"به نام خدا"

پیاده سازی:

شیفت رجیستر 4بیتی در مدهای:

SIPO,SISO,PISO,PIPO

کاربردهای ثبات در کامپیوترها

ارائه طراحی یک رجیستر

اعضای گروه: نگین حقیقی، هلیا وفایی، ستاره باباجانی

استاد درس: دکترمریم محبتی

نیم سال اول 1402-1401

موضوع و اهداف:

این جلسه در تاریخ 1401/9/16، کلاس ساعت 10:30 الی 12:00 برگزار شد.

در این جلسه، در ابتدا شیفت رجیستر 4 بیتی در مد SIPO به همراه تست مربوطه طراحی شد. در این گزارشکار با هم شیفت رجیسترهای 4 بیتی در مدهای SIPO, SISO, PISO, PIPO را طراحی کرده و صحت عملکرد هریک را بررسی میکنیم و تصویری از شبیه سازی صحیح آنها نیز ارائه میدهیم.

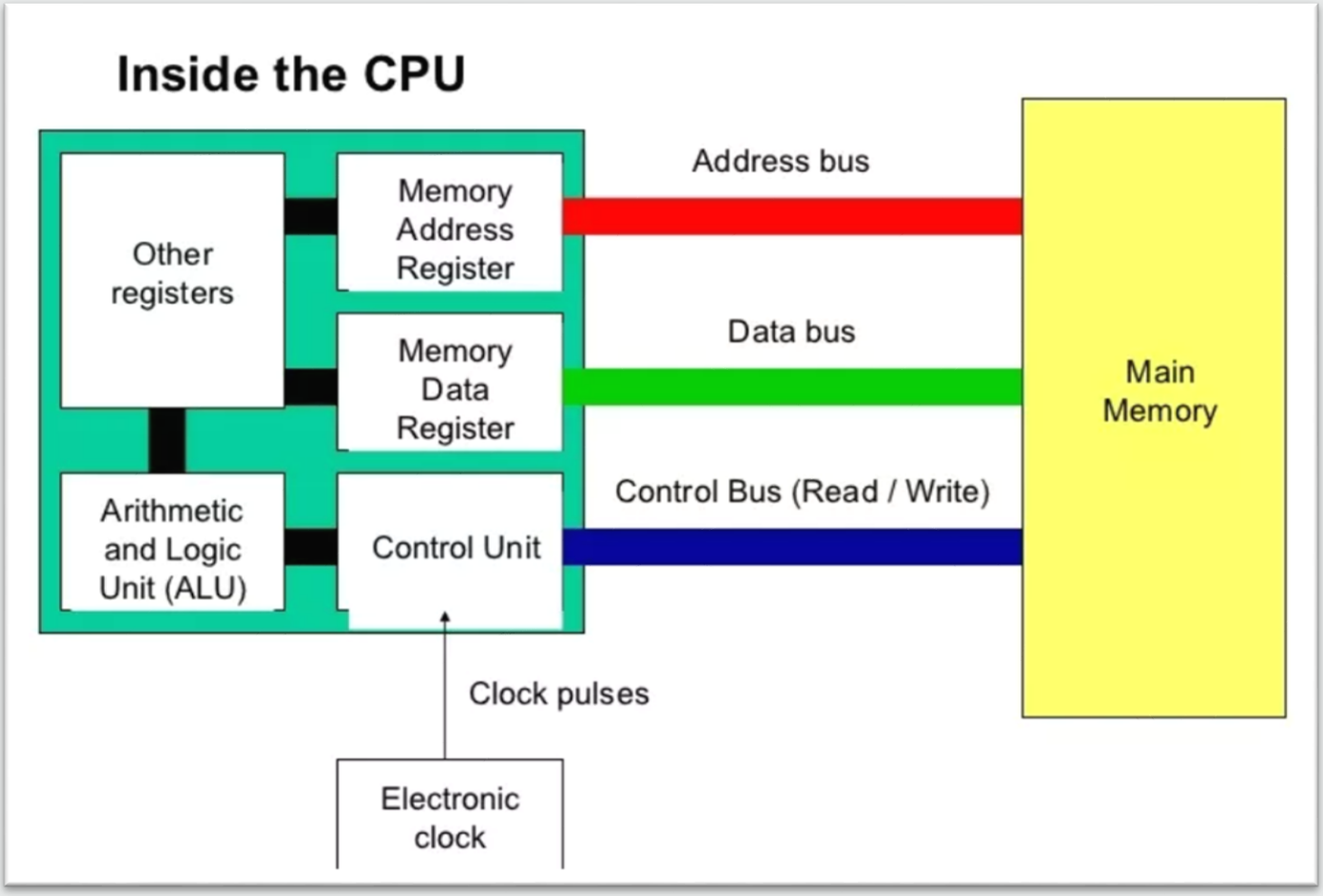
سپس کاربرد ثباتها در کامپیوتر را ذکر میکنیم و در آخر نیز طراحی ای از یک رجیستر دلخواه ارائه خواهیم داد.

هر یک از مدارات فوق، در ise پیاده سازی میشوند و کد تمامی آن ها نیز به پیوست ارسال میشود.

ثبات پردازنده:

ثبات پردازنده(رجیستر پردازنده) یا ثباتهای حافظه پردازنده، جهت ذخیره داده ها و دستورالعمل ها و انتقال آنها استفاده میشوند. CPUکامپیوتر از مهمترین و پیچیده ترین جزهای یک کامپیوتر هست و برای همین یک سخت افزار یک بعدی و تک منظوره نیست و وظایف زیادی برعهده دارد، به همین منظور رجیستر در پردازنده انواع مختلفی دارد که هرکدام در هندل و پردازش کردن دستورالعملها برای CPU بسیار مهم اند. ثباتها معمولا شامل مقدار اندکی حافظه سریع هستند و بعضی از رجیسترها، عملیات سخت افزاری خاصی دارند مثلا میتوانند فقط خواندنی یا فقط نوشتنی باشند. ثباتها را

میتوان به دو دسته کلی "ثباتهای نشانی پذیر" و "ثباتهای آدرس ناپذیر" تقسیم کرد.



در زیر چند رجیستر مهم در پردازنده و ریزپردازنده به همراه توضیح مختصری می آوریم:

❖ Memory Address Register یا MAR

این Register همانطور که از نامش نیز پیداست آدرس های حافظه از داده ها و دستورالعمل ها را در خود نگه میدارد. این Register برای دسترسی به داده ها و دستورالعمل ها از حافظه RAM در طی اجرا شدن دستورالعمل ها مورد استفاده قرار میگیرد.

❖ Program Counter یا PC

رجیستری است که به آن Instruction Pointer یا IP نیز میگویند. به این رجیستر گاهاً Instruction Address Register نیز گفته میشود. این رجیستر مسیر آدرس حافظه دستورالعملی که بعد از تمام شدن پردازش دستورالعمل فعلی بایستی مورد پردازش قرار بگیرد را در خود ذخیره میکند. به عبارت دیگر این Register تا زمانیکه پردازش روی دستورالعمل فعلی به اتمام نرسیده است آدرس حافظه دستورالعمل بعدی را در خود نگه میدارد.

❖Accumulator Register یا AC

این رجیستر برای ذخیره سازی نتایج دستوراتی که توسط واحد ALU پردازنده سیستم انجام شده است مورد استفاده قرار میگیرد. هنگامی که CPUدستورات را مورد پردازش قرار داد و تمام شد نتیجه دستورات در رجیستر AC به صورت موقت ذخیره میگردد.

❖Memory Data Register یا MDR

این رجیستر یکی از مهم ترین رجیستر های CPU است رجیستر MDR رجیستر واحد CUاز پردازنده میباشد و شامل اطلاعاتی است که باید در حافظه RAM سیستم یا سایر حافظه ها ذخیره شود، همچنین این رجیستر میتواند شامل داده هایی باشد که با عملیات Fetch یا واکشی داده ها از یک دستگاه ذخیره سازی بدست آمده باشد. رجیستر MDR همانند بافر عمل میکند و شامل کپی اطلاعاتی است که از حافظه RAM طی عملیات Fetch به این رجیستر منتقل شده است تا توسط CPU مورد پردازش قرار گیرد.

❖Index Register

این رجیستر در پردازنده اعداد یا مقادیری را در خود نگه داری میکند که میتواند از بخشی از آدرس یک دستورالعمل کم یا به آن اضافه شود تا به یک آدرس موثر و کارامد تبدیل شود.

❖Memory Buffer Register یا MBR

این Register محتویات داده یا دستورالعمل هایی که از حافظه خوانده یا روی آن نوشته میشوند را در خود نگهداری میکند.

❖Data Register

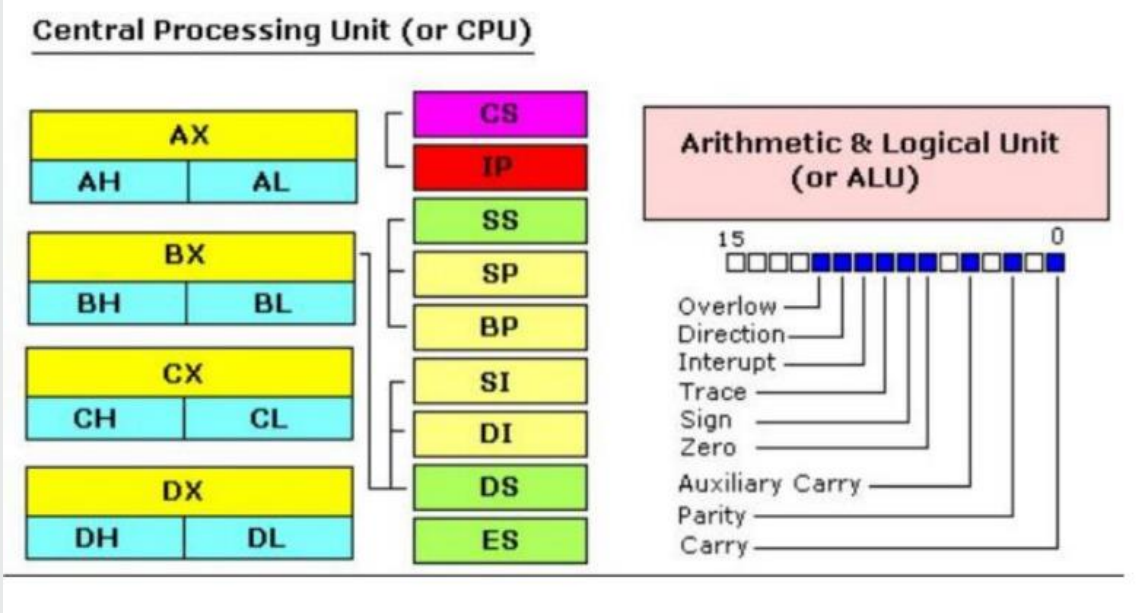
این رجیستر برای ذخیره سازی موقتی داده هایی که از دستگاه های ذخیره سازی خوانده یا نوشته میشوند مورد استفاده قرار میگیرد.

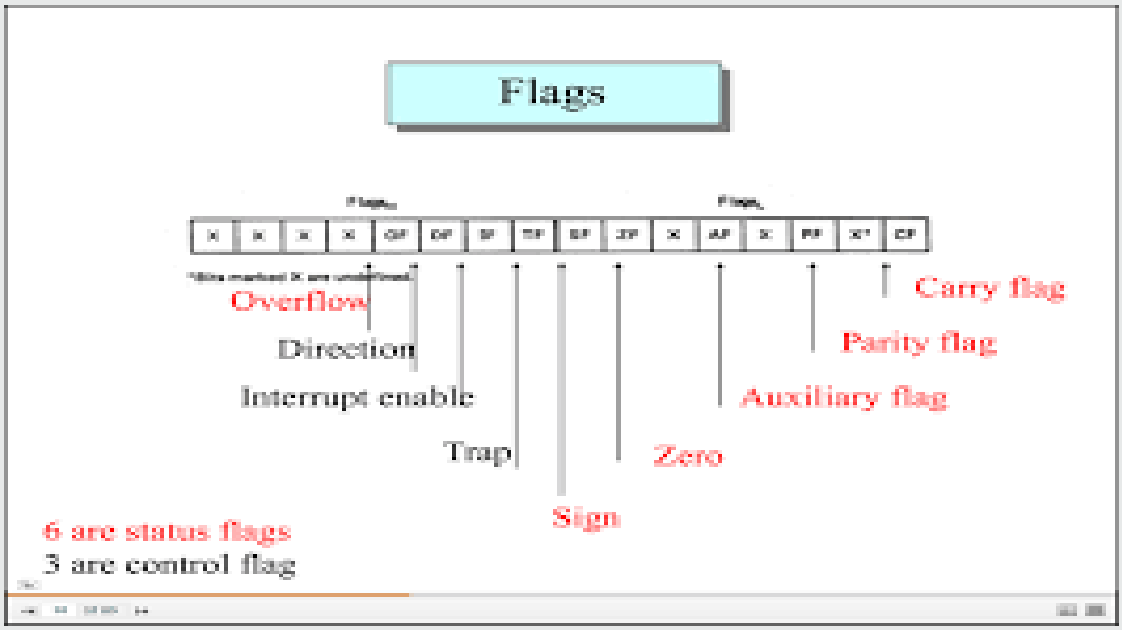
❖Flag Register

یک ثبات وضعیت در ریزپردازنده‌ها اینتل x86 است که شامل وضعیت فعلی پردازنده است .

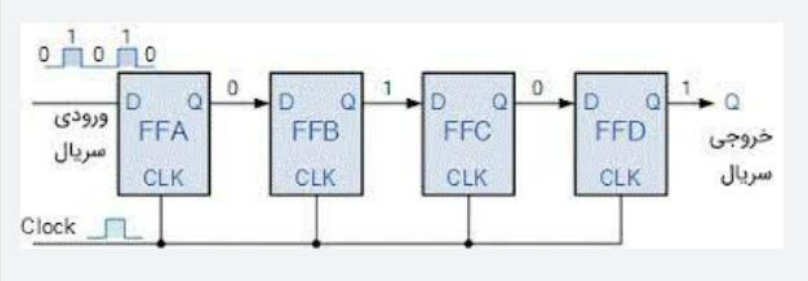
طراحی یک رجیستر:

ALU: Arithmetic and Logical Unit





طراحی شیفت رجیستر(یک نمونه):



SIPO:

ابتدا دو ورودی d و clk و یک خروجی output(از نوع STD_LOGIC_VECTOR(3 DOWNTO 0)) برای این مدار در نظر میگیریم کد مدار به صورت زیر است:

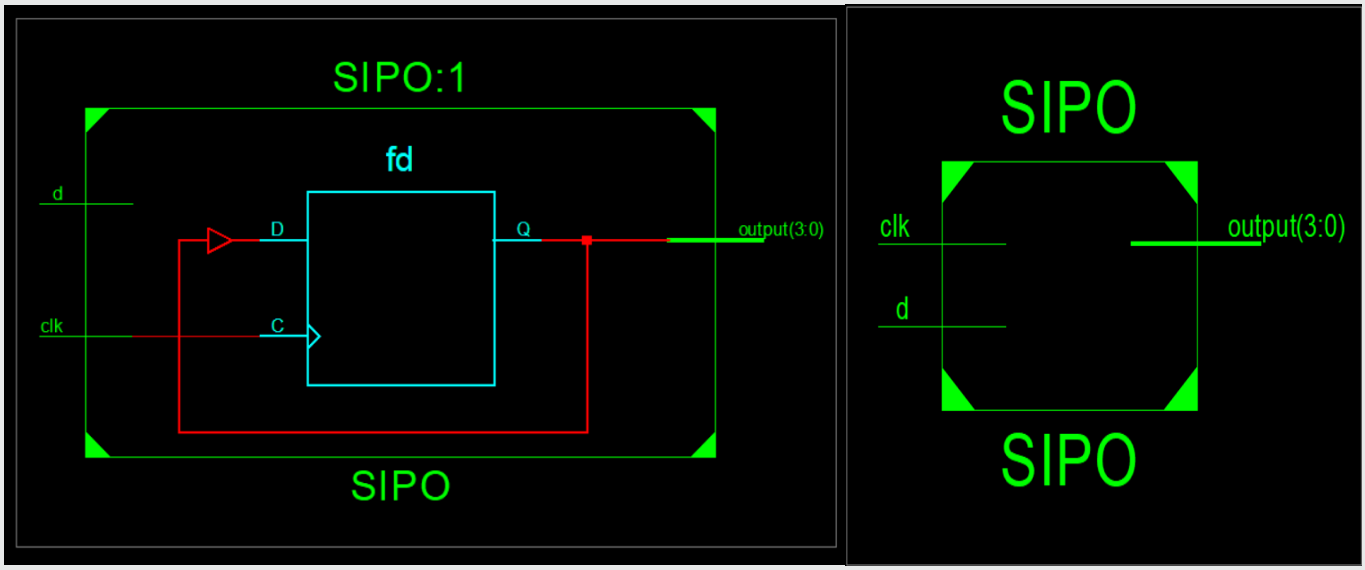
(فایل کدها به پیوست ارسال میگردد)

```
19  ----------------------------------------------------------------------------------
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22
23  -- Uncomment the following library declaration if using
24  -- arithmetic functions with Signed or Unsigned values
25  --use IEEE.NUMERIC_STD.ALL;
26
27  -- Uncomment the following library declaration if instantiating
28  -- any Xilinx primitives in this code.
29  --library UNISIM;
30  --use UNISIM.VComponents.all;
31
32  entity SIPO is
33      generic(n:positive := 4);
34      Port ( d : in  STD_LOGIC;
35             clk : in  STD_LOGIC;
36             output : out  STD_LOGIC_VECTOR(n-1 downto 0));
37  end SIPO;
38
39  architecture Behavioral of SIPO is
40  signal temp_out : STD_LOGIC_VECTOR(n-1 downto 0);
41  begin
42  process(clk)
43  begin
44  if rising_edge(clk) then
45  temp_out <= d & temp_out(n-1 downto 1);
46  end if;
47  end process;
48  output <= temp_out;
49
50  end Behavioral;
51
```

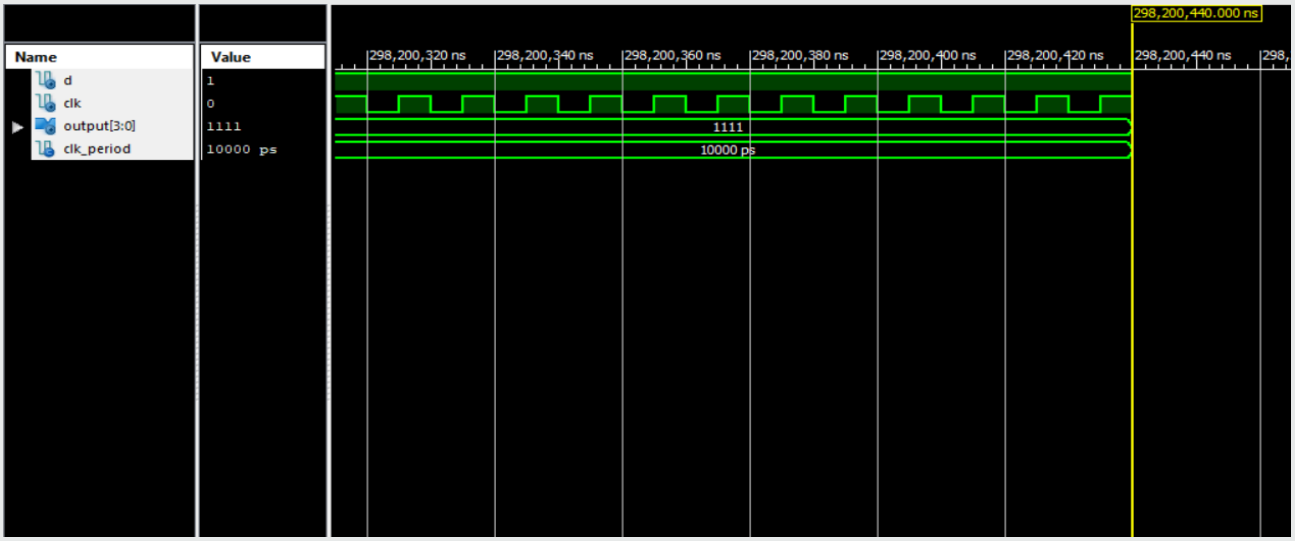
کد فایل تست نیز به صورت زیر است. و فایل کد کامل آن نیز در پیوست قرار دارد:

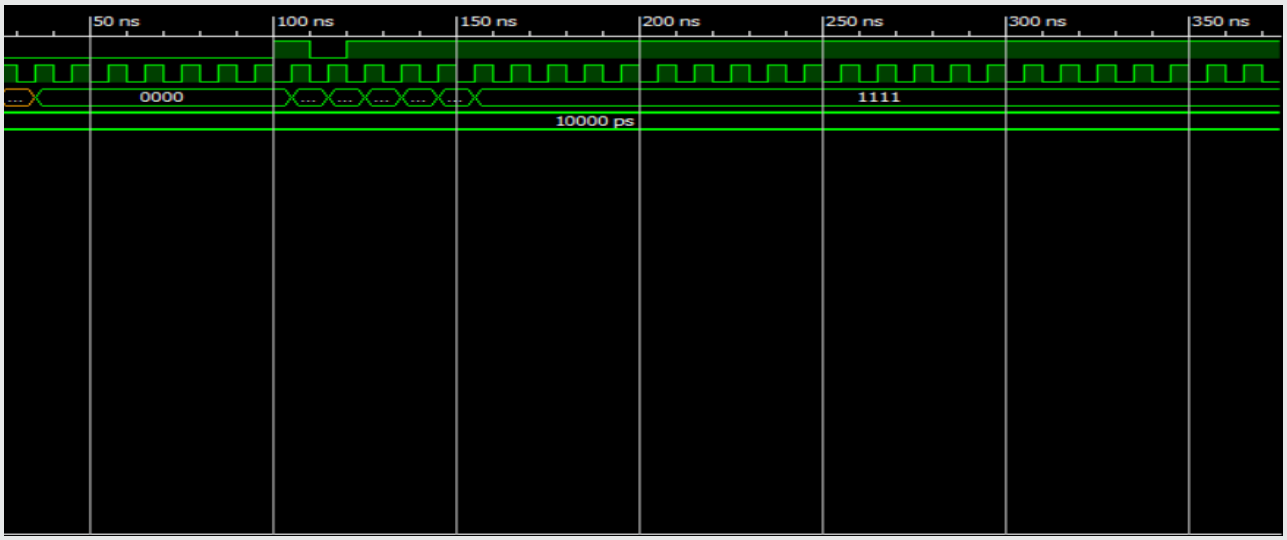
```
53      signal clk : std_logic := '0';
54
55      --Outputs
56      signal output : std_logic_vector(3 downto 0);
57
58      -- Clock period definitions
59      constant clk_period : time := 10 ns;
60
61  BEGIN
62
63      -- Instantiate the Unit Under Test (UUT)
64      uut: SIPO PORT MAP (
65             d => d,
66             clk => clk,
67             output => output
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         -- hold reset state for 100 ns.
84         wait for 100 ns;
85         d <= '1';
86         wait for clk_period;
87         d <= '0';
88         wait for clk_period;
89         d <= '1';
90         wait for clk_period*10;
91         -- insert stimulus here
92
93         wait;
94      end process;
95
96  END;
97
```

و همچنین شکل مدار به صورت زیر میباشد:



و سپس، صحت عملکرد این مدار را سنجیدیم:





:PIPO

ابتدا دو ورودی d (از نوع STD_LOGIC_VECTOR(3 DOWNTO 0)) و clk و یک خروجی output (از نوع STD_LOGIC_VECTOR(3 DOWNTO 0)) برای این مدار در نظر میگیریم کد مدار به صورت زیر است:

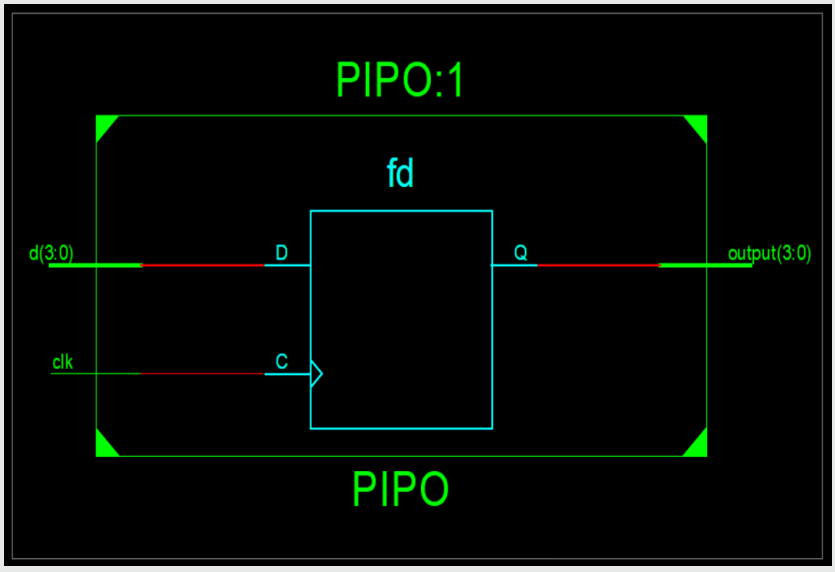
(فایل کدها به پیوست ارسال میگردد)

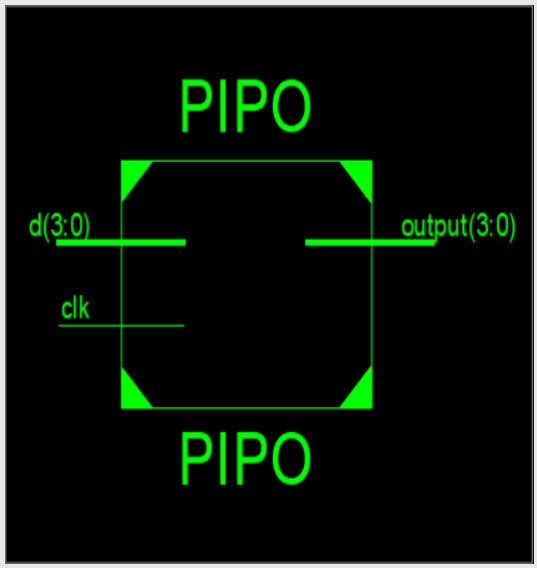
```
19  ----------------------------------------------------------------------------
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22
23  -- Uncomment the following library declaration if using
24  -- arithmetic functions with Signed or Unsigned values
25  --use IEEE.NUMERIC_STD.ALL;
26
27  -- Uncomment the following library declaration if instantiating
28  -- any Xilinx primitives in this code.
29  --library UNISIM;
30  --use UNISIM.VComponents.all;
31
32  entity PIPO is
33  Port( d: in STD_LOGIC_VECTOR(3 downto 0);
34  clk : in STD_LOGIC;
35  output : out STD_LOGIC_VECTOR(3 downto 0));
36  end PIPO;
37
38  architecture Behavioral of PIPO is
39  signal mid : STD_LOGIC_VECTOR(3 downto 0) := (others => '0');
40
41  begin
42  process(clk)
43  begin
44  if rising_edge(clk) then
45  mid <= d;
46  end if;
47  end process;
48  output <= mid;
49
50  end Behavioral;
51
```

کد فایل تست نیز به صورت زیر است. و فایل کد کامل آن نیز در پیوست قرار دارد:

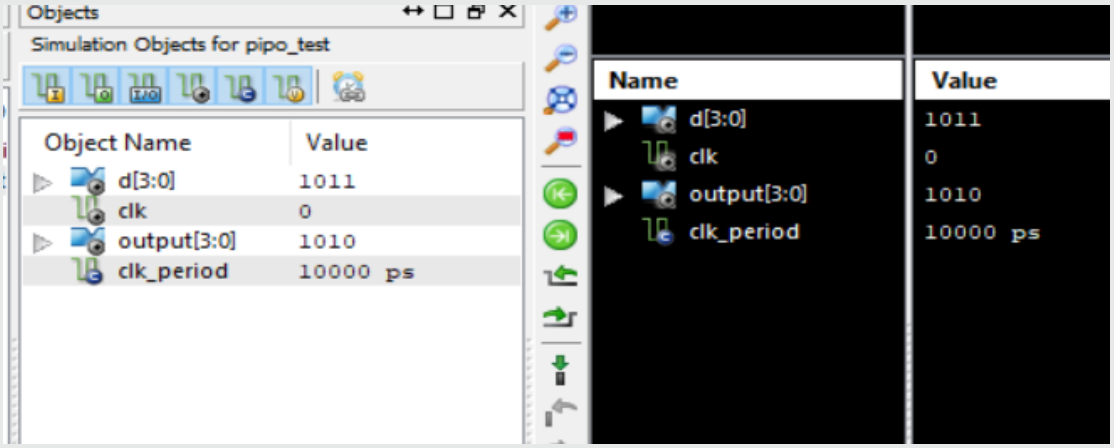
```
   -- Clock process definitions
   clk_process :process
   begin
      clk <= '0';
      wait for clk_period/2;
      clk <= '1';
      wait for clk_period/2;
   end process;


   -- Stimulus process
   stim_proc: process
   begin
      -- hold reset state for 100 ns.
      wait for 100 ns;
      d <= "1011";
      wait for clk_period*2;
      d<= "1010";
      wait for clk_period;

      -- insert stimulus here
      wait for clk_period*5;
   end process;

END;
```

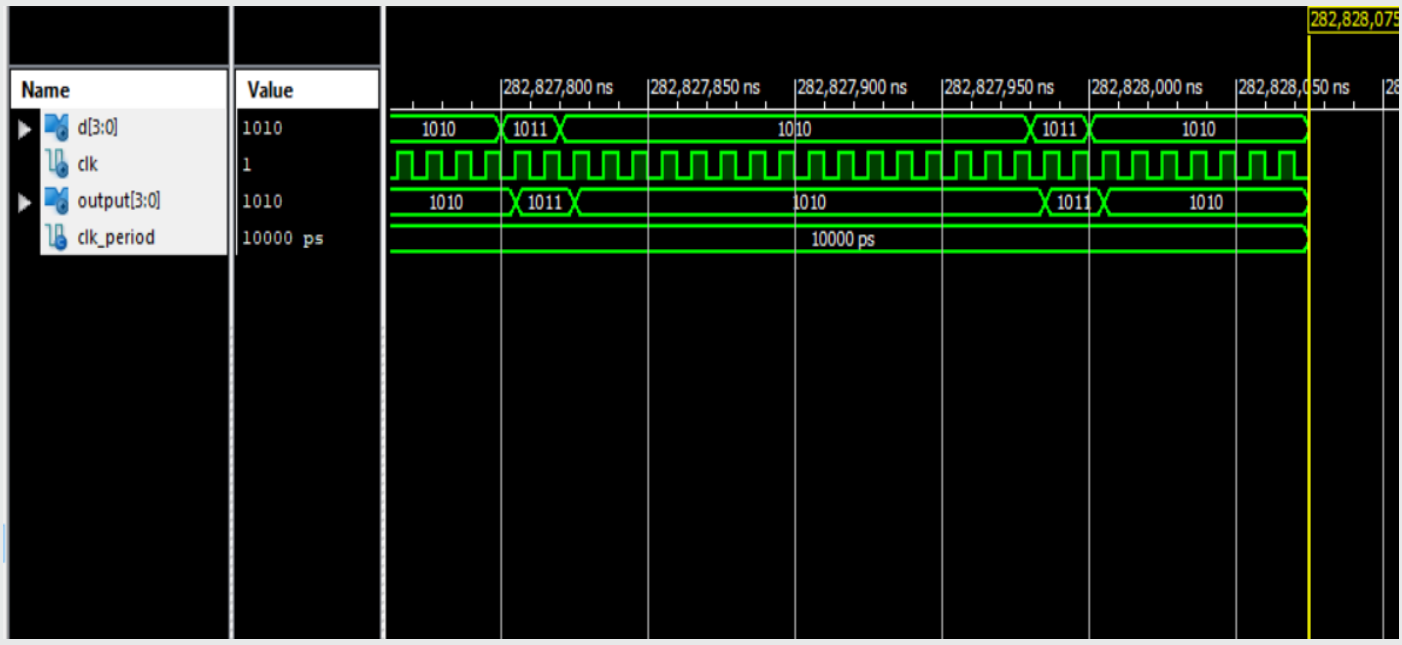
و همچنین شکل مدار به صورت زیر میباشد:

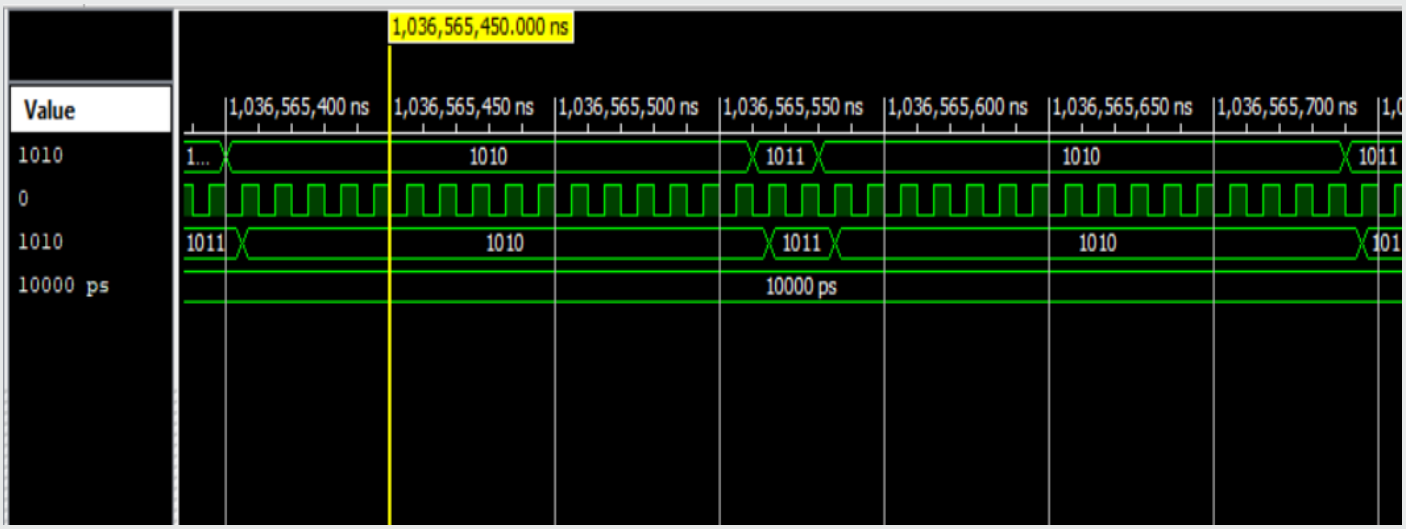




و سپس، صحت عملکرد این مدار را سنجیدیم:







PISO:

ابتدا دو ورودی d(از نوع STD_LOGIC_VECTOR(3 DOWNTO 0)) و clk و یک خروجی output برای این مدار در نظر میگیریم کد مدار به صورت زیر است:

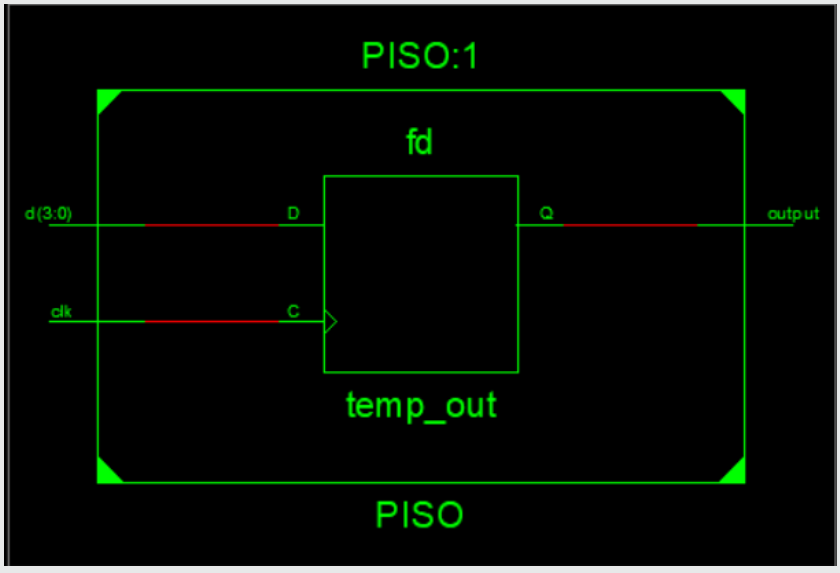
(فایل کدها به پیوست ارسال میگردد)

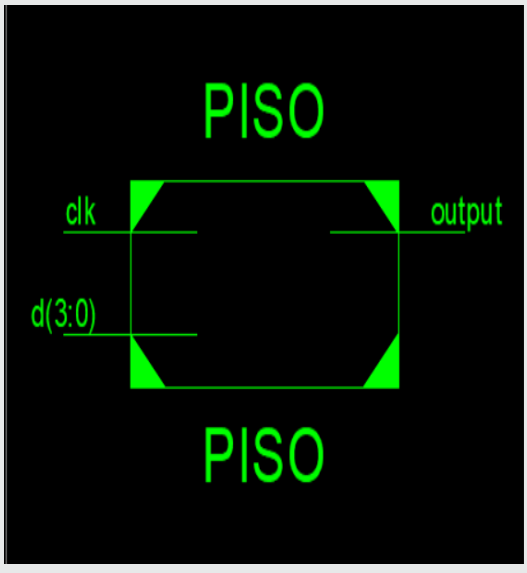
```
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22
23  -- Uncomment the following library declaration if using
24  -- arithmetic functions with Signed or Unsigned values
25  --use IEEE.NUMERIC_STD.ALL;
26
27  -- Uncomment the following library declaration if instantiating
28  -- any Xilinx primitives in this code.
29  --library UNISIM;
30  --use UNISIM.VComponents.all;
31
32  entity PISO is
33      Port ( d : in  STD_LOGIC_VECTOR(3 DOWNTO 0);
34             clk : in  STD_LOGIC;
35             output : out  STD_LOGIC);
36  end PISO;
37
38  architecture Behavioral of PISO is
39  SIGNAL temp_out : STD_LOGIC_VECTOR(3 DOWNTO 0);
40
41  begin
42  PROCESS(clk)
43  BEGIN
44  IF rising_edge(clk) then
45  temp_out <= d;
46  END IF;
47  END PROCESS;
48  output <= temp_out(0);
49
50  end Behavioral;
```

کد فایل تست نیز به صورت زیر است. و فایل کد کامل آن نیز در پیوست قرار دارد:

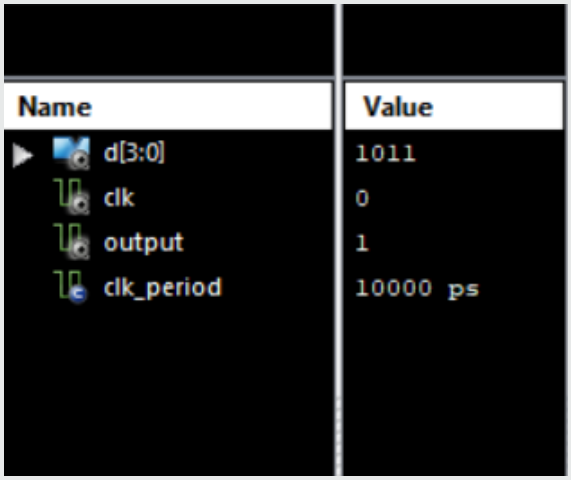
```
70      -- Clock process definitions
71      clk_process :process
72      begin
73          clk <= '0';
74          wait for clk_period/2;
75          clk <= '1';
76          wait for clk_period/2;
77      end process;
78
79
80      -- Stimulus process
81      stim_proc: process
82      BEGIN
83          -- hold reset state for 100 ns.
84          wait for 100 ns;
85          d <= "1011";
86          wait for clk_period*2;
87          d <= "1001";
88          wait for clk_period;
89          -- insert stimulus here
90
91          wait for clk_period*5;
92      END PROCESS;
93
94  END;
```

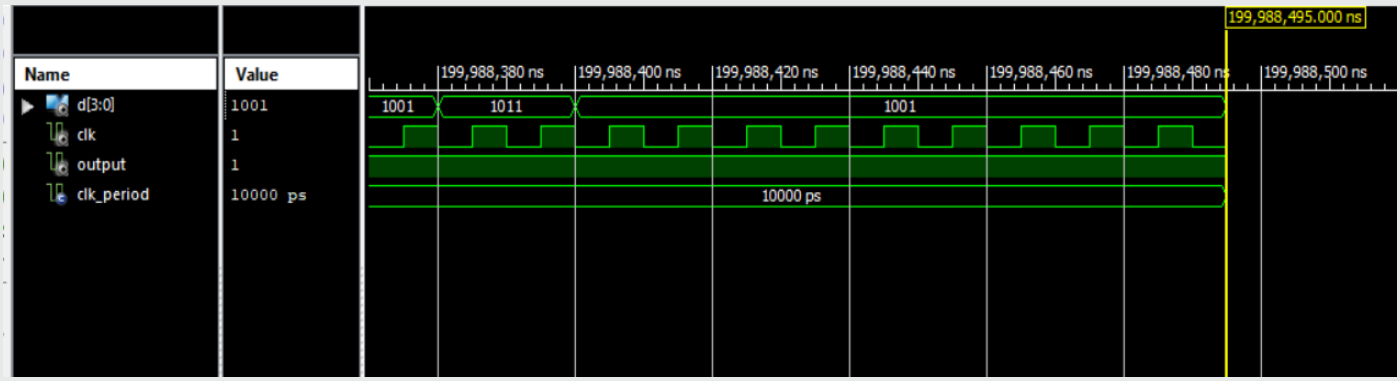
و همچنین شکل مدار به صورت زیر میباشد:

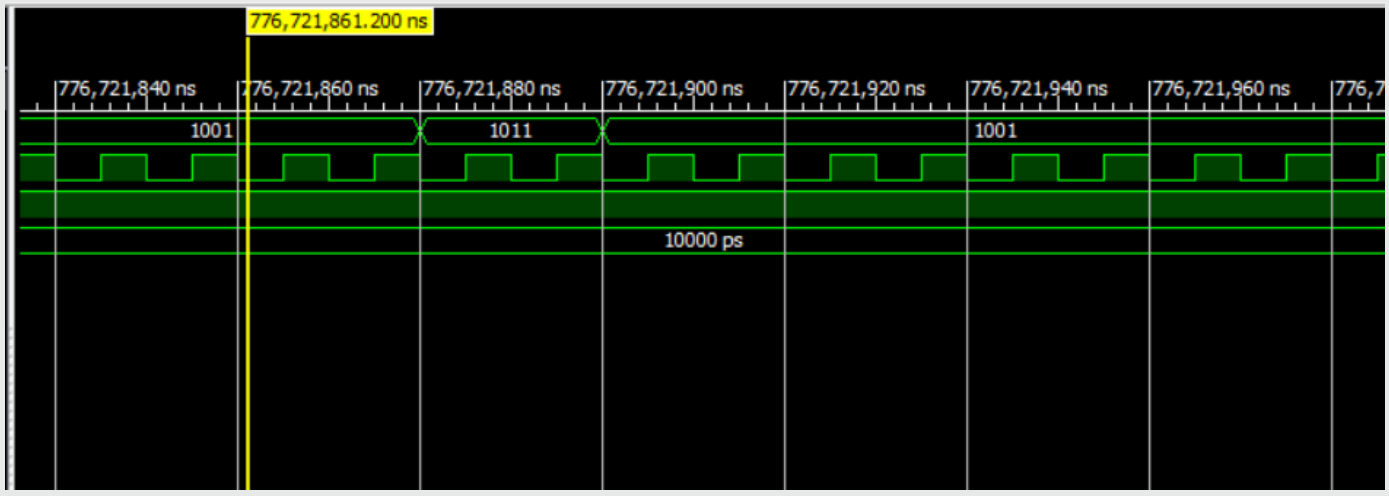


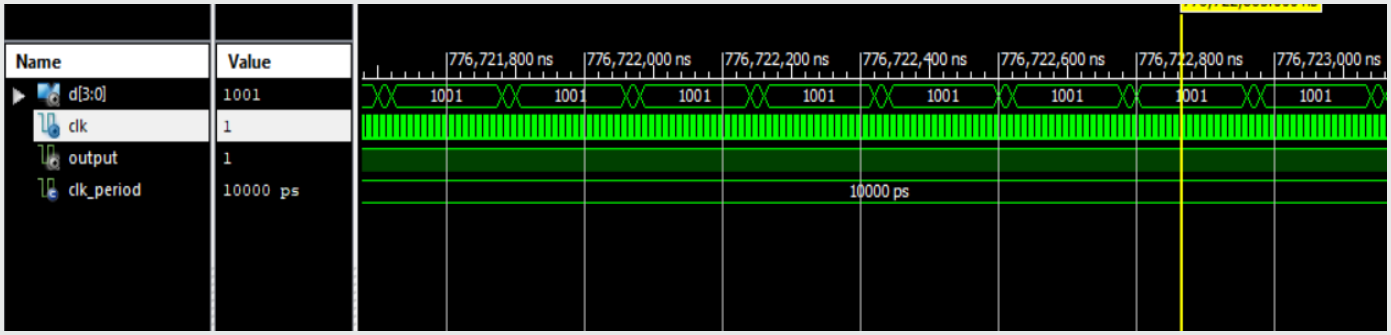


و سپس، صحت عملکرد این مدار را سنجیدیم:









SISO:

ابتدا دو ورودی d و clk و یک خروجی output برای این مدار در نظر میگیریم کد مدار به صورت زیر است:

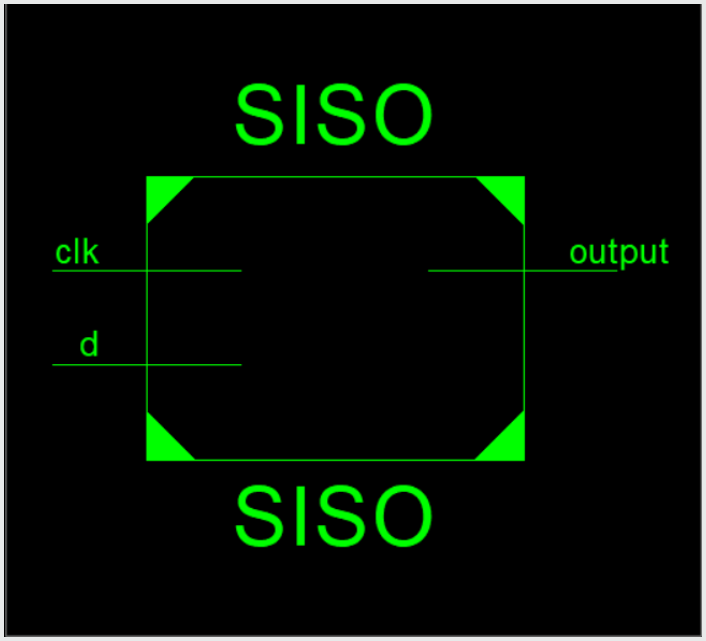
(فایل کدها به پیوست ارسال میگردد)

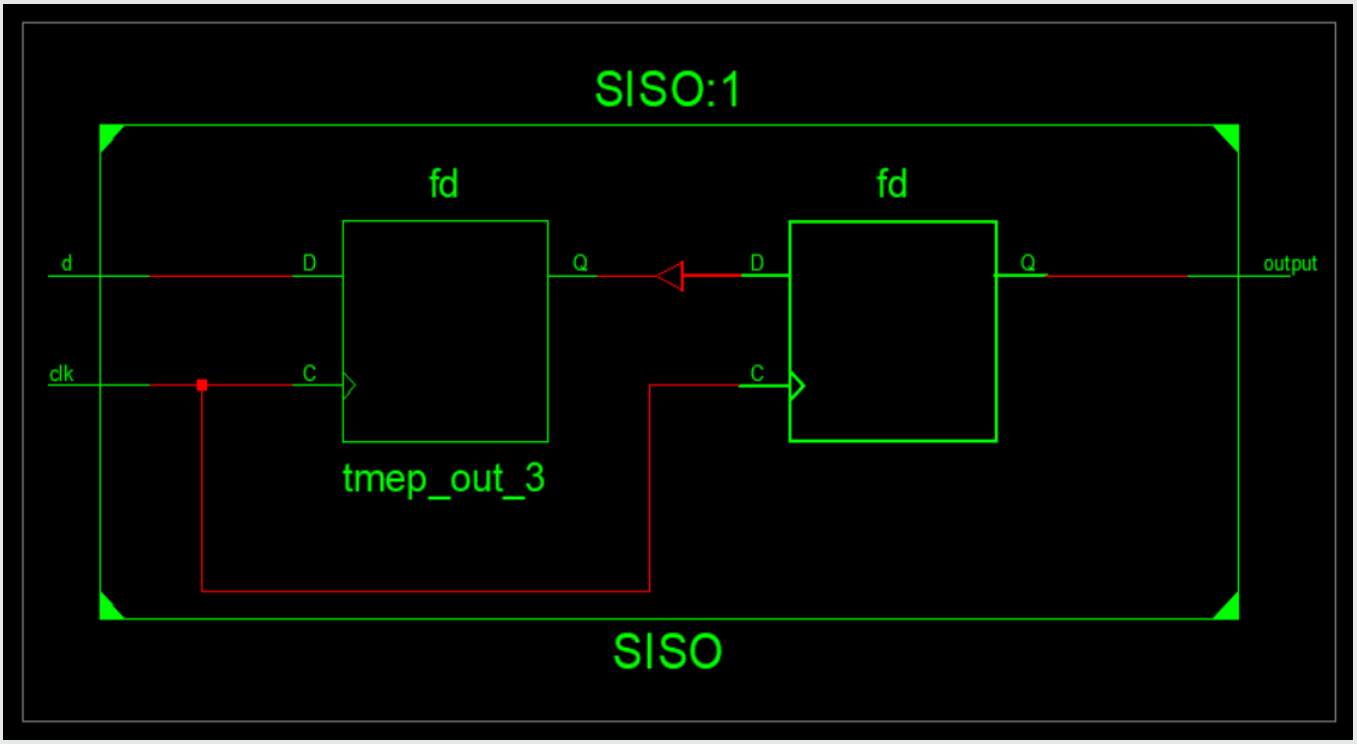
```
----------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- 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 primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity SISO is
    Port ( d : in  STD_LOGIC;
           clk : in  STD_LOGIC;
           output : out  STD_LOGIC);
end SISO;

architecture Behavioral of SISO is
signal tmep_out : STD_LOGIC_VECTOR(3 DOWNTO 0);


begin
PROCESS(clk)
BEGIN
IF RISING_EDGE(clk) THEN
tmep_out <= d & tmep_out(3 DOWNTO 1);
END IF;
END PROCESS;
output <= tmep_out(0);

end Behavioral;
```

کد فایل تست نیز به صورت زیر است. و فایل کد کامل آن نیز در پیوست قرار دارد:

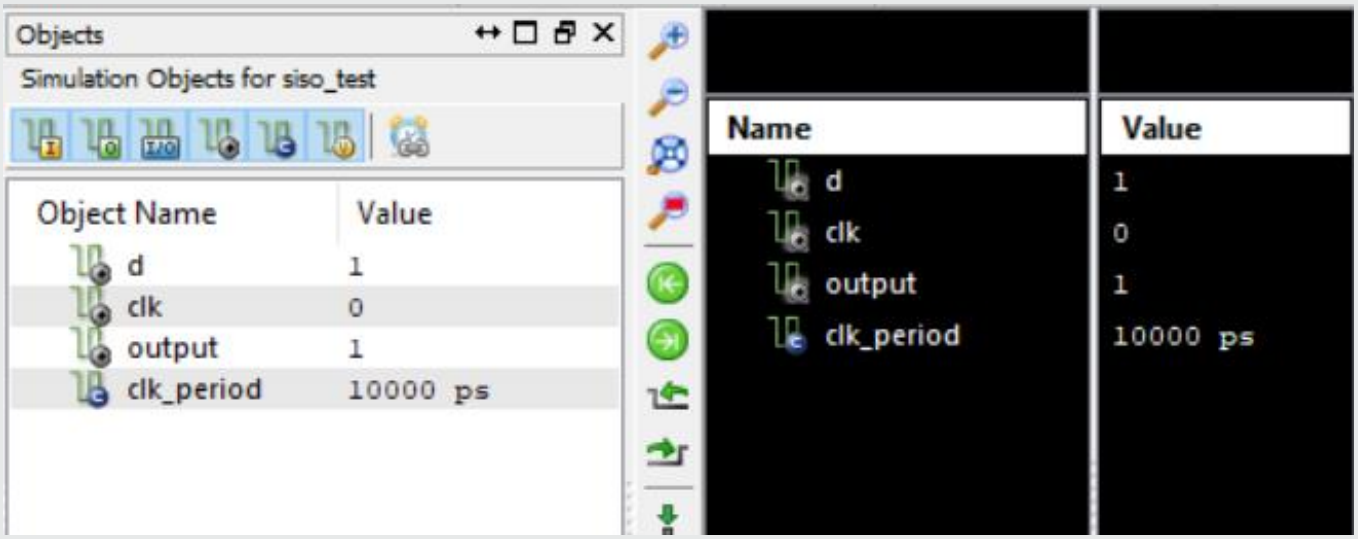
```
71      clk_process :process
72      begin
73          clk <= '0';
74          wait for clk_period/2;
75          clk <= '1';
76          wait for clk_period/2;
77      end process;
78
79
80      -- Stimulus process
81      stim_proc: process
82      begin
83          -- hold reset state for 100 ns.
84          wait for 100 ns;
85          d <= '1';
86          wait for clk_period;
87          d <= '0';
88          wait for clk_period;
89          d <= '1';
90  wait for clk_period*10;
91          -- insert stimulus here
92
93          wait;
94      end process;
95
96  END;
```

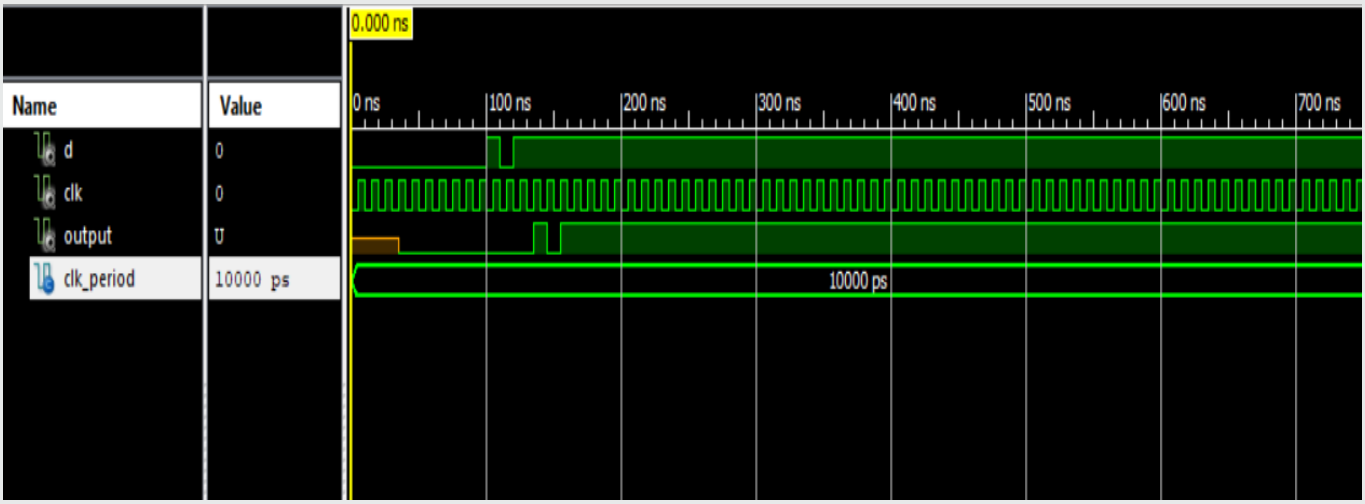
و همچنین شکل مدار به صورت زیر میباشد:

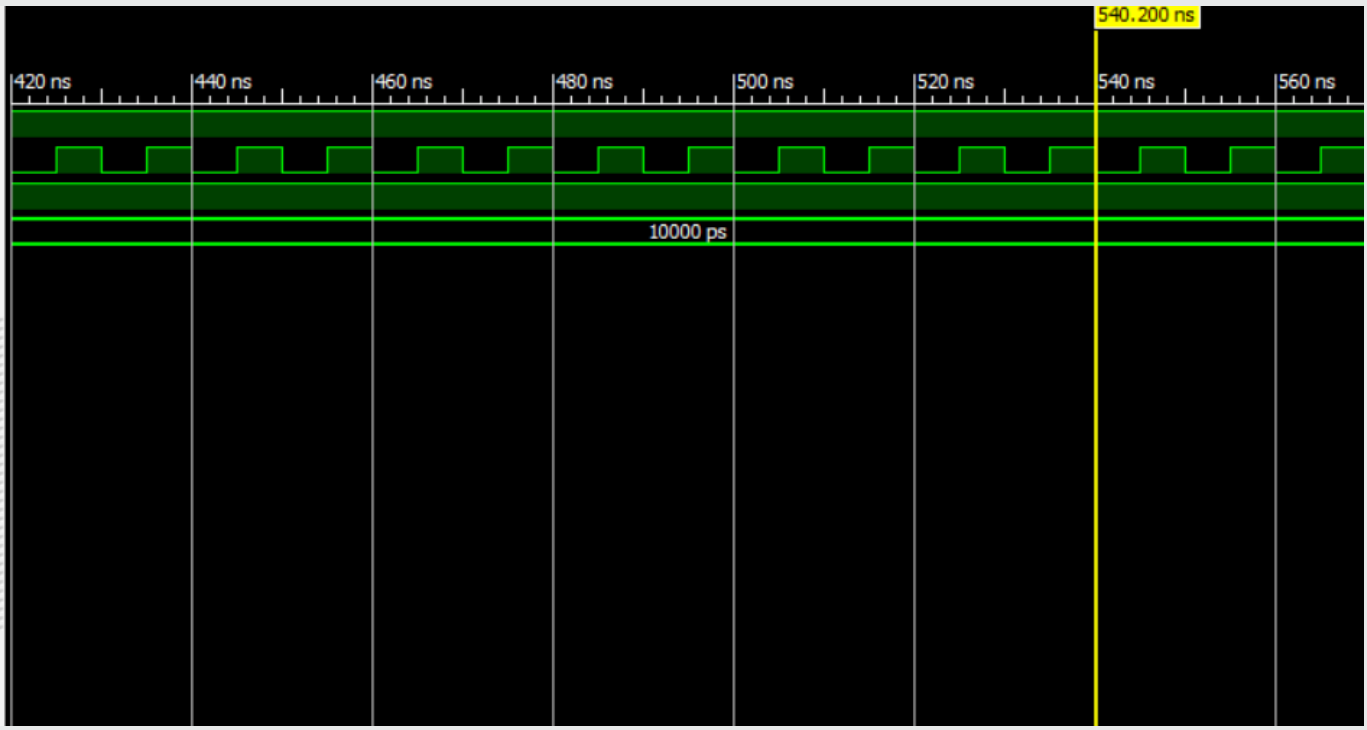


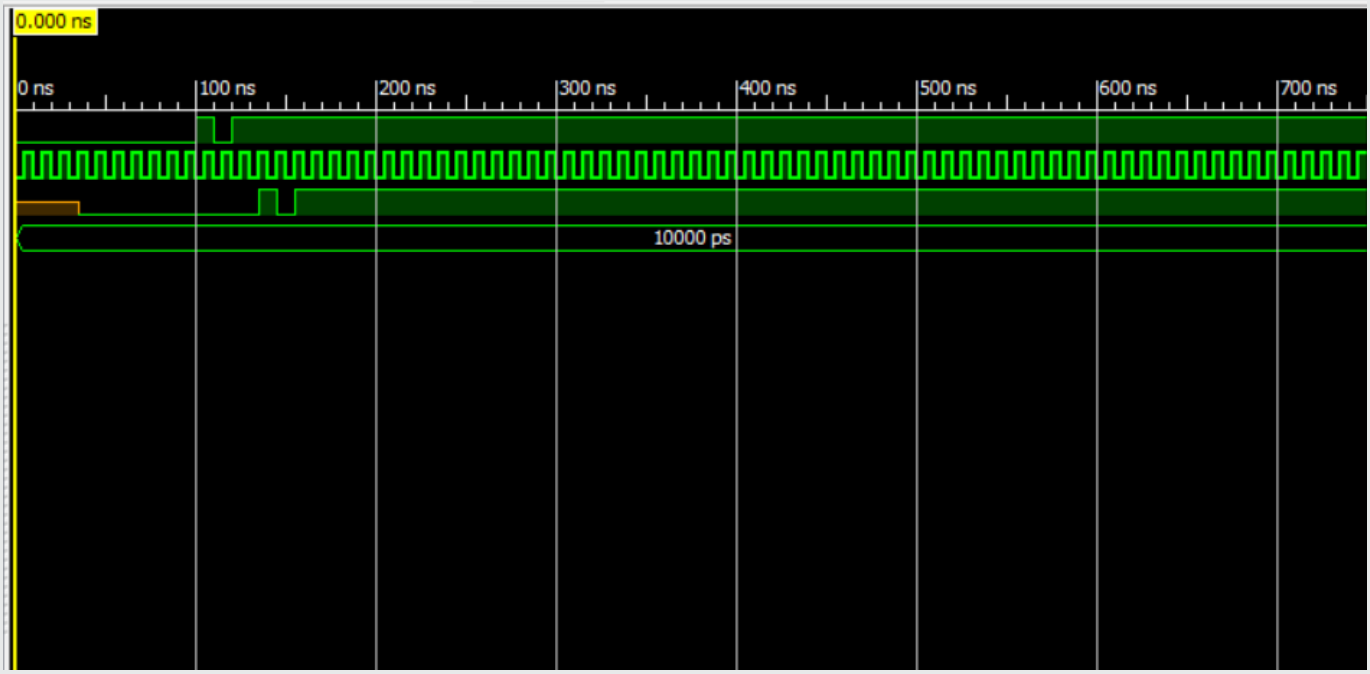


و سپس، صحت عملکرد این مدار را سنجیدیم:






540.200 ns
420 ns
440 ns
460 ns
480 ns
500 ns
520 ns
540 ns
560 ns
10000 ps


0.000 ns
0 ns
100 ns
200 ns
300 ns
400 ns
500 ns
600 ns
700 ns
10000 ps

"پایان"