Lecture08Quiz

Due Nov 5 at 11:55pm **Points** 8 **Questions** 8

Available Oct 29 at 9:55am - Nov 5 at 11:59pm **Time Limit** 11 Minutes

This quiz was locked Nov 5 at 11:59pm.

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	6 minutes	8 out of 8

ⓘ Correct answers are hidden.

Score for this quiz: **8** out of 8

Submitted Nov 4 at 10:41pm

This attempt took 6 minutes.

Question 1

1 / 1 pts

Which of the following **constraint** types for **rand** numbers is supported by system verilog?

☑ Conditional (constraint is conditioned off value of another rand signal)

☑ Relational (values follow some specified relation to each other)

☑ Equation (rand values fit an equation)

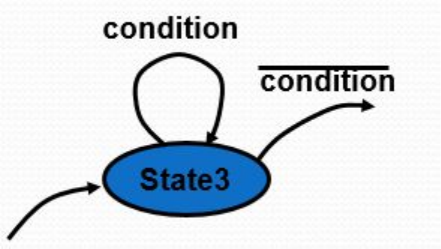
☑ Range Sets (rand value fit inside specified range set)

☑ Distribution (values fit a specified PDF)

Question 2

1 / 1 pts

The following excerpt from a state diagram represents:



○ A "for" loop with a fixed number of iterations

● How "while" loops should be done for code we intend to synthesize

○ A bad way of implementing a "while" loop

○ An infinite loop

Question 3

1 / 1 pts

Is the following sequence of 3-bit numbers from a **rand** or **randc** sequence? could be either

001 010 110 011 101 000 100 111

Is the following sequence of 3-bit numbers from a **rand** or **randc** sequence? rand

001 010 110 000 101 010 100 111

Answer 1:

could be either

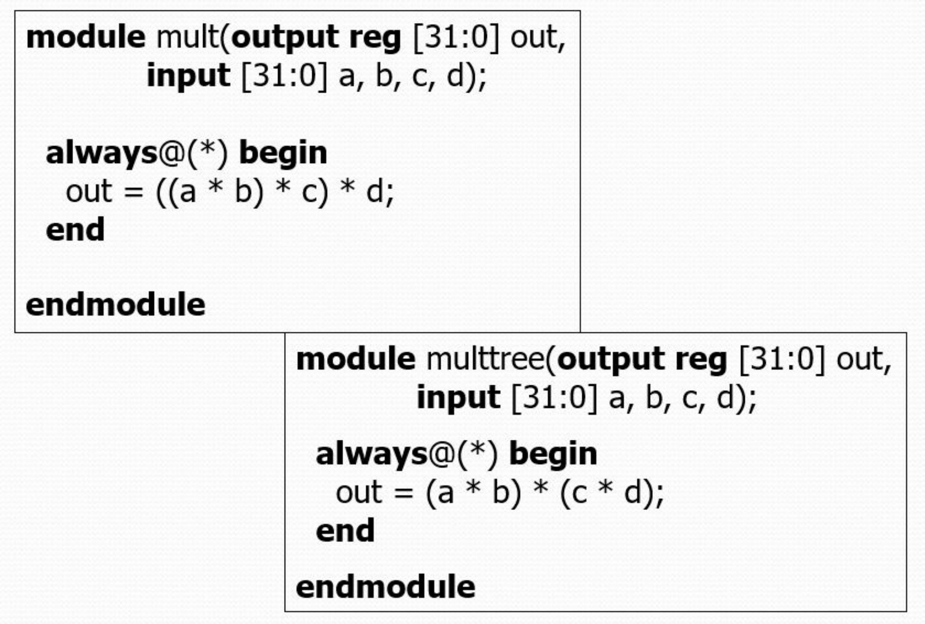
Answer 2:

rand

Question 4

1 / 1 pts

Regarding how the two code snippets shown would synthesize:



```
module mult(output reg [31:0] out,
         input [31:0] a, b, c, d);

  always@(*) begin
    out = ((a * b) * c) * d;
  end

endmodule
```

```
module multtree(output reg [31:0] out,
         input [31:0] a, b, c, d);

  always@(*) begin
    out = (a * b) * (c * d);
  end

endmodule
```

□ They would synthesize the same because Synopsys ignores operator precedence with regard to arithmetic ops

□ The first one ((a*b)*c)*d will be smaller

☑ The second one (a*b)*(c*d) is faster (longest (worst case) path is shorter))

☑

There might be situations where the first one is better than the second one.

Question 5

1 / 1 pts

System verilog suports object oriented programming constucts True

Hoffman expects you to use CRV (Constrained Random Verification) on the project False

CRV methodology tends to get project management nervous because they don't see immediate progress on verification coverage True

Answer 1:

True

Answer 2:

False

Answer 3:

True

Question 6

1 / 1 pts

There are multiple possible true statements you could form here, but form the one that was explicitly mentioned in Lec08.

If you stare at a block of code for more than 15 seconds and you can't figure out what is synthesizes to then the code is wrong.

Answer 1:

more than 15 seconds

Answer 2:

can't figure out what is synthesizes to

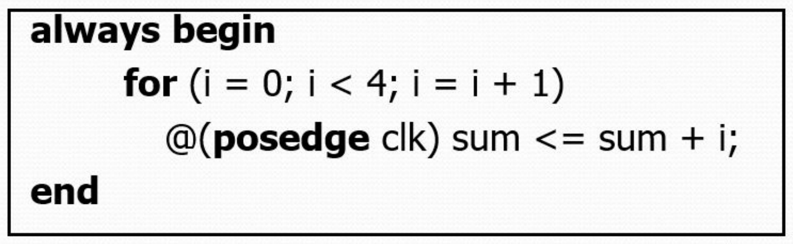
Answer 3:

the code is wrong.

Question 7

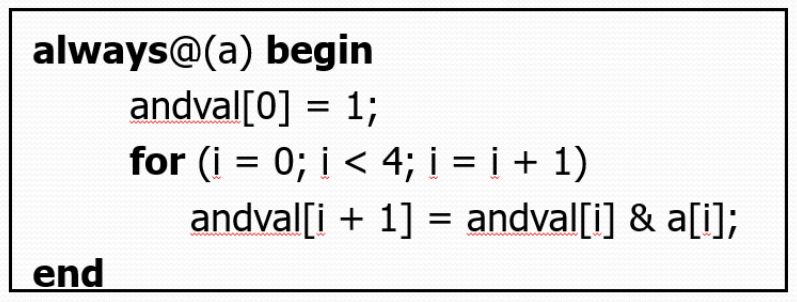
1 / 1 pts

When the following code segment is synthesized, what will the loop index (**i**) synthesize to? **i** is declared as type integer. a 32-bit register



```
always begin
        for (i = 0; i < 4; i = i + 1)
            @(posedge clk) sum <= sum + i;
end
```

When the following code segment is synthesized, what will the loop index (i) synthesize to? i is declared as type integer. Nothing...it is just used for loop unrolling



```
always@(a) begin
        andval[0] = 1;
        for (i = 0; i < 4; i = i + 1)
             andval[i + 1] = andval[i] & a[i];
end
```

Answer 1:

a 32-bit register

Answer 2:

Nothing...it is just used for loop unrolling

Question 8

1 / 1 pts

A **generate** statement can be used to make an architectual choice like choosing between a CLA based multiplier vs a Wallace tree multiplier. This decision can be made based on the width of the operands being multiplied. The instructor considers this a good example of the use of **generate**. False

Answer 1:

width

Answer 2:

False

Yes. Synopsys is very smart about implementation choices for arithmetic blocks. It knows the timing constraints better than you. Let it choose the multiplier implementation based meeting timing.

Quiz Score: **8** out of 8