HOMEWORK1

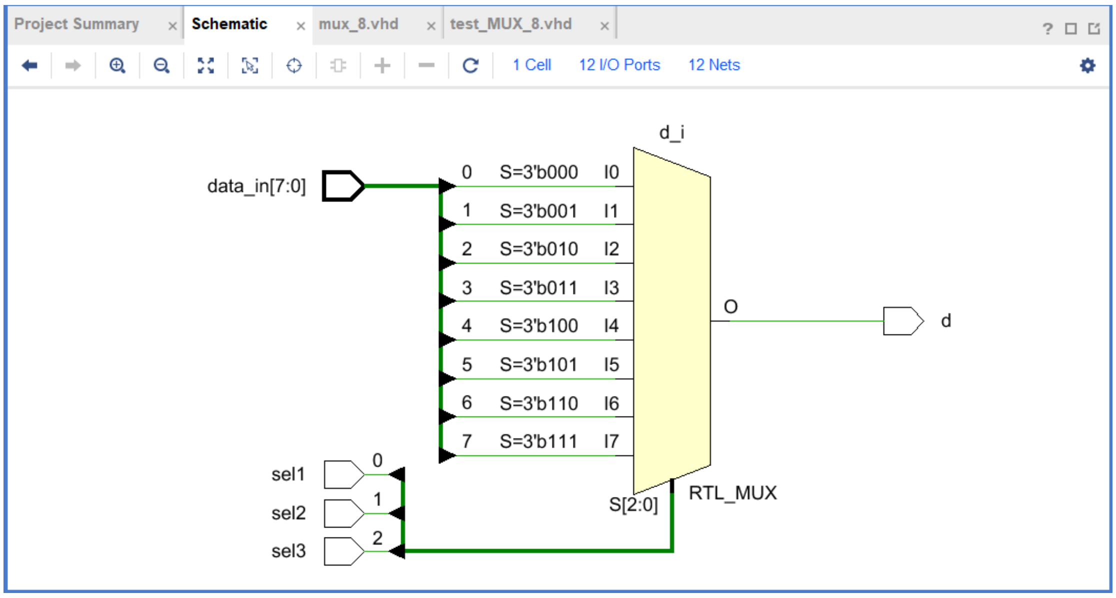
Source code

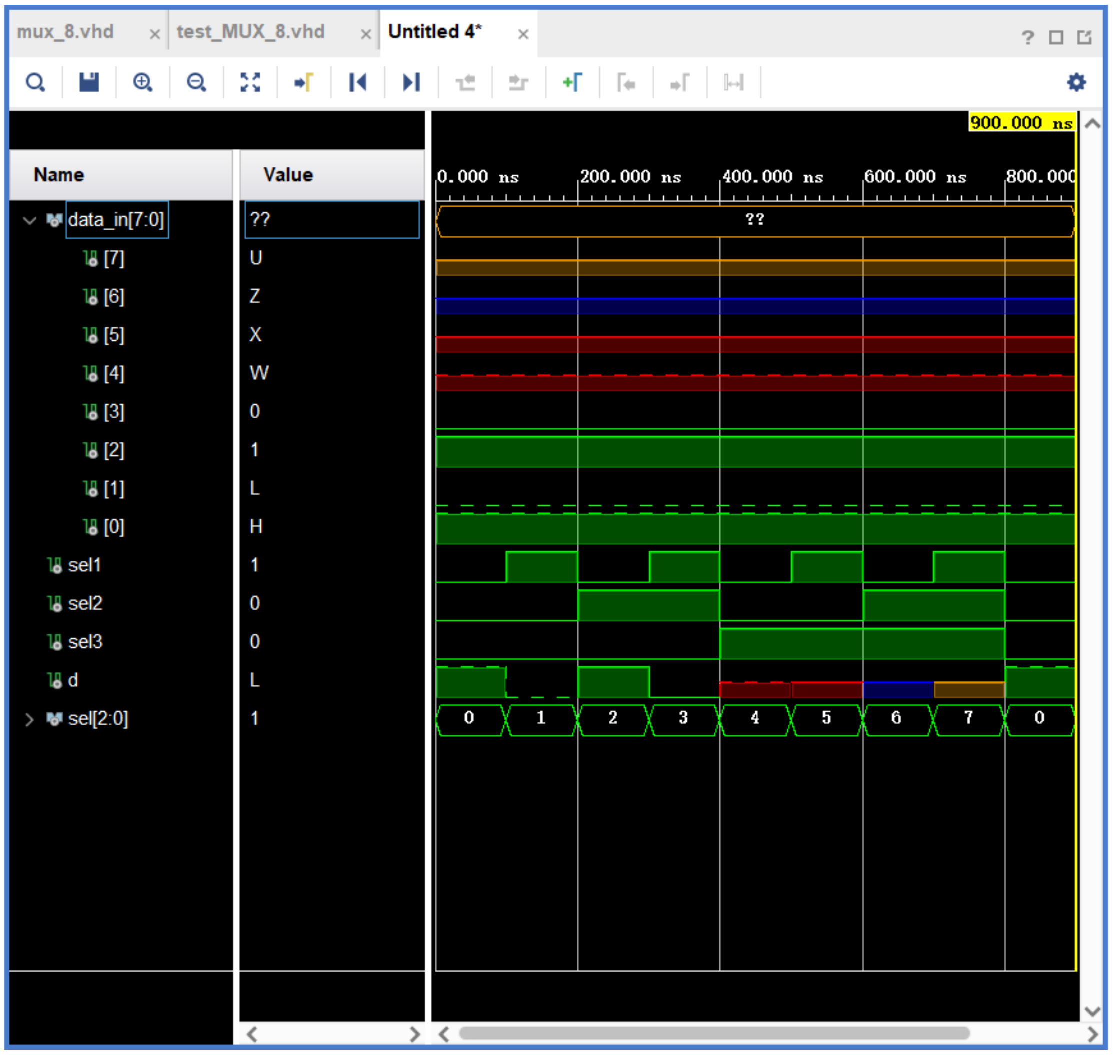
```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity MUX_8 is
    port(data_in:       in STD_LOGIC_VECTOR(7 downto 0);
         sel1,sel2,sel3:in STD_LOGIC;
         d:             out STD_LOGIC
    );
end MUX_8;

architecture Behavioral of MUX_8 is
    signal sel:STD_LOGIC_VECTOR(2 downto 0);
begin
    sel<=sel3 & sel2 & sel1;
    with sel select
        d<=data_in(0) when "000",
           data_in(1) when "001",
           data_in(2) when "010",
           data_in(3) when "011",
           data_in(4) when "100",
           data_in(5) when "101",
           data_in(6) when "110",
           data_in(7) when "111",
           'Z' when others;
end Behavioral;
```

Testbench code

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

entity test_MUX_8 is
end test_MUX_8;

architecture Behavioral of test_MUX_8 is
    component MUX_8 is
        port(data_in:       in STD_LOGIC_VECTOR(7 downto 0);
             sel1,sel2,sel3:in STD_LOGIC;
             d:             out STD_LOGIC
        );
    end component;
    signal data_in:        STD_LOGIC_VECTOR(7 downto 0);
    signal sel1,sel2,sel3: STD_LOGIC;
    signal d:              STD_LOGIC;
begin
    uut:MUX_8
    port map(data_in,sel1,sel2,sel3,d
    );
    data_in<=('U','Z','X','W','0','1','L','H');
    process
        variable sel:STD_LOGIC_VECTOR(2 downto 0);--used for better 
expressing
    begin
        for i in 0 to 7 loop
            sel:=std_logic_vector(to_unsigned(i,3));
            sel1<=sel(0);
            sel2<=sel(1);
            sel3<=sel(2);
            wait for 100 ns;
        end loop;
    end process;
end Behavioral;
```

Schematic & waveform





Name	Value
data_in[7:0]	??
[7]	U
[6]	Z
[5]	X
[4]	W
[3]	0
[2]	1
[1]	L
[0]	H
sel1	1
sel2	0
sel3	0
d	L
sel[2:0]	1

HOMEWORK2

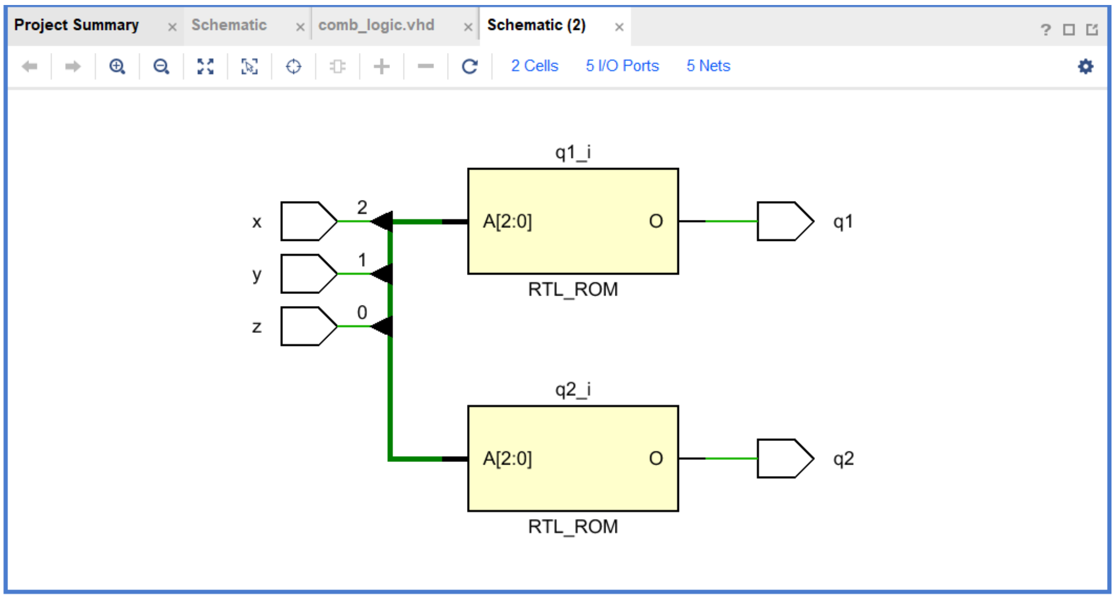
Source code

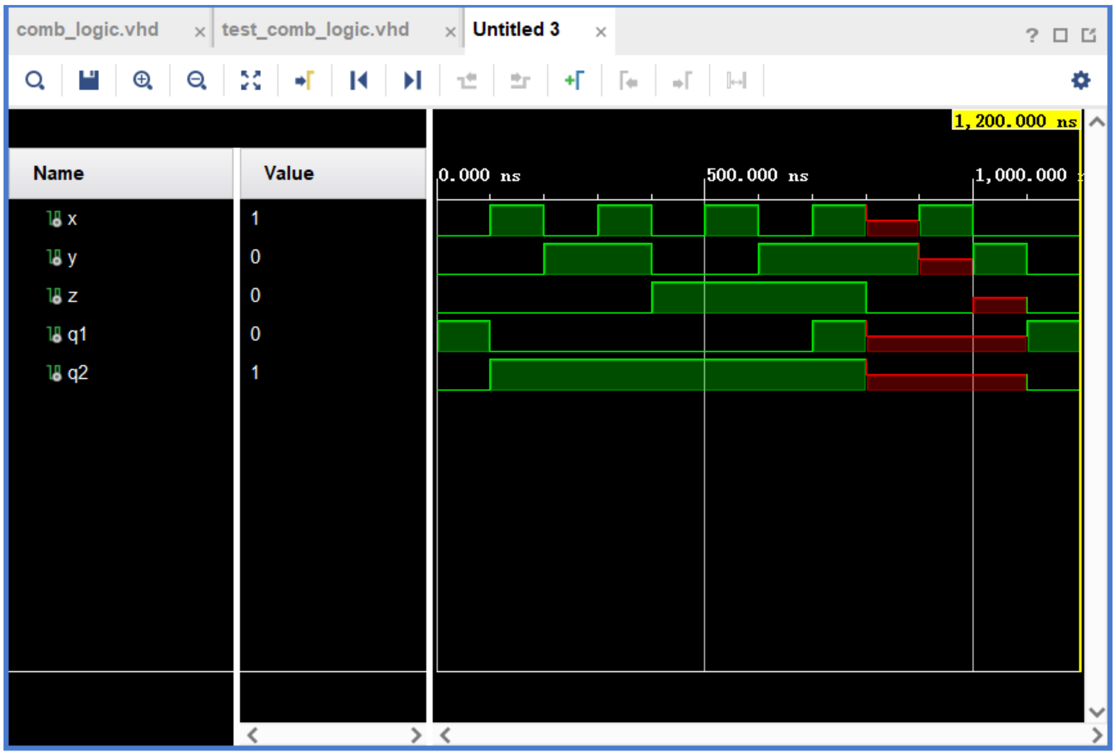
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity comb_logic is
    port(x,y,z:in STD_LOGIC;
         q1,q2:out STD_LOGIC
    );
end comb_logic;

architecture Behavioral of comb_logic is
    signal data_in:STD_LOGIC_VECTOR(2 downto 0);
begin
    data_in<=x & y & z;
    with data_in select
        q1<='1' when "000" | "111",
            '0' when "001" | "010" | "011" | "100" | "101" | "110",
            'X' when others;
    with data_in select
        q2<='1' when "001" | "010" | "011" | "100" | "101" | "110" |
"111",
            '0' when "000",
            'X' when others;
end Behavioral;
```

Testbench code

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

entity test_comb_logic is
end test_comb_logic;

architecture Behavioral of test_comb_logic is
    component comb_logic is
    port(x,y,z:in STD_LOGIC;
         q1,q2:out STD_LOGIC
    );
    end component;
    signal x,y,z:STD_LOGIC;
    signal q1,q2:STD_LOGIC;
begin
    uut:comb_logic
    port map(x,y,z,q1,q2
    );
    process
        variable data_in:STD_LOGIC_VECTOR(2 downto 0);
    begin
        for i in 0 to 7 loop
            data_in:=std_logic_vector(to_unsigned(i,3));
            x<=data_in(0);
            y<=data_in(1);
            z<=data_in(2);
            wait for 100 ns;
        end loop;
        x<='X'; y<='1'; z<='0';
        wait for 100 ns;
        x<='1'; y<='X'; z<='0';
        wait for 100 ns;
        x<='0'; y<='1'; z<='X';
        wait for 100 ns;
    end process;
end Behavioral;
```

Schematic & waveform





Name	Value
x	1
y	0
z	0
q1	0
q2	1

HOMEWORK3

Source code

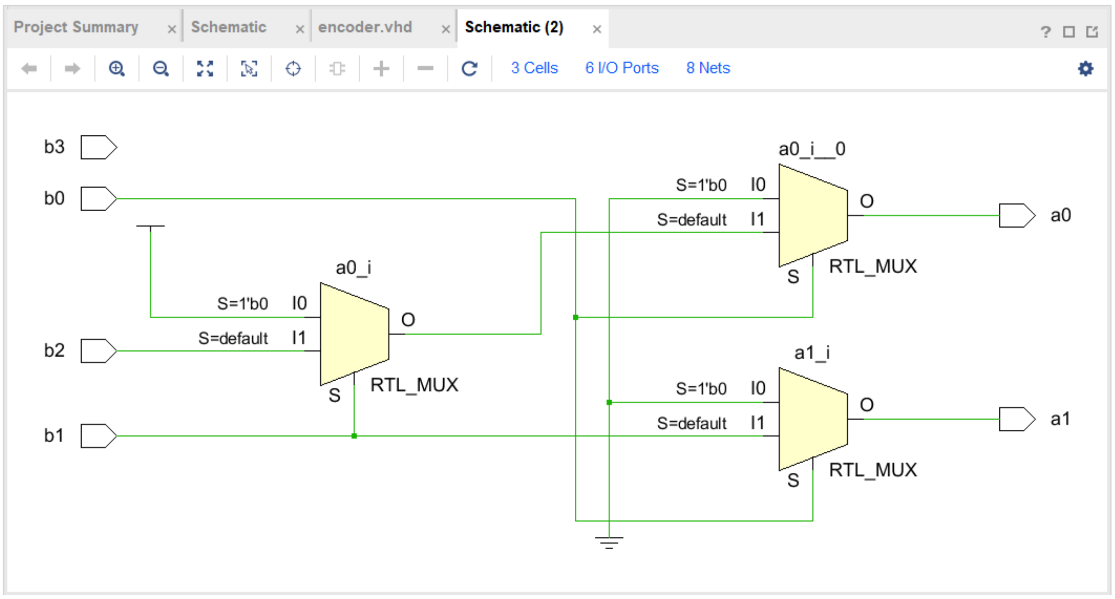
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

--b3~b0:active low
--b0 has the highest priority
--a1~a0:active high
--input can only be '1' or '0'

entity encoder is
    Port(b3,b2,b1,b0:in STD_LOGIC;
         a1,a0:      out STD_LOGIC
    );
end encoder;

architecture Behavioral of encoder is
begin
    process(b3,b2,b1,b0)
    begin
        if b0='0' then
            a1<='0';    a0<='0';
        elsif b1='0' then
            a1<='0';    a0<='1';
        elsif b2='0' then
            a1<='1';    a0<='0';
        else
            a1<='1';    a0<='1';
        end if;
    end process;
--    process(b3,b2,b1,b0)
--        variable data_in:STD_LOGIC_VECTOR(3 downto 0);
--    begin
--        data_in:=b3 & b2 & b1 & b0;
--        case data_in is
--            when
"0000"|"0010"|"0100"|"0110"|"1000"|"1010"|"1100"|"1110" =>
--                a1<='0';    a0<='0';
--            when "0001"|"0101"|"1001"|"1101"=>
--                a1<='0';    a0<='1';
--            when "0011"|"1011"=>
--                a1<='1';    a0<='0';
```

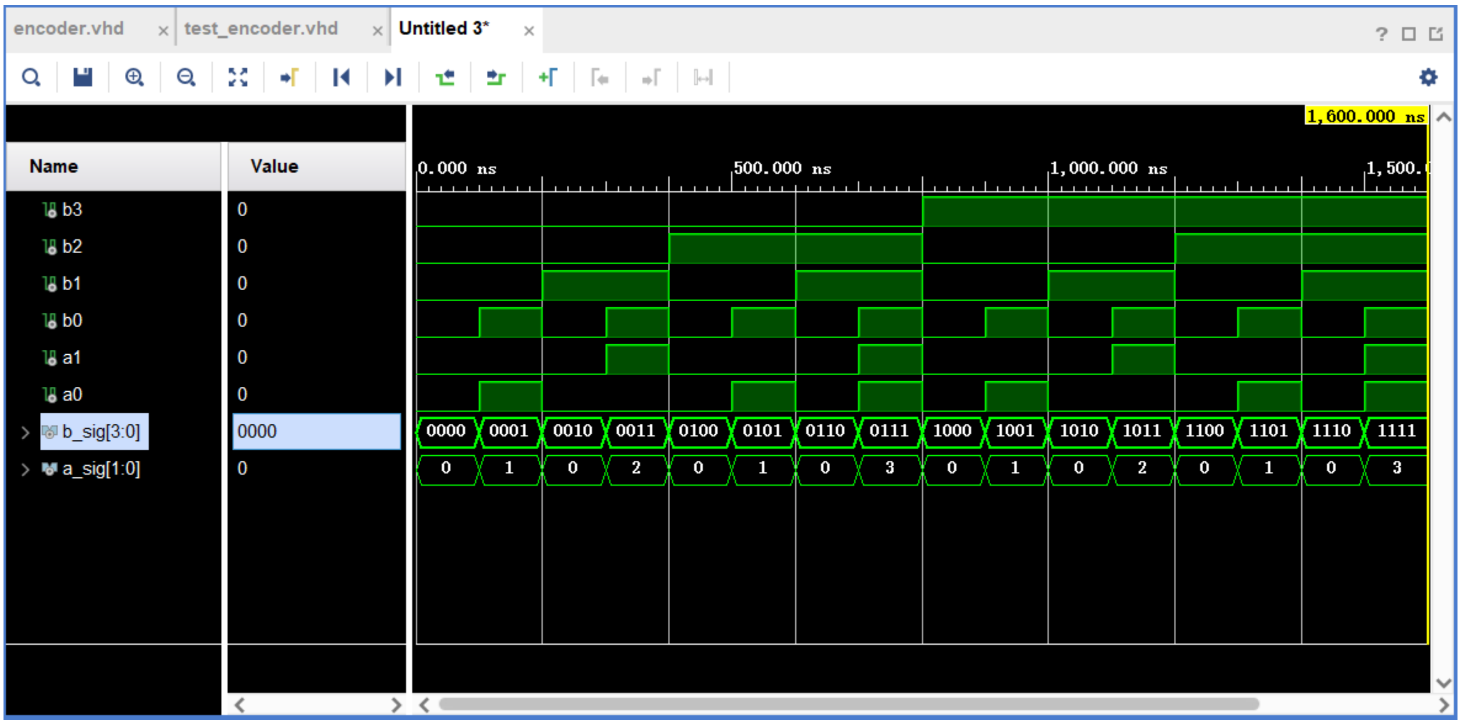
```vhdl
--              when others=>
--                  a1<='1';    a0<='1';
--          end case;
--      end process;
end Behavioral;
```

Testbench code

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

entity test_encoder is
end test_encoder;

architecture Behavioral of test_encoder is
    component encoder is
        Port(b3,b2,b1,b0:in STD_LOGIC;
             a1,a0:      out STD_LOGIC
        );
    end component;
    signal b3,b2,b1,b0:STD_LOGIC;
    signal a1,a0:      STD_LOGIC;

    signal b_sig:STD_LOGIC_VECTOR(3 downto 0);
    signal a_sig:STD_LOGIC_VECTOR(1 downto 0);
begin
    uut:encoder
    port map(b3,b2,b1,b0,a1,a0
    );
    a_sig<=a1 & a0;
    process
        variable b:STD_LOGIC_VECTOR(3 downto 0);
    begin
        for i in 0 to 15 loop
            b:=std_logic_vector(to_unsigned(i,4));
            b_sig<=b;
            b3<=b(3);
            b2<=b(2);
            b1<=b(1);
            b0<=b(0);
            wait for 100 ns;
        end loop;
    end process;
end Behavioral;
```

Schematic & waveform





HOMEWORK4

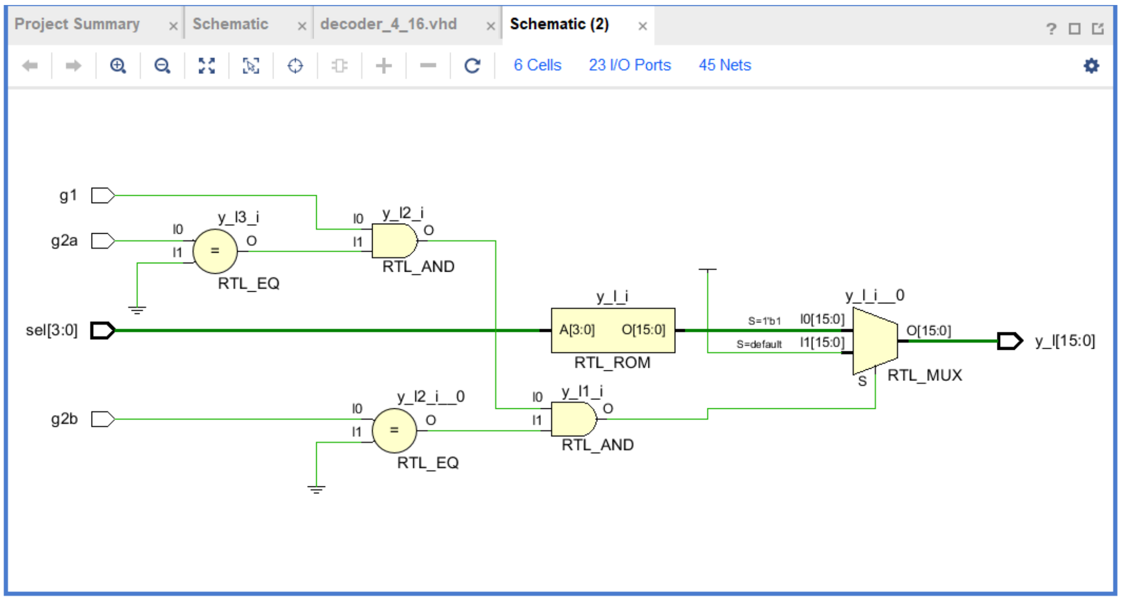
Source code

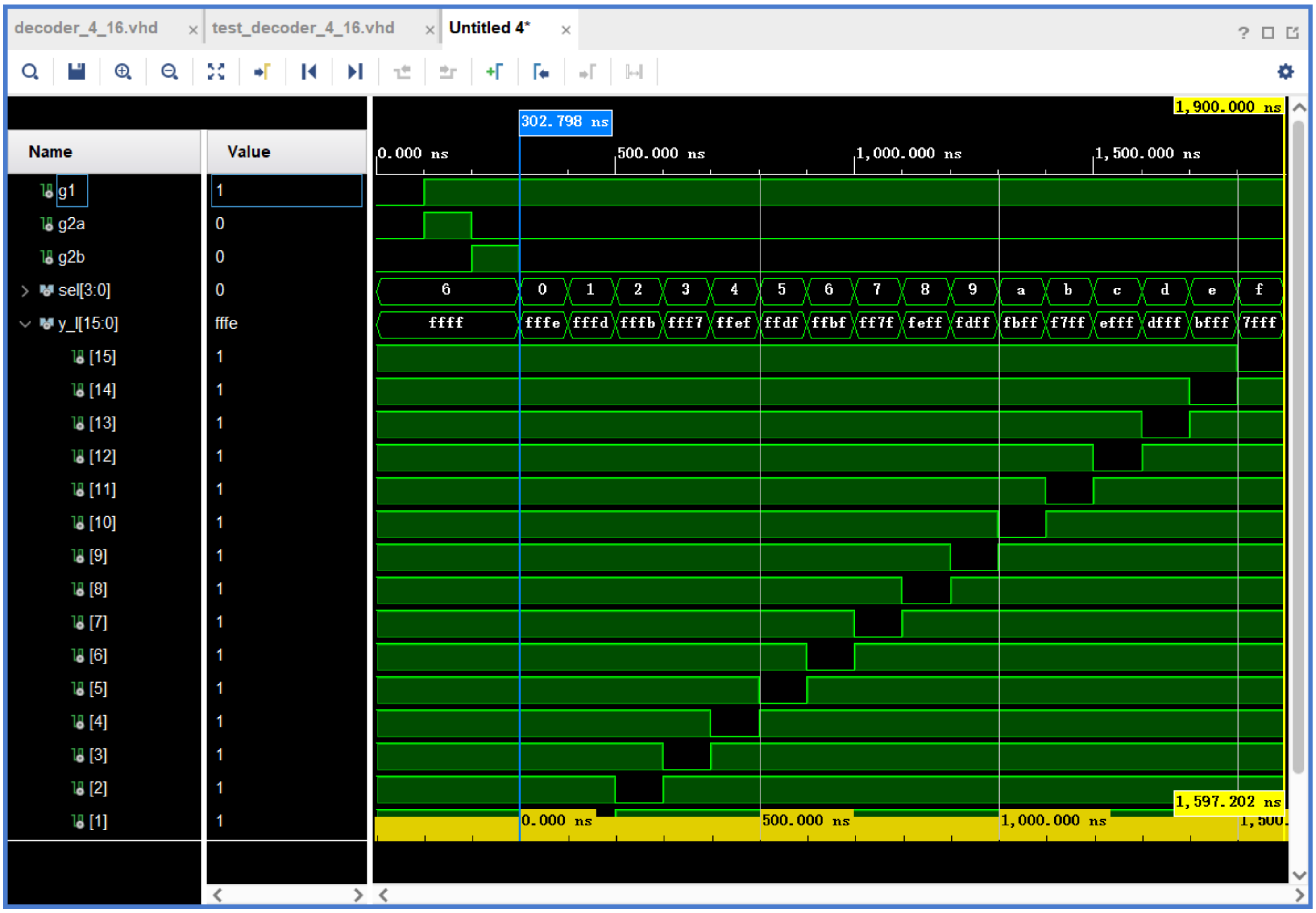
```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

--g1:enable,high active
--g2a,g2b:enable,low active
--sel:select,high active
--y_l:decode,low active

entity decoder_4_16 is
    Port(g1,g2a,g2b: in STD_LOGIC;
         sel :      in STD_LOGIC_VECTOR (3 downto 0);
         y_l :      out STD_LOGIC_VECTOR (15 downto 0)
    );
end decoder_4_16;

architecture Behavioral of decoder_4_16 is

begin
    process(g1,g2a,g2b,sel)
    begin
        if g1='1' and g2a='0' and g2b='0' then
            case sel is
                when "0000"=>   y_l<=(0=>'0',others=>'1');
                when "0001"=>   y_l<=(1=>'0',others=>'1');
                when "0010"=>   y_l<=(2=>'0',others=>'1');
                when "0011"=>   y_l<=(3=>'0',others=>'1');
                when "0100"=>   y_l<=(4=>'0',others=>'1');
                when "0101"=>   y_l<=(5=>'0',others=>'1');
                when "0110"=>   y_l<=(6=>'0',others=>'1');
                when "0111"=>   y_l<=(7=>'0',others=>'1');
                when "1000"=>   y_l<=(8=>'0',others=>'1');
                when "1001"=>   y_l<=(9=>'0',others=>'1');
                when "1010"=>   y_l<=(10=>'0',others=>'1');
                when "1011"=>   y_l<=(11=>'0',others=>'1');
                when "1100"=>   y_l<=(12=>'0',others=>'1');
                when "1101"=>   y_l<=(13=>'0',others=>'1');
                when "1110"=>   y_l<=(14=>'0',others=>'1');
                when "1111"=>   y_l<=(15=>'0',others=>'1');
                when others =>   y_l<=(others=>'1');
            end case;
        else
            y_l<=(others=>'1');
        end if;
    end process;
end Behavioral;
```

Testbench code

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

entity test_decoder_4_16 is
end test_decoder_4_16;

architecture Behavioral of test_decoder_4_16 is
    component decoder_4_16 is
        Port(g1,g2a,g2b: in STD_LOGIC;
             sel :      in STD_LOGIC_VECTOR (3 downto 0);
             y_l :      out STD_LOGIC_VECTOR (15 downto 0)
        );
    end component;
    signal g1,g2a,g2b:STD_LOGIC;
    signal sel :      STD_LOGIC_VECTOR (3 downto 0);
    signal y_l :      STD_LOGIC_VECTOR (15 downto 0);
begin
    uut:decoder_4_16
    port map(g1,g2a,g2b,sel,y_l
    );
    process
        variable en_test:boolean:=false;
    begin
        --test for enable port
        if en_test=false then
            sel<=(2 downto 1=>'1',others=>'0');
            g1<='0'; g2a<='0';  g2b<='0';
            wait for 100 ns;
            g1<='1'; g2a<='1';  g2b<='0';
            wait for 100 ns;
            g1<='1'; g2a<='0';  g2b<='1';
            wait for 100 ns;
            en_test:=true;
        else
            --test for select port
            g1<='1'; g2a<='0';  g2b<='0';
            for i in 0 to 15 loop
                sel<=std_logic_vector(to_unsigned(i,4));
                wait for 100 ns;
            end loop;
        end if;
    end process;
end Behavioral;
```

Schematic & waveform





Name	Value
g1	1
g2a	0
g2b	0
sel[3:0]	0
y_I[15:0]	fffe
[15]	1
[14]	1
[13]	1
[12]	1
[11]	1
[10]	1
[9]	1
[8]	1
[7]	1
[6]	1
[5]	1
[4]	1
[3]	1
[2]	1
[1]	1

HOMEWORK5

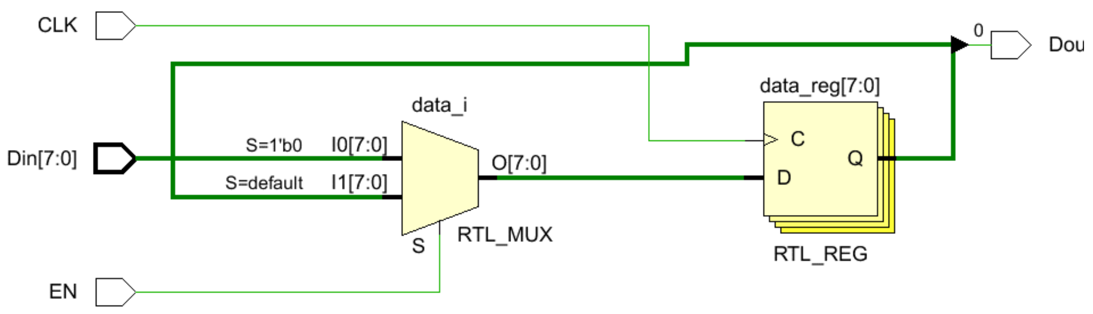
Source code

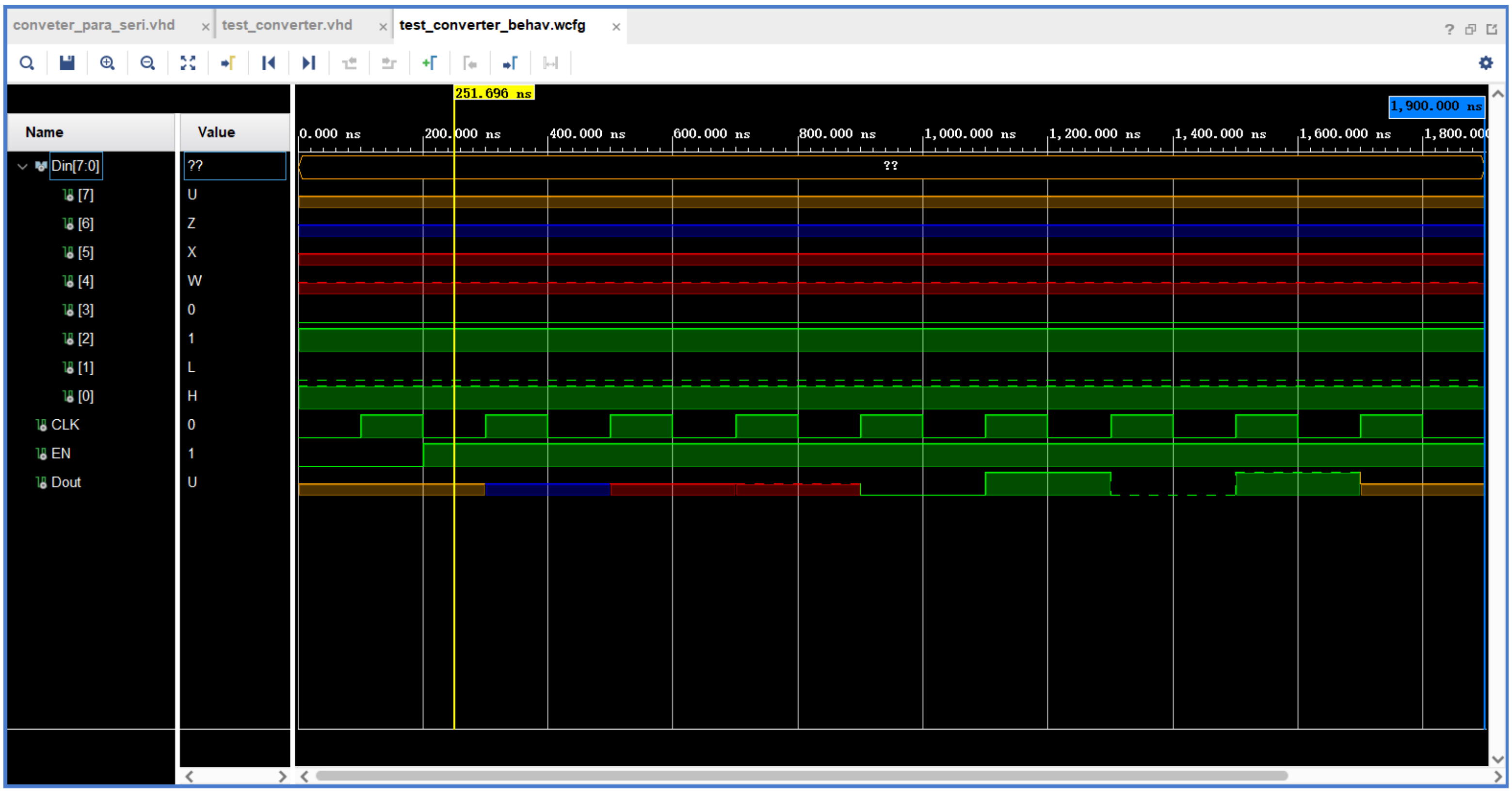
```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity conveter_para_seri is
    Port(Din : in STD_LOGIC_VECTOR (7 downto 0);
         CLK,EN : in STD_LOGIC;
         Dout : out STD_LOGIC
    );
end conveter_para_seri;

architecture Behavioral of conveter_para_seri is
    signal data:STD_LOGIC_VECTOR(7 downto 0);
begin
    Dout<=data(7);
    process(CLK)
    begin
        if CLK'event and CLK='1' then
            if EN='0' then --parallel loading
                data<=Din;
            else --shift to serial out
                data<=data(6 downto 0) & data(7);--rotate left
            end if;
        end if;
    end process;
end Behavioral;
```

Testbench code

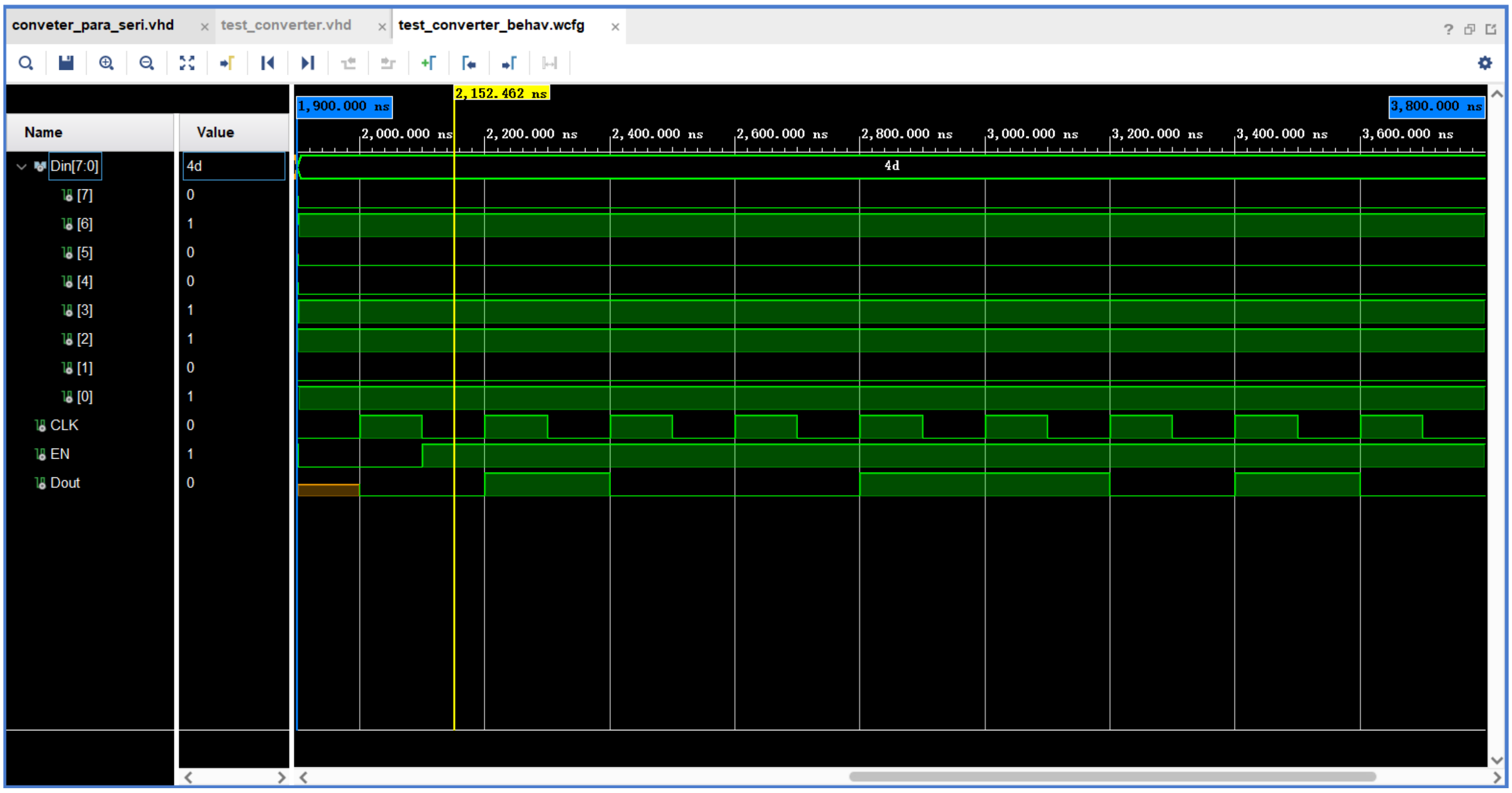
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity test_converter is
end test_converter;

architecture Behavioral of test_converter is
    component conveter_para_seri is
        Port(Din : in STD_LOGIC_VECTOR (7 downto 0);
             CLK,EN : in STD_LOGIC;
             Dout : out STD_LOGIC
        );
    end component;
    signal Din:   STD_LOGIC_VECTOR (7 downto 0);
    signal CLK,EN:STD_LOGIC;
    signal Dout:  STD_LOGIC;
begin
    uut:conveter_para_seri
    port map(Din,CLK,EN,Dout
    );
    process
    begin
        for1:for m in 0 to 1 loop
            --EN='0'--parallel loading
            CLK<='0';
            EN<='0';
            --choose 2 inputs
            if m=0 then
                Din<="UZXW01LH"; --input 1:"UZXW01LH"
            else
                Din<="01001101"; --input 2:"01001101"
            end if;
            wait for 100 ns;
            CLK<='1';
            wait for 100 ns;
            CLK<='0';
            --EN='1'--convert parallel to serial
            EN<='1';
            wait for 100 ns;
            for2:for i in 0 to 7 loop
                CLK<='1';
                wait for 100 ns;
                CLK<='0';
                wait for 100 ns;
            end loop for2;
        end loop for1;
    end process;
end Behavioral;
```

Schematic & waveform





Name	Value
Din[7:0]	??
[7]	U
[6]	Z
[5]	X
[4]	W
[3]	0
[2]	1
[1]	L
[0]	H
CLK	0
EN	1
Dout	U



Name	Value
Din[7:0]	4d
[7]	0
[6]	1
[5]	0
[4]	0
[3]	1
[2]	1
[1]	0
[0]	1
CLK	0
EN	1
Dout	0

HOMEWORK6

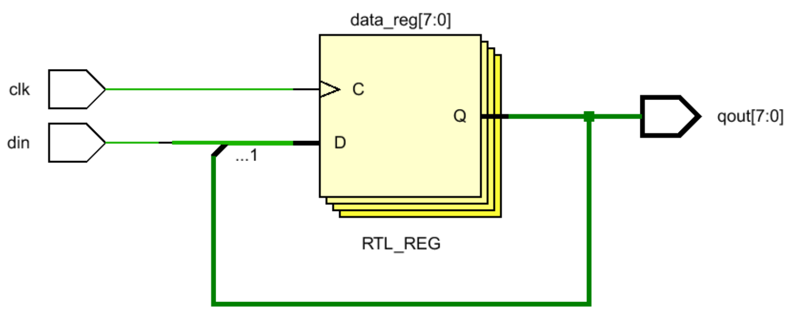
Source code

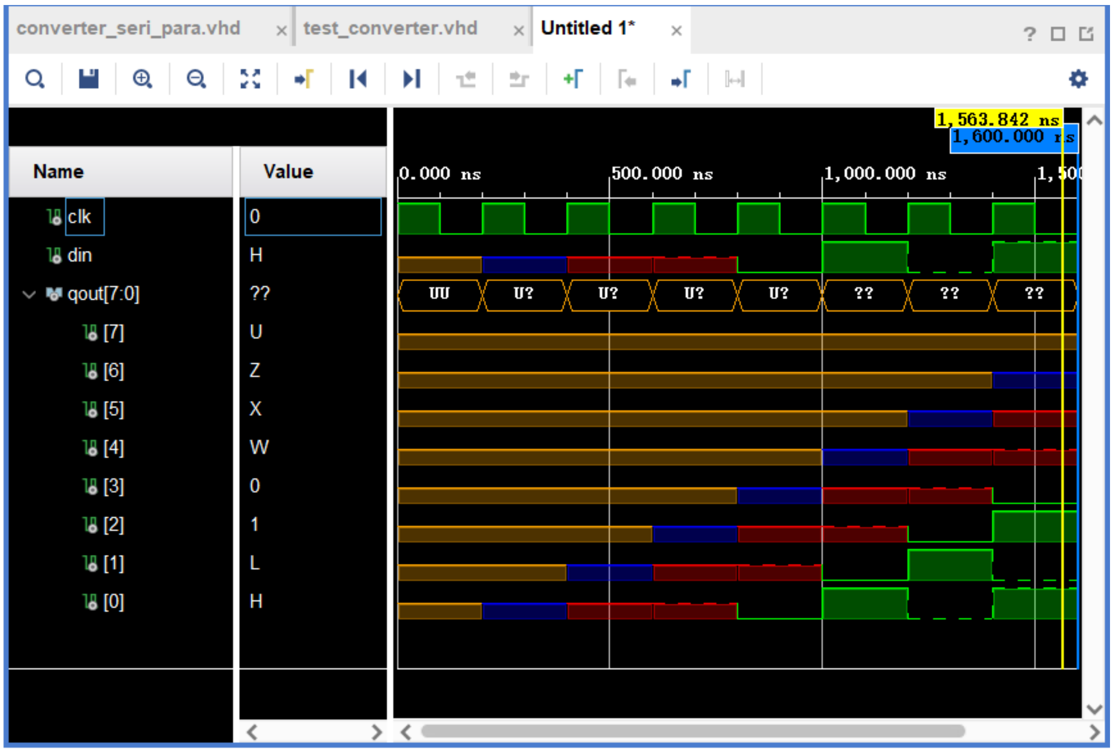
```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity converter_seri_para is
    Port(clk,din: in STD_LOGIC;
         qout:    out STD_LOGIC_VECTOR (7 downto 0));
end converter_seri_para;

architecture Behavioral of converter_seri_para is
    signal data:STD_LOGIC_VECTOR(7 downto 0);
begin
    qout<=data;
    process(clk)
    begin
        if clk'event and clk='1' then
            data<=data(6 downto 0) & din;
        end if;
    end process;
end Behavioral;
```

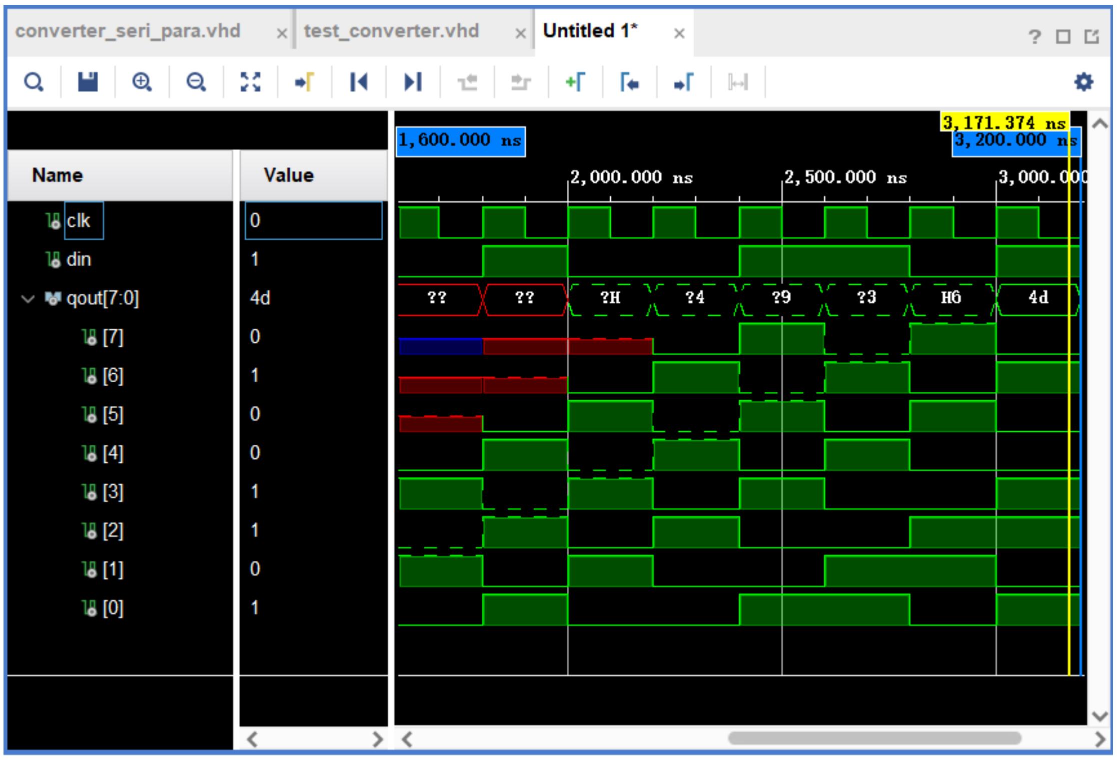
Testbench code

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

entity test_converter is
end test_converter;

architecture Behavioral of test_converter is
    component converter_seri_para is
        Port(clk,din: in STD_LOGIC;
             qout:    out STD_LOGIC_VECTOR (7 downto 0));
    end component;
    signal clk,din:STD_LOGIC;
    signal qout:STD_LOGIC_VECTOR(7 downto 0);
begin
    uut:converter_seri_para
    port map(clk,din,qout
    );
    process
        type din_type1 is array (0 to 1) of STD_LOGIC_VECTOR(0 to 7);
        constant din_cons1:din_type1:=("UZXW01LH","01001101");
--        type din_type2 is array (0 to 1,0 to 7) of STD_LOGIC;
--        constant din_cons2:din_type2:=("UZXW01LH","01001101");
    begin
        clk<='0';
        din<='0';
        for i in 0 to 1 loop
            for j in 0 to 7 loop
                din<=din_cons1(i)(j);
--                din<=din_cons2(i,j);
                clk<='1';    wait for 100 ns;
                clk<='0';    wait for 100 ns;
            end loop;
        end loop;
    end process;
end Behavioral;
```

Schematic & waveform







HOMEWORK7

Source code

Adder_4bit.vhd

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

entity adder_4bit is
    Port ( x : in STD_LOGIC_VECTOR (3 downto 0);
           y : in STD_LOGIC_VECTOR (3 downto 0);
           cin : in STD_LOGIC;
           sum : out STD_LOGIC_VECTOR (3 downto 0);
           cout : out STD_LOGIC);
end adder_4bit;

architecture Behavioral of adder_4bit is

begin
    process(x,y,cin)
        variable sum_tmp:STD_LOGIC_VECTOR(4 downto 0); --5 bit
    begin
        sum_tmp:=('0' & x)+('0' & y)+("0000" & cin);
--        sum_tmp:=x+y+cin; --wrong,width dismatch!
        sum<=sum_tmp(3 downto 0);
        cout<=sum_tmp(4);
    end process;
end Behavioral;
```

Source code

Adder_8bit.vhd

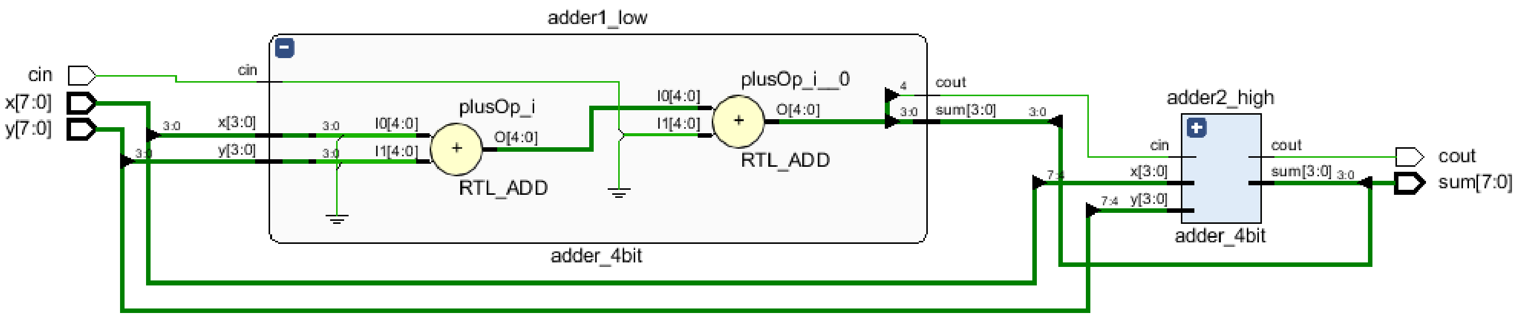
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity adder_8bit is
    Port ( x : in STD_LOGIC_VECTOR (7 downto 0);
           y : in STD_LOGIC_VECTOR (7 downto 0);
           cin : in STD_LOGIC;
           sum : out STD_LOGIC_VECTOR (7 downto 0);
           cout : out STD_LOGIC);
end adder_8bit;

architecture Behavioral of adder_8bit is
    component adder_4bit is
        Port ( x : in STD_LOGIC_VECTOR (3 downto 0);
               y : in STD_LOGIC_VECTOR (3 downto 0);
               cin : in STD_LOGIC;
               sum : out STD_LOGIC_VECTOR (3 downto 0);
               cout : out STD_LOGIC);
    end component;
    signal carry:STD_LOGIC;
begin
    adder1_low:adder_4bit
    port map(
        x=>x(3 downto 0),
        y=>y(3 downto 0),
        cin=>cin,
        sum=>sum(3 downto 0),
        cout=>carry
    );
    adder2_high:adder_4bit
    port map(
        x=>x(7 downto 4),
        y=>y(7 downto 4),
        cin=>carry,
        sum=>sum(7 downto 4),
        cout=>cout
    );
end Behavioral;
```

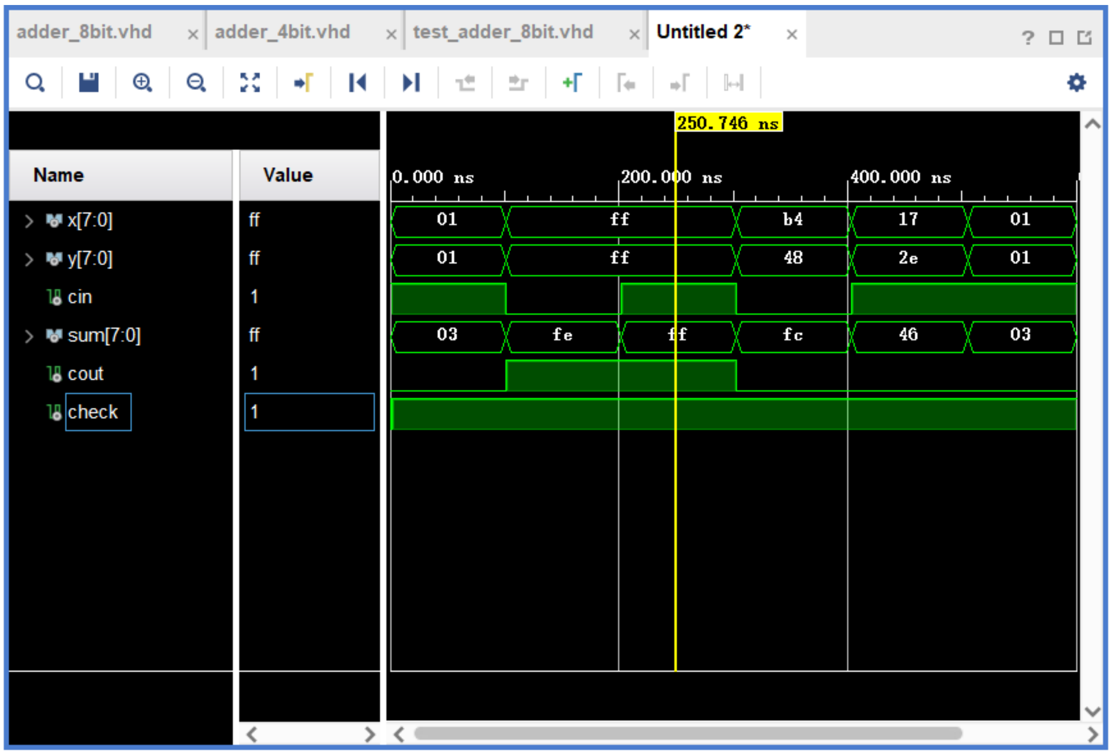
Testbench code

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

entity test_adder_8bit is
end test_adder_8bit;

architecture Behavioral of test_adder_8bit is
    component adder_8bit is
        Port ( x : in STD_LOGIC_VECTOR (7 downto 0);
               y : in STD_LOGIC_VECTOR (7 downto 0);
               cin : in STD_LOGIC;
               sum : out STD_LOGIC_VECTOR (7 downto 0);
               cout : out STD_LOGIC);
    end component;
    signal x,y,sum : STD_LOGIC_VECTOR (7 downto 0);
    signal cin,cout : STD_LOGIC;

    signal check:STD_LOGIC;
begin
    uut:adder_8bit
    port map(x,y,cin,sum,cout
    );
    process
        variable check1,check2:STD_LOGIC_VECTOR(8 downto 0);
        type add_type is array(0 to 4) of STD_LOGIC_VECTOR(7 downto 0);
        type carry_type is array(0 to 4) of STD_LOGIC;
        constant
x_cons:add_type:=("00000001","11111111","11111111","10110100","00010111
");
        constant
y_cons:add_type:=("00000001","11111111","11111111","01001000","00101110
");
        constant cin_cons:carry_type:=('1','0','1','0','1');
    begin
        for i in 0 to 4 loop
            x<=x_cons(i);   y<=y_cons(i);   cin<=cin_cons(i);
            wait for 1 ns;
            check1:=('0' & x)+('0' & y)+("00000000" & cin);
            check2:=cout & sum;
            if check1=check2 then
```

```vhdl
                check<='1';
            else
                check<='0';
            end if;
            wait for 100 ns;
        end loop;
    end process;
end Behavioral;
```

Schematic & waveform





Name	Value							
x[7:0]	ff	01	ff		b4	17	01
y[7:0]	ff	01	ff		48	2e	01
cin	1						
sum[7:0]	ff	03	fe	ff	fc	46	03
cout	1						
check	1						

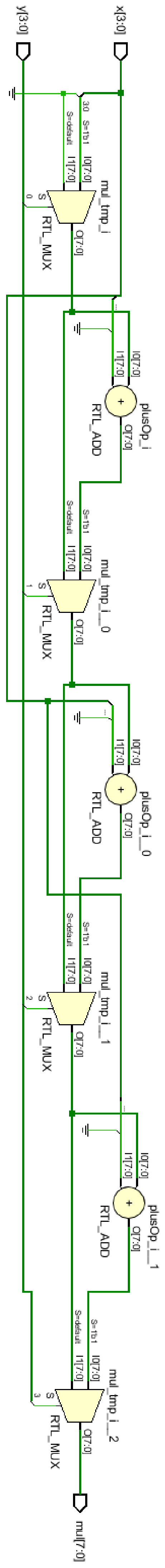
HOMEWORK8

Source code

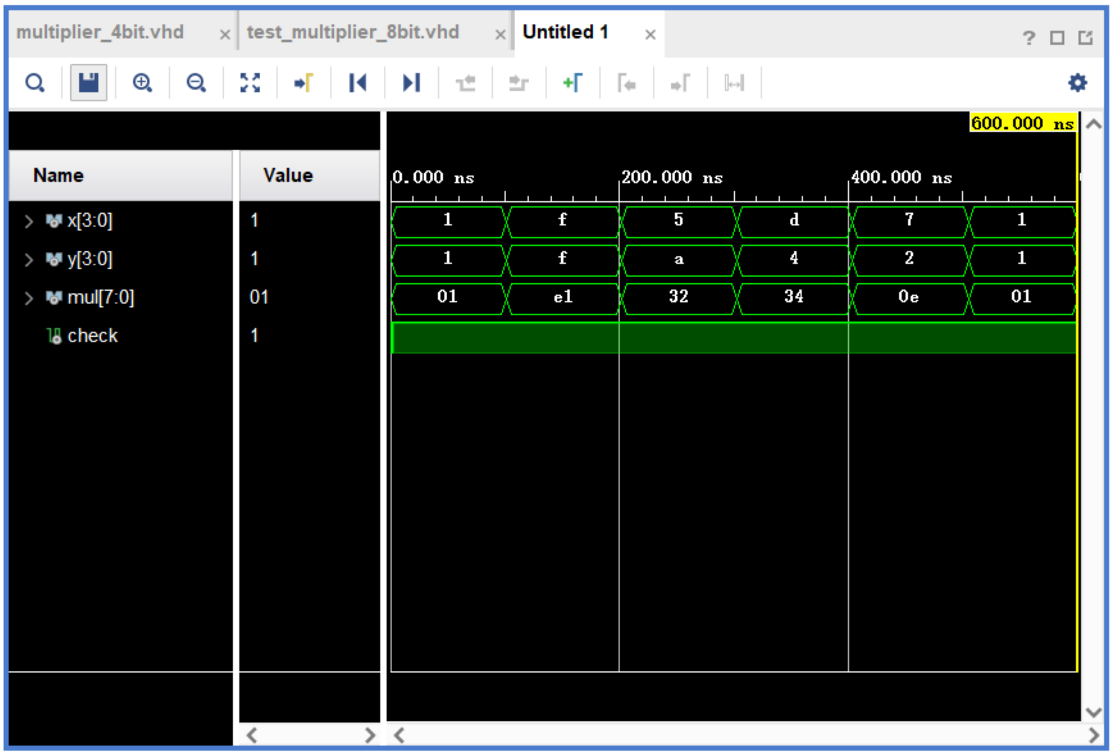
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity multiplier_4bit is
    Port ( x : in STD_LOGIC_VECTOR (3 downto 0);
           y : in STD_LOGIC_VECTOR (3 downto 0);
           mul : out STD_LOGIC_VECTOR (7 downto 0));
end multiplier_4bit;

architecture Behavioral of multiplier_4bit is

begin
    process(x,y)
        variable sum_tmp:STD_LOGIC_VECTOR(6 downto 0); --7 bit
        variable mul_tmp:STD_LOGIC_VECTOR(7 downto 0); --8 bit
        variable i_count:integer;
    begin

        mul_tmp:=(others=>'0');
        sum_tmp:="000" & x;
        for i in 0 to 3 loop --multiply x*y,check every bit of y to
summation
            if y(i)='1' then
                mul_tmp:=mul_tmp+('0' & sum_tmp);
            end if;
            sum_tmp:=sum_tmp(5 downto 0) & '0';
        end loop;
        mul<=mul_tmp;
    end process;
end Behavioral;
```

Testbench code

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

entity test_multiplier_8bit is
end test_multiplier_8bit;

architecture Behavioral of test_multiplier_8bit is
    component multiplier_4bit is
        Port ( x : in STD_LOGIC_VECTOR (3 downto 0);
               y : in STD_LOGIC_VECTOR (3 downto 0);
               mul : out STD_LOGIC_VECTOR (7 downto 0));
    end component;
    signal x,y:STD_LOGIC_VECTOR(3 downto 0);
    signal mul:STD_LOGIC_VECTOR(7 downto 0);
    signal check:STD_LOGIC;
begin
    uut:multiplier_4bit
    port map(x,y,mul
    );
    process
        type mul_type is array(0 to 4) of STD_LOGIC_VECTOR(3 downto 0);
        constant x_cons:mul_type:=("0001","1111","0101","1101","0111");
        constant y_cons:mul_type:=("0001","1111","1010","0100","0010");
    begin
        for i in 0 to 4 loop
            x<=x_cons(i);   y<=y_cons(i);
            wait for 1 ns;
            if x*y=mul then
                check<='1';
            else
                check<='0';
            end if;
            wait for 100 ns;
        end loop;
    end process;
end Behavioral;
```



Schematic & waveform



multiplier_4bit.vhd
test_multiplier_8bit.vhd
Untitled 1
600.000 ns
Name
Value
0.000 ns
200.000 ns
400.000 ns
x[3:0] 1 | 1 f 5 d 7 1
y[3:0] 1 | 1 f a 4 2 1
mul[7:0] 01 | 01 e1 32 34 0e 01
check 1