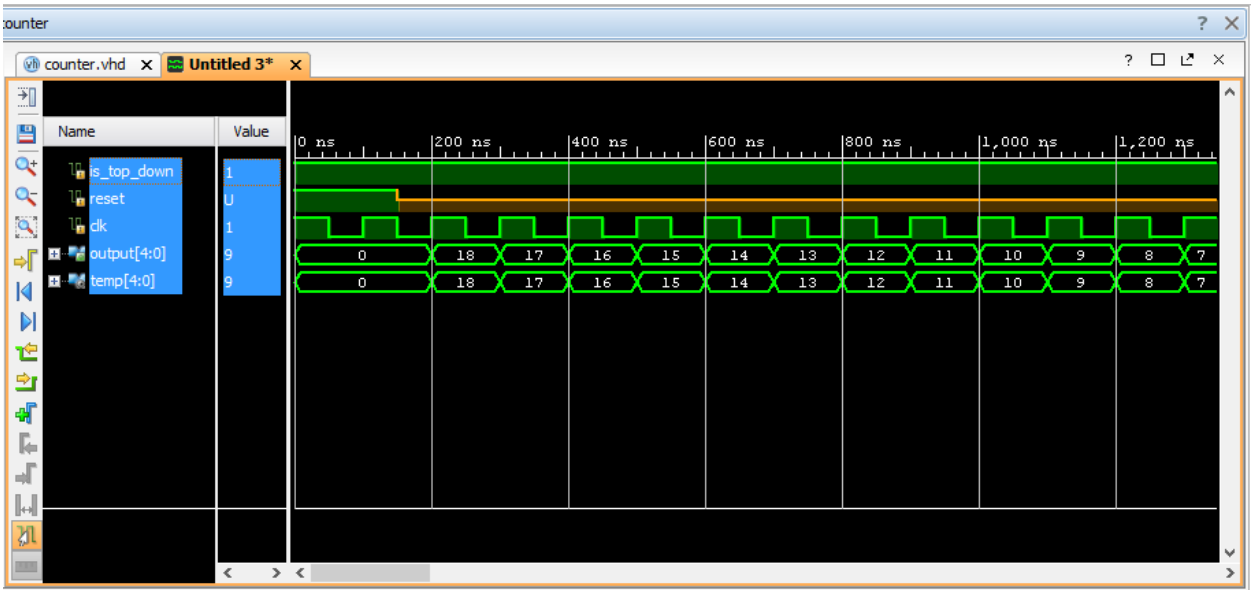
سوال ۵ :

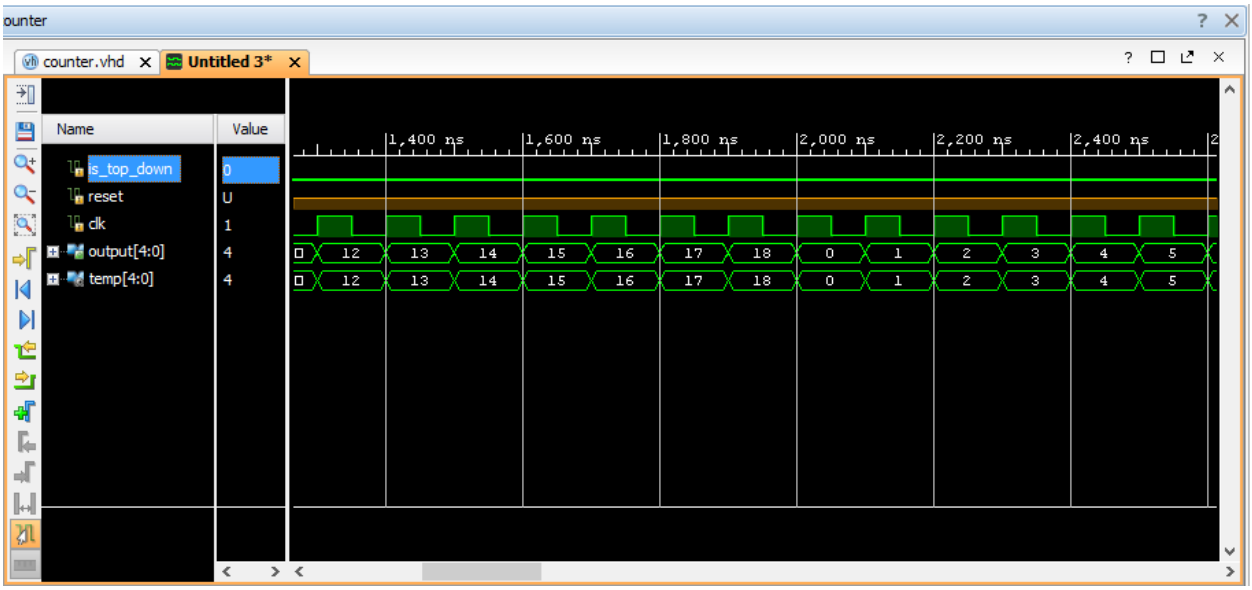
تعداد حالات این ماشین ۱۹ حالت می باشد که با شماره شمارنده نام گذاری شده اند . کد زیر در واقع یک شمارنده را پیاده سازی میکند:

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

entity counter is Port (
is_top_down : in std_logic ;
reset : in std_logic ;
clk : in std_logic ;
output : out std_logic_vector( 4 downto 0)
);
end counter;

architecture Behavioral of counter is
signal temp : std_logic_vector( 4 downto 0 ) ;
begin
process(clk)
begin
    if( clk'event and clk = '1')then
        if( reset = '1' )then
           temp <= "00000" ;
        else
            if( is_top_down = '1' ) then
               case temp is
                                when "00001" => temp <= "00000";
                                when "00010" => temp <= "00001";
                                when "00011" => temp <= "00010";
                                when "00100" => temp <= "00011";
                                when "00101" => temp <= "00100";
                                when "00110" => temp <= "00101";
                                when "00111" => temp <= "00110";
                                when "01000" => temp <= "00111";
                                when "01001" => temp <= "01000";
                                when "01010" => temp <= "01001";
                                when "01011" => temp <= "01010";
                                when "01100" => temp <= "01011";
```

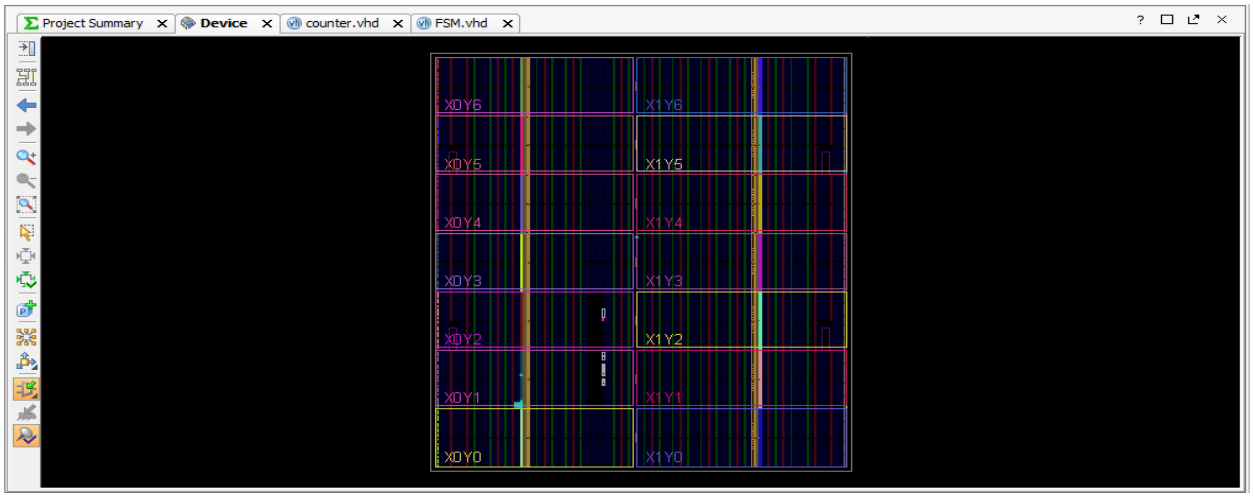
```
                                when "01101" => temp <= "01100";
                                when "01110" => temp <= "01101";
                                when "01111" => temp <= "01110";
                                when "10000" => temp <= "01111";
                                when "10001" => temp <= "10000";
                                when "10010" => temp <= "10001";
                                when "00000" => temp <= "10010";
                                when others => temp <= "00000";
                            end case;
            else
                case temp is
                    when "10010" => temp <= "00000";
                    when "00000" => temp <= "00001";
                    when "00001" => temp <= "00010";
                    when "00010" => temp <= "00011";
                    when "00011" => temp <= "00100";
                    when "00100" => temp <= "00101";
                    when "00101" => temp <= "00110";
                    when "00110" => temp <= "00111";
                    when "00111" => temp <= "01000";
                    when "01000" => temp <= "01001";
                    when "01001" => temp <= "01010";
                    when "01010" => temp <= "01011";
                    when "01011" => temp <= "01100";
                    when "01100" => temp <= "01101";
                    when "01101" => temp <= "01110";
                    when "01110" => temp <= "01111";
                    when "01111" => temp <= "10000";
                    when "10000" => temp <= "10001";
                    when "10001" => temp <= "10010";
                    when others => temp <= "00000";
                end case;
            end if;
        end if;
    end if;
end process;
output <= temp ;
end Behavioral;
```

شکل موج های کد بالا به صورت زیر است :

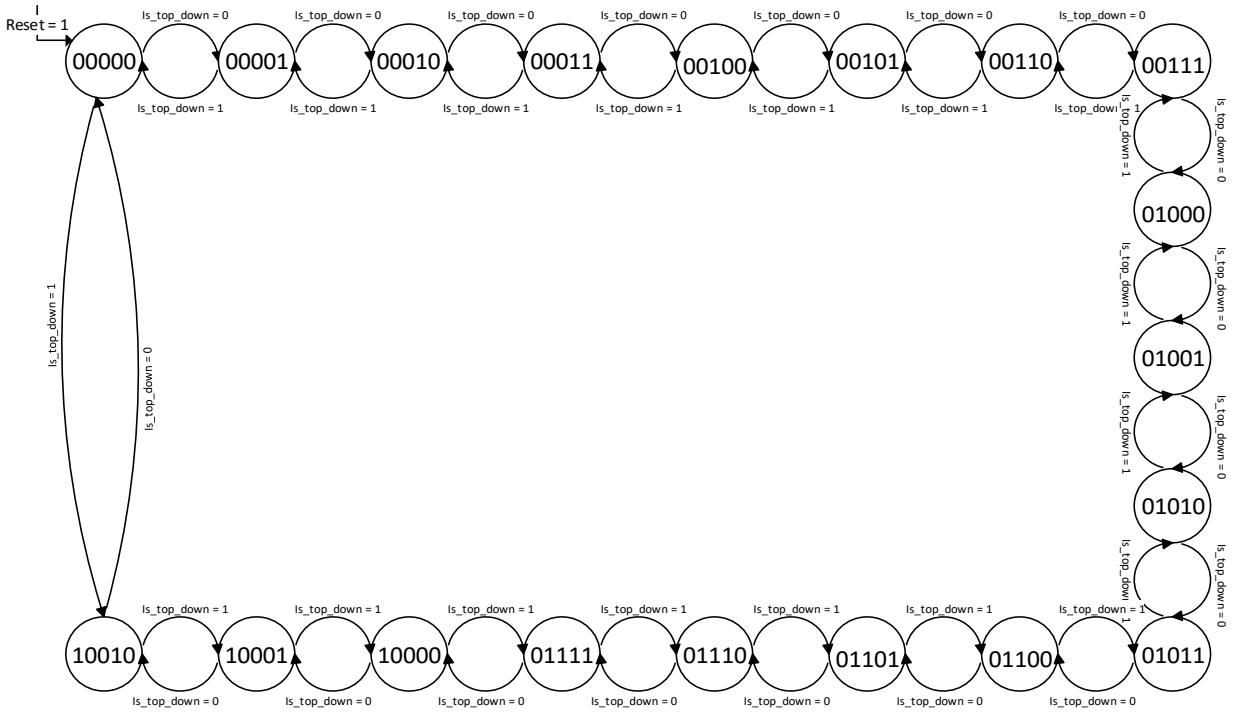




همچنین تصویر زیر نشان دهنده سنتز کردن کد بالا می باشد :



برای کد بالا FSM زیر را طراحی کرده ایم :



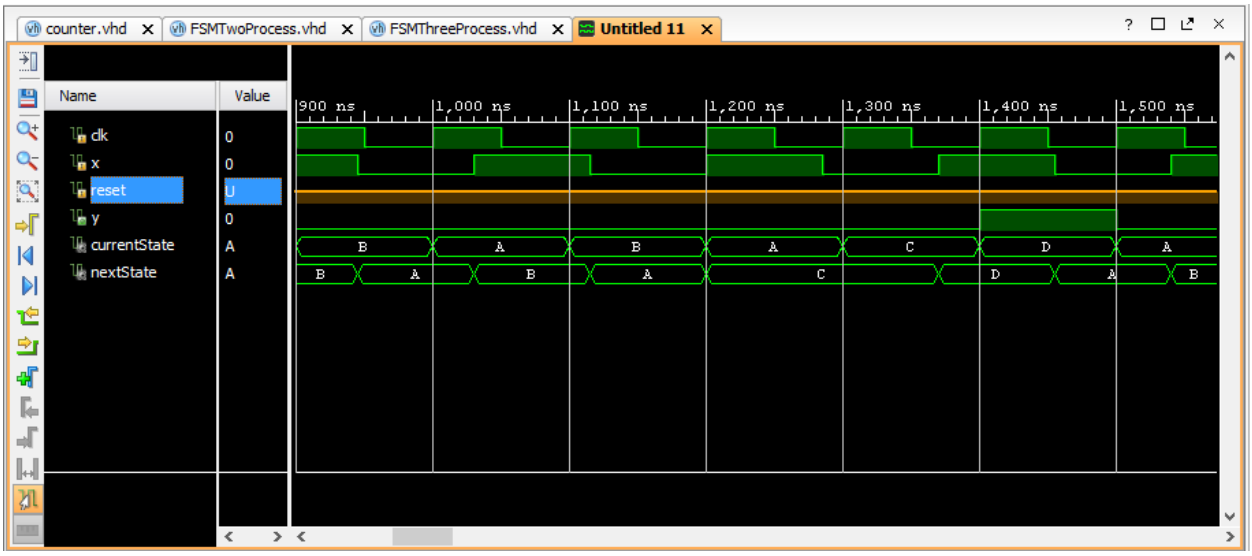
سوال ۶ :

در ابتدا کد آن را با استفاده از ۳ process مینویسیم :

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

entity FSMThreeProcess is Port (
clk : in std_logic;
x : in std_logic ;
reset : in std_logic ;
y : out std_logic );
end FSMThreeProcess;

architecture Behavioral of FSMThreeProcess is
type State_type IS (A, B, C, D);
      signal currentState , nextState : State_Type;
begin
process (x)
  begin
    if (reset = '1') then
      nextState <= A;
    else

      case currentState is
            when A =>
                  if x='1' then
                        nextState <= B;
                  end if;

            when B =>
                  if x='1' then
                  nextState <= C;
             elsif x='0' then
                  nextState <= A;
             end if;

            when C =>
                  if x='1' then
                        nextState <= D;
```

```
                elsif x='0' then
                    nextState <= A;
                end if;

            when D=>
                if x='1' then
                      nextState <= D;
                   elsif x='0' then
                      nextState <= A;
                   end if;
            when others =>
                nextState <= A;
      end case;
    end if;
  end process;

    process
      begin
          wait until rising_edge(clk);
          currentState <= nextState;
     end process;

  process(currentState)
  begin
  case currentState is
    when D => y <= '1';
    when others => y <= '0' ;
    end case;
  end process;
end Behavioral;
```

شکل موج آن به صورت زیر است :

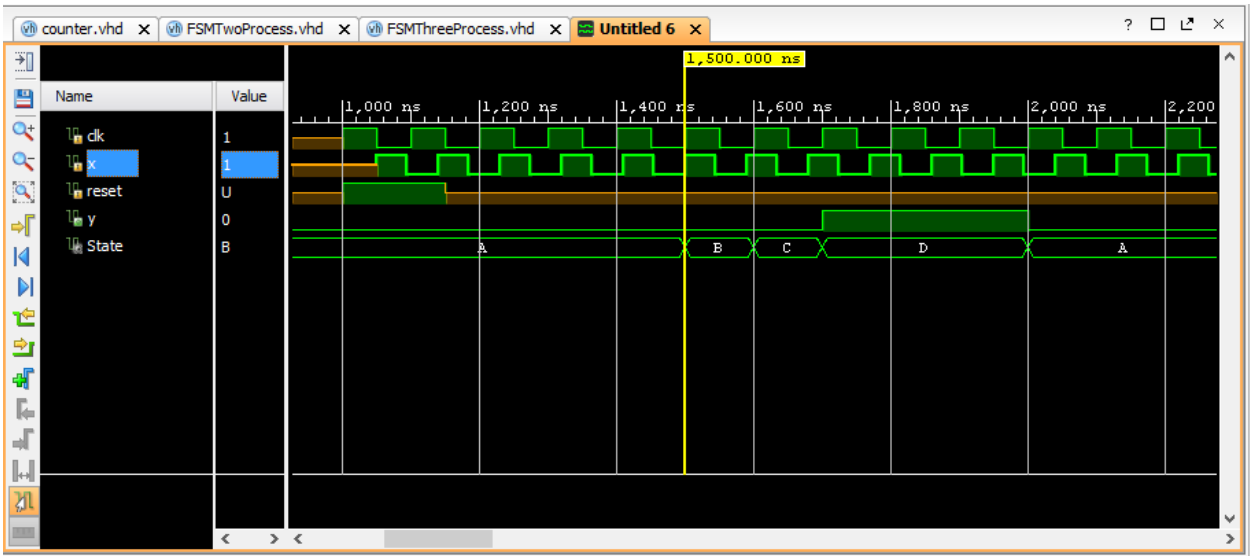


حال آن را با ۲ process پیاده سازی می کنیم :

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

entity FSMTwoProcess is Port (
clk : in std_logic;
x : in std_logic ;
reset : in std_logic ;
y : out std_logic );
end FSMTwoProcess;

architecture Behavioral of FSMTwoProcess is
type State_type IS (A, B, C, D);
      signal State : State_Type;
begin
process (clk, reset)
  begin
    if (reset = '1') then
      State <= A;

    elsif (clk'event and clk = '1') then
      case State is
            when A =>
                  if x='1' then
                        State <= B;
```

```
                end if;

            when B =>
                if x='1' then
                State <= C;
             elsif x='0' then
                State <= A;
             end if;

            when C =>
                if x='1' then
                      State <= D;
                elsif x='0' then
                    State <= A;
                end if;

            when D=>
                if x='1' then
                       State <= D;
                   elsif x='0' then
                       State <= A;
                   end if;
            when others =>
                State <= A;
      end case;
    end if;
  end process;

  process(State)
  begin
  case State is
    when D => y <= '1';
    when others => y <= '0' ;
    end case;
  end process;
end Behavioral;
```

شکل موج آن هم به صورت زیر است :



سوال ۷ :

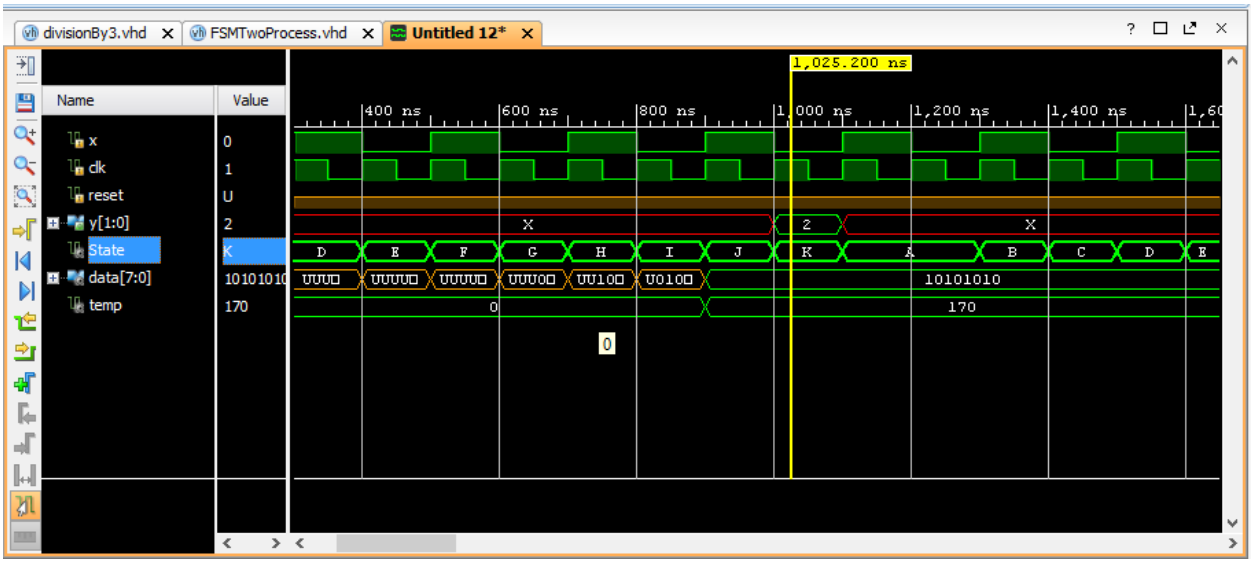
کد ماژول به صورت زیر است :

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

entity divisionBy3 is Port (
x : in std_logic;
clk : in std_logic;
reset : in std_logic;
y : out std_logic_vector(1 downto 0 )
);
end divisionBy3;

architecture Behavioral of divisionBy3 is
type State_type IS (A, B, C, D , E, F , G , H , I , J , K);
      signal State : State_Type;
signal data : std_logic_vector( 7 downto 0 );
signal temp : integer range 0 to 255;

begin
process (clk)
```

```
begin
  if (reset = '1') then
    State <= A;

  elsif (clk'event and clk = '1') then
    case State is
          when A =>
                if x='1' then
                State <= B;
           elsif x='0' then
                State <= A;
           end if;

          when B =>
              data(0) <= x;
                State <= C;

          when C =>
          data(1) <= x;
                State <= D;

          when D =>
          data(2) <= x;
          State <= E;

       when E =>
       data(3) <= x;
               State <= F;
       when F =>
              data(4) <= x;
              State <= G;
       when G =>
              data(5) <= x;
              State <= H;
       when H =>
              data(6) <= x;
              State <= I;
       when I =>
              data(7) <= x;
              State <= J;
       when J =>
              if x='0' then
```

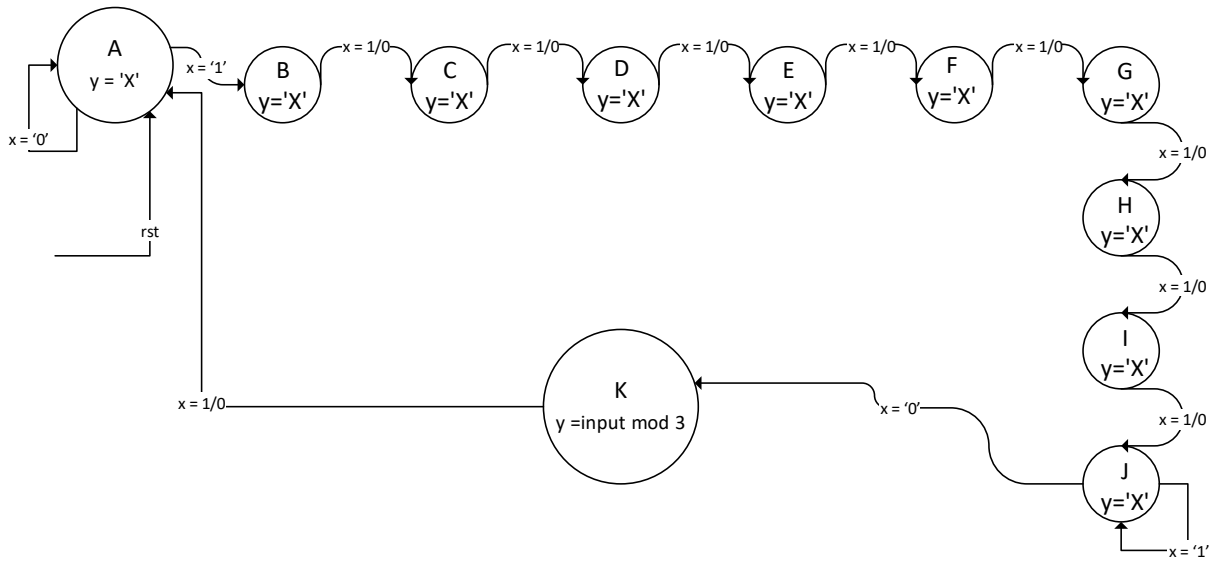
```
                State <= K;
               elsif x='1' then
                State <= J;
               end if;
        when K =>
               State <= A;
            when others =>
                  State <= A;
      end case;
    end if;
  end process;

  temp <= to_integer(unsigned(data)) ;


  process(State)
  variable remember : integer range 0 to 2 ;
  begin
  case State is
    when K => remember := temp mod 3 ;
     y <= std_logic_vector(to_unsigned(remember, 2));
    when others => y <= "XX" ;
    end case;
  end process;
end Behavioral;
```

شکل موج آن به صورت زیر است :



ب) در این توصیف از حالت دو فرآیندی استفاده کردیم که دارای حجم بیشتر است ولی به رفتار نزدیک تر می باشد.

برای این سوال FSM زیر را طراحی کرده ایم :



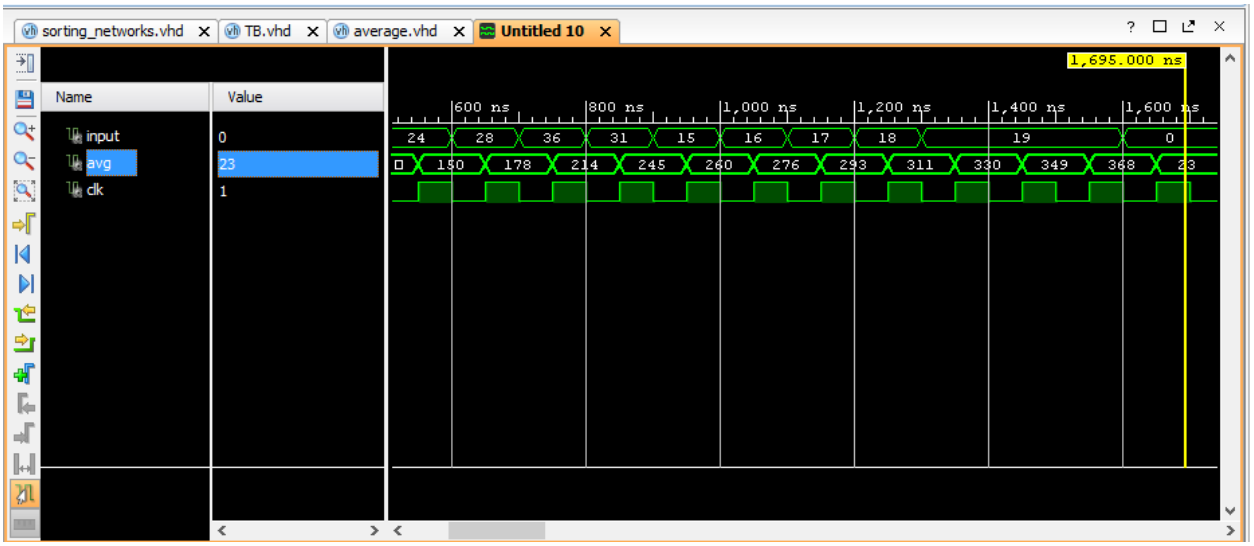
حالات B , C , … , J **به ترتیب مربوط به دریافت بیت های اول تا هشتم است.**

سوال ۸ :

کد این سوال به صورت زیر است :

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

entity average is Port (
input : in natural range 15 to 40;
avg : inout integer := 0 ;
clk : in std_logic
);
end average;

architecture Behavioral of average is

function findAVG (a,b : in natural) return integer is
begin
    if( a = 0 ) then
        return b/16;
    else
        return a+b;
    end if;
end findAVG;

begin

process(clk)
begin
if( clk'event and clk = '1')then
        avg <= findAVG(input , avg);
end if;
end process;
end Behavioral;
```

در اینجا ورودی های اعداد 15,16,17,18,19,19,19,24,27,28,31,32,33,34,35,36 میباشند و همانطور که میبینید میانگین برابر ۲۳ شده است :



سوال نهم :

کد این سوال به صورت زیر است :

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

entity Genom is Port (
input : in std_logic_vector( 1 downto 0 );
reset : in std_logic ;
clk : in std_logic ;
output : out std_logic_vector( 1 downto 0 )
);
end Genom;

architecture Behavioral of Genom is
type State_type IS (start , A, B, C, D , E , F , G , H , I , J , P , Q );
      signal State : State_Type;
signal foundP : std_logic := '0';
signal foundQ : std_logic := '0';
begin
process (clk, reset)
  begin
```

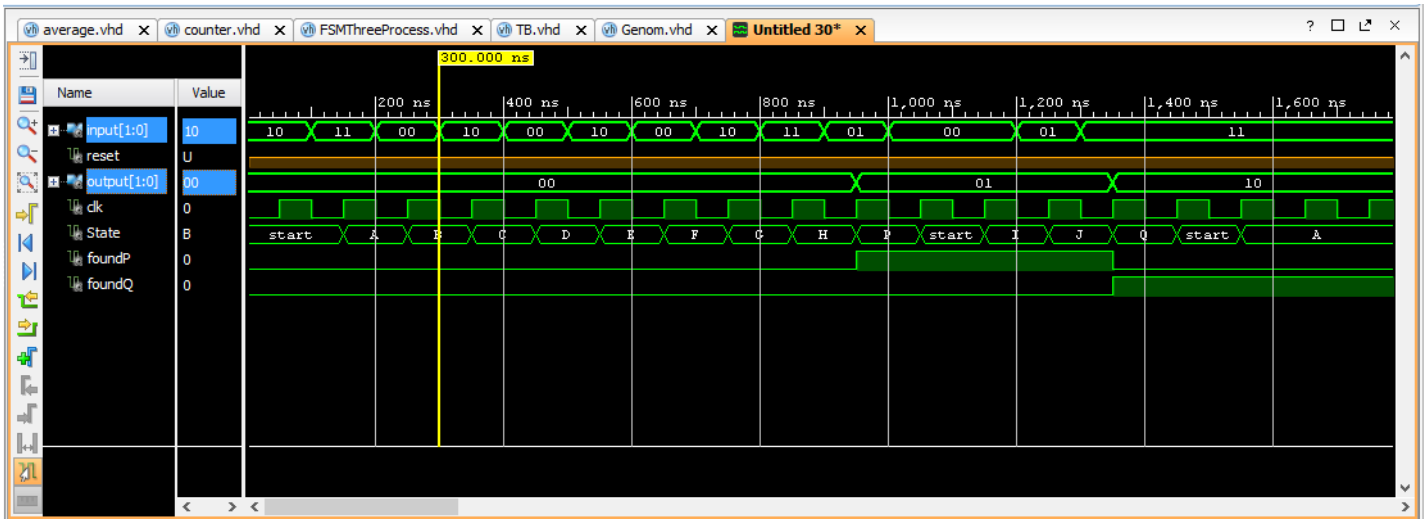
```
if (reset = '1') then
foundP <= '0';
foundQ <= '0';
  State <= start;

elsif (clk'event and clk = '1') then
  case State is
  when start =>
  if input = "11" then
  State <= A ;
  elsif input = "00" then
  State <= I ;
  elsif input = "01" then
  State <= start;
  else
  State <= start;
  end if;
        when A =>
              if input = "11" then
         State <= A ;
         elsif input = "00" then
         State <= B ;
         elsif input = "01" then
         State <= start;
         else
         State <= start;
         end if;
        when B =>
              if input = "11" then
         State <= A ;
         elsif input = "00" then
         State <= I ;
         elsif input = "01" then
         State <= J;
         else
         State <= C;
         end if;
        when C =>
              if input = "11" then
         State <= A ;
         elsif input = "00" then
         State <= D ;
         elsif input = "01" then
```

```
    State <= start;
    else
    State <= start;
    end if;
    when D=>
         if input = "11" then
    State <= A ;
    elsif input = "00" then
    State <= I ;
    elsif input = "01" then
    State <= J;
    else
    State <= E;
    end if;
when E =>
if input = "11" then
            State <= A ;
            elsif input = "00" then
            State <= F ;
            elsif input = "01" then
            State <= start;
            else
            State <= start;
            end if;
when F =>
if input = "11" then
            State <= A ;
            elsif input = "00" then
            State <= I ;
            elsif input = "01" then
            State <= J;
            else
            State <= G;
            end if;
when G =>
if input = "11" then
            State <= H ;
            elsif input = "00" then
            State <= I ;
            elsif input = "01" then
            State <= start;
            else
            State <= start;
```

```
                end if;
    when H =>
    if input = "11" then
                State <= A ;
                elsif input = "00" then
                State <= B ;
                elsif input = "01" then
                foundP <= '1';
                foundQ <= '0';
                State <= P;
                else
                State <= start;
                end if;
    when I =>
    if input = "11" then
                State <= A ;
                elsif input = "00" then
                State <= I ;
                elsif input = "01" then
                State <= J;
                else
                State <= start;
                end if;
    when J =>
    if input = "11" then
                foundQ <= '1';
                foundP <= '0';
                State <= Q ;
                elsif input = "00" then
                State <= I ;
                elsif input = "01" then
                State <= start;
                else
                State <= start;
                end if;
    when P =>
    State <= start ;
    when Q =>
    State <= start ;
        when others =>
              State <= A;
  end case;
end if;
```

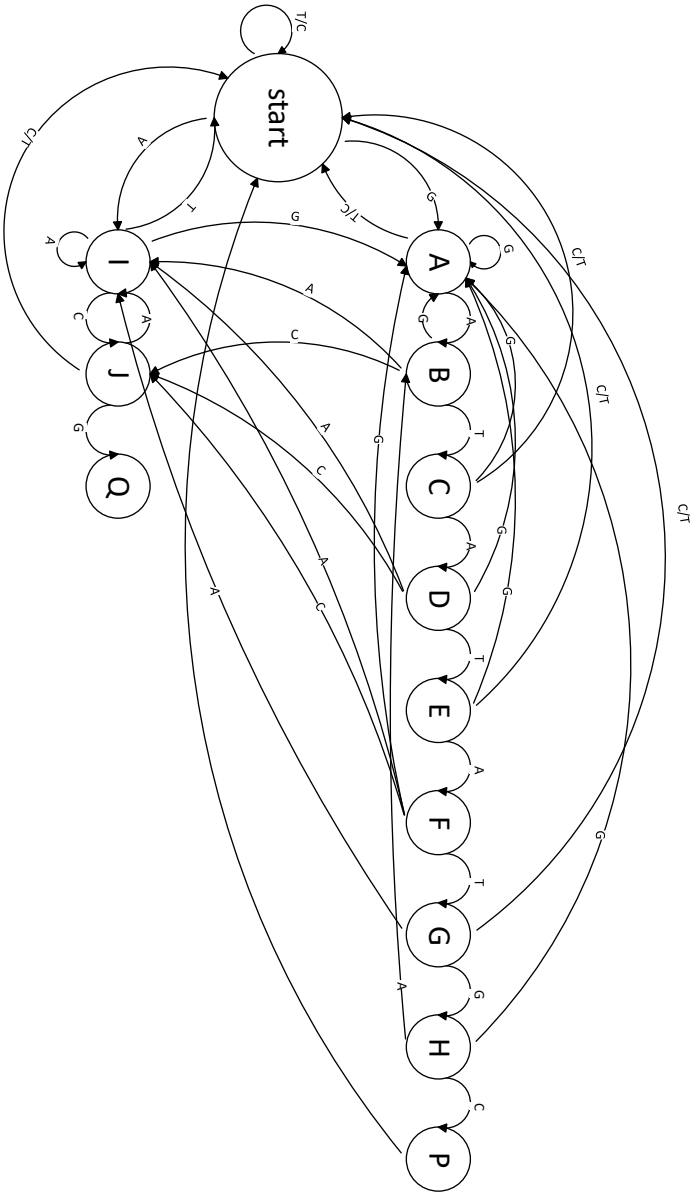
```
  end process;

process(State)
  begin
  case State is
    when P =>
    output <= "01";
    when Q =>
        output <= "10";
    when others =>
        if( foundP = '1' )then
            output <= "01";
            elsif foundQ = '1' then
            output <= "10";
            else
            output <= "00";
            end if;
    end case;
  end process;
end Behavioral;
```

در این سوال استیت های A تا H مربوط به تشخیص رشته P میباشند و استیت های I , J مربوط به تشخیص رشته Q هستند.همچنین استیت P مربوط به هنگامی است که رشته P تشخیص داده شده و استیت Q مربوط به هنگامی است که رشته Q تشخیص داده شده باشد.استیت اولیه برابر با start است . سیگنال های کنترلی foundP , foundQ سیگنال هایی کنترلی هستند که با آن ها می توانیم وقتی یک رشته P را تشخیص دادیم خروجی را در استیت های دیگر ثابت نگه داریم همچنین



برای تشخیص رشته Q نیز سیگنال foundQ همین کار را میکند.برای کد کردن حروف از قاعده زیر پیروی میکنیم که حرف A برابر 00 ، حرف C برابر 01 ، حرف T برابر 10 ، حرف G برابر 11 است.شکل موج زیر مربوط به رشته ورودی TGATATATGCAACG می باشد.که در آن هم رشته P یافت می شود و هم رشته .Q

برای این سوال FSM زیر را طراحی کرده ایم که در صفحه بعد مشاهده میکنید :


start
A
B
C
D
E
F
G
H
I
J
P
Q
T/C
C/T
A
G
T
C

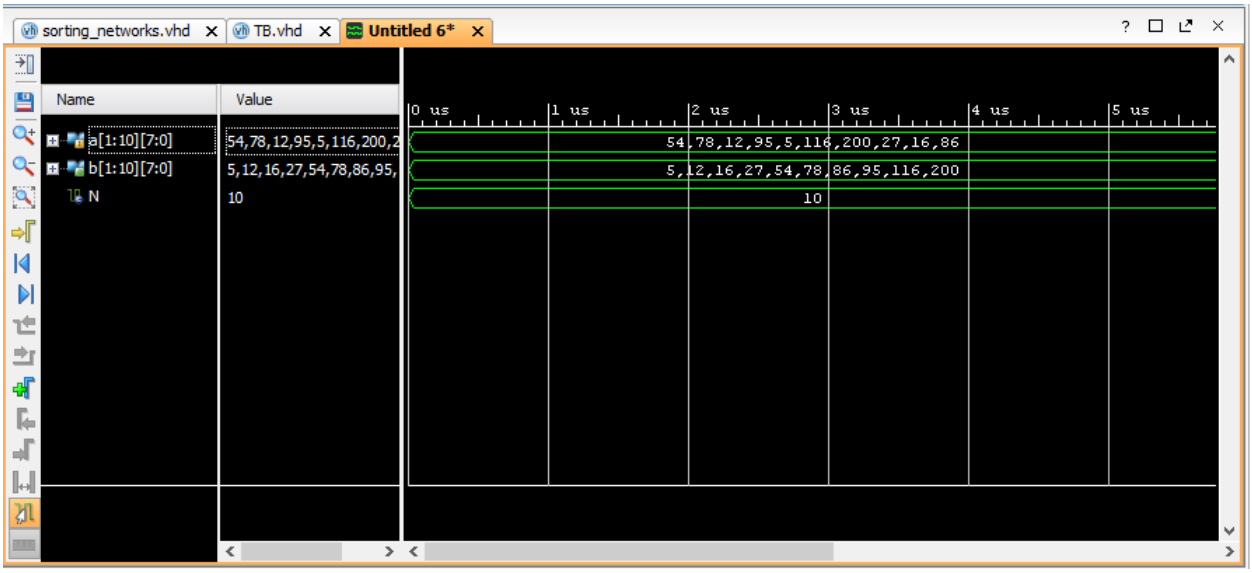
سوال ۱۰ :

کد این سوال به صورت زیر است :

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

package pkg is
  type array_inout is array (natural range <>) of std_logic_vector(7
downto 0);
end package;

package body pkg is
end package body;

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
library work;
use work.pkg.all;

entity sorting_networks is
generic ( N : integer := 10 );
Port (
a : in array_inout( 1 to N );
b : out array_inout( 1 to N )
);
end sorting_networks;

architecture Behavioral of sorting_networks is
component comprator is port (
  a : in std_logic_vector( 7 downto 0 ) ;
  b : in std_logic_vector( 7 downto 0 ) ;
  out1 : out std_logic_vector( 7 downto 0 );
  out2 : out std_logic_vector( 7 downto 0 )
  );
end component;

signal temp1 : array_inout ( 1 to N * (N-1) / 2);
signal temp2 : array_inout ( 1 to N * (N-1) / 2);
begin
```

```
temp1(1) <= a(1);

Assign: for I in 1 to N-1 generate
begin
    temp2(I) <= a( I+1);
end generate;

F: for I in 1 to N-1 generate
begin
G: for J in 1 to N-I generate
begin
      if_G0: if N-I = 1 generate
        comp: comprator port map(temp1(((2*N-I)*(I-1)/2)+J), temp2(((2*N-
I)*(I-1)/2)+J),b(1) , b(2));
      end generate if_G0;
      if_G1: if J = 1 and N-I /= 1 generate
        comp: comprator port map(temp1(((2*N-I)*(I-1)/2)+J), temp2(((2*N-
I)*(I-1)/2)+J), temp1(((2*N-I)*(I-1)/2)+J+N-I), temp1(((2*N-I)*(I-
1)/2)+J+1) );
      end generate if_G1;
      if_G2: if J = N-I and J /= 1 generate
       comp: comprator port map(temp1(((2*N-I)*(I-1)/2)+J), temp2(((2*N-
I)*(I-1)/2)+J), temp2(((2*N-I)*(I-1)/2)+J+N-I-1), b(N-I+1) );
      end generate if_G2;
      if_G3: if J /= N-I and J /= 1 generate
        comp: comprator port map(temp1(((2*N-I)*(I-1)/2)+J), temp2(((2*N-
I)*(I-1)/2)+J), temp2(((2*N-I)*(I-1)/2)+J+N-I-1), temp1(((2*N-I)*(I-
1)/2)+J+1) );
      end generate if_G3;
   end generate G;
end generate F;

end Behavioral;
```

شکل موج آن به صورت زیر میباشد(مقدار N برابر ۱۰ است) :


sorting_networks.vhd
TB.vhd
Untitled 6*
Name
Value
a[1:10][7:0]
54,78,12,95,5,116,200,2
b[1:10][7:0]
5,12,16,27,54,78,86,95,
N
10
0 us
1 us
2 us
3 us
4 us
5 us
54,78,12,95,5,116,200,27,16,86
5,12,16,27,54,78,86,95,116,200
10