به نام خدا

تمرین پنجم

محمدمهدی آقاجانی

۹۳۳۱۰۵۶

استاد : دکتر صاحب الزمانی

سوال ۶

الف) طراح برای طراحی توامان ابتدا باید سیستم را به دو بخش نرم افزار و سخت افزار افراز کند . تصمیم گیری در افزار مناسب گاهی با روش آزمون و خطا انجام میشود . طراحی با جا به جایی بخش هایی بین سخت افزا رو نرم افزار و تحلیل هزینه و توان مصرفی و سرعت در نهایت بهترین تصمیم را میگیرد و بهترین افراز را انتخاب میکند . البته در افراز عوامل دیگری همچون راحتی پیاده سازی هم موثر می باشد مثلا ممکن است بخشی را به صرفه باشد که به صورت سخت افزاری پیاده سازی کنیم ولی چون از لحاظ پیاده سازی بسیار پیچیده میشود ترجیح میدهیم که آن را نرم افزاری پیاده سازی کنیم یا اینکه ممکن است پیاده سازی یک ماژول به صورت سخت افزاری سریع باشد ولی چون ارتباطات آن ماژول کند است در عمل بهبود سرعت حس نشود.

اما یک فاکتور بسیار مهم در افراز میزان زمان اجرای یک ماژول نسبت به کل زمان اجرا میباشد . مثلا فرض کنید که ماژول A برای اجرا ۱۰ میلی ثانیه زمان نیاز داشته باشد ولی ۱۰۰۰ بار اجرا شود ولی ماژول B ۲۰ میلی ثانیه زمان نیاز به اجرا داشته باشد ولی ۱ بار اجرا شود . در این صورت پر واضح است که با اینکه ماژول A سریع تر از B اجرا میشود ولی باید ماژول A را به صورت سخت افزاری پیاده سازی کنیم زیرا ۹۹ درصد زمان اجرا را به خود اختصاص داده است . در نتیجه میتوان همه ماژول ها را ابتدا نرم افزاری پیاده کرد و بعد با استفاده از نرم افزار های profiling مشخص نمود که سهم هر زیر برنامه در کل زمان اجرا چقدر است و بعد بیشتر سهم ها را به صورت سخت افزاری پیاده سازی نمود که به این عمل profiling میگویند.

ب) خب پیاده سازی نرم افزاری در اجرا به وضوح کند تر از سخت افزاری می باشد اما یک فاکتور مهم نحوه تبادل اطلاعات بین ماژول هایی ست که سخت افزاری و نرم افزاری پیاده سازی شده اند گاهی اوقات ماژولی که سخت افزاری پیاده سازی شده است به علت محدودیت نرخ انتقال در درگاه های خروجی یا ورودی کند تر از حالتی که نرم افزاری پیاده سازی میشود کارایی دارد . پس این یکی از عوامل مهم در افراز است. هم چنین با استفاده از پروفایلینگ باید ببینیم کدام بلاک ها در زمان اجرا سهم بیشتری دارند و آن ها را پیاده سازی سخت افزاری کنیم.

ج) پردازنده های سخت با سرعت بسیار بالایی کار میکنند و همچنین مساحت بسیار کمتری را بر روی تراشه اشغال میکنند اما اگر طراح به دلایلی تراشه ای را انتخاب کرده که خودش پردازنده سخت دارد دلیلی ندارد که از پردازنده نرم استفاده کند زیرا به هر حال این مساحت اشغال شده است . در ضمن توان مصرفی پردازنده سخت بسیار پایین تر از پردازنده نرم می باشد.

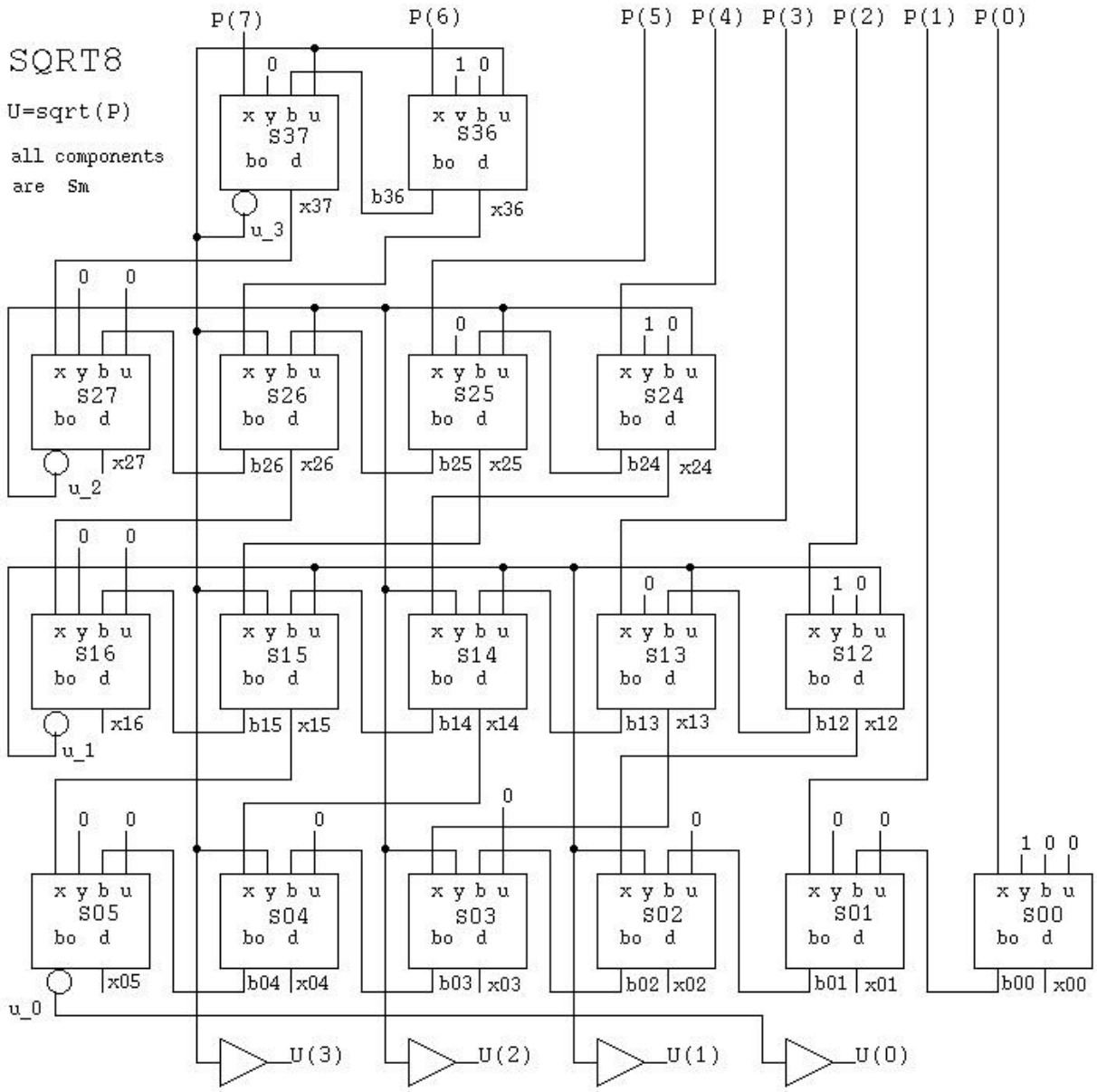
در عوض پردازنده های نرم از انعطاف پذیری بالاتری برخوردارند زیرا طراح میتواند بسته به نیاز آن ها را پیکربندی کند و همچنین میتوانند مثلا مجموعه دستوراتش را گسترش دهد . از طرفی استفاده از پردازنده های نرم باعث میشود که بتوان به راحتی طرح را از روی یک تراشه به تراشه دیگر منتقل نمود.

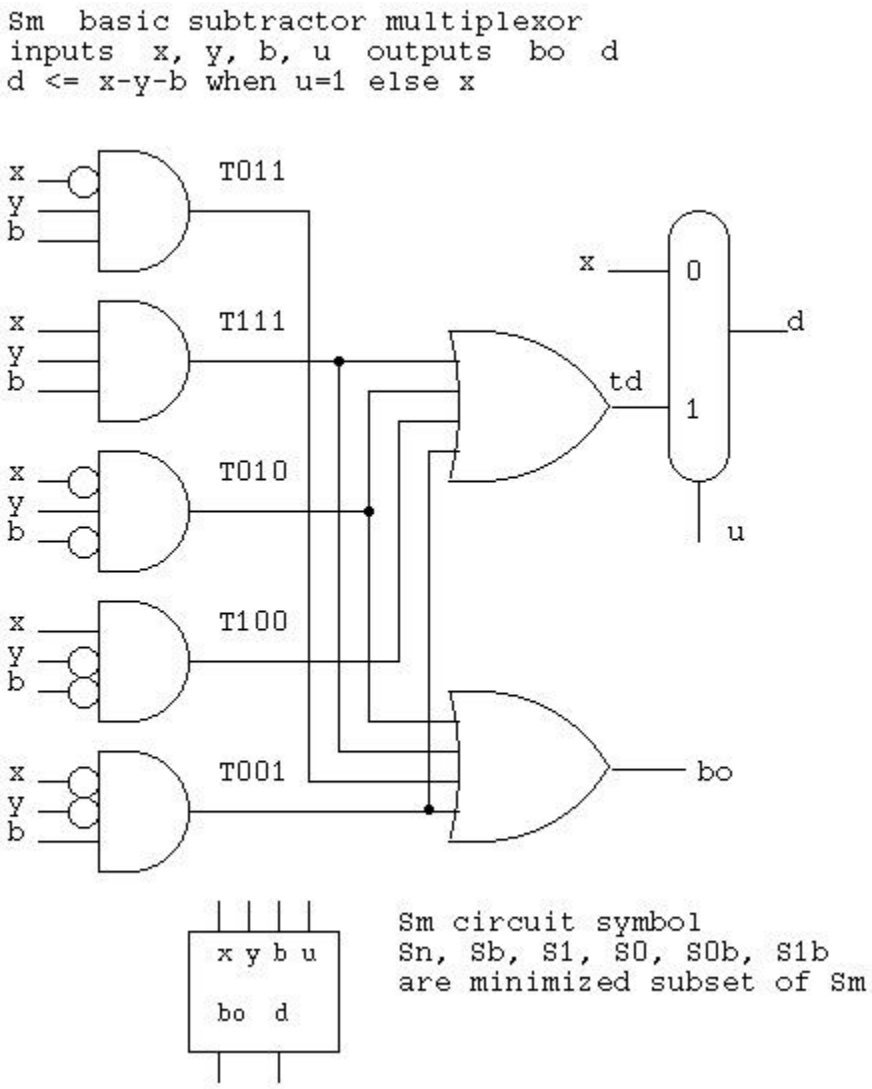
اگر خودمان بخواهیم پردازنده ای توصیف کنیم قطعا بیشترین سر بار آن هزینه زمانی آن است به علاوه اینکه ممکن است از جهت کارایی مناسب نباشد و توان مصرفی حتی بیشتری از پردازنده توصیف شده خود تراشه داشته باشد.

د) دو نوع ارتباط کلی وجود دارد : یکی ارتباط نقطه به نقطه و دیگری ارتباط گذرگاه مشترک . در مقام مقایسه باید گفت که سیستم با گذرگاه مشترک در واقع هزینه کمتری را در بر دارد ولی نقطه به نقطه هزینه بیشتری را در بر میگیرد در عوض در حالت گذرگاه مشترک سرعت ابید با کمترین سرعت هماهنگ باشد ولی در نقطه به نقطه هر دستگاه میتواند با سرعت مربوط به خود کار کند . البته در حالت گذرگاه مشترک در بعضی اوقات که سرعت دستگاه ها با هم فاصله زیاد دارند از دو گذرگاه استفاده میکنند که دستگاه های سریع را بر روی یکی و کند ها را بر روی دیگری سوار میکنند و این دو گذرگاه را از طریق پل به یکدیگر متصل می نمایند.

سوال ۷

شکل مدار استفاده شده به صورت زیر است :





پیاده سازی های آن به صورت زیر می باشد

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
USE ieee.numeric_std.ALL;

entity sqrt is Port
( x : in  STD_LOGIC_VECTOR (4 downto 0);
          y : out  STD_LOGIC_VECTOR (2 downto 0));
end sqrt;
```

```
architecture Memory of sqrt is
component MB is Port (
index : in integer;
sqrt : out integer
);
end component;

signal output : integer;
begin

MB1: MB port map ( to_integer(unsigned(x)), output );
y <= std_logic_vector(to_unsigned(output,3));

end  Memory;

architecture RTL of sqrt is

    component SubMul is port (
x,y,bin,control  : in  std_logic;
sub ,bout : out std_logic
);
end component;

  signal zer : std_logic := '0';
  signal one : std_logic := '1';
  signal x00, x01, x02, x03, x04, x05, u_0 : std_logic;
  signal b00, b01, b02, b03, b04, b05 : std_logic;
  signal x12, x13, x14, x15, x16, u_1 : std_logic;
  signal b12, b13, b14, b15, b16 : std_logic;
  signal x24, x25, x26, x27, u_2 : std_logic;
  signal b24, b25, b26, b27 : std_logic;
  signal x36, x37, u_3 : std_logic;
  signal b36, b37 : std_logic;
begin

  --
  s36: SubMul port map('0', one, zer, u_3, x36, b36);
  s37: SubMul port map('0', zer, b36, u_3, x37, b37);

  s24: SubMul port map(x(4), one, zer, u_2, x24, b24);
  s25: SubMul port map('0', zer, b24, u_2, x25, b25);
```

```
  s26: SubMul port map(x36 , u_3, b25, u_2, x26, b26);
  s27: SubMul port map(x37 , zer, b26, zer, x27, b27);

  s12: SubMul port map(x(2), one, zer, u_1, x12, b12);
  s13: SubMul port map(x(3), zer, b12, u_1, x13, b13);
  s14: SubMul port map(x24 , u_2, b13, u_1, x14, b14);
  s15: SubMul port map(x25 , u_3, b14, u_1, x15, b15);
  s16: SubMul port map(x26 , zer, b15, zer, x16, b16);

  s00: SubMul port map(x(0), one, zer, zer, x00, b00);
  s01: SubMul port map(x(1), zer, b00, zer, x01, b01);
  s02: SubMul port map(x12 , u_1, b01, zer, x02, b02);
  s03: SubMul port map(x13 , u_2, b02, zer, x03, b03);
  s04: SubMul port map(x14 , u_3, b03, zer, x04, b04);
  s05: SubMul port map(x15 , zer, b04, zer, x05, b05);
  process( b05 , b16 , b27 , b37 , u_0 , u_1 , u_2)
  begin
    u_0  <= not b05;
  u_1  <= not b16;
  u_2  <= not b27;
  u_3  <= not b37;
  y(0) <= u_0;
  y(1) <= u_1;
  y(2) <= u_2;
  end process;
end RTL;
```

بخش memory block هم به صورت زیر است :

```
entity MB is Port (
index : in integer range 0 to 31;
sqrt : out integer
);
end MB;

architecture Behavioral of MB is

begin

with index select sqrt <=
    0 when 0 ,
    1 when 1 ,
    1 when 2 ,
```

```
    1 when 3 ,
    2 when 4 ,
    2 when 5 ,
    2 when 6 ,
    2 when 7 ,
    2 when 8 ,
    3 when 9 ,
    3 when 10 ,
    3 when 11 ,
    3 when 12 ,
    3 when 13 ,
    3 when 14 ,
    3 when 15 ,
    4 when 16 ,
    4 when 17 ,
    4 when 18 ,
    4 when 19 ,
    4 when 20 ,
    4 when 21 ,
    4 when 22 ,
    4 when 23 ,
    4 when 24 ,
    5 when 25 ,
    5 when 26 ,
    5 when 27 ,
    5 when 28 ,
    5 when 29 ,
    5 when 30 ,
    5 when 31 ;
end Behavioral;
```

بخش SubMul هم به صورت زیر پیاده سازی شده است :

```
library IEEE;
use IEEE.std_logic_1164.all;

entity SubMul is port (
x,y,bin,control  : in  std_logic;
sub ,bout : out std_logic
);
end SubMul;
```

```
architecture Structural of SubMul is
  signal temp3, temp7, temp2, temp1, temp4, temp5 : std_logic;
begin
  temp1 <= (not x) and (not y) and bin;
  temp2 <= (not x) and y and (not bin);
  temp3 <= (not x) and y and bin;
  temp4 <= x and (not y) and (not bin);
  temp7 <= x and y and bin;
  bout   <= temp1 or temp2 or temp3 or temp7;
  temp5  <= temp1 or temp2 or temp4 or temp7;
  sub    <= temp5 when control='1' else x;
end Structural;
```

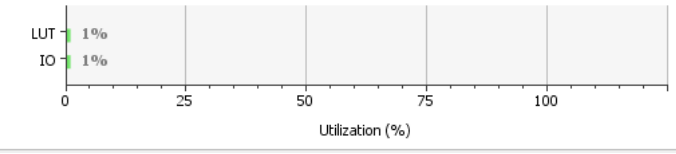
برای مقایسه هر دو طرح را سنتز میکنیم :

urses/University courses/Design Automation/Homeworks/HW5/codes/codes.runs/synth_1/sqrt_utilization_synth.rpt **Read-only**

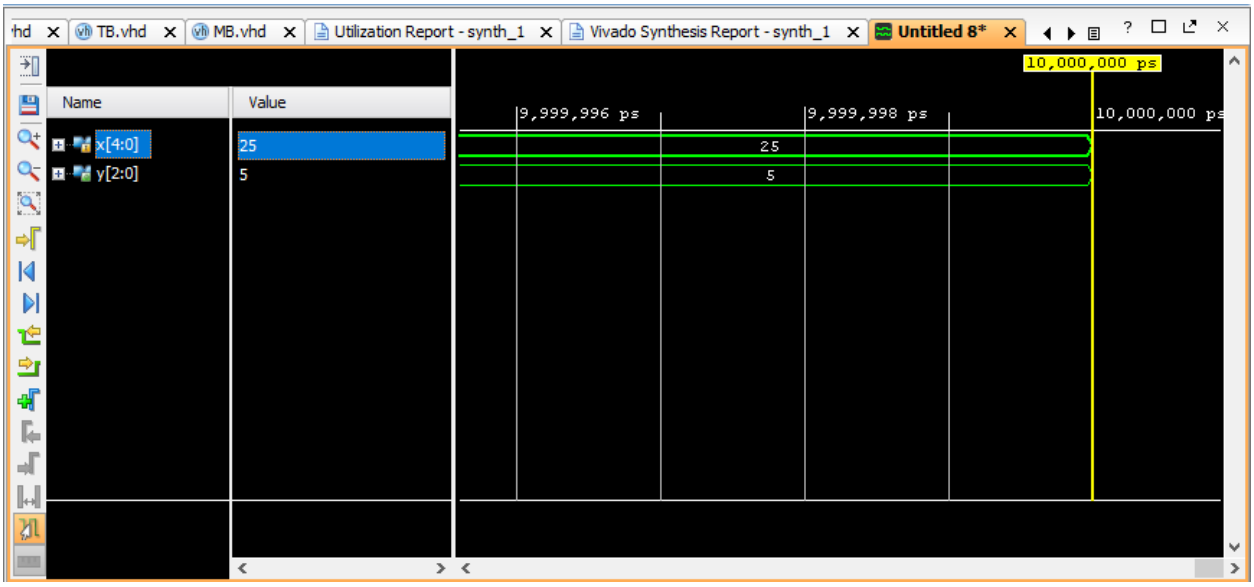
```
-------------

+-------------------------+------+-------+-----------+-------+
|        Site Type        | Used | Fixed | Available | Util% |
+-------------------------+------+-------+-----------+-------+
| Slice LUTs*             |    1 |     0 |    303600 | <0.01 |
|   LUT as Logic          |    1 |     0 |    303600 | <0.01 |
|   LUT as Memory         |    0 |     0 |    130800 |  0.00 |
| Slice Registers         |    0 |     0 |    607200 |  0.00 |
|   Register as Flip Flop |    0 |     0 |    607200 |  0.00 |
|   Register as Latch     |    0 |     0 |    607200 |  0.00 |
| F7 Muxes                |    0 |     0 |    151800 |  0.00 |
| F8 Muxes                |    0 |     0 |     75900 |  0.00 |
+-------------------------+------+-------+-----------+-------+
* Warning! The Final LUT count, after physical optimizations and full implementation, is
```

Summary

Resource	Utilization	Available	Utilization %
LUT	1	303600	0.00
IO	8	600	1.33



برای درستی هم شکل موج های زیر را بررسی میکنیم :



سوال ۸

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

entity controller is Port (
input : in std_logic_vector( 1 downto 0 );
clk,rst : in std_logic;
output : out std_logic );
end controller;

architecture Medvedov of controller is
signal state : std_logic_vector( 2 downto 0 );

begin
process(clk)
begin
if( rst = '1' )then
    state <= "000";
end if;

case state is
when "000" =>
output <= state(0);
if( input = "00" or input = "11")then
    state <= "000";
elsif( input = "10") then
    state <= "100";
elsif( input = "01" ) then
    state <= "010";
else
    state <= "000";
end if;
when "001" =>
output <= state(0);
if( input = "00" or input = "11" )then
    state <= "001";
elsif( input = "01")then
    state <= "011";
elsif( input = "10")then
    state <= "101";
```

```
else
    state <= "001";
end if;
when "010" =>
output <= state(0);
if( input = "00" or input = "11")then
    state <= "000";
elsif( input = "10" )then
    state <= "100";
elsif( input = "01" )then
    state <= "010";
else
    state <= "010";
end if;
when "011" =>
output <= state(0);
if( input = "01" ) then
    state <= "011";
elsif( input = "00" ) then
    state <= "001" ;
elsif( input = "11" ) then
    state <= "000";
elsif( input = "10" ) then
    state <= "101";
else
    state <= "011";
end if;
when "100" =>
output <= state(0);
if( input = "01" ) then
    state <= "011";
elsif( input = "10" ) then
    state <= "100";
elsif( input = "11" ) then
    state <= "001";
elsif( input ="00" ) then
    state <= "000";
else
    state <= "100";
end if;
when "101" =>
output <= state(0);
if( input = "00" or input = "11" ) then
```

```
    state <= "001";
elsif( input = "10" ) then
    state <= "101";
elsif( input = "01" ) then
    state <= "010";
else
    state <= "101";
end if;
when others =>
    state <= "000";
end case ;
end process;

end Medvedov;

architecture Behavioral of controller is
type State_type is ( s0 , s1 , s2 , s3 , s4 , s5 );
signal state : State_type;
attribute fsm_encoding : string;
attribute fsm_encoding of STATE : signal is "sequential";

begin
process(clk)
begin
if( rst = '1' )then
    state <= s0;
end if;

case state is
when s0 =>
output <= '0';
if( input = "00" or input = "11")then
    state <= s0;
elsif( input = "10") then
    state <= s4;
elsif( input = "01" ) then
    state <= s2;
else
    state <= s0;
end if;
when s1 =>
output <= '1';
if( input = "00" or input = "11" )then
```

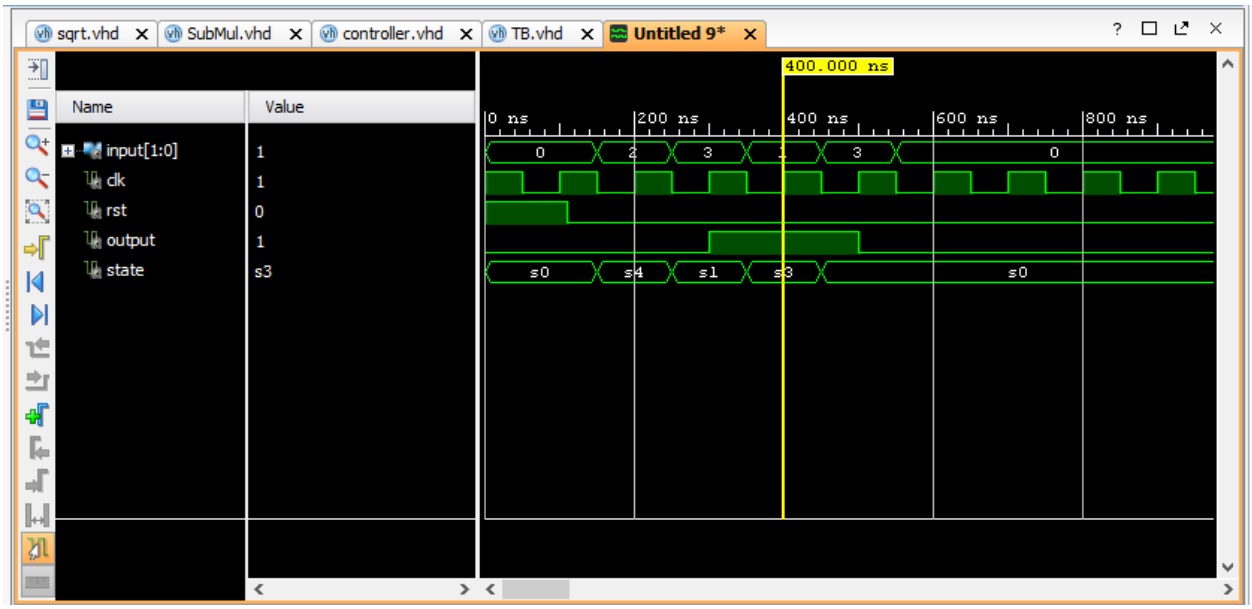
```
    state <= s1;
elsif( input = "01")then
    state <= s3;
elsif( input = "10")then
    state <= s5;
else
    state <= s1;
end if;
when s2 =>
output <= '0';
if( input = "00" or input = "11")then
    state <= s0;
elsif( input = "10" )then
    state <= s4;
elsif( input = "01" )then
    state <= s2;
else
    state <= s2;
end if;
when s3 =>
output <= '1';
if( input = "01" ) then
    state <= s3;
elsif( input = "00" ) then
    state <= s1 ;
elsif( input = "11" ) then
    state <= s0;
elsif( input = "10" ) then
    state <= s5;
else
    state <= s3;
end if;
when s4 =>
output <= '0';
if( input = "01" ) then
    state <= s3;
elsif( input = "10" ) then
    state <= s4;
elsif( input = "11" ) then
    state <= s1;
elsif( input ="00" ) then
    state <= s0;
else
```

```
    state <= s4;
end if;
when s5 =>
output <= '1';
if( input = "00" or input = "11" ) then
    state <= s1;
elsif( input = "10" ) then
    state <= s5;
elsif( input = "01" ) then
    state <= s2;
else
    state <= s5;
end if;
end case ;
end process;

end Behavioral;
```

همانطور که نشان داده شده است میتوان از کد مدودف استفاده کرد به طوریکه باید کد گذاری state ها را یک بیت افزایش دهیم . برای قسمت الف testbench زیر را نوشته ایم:

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

entity TB is
--  Port ( );
end TB;

architecture Controller of TB is
component controller is Port (
input : in std_logic_vector( 1 downto 0 );
clk,rst : in std_logic;
output : out std_logic );
end component;

signal input : std_logic_vector( 1 downto 0);
signal clk,rst,output : std_logic := '1';
begin
rst <= '0' after 110ns;
clk <= not clk after 50ns;

input <= "00",
```

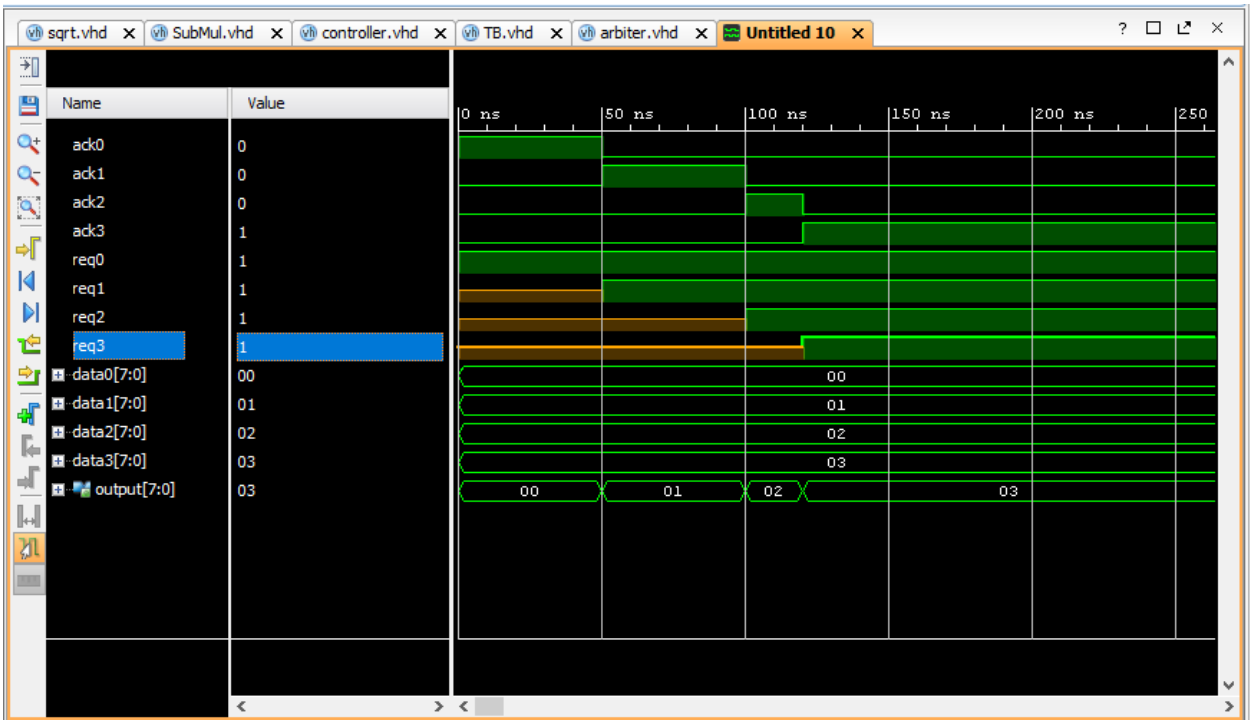
```
        "10" after 150ns,
        "11" after 250ns,
        "01" after 350ns,
        "11" after 450ns,
        "00" after 550ns;

MODULE: controller port map( input , clk ,rst , output);
end Controller;
```



سوال ۹

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

entity arbiter is Port (
ack0,ack1,ack2,ack3 : inout std_logic;
req0,req1,req2,req3: inout std_logic;
data0 : inout std_logic_vector(7 downto 0):="00000000";
data1 : inout std_logic_vector(7 downto 0):="00000001";
data2 : inout std_logic_vector(7 downto 0):="00000010";
data3 : inout std_logic_vector(7 downto 0):="00000011";
output : out std_logic_vector(7 downto 0));
end arbiter;

architecture Behavioral of arbiter is



begin
process(req0,req1,req2,req3)
begin
    if(req3='1') then
        ack0 <= '0';
        ack1 <= '0';
        ack2 <= '0';
        ack3 <= '1';
    elsif(req2='1') then
     ack0 <= '0';
           ack1 <= '0';
           ack2 <= '1';
           ack3 <= '0';
    elsif(req1='1') then
     ack0 <= '0';
           ack1 <= '1';
           ack2 <= '0';
           ack3 <= '0';
    elsif(req0='1') then
    ack0 <= '1';
              ack1 <= '0';
              ack2 <= '0';
```

```
                ack3 <= '0';
    else
        ack0 <= '0';
                ack1 <= '0';
                ack2 <= '0';
                ack3 <= '0';
    end if;
end process;

process(ack0,ack1,ack2,ack3)
begin
    if(ack3='1') then
        output <= data3;
    elsif(ack2='1') then
        output <= data2;
    elsif(ack1='1') then
        output <= data1;
    elsif(ack0='1') then
        output <= data0;
    else
        output <= "XXXXXXXX";
    end if;
end process;
end Behavioral;
```



و حالا طرح را سنتز میکنیم :