VLSI PRACTICE QUESTIONS SOLUTION

Done By T Tushar Shenoy

NOTE:-Add Compiler Directives as given in Lab Manual if in case they mention, also make sure you know about Tasks and Repeat functions, They may ask to draw the Block Diagram of the Verilog and Testbench Code.

1) Verilog codes for logic operations on 3 or more input bits, Verilog gate level model for a given Boolean expression and relevant Verilog testbenches.

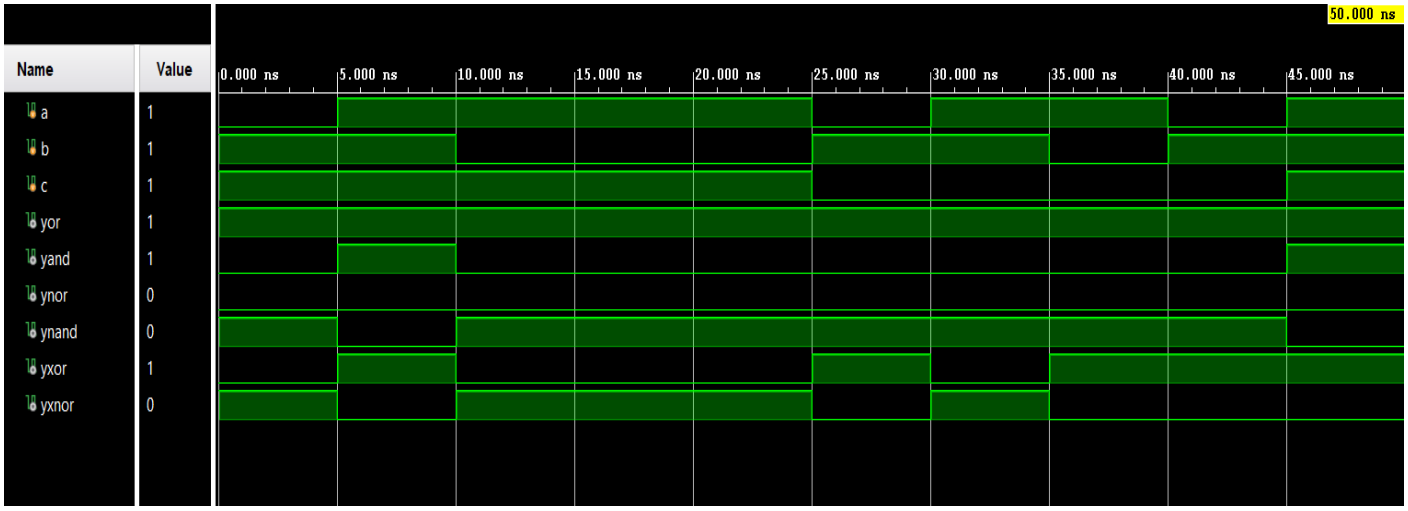
Verilog Code

```
module logic_gates_3in(a,b,c,yor,yand,ynor,ynand,yxor,yxnor);
input a,b,c;
output yor,yand,ynor,ynand,yxor,yxnor;

or(yor,a,b,c);
and(yand,a,b,c);
nor(ynor,a,b,c);
nand(ynand,a,b,c);
xor(yxor,a,b,c);
xnor(yxnor,a,b,c);

endmodule
```

Testbench

```
module logic_gates_3in_tb;
reg a,b,c;
wire yor,yand,ynor,ynand,yxor,yxnor;

logic_gates_3in
dut(.a(a),.b(b),.c(c),.yor(yor),.yand(yand),.ynor(ynor),.ynand(ynand),.
yxor(yxor),.yxnor(yxnor));

initial begin

repeat(10)
  begin
  a=$random;
  b=$random;
  c=$random;
  #5;
  end
  $finish;
end
endmodule
```

OUTPUT WAVEFORM



Consider the Expression **Y=AB+CD+ABCD**

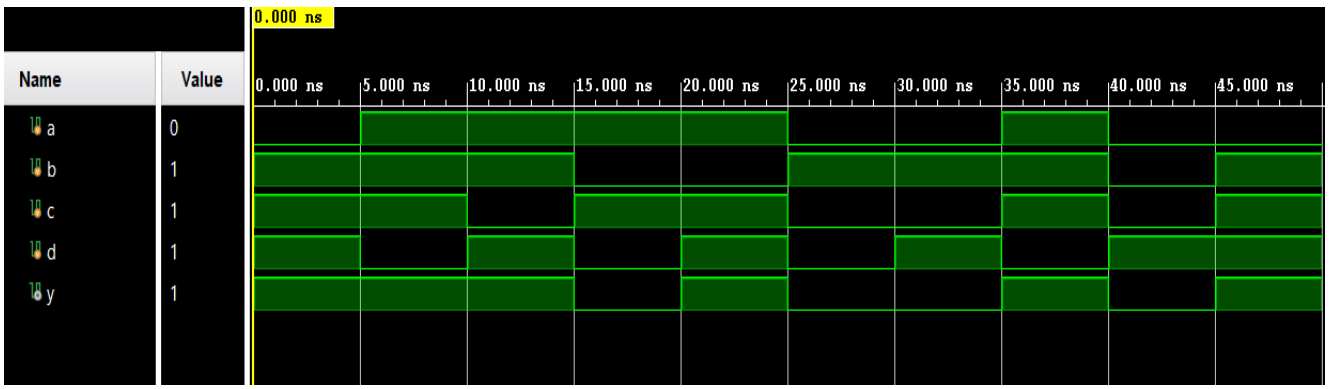
Verilog Code

```
module logic_expression(a,b,c,d,y);
//Consider y=ab+cd+abcd
input a,b,c,d;
output y;

wire w1,w2,w3;

and(w1,a,b);
and(w2,c,d);
and(w3,a,b,c,d);

or(y,w1,w2,w3);

endmodule
```

TESTBENCH

```
module logic_expression_tb;
reg a,b,c,d;
wire y;
logic_expression dut(.a(a),.b(b),.c(c),.d(d),.y(y));

initial begin
repeat(10)
  begin
  a=$random;
  b=$random;
  c=$random;
  d=$random;
  #5;
  end
  $finish;
end
endmodule
```

OUTPUT WAVEFORM



2) Verilog code to compute 2’s complement of a nibble data. Write relevant Verilog testbenches

Verilog Code

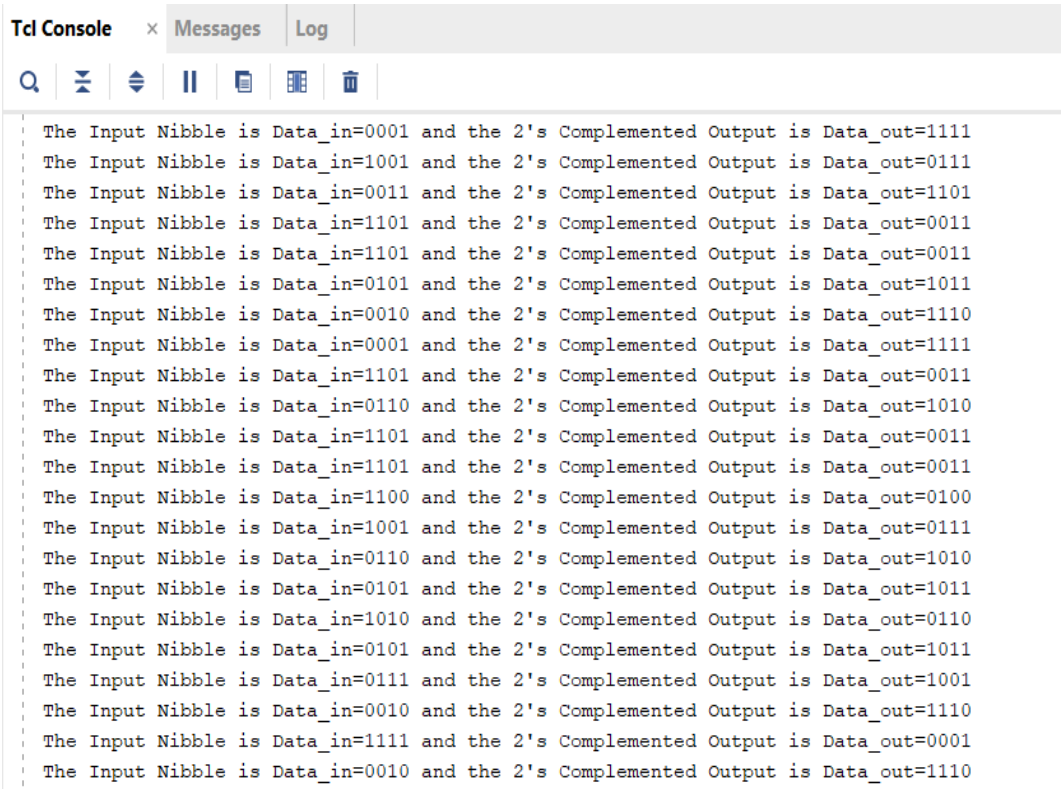
```
module nibble_2s_complement(
    input [3:0]data_in,
    output [3:0]data_out
    );
    assign data_out=~(data_in)+1'b1;
endmodule
```

Testbench

```
module nibble_2s_complement_tb();
reg [3:0]data_in;
wire [3:0]data_out;

nibble_2s_complement dut(.data_in(data_in),.data_out(data_out));

initial begin
  repeat(25)
      begin
      data_in=$random;
      #5$display("The Input Nibble is Data_in=%b and the 2's Complemented Output is Data_out=%b",data_in,data_out);
      #5;
      end
   $finish;
end
endmodule
```

OUTPUT WAVEFORM

CONSOLE OUTPUT



3) Verilog code for MOD-N up counter with synchronous RESET control, MOD-N down counter with synchronous RESET control, given the value of N. Write relevant Verilog testbenches

MOD-N UP Counter (Here MOD Value N is chosen as 8)

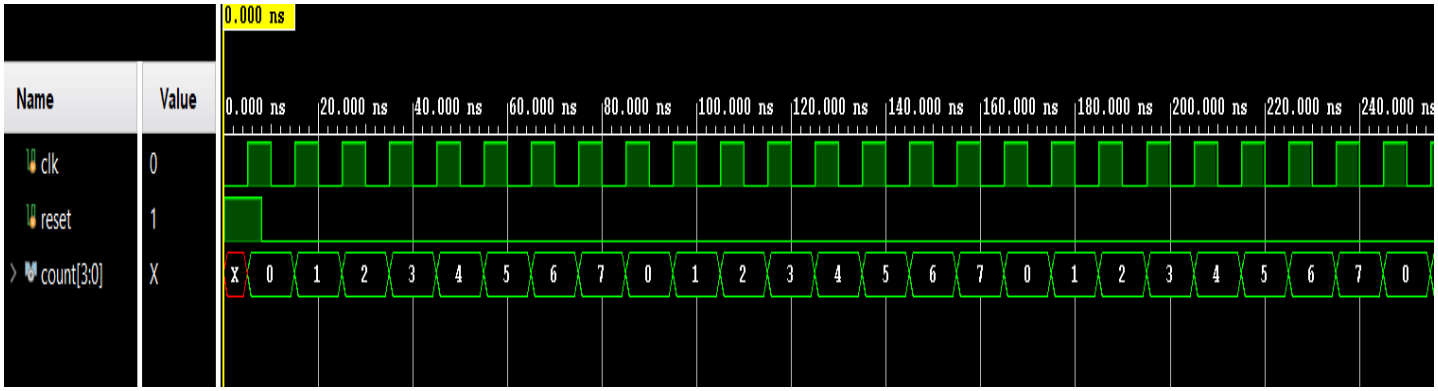
Verilog Code

```
module modnupcounter(clk,reset,count);
 integer i=0;
 parameter N=8;/*MOD*/
 input clk,reset;
 output reg [3:0]count;// Change the Count Size if the N value is greater than 16


        always@(posedge clk)
          begin
              if(reset)
                 begin
                count=0;
                 end
              else if(count<N-1)
                 begin
                       count=count+1;
                 end
              else count=4'b0000;
           end
      endmodule
```

Testbench

```
module modnupcounter_tb();

reg clk,reset;
wire [3:0]count;// Change the Count Size if the N value is greater than 16

modnupcounter dut(.clk(clk),.reset(reset),.count(count));

initial begin
reset=1'b1;
clk=1'b0;
#8 reset=1'b0;
end

always #5 clk=~clk;
endmodule
```

OUTPUT WAVEFORM



MOD N DOWN Counter (Here MOD Value N is chosen as 10)

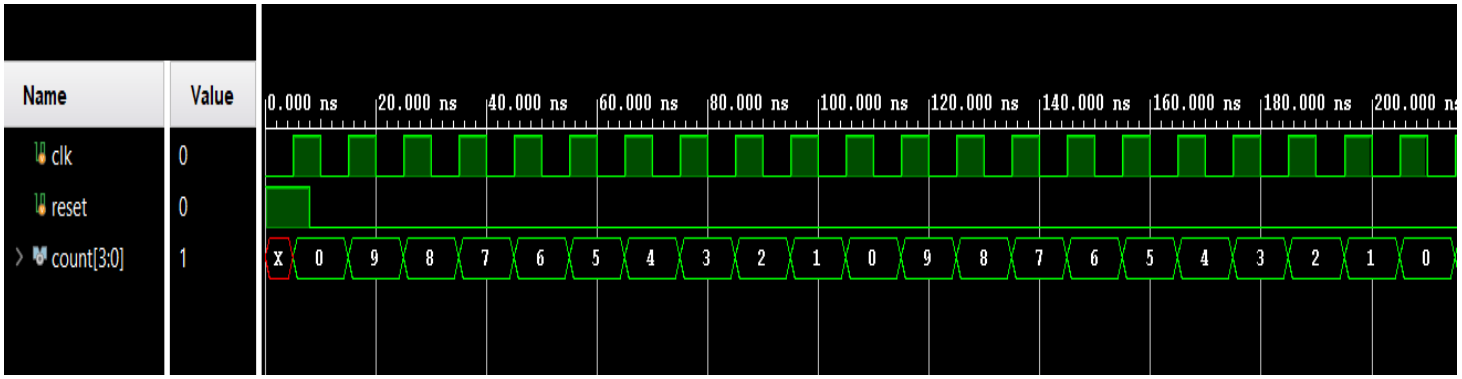
Verilog Code

```
module modndowncounter(clk,reset,count);
 integer i=0;
 parameter N=10;/*MOD*/
 input clk,reset;
 output reg [3:0]count;// Change the Count Size if the N value is greater than 16


    always@(posedge clk)
      begin
           if(reset)
              begin
             count=0;
              end
           else if(count>0)
              begin
                   count=count-1;
              end
           else count=N-1;
       end
endmodule
```

Testbench

```
module modndowncounter_tb();

reg clk,reset;
wire [3:0]count;// Change the Count Size if the N value is greater than 16

modndowncounter dut(.clk(clk),.reset(reset),.count(count));

initial begin
reset=1'b1;
clk=1'b0;
#8 reset=1'b0;

end

always #5 clk=~clk;
endmodule
```

OUTPUT WAVEFORM



MOD N UP/DOWN Counter

Verilog Code (Here MOD Value N is chosen as 10**)**

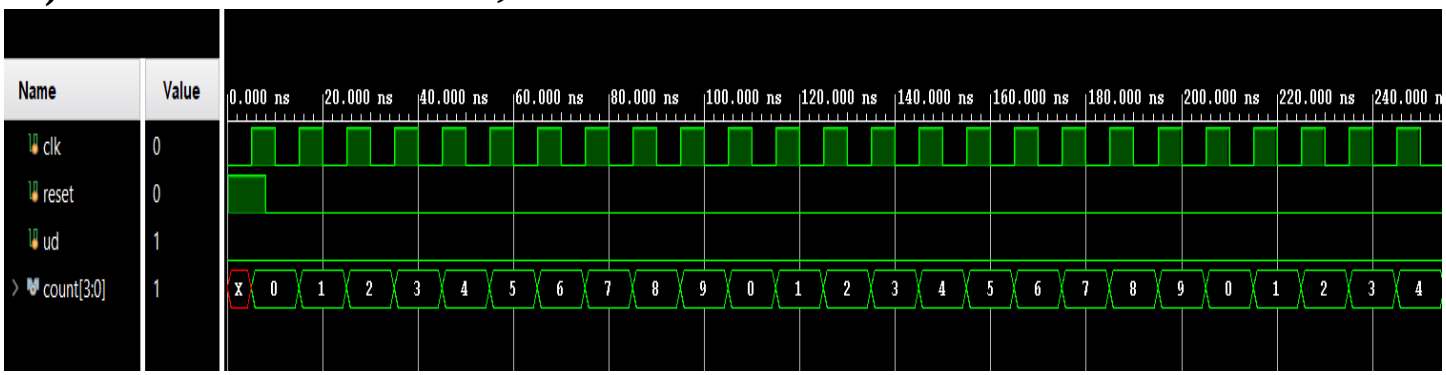
```
module modnupdowncounter(clk,reset,ud,count);
 integer i=0;
 parameter N=10;/*MOD*/
 input clk,reset,ud;//if ud=0 Up Count else down count
 output reg [3:0]count;// Change the Count Size if the N value is greater than 16


   always@(posedge clk)
     begin
          if(reset)
             begin
             count=0;
             end
          else
            if(ud==0)
            begin
              if(count<N-1)
                  count=count+1;
              else  count=4'b0000;
            end
            else if(ud==1)
            begin
              if(count>0)
                  count=count-1;
              else count=N-1;
            end

       end
endmodule
```

Testbench

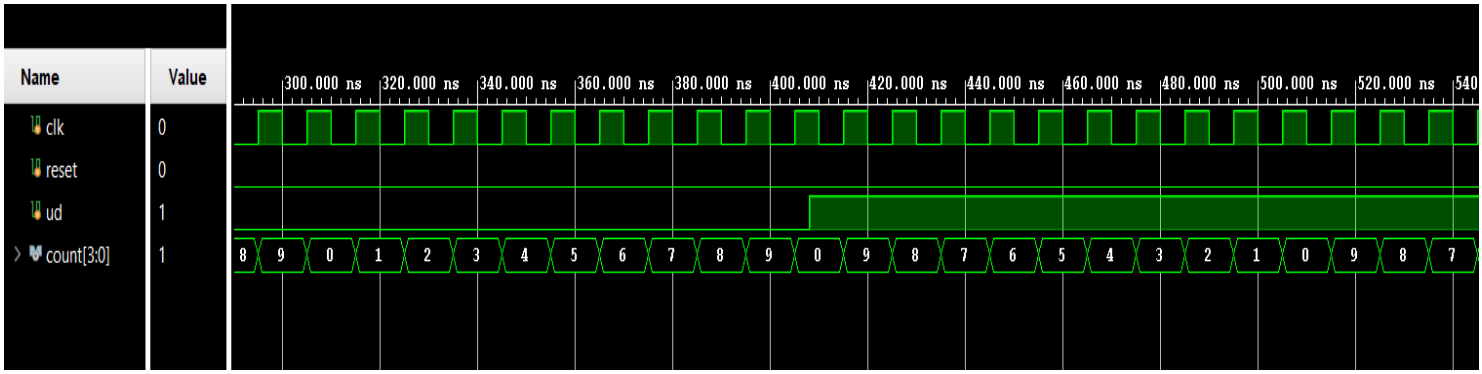
```
module modnupdowncounter_tb();

reg clk,reset,ud;
wire [3:0]count;// Change the Count Size if the N value is greater than 16

modnupdowncounter dut(.clk(clk),.reset(reset),.ud(ud),.count(count));

initial begin
ud=0;
reset=1'b1;
clk=1'b0;
#8 reset=1'b0;
#400 ud=1;
end

always #5 clk=~clk;
endmodule
```

OUTPUT WAVEFORMS

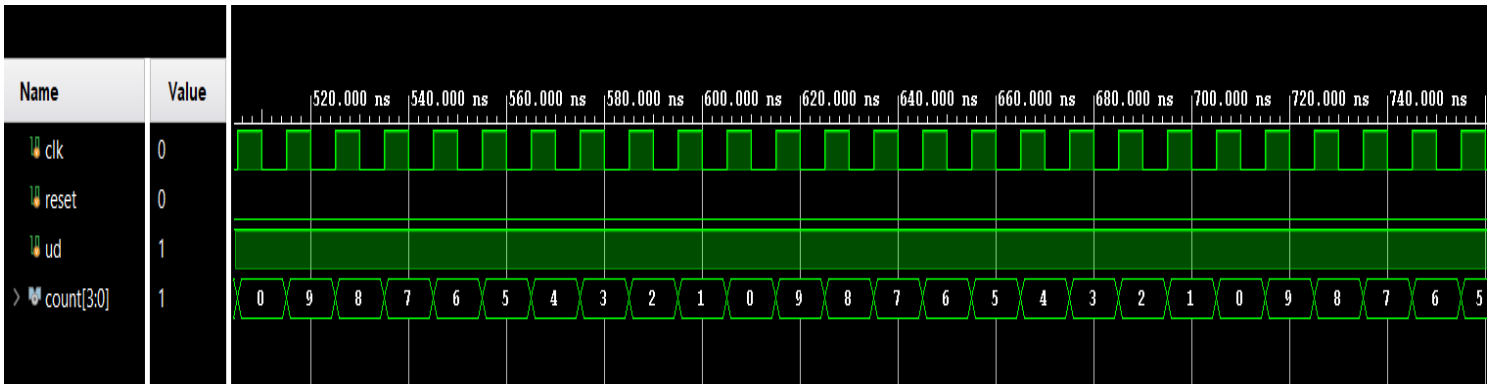
1) When ud=0 UP Count , MOD N UP Counter



2) When ud changes from 0 to 1 Down Count starts



3) When ud=1, DOWN Count, MOD N DOWN Counter



Github:-
https://github.com/tusharshenoy/VLSI_LAB_Practice_Questions