LUCID VLSI

Learning Unlimited. Coding InDelible.

```
module memory_rtl
(clk,reset,wr,rd,addr,wdata,rdata,response);

//Synchronous write read memory
parameter reg [15:0] ADDR_WIDTH=8;
parameter reg [15:0] DATA_WIDTH=32;
parameter reg [15:0] MEM_SIZE=16;

input   clk,reset;
input   wr;// for write wr=1;
input   rd;// for read  rd=1;
input   [ADDR_WIDTH-1:0] addr;
input   [DATA_WIDTH-1:0] wdata;
output  [DATA_WIDTH-1:0] rdata;
output response;

wire    [DATA_WIDTH-1:0] rdata;
reg     [DATA_WIDTH-1:0] mem [MEM_SIZE];
reg     [DATA_WIDTH-1:0] data_out;

reg response ;//Provides response to master on successful
write
reg out_enable;//controls when to pass read data on rdata
pin

//if rd=0 rdata should be in high impedance state
//if rd=1 rdata should be content of memory with given
address
assign rdata = out_enable ? data_out : 'bz;

//asynchronous reset and synchronous write
always @(posedge clk or posedge reset)
begin
    if (reset) begin
        for(int i=0;i<MEM_SIZE;i++)
            mem[i] <= 'b0;
     end
    else if(wr ) begin
            mem[addr] <= wdata ;
             response <=1'b1;
            end
      else response <=1'b0;
end//end_of_write
```

LUCID VLSI

Learning Unlimited. Coding InDelible.

```
//Synchronous Read
always @(posedge clk )
begin
 if (reset) begin
                data_out<='b0;
                out_enable<=1'b0;
                end
else
     begin
       if(rd==1) begin
            data_out <= mem[addr[3:0]] ;
            out_enable <= 1'b1;
        end
        else
            out_enable <=1'b0;
     end
end//end_of_read

endmodule/*****************************
File Name:  ./2x1Mux/dut.sv
Author : Raja Bandi
Organization: Lucid VLSI
Email ID: raja@lucidvlsi.com
Phone No: +91 994 995 4576
*****************************/
module mux (y,s,i1,i0);
input s,i1,i0;
output y;

reg y;

always @(s,i1,i0)
//always @(s or i1 or i0)
//always @*
//always @(*)
//always_comb
     begin
          if (s==0) y = 1'b1;
          else      y = i1;
     end

endmodule
/*****************************
```

LUCID VLSI

Learning Unlimited. Coding InDelible.

```
File Name:  ./2x1Mux/mem.txt
Author : Raja Bandi
Organization: Lucid VLSI
Email ID: raja@lucidvlsi.com
Phone No: +91 994 995 4576
*****************************/
0000
0011
0100
0111
1000
1010
1101
1111
/*****************************
File Name:  ./2x1Mux/run.do
Author : Raja Bandi
Organization: Lucid VLSI
Email ID: raja@lucidvlsi.com
Phone No: +91 994 995 4576
*****************************/
vlib work
vdel -all
vlib work
vlog tb.sv +acc
vlog dut.sv +acc
vsim work.tb
add wave -r *
run -all
/*****************************
File Name:  ./2x1Mux/tb.sv
Author : Raja Bandi
Organization: Lucid VLSI
Email ID: raja@lucidvlsi.com
Phone No: +91 994 995 4576
*****************************/
module tb;

reg s,i1,i0;
logic y;
reg Yexp;
integer errcnt=0;
integer matched_count=0;
```

```
mux mux_inst
( //port by name
.i0(i0),
.y(y),
.i1(i1),
.s (s)
);
reg [3:0] mem[1:8];
integer j;

int golden [$];
//Queue
//push_front
initial
begin
      $readmemb("mem.txt",mem);

      repeat (200000)
            begin
                  {s,i1,i0}=$random;
                  #1;
                  generate_golden_result;
                  compare;//Lin32 executes 2 lakhs
            end
            report;
end

task generate_golden_result;
begin
      for (j=1;j<=8;j=j+1)
            begin
                  if (mem[j][3:1]=={s,i1,i0}) Yexp=mem[j][0];
            end
end
endtask

task compare;
begin
            if (Yexp != y)//checking
                  begin
                        errcnt=errcnt+1;
```


```
                    $display
("%d:FAIL : %b%b%b %b",errcnt,s,i1,i0,y);
                    $finish;//
                //reporting
                end
                else matched_count=matched_count+1;
end
endtask

final
begin
     $display ("I am in the final Block");
     $display ("Please go to LIC office and get money");
end

task report;
begin
$display ("matched count=%d",matched_count);
     $display ("mismatched count=%d",errcnt);
     if (errcnt==0) $display ("************Test Passed
**************");
     else $display ("**Test FAILED : FIX THE BUGS : PRAY
GOD *******");
end
endtask
endmodule

/*****************************
File Name:   ./Counting1s_100bits/counting1s.v
Author : Raja Bandi
Organization: Lucid VLSI
Email ID: raja@lucidvlsi.com
Phone No: +91 994 995 4576
*****************************/
module counting1s (I,Y);//37800 37800
output wire [6:0] Y;
input [99:0] I;
assign Y =
 I[0]+
 I[1]+
 I[2]+
 I[3]+
 I[4]+
 I[5]+
```


```
I[6]+
I[7]+
I[8]+
I[9]+
I[10]+
I[11]+
I[12]+
I[13]+
I[14]+
I[15]+
I[16]+
I[17]+
I[18]+
I[19]+
I[20]+
I[21]+
I[22]+
I[23]+
I[24]+
I[25]+
I[26]+
I[27]+
I[28]+
I[29]+
I[30]+
I[31]+
I[32]+
I[33]+
I[34]+
I[35]+
I[36]+
I[37]+
I[38]+
I[39]+
I[40]+
I[41]+
I[42]+
I[43]+
I[44]+
I[45]+
I[46]+
I[47]+
I[48]+
I[49]+
```


```
I[50]+
I[51]+
I[52]+
I[53]+
I[54]+
I[55]+
I[56]+
I[57]+
I[58]+
I[59]+
I[60]+
I[61]+
I[62]+
I[63]+
I[64]+
I[65]+
I[66]+
I[67]+
I[68]+
I[69]+
I[70]+
I[71]+
I[72]+
I[73]+
I[74]+
I[75]+
I[76]+
I[77]+
I[78]+
I[79]+
I[80]+
I[81]+
I[82]+
I[83]+
I[84]+
I[85]+
I[86]+
I[87]+
I[88]+
I[89]+
I[90]+
I[91]+
I[92]+
I[93]+
```

```
        I[94]+
        I[95]+
        I[96]+
        I[97]+
        I[98]+
        I[99];

    endmodule


    /*****************************
    File Name:  ./Counting1s_100bits/run.do
    Author : Raja Bandi
    Organization: Lucid VLSI
    Email ID: raja@lucidvlsi.com
    Phone No: +91 994 995 4576
    *****************************/
    vlib work
    vdel -all
    vlib work

    vlog counting1s.v
    vlog tb.v
    vsim work.tb
    run -all
    /*****************************
    File Name:  ./Counting1s_100bits/tb.v
    Author : Raja Bandi
    Organization: Lucid VLSI
    Email ID: raja@lucidvlsi.com
    Phone No: +91 994 995 4576
    *****************************/
    module tb;
    reg [99:0] I;
    wire [6:0] Y;
    reg [6:0] Yexp;
    integer j;
    integer errcnt=0;
    integer matched_count=0;

    counting1s  KABALI (I,Y);

    initial
    begin
```

```
        repeat (50)
        begin
             I={5{$random}};
             #1;
             generate_golden_result;
             compare;
        end
             report;
end
task generate_golden_result;
begin
        Yexp=0;
             for (j=0;j<=99;j=j+1)
                  begin
                       Yexp=Yexp+I[j];
                  end
end
endtask

task compare;
begin
             if (Yexp != Y)//checking
                  begin
                       errcnt=errcnt+1;
                      $display ("%d:FAIL : %b %d",errcnt,I,Y);
//reporting
                  end
                  else matched_count=matched_count+1;
end
endtask

task report;
begin
$display ("matched count=%d",matched_count);
        $display ("mismatched count=%d",errcnt);
        if (errcnt==0) $display ("************Test Passed
**************");
        else $display ("**Test FAILED : FIX THE BUGS : PRAY
GOD *******");
end
endtask
endmodule
```