Verilog/Digital FAQS

By

Verilog Course Team

Email:info@verilogcourseteam.com

www.vlsi-faqs.blogspot.com

www.verilogcourseteam.com

DISCLAIMER

Verilog Course Team does not warrant or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed. No warranty of any kind, implied, expressed or statutory, including to fitness for a particular purpose and freedom from computer virus, is given with respect to the contents of this document or its hyperlinks to other Internet resources. Reference in this document to any specific commercial products, processes, or services, or the use of any trade, firm or corporation name is for the information, and does not constitute endorsement, recommendation, or favoring.

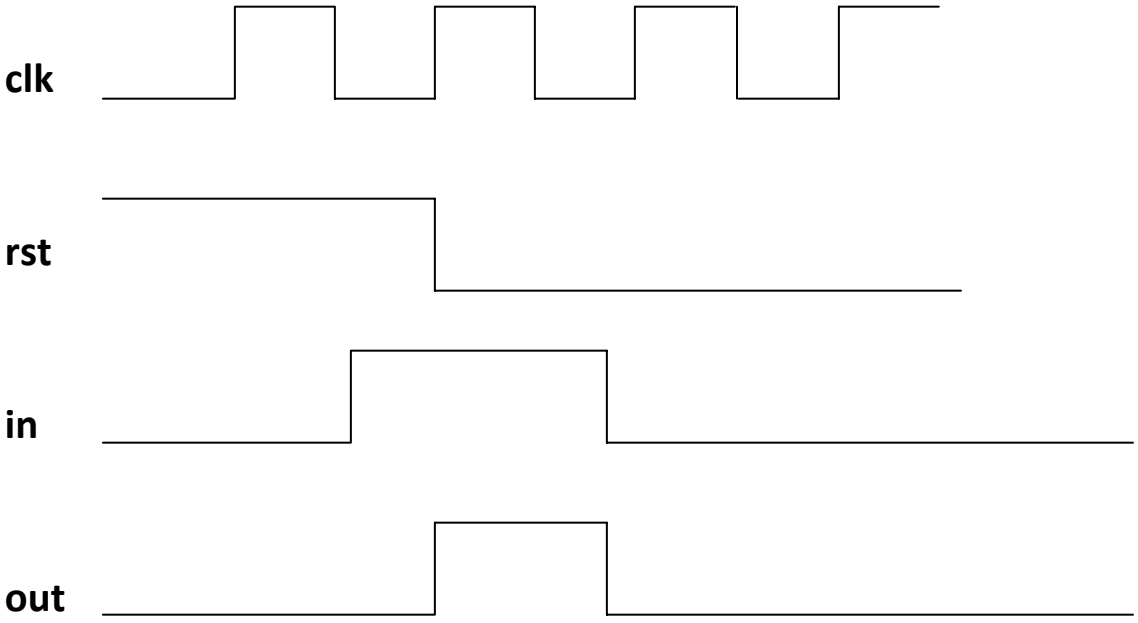
About Verilog Course Team

Verilog Course Team is a Electronic Design Services (EDS) for **VLSI / EMBEDDED and MATLAB**, delivering a wide variety of end-to-end services , including design , development, & testing for customers around the world .With proven expertise across multiple domains such as Consumer Electronics Market ,Infotainment, Office Automation, Mobility and Equipment Controls. Verilog Course Team is managed by Engineers / Professionals possessing significant industrial experience across various application domains and engineering horizontals . Our engineers have expertise across a wide range of technologies, to the engineering efforts of our clients. Leveraging standards based components and investments in dedicated test lab infrastructure; we offer innovative, flexible and cost-effective Services and solutions.

Our Mission

Our mission is to provide cost effective, technology independent, good quality reusable Intellectual Property cores with quality and cost factor are our important constraints so as to satisfy our customers ultimately. We develop and continuously evaluate systems so as to pursue quality in all our deliverables. At our team, we are completely dedicated to customer's requirements. Our products are designed and devoted to empower their competitive edge and help them succeed.

1.Design a circuit(positive edge) that detect the sequence when input changes from 0 to 1,the output should go high for only one clock pulse.



2. Design a circuit to detect when the 2-16 bits inputs are same.

3. Assume b = 3 and c = 5, after the first @ (posedge clk) what is the value of a?

```
always @ (posedge clk)
        a = b;

always @ (posedge clk)
        b = c;
```

4. After the first @ (posedge clk), does this do a swap?

```
always @ (posedge clk)
 begin
        a = b;
        b = a;
 end
```

5. Consider the following code:

```
`define FALSE 0
`define TRUE 1
initial
  begin
a = `FALSE;
a <= `TRUE;
if (a == `TRUE)
$display ("True");
else
$display ("False");
 end
```

What will print out? True or False?

6. Design AND, OR gate using 2:1 mux.

7. Draw the circuit to avoid the Set-up and Hold-time violation.

8. Consider the following code:

```
always@(posedge clk)
if(rst==0)
out<=1'b0;
else
out<=data_in;
```

a. Draw the synthesis view for the above code.
b. Modify the code for Asynchronous Reset.
c. Draw the timing diagram for synchronous and asynchronous reset.

9. How to test the functionality (test cases) of a FIFO.

10. What is the advantage of using Gray code instead of Binary code while designing FIFO.

11. What are the parameters to be considered before starting the design work.

12. Design EX-OR gate using 4 NAND gates.

13. Design EX-NOR gate using 4 NOR gates.

14. What will be the synthesis structure?

```
module two_stage(Q, D, CLK);
input D, CLK;
output Q;

reg Q, P;

always @ (posedge CLK)
  begin
     Q = P;
     P = D;
 end
```

```
module two_stage(Q, D, CLK);
input D, CLK;
output Q;

reg Q, P;

always @ (posedge CLK)
  begin
     Q <= P;
     P <= D;
 end
```

```
module two_stage(Q, D, CLK);
input D, CLK;
output Q;

reg Q, P;

always @ (posedge CLK)
  begin
     P = D;
     Q = P;
end
```

```
module two_stage(Q, D, CLK);
input D, CLK;
output Q;

reg Q, P;

always @ (posedge CLK)
  begin
     P <= D;
     Q <= P;
end
```

15. Consider the following code,

```
always@(posedge clk)
 begin
      a=b;
      b=c;
      c=a;
end
```

What logic does the code implies.