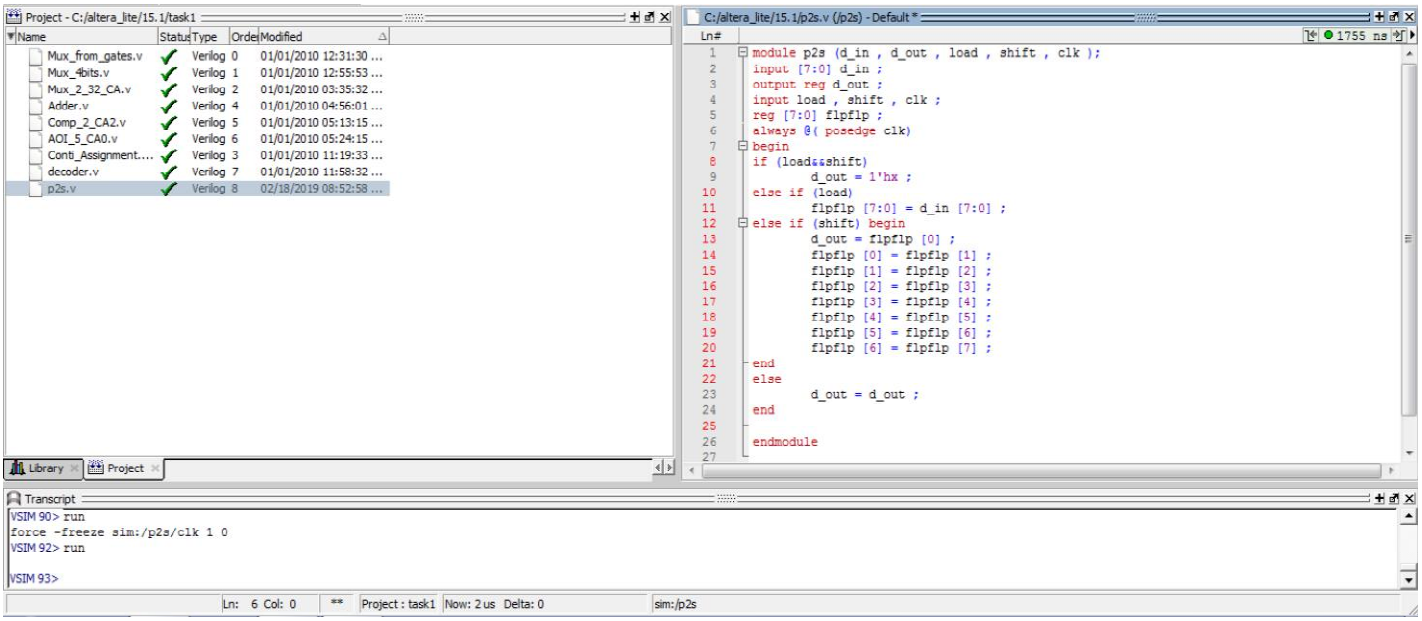
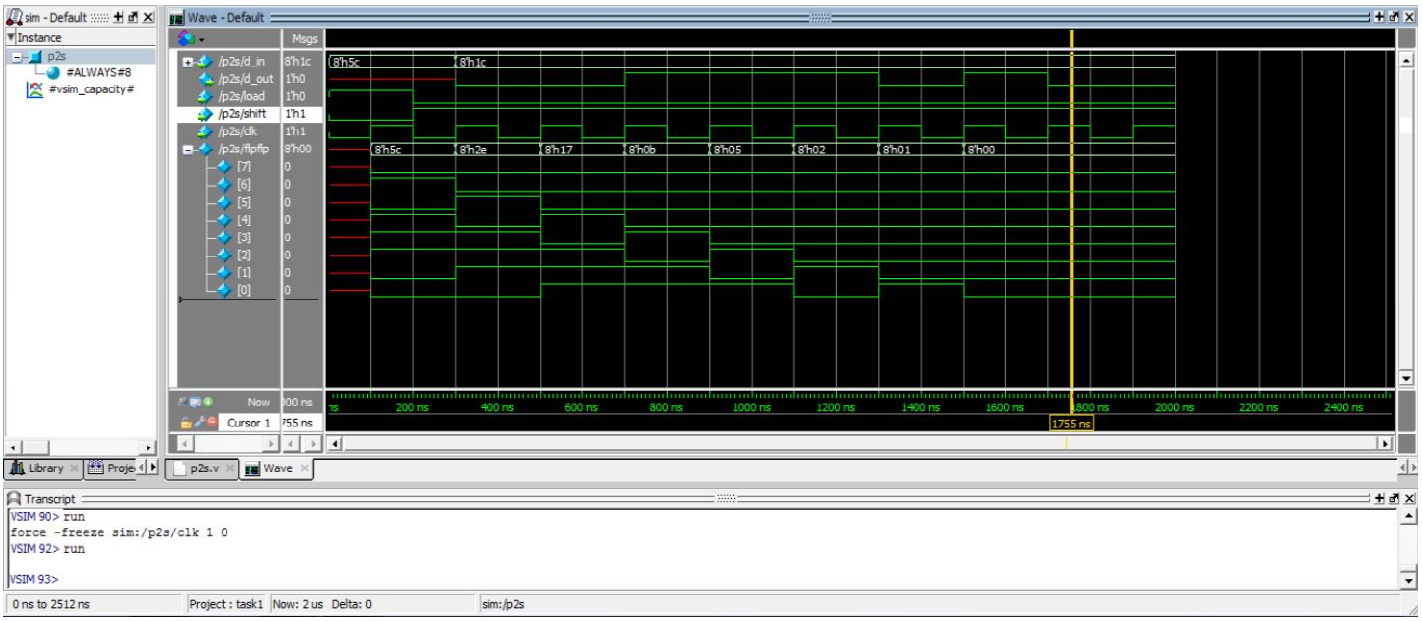
Converter


Project - C:/altera_lite/15.1/task1
Name
Status
Type
Order
Modified
Mux_from_gates.v
Mux_4bits.v
Mux_2_32_CA.v
Adder.v
Comp_2_CA2.v
AOI_5_CA0.v
Conti_Assignment....
decodcr.v
p2s.v
Verilog
01/01/2010 12:31:30 ...
01/01/2010 12:55:53 ...
01/01/2010 03:35:32 ...
01/01/2010 04:56:01 ...
01/01/2010 05:13:15 ...
01/01/2010 05:24:15 ...
01/01/2010 11:19:33 ...
01/01/2010 11:58:32 ...
02/18/2019 08:52:58 ...
Library
Project
C:/altera_lite/15.1/p2s.v (/p2s) - Default *
Ln#
1755 ns
module p2s (d_in , d_out , load , shift , clk);
input [7:0] d_in ;
output reg d_out ;
input load , shift , clk ;
reg [7:0] flpflp ;
always @(posedge clk)
begin
if (load&&shift)
d_out = 1'hx ;
else if (load)
flpflp [7:0] = d_in [7:0] ;
else if (shift) begin
d_out = flpflp [0] ;
flpflp [0] = flpflp [1] ;
flpflp [1] = flpflp [2] ;
flpflp [2] = flpflp [3] ;
flpflp [3] = flpflp [4] ;
flpflp [4] = flpflp [5] ;
flpflp [5] = flpflp [6] ;
flpflp [6] = flpflp [7] ;
end
else
d_out = d_out ;
end
endmodule
Transcript
VSIM 90> run
force -freeze sim:/p2s/clk 1 0
VSIM 92> run
VSIM 93>
Ln: 6 Col: 0
Project : task1
Now: 2 us Delta: 0
sim:/p2s


sim - Default
Instance
p2s
#ALWAYS#8
#vsim_capacity#
Wave - Default
Msgs
/p2s/d_in
/p2s/d_out
/p2s/load
/p2s/shift
/p2s/clk
/p2s/flpflp
[7]
[6]
[5]
[4]
[3]
[2]
[1]
[0]
8'h1c
1'h0
1'h0
1'h1
1'h1
8'h00
8'h5c
8'h2e
8'h17
8'h0b
8'h05
8'h02
8'h01
8'h00
Now
Cursor 1
200 ns
400 ns
600 ns
800 ns
1000 ns
1200 ns
1400 ns
1600 ns
1800 ns
2000 ns
2200 ns
2400 ns
1755 ns
Library
p2s.v
Wave
Transcript
VSIM 90> run
force -freeze sim:/p2s/clk 1 0
VSIM 92> run
VSIM 93>
0 ns to 2512 ns
Project : task1
Now: 2 us Delta: 0
sim:/p2s

Ram

```
Ln#
 1  module dual_ram (chip_sel1,chip_sel2,data_bus1,data_bus2,addr1,addr2,wr1,wr2,rd1,rd2,clk,out1,out2);
 2  input [7:0] addr1,addr2;  // input address port
 3  input chip_sel1,chip_sel2,wr1,wr2,rd1,rd2,clk; //contro; signals
 4  inout [7:0] data_bus1,data_bus2; //in-out data port
 5
 6  output reg [7:0] out1,out2; //output of ram
 7  reg [7:0] ram1 [0:255]; //ram 1 declearation
 8  reg [7:0] ram2 [0:255]; //ram 2 declearation
 9
10  //code of ram 1
11  always @(posedge clk) begin
12  if (wr1&&rd1) //to be sure there is read-write in the same time
13          out1[7:0]=8'hxx;
14  else if (chip_sel1&&rd1&&(addr1!=((8'hxx)||(8'hzz)))) //to be sure there is an input address while readind data
15          out1[7:0]=ram1[addr1];
16  else if (chip_sel1&&wr1) //write data
17          ram1[addr1]=data_bus1[7:0];
18  else
19          out1=8'hxx; //to avoid any latch
20  end
21
22  assign data_bus1[7:0]= (chip_sel1&&rd1&& !wr1)?out1[7:0]:8'hzz ; //buffer to put output in the in-out port
23
24  //code of ram 2
25  always @(posedge clk) begin
26  if (wr2&&rd2) //to be sure there is read-write in the same time
27          out2[7:0]=8'hxx;
28  else if (chip_sel2&&rd2&&(addr2!=((8'hxx)||(8'hzz)))) //to be sure there is an input address while readind data
29          out2[7:0]=ram2[addr2];
30  else if (chip_sel2&&wr2) //write data
31          ram2[addr2]=data_bus2[7:0];
32  else
33          out2=8'hxx; //to avoid any latch
34  end
35
36  assign data_bus2[7:0]= (chip_sel2&&rd1&& !wr2)?out2[7:0]:8'hzz ; //buffer to put output in the in-out port
37
38  endmodule
                                                                                    Ln: 37 Col: 0   **
```