1) Which of the following is not a net data type?

1. Triand;
2. **Time;**
3. Wire;
4. Tri;
5. Which of the following examples are true /correct for continuous assignments statements?

**Ex 1.**

**wire** out;  
**assign** out = In\_A & In\_B ;

**Ex 2.**

**wire** #25 out = In\_A & In\_B ;

**Ex3.**

**assign** #25 out = In\_A & In\_B ;

1. 1;
2. 2;
3. 3;
4. **All of these;**
5. Out of the following operators, which one is a “replication operator”?
6. ( );
7. { };
8. **{{ }};**
9. [ ];
10. Turn off delay means, gate output transition to
11. 0;
12. **Z;**
13. 1;
14. X;
15. In continuous assignment statements, LHS is
16. Scalar net;
17. Vector net;
18. **Concatenation of A and B;**
19. All of the above;
20. Which of these statements is NOT true regarding Functions in Verilog?
21. **Blocking and Non Blocking statements both are allowed.**
22. Automatic keyword allows each function call to operate in an independent space.
23. They cannot have output arguments.
24. Used to describe purely combinational logic.
25. Which of these statements is NOT true regarding Tasks in Verilog?
26. Can have zero or more arguments of any type.
27. They do not return any value.
28. **Automatic keyword cannot be used unlike functions**.
29. Used to describe Sequential logic and Combinational logic.
30. Which of the following is NOT a predefined keyword in Verilog?
31. small
32. assign
33. liblist
34. vector
35. Which of the following is NOT a compiler directive in Verilog?
36. **`elseif**
37. `ifndef
38. `autoexpand\_vectornets
39. `accelerate
40. Which of these statements is NOT true regarding “disable” statement in Verilog?
41. It is used to terminate tasks and blocks.
42. **It can be used in anywhere inside module.**
43. If a task enables other tasks then all enabled tasks will be disabled.
44. It cannot disable functions.
45. Which of the following is an INCORRECT way for overriding parameters in following code?

**module** xor\_array(y, a, b);

**parameter** SIZE = 8, DELAY = 15; // parameter defaults

**output** [SIZE-1:0] y;

**input** [SIZE-1:0] a,b;

**wire** #DELAY y = a ^ b;

**endmodule**

1. **xor\_array #(.SIZE(4), .DELAY(5));**

**xor\_array G3(y2, a2, b2);**

1. xor\_array #(.SIZE(4), .DELAY(5)) G3(y2, a2, b2);
2. defparam G4.SIZE = 4, G4.DELAY = 15;

xor\_array G4(y2, a2, b2);

1. xor\_array #(4, 5) G2(y2, a2, b2);
2. Verilog allows constant to be defined in a module using which keyword?
3. generic
4. constant
5. define
6. **parameter**
7. Initially a=5 and b=3. What will be the output of a and b after execution of the following program?

always @(posedge clock)

a=b;

always @(posedge clock)

b=a;

A. a=3 and b=5

B**. a=3 and b=3**

C. a=5 and b=3

D. None of the above

1. What %g string format specifies?

A. Display real number in scientific format.

B. Display real number in scientific or decimal, whichever is shortest.

C. Display real number in decimal format.

D. Display hierarchical name.

1. What will be the output of a, b and c after the execution of the following program?

**Initial**

**begin**

**a=1; b=0; c=0;**

**a<= #2 b;**

**c<= #2 a;**

**end**

A. **a=0; b=0; c=1;**

B. a=1; b=0; c=1;

C. a=1; b=0; c=0;

D. None of the above

1. To eliminate race condition in Verilog which type of statement is use in Verilog
2. conditional Statements
3. **nonblocking statements**
4. delay control statements
5. blocking statements
6. how many flops will be synthesis by given code

always @(posedge clock) begin

q1<=d;

q2<=q1;

q3<=q2;

end

1. 1
2. 2
3. **3**
4. NONE
5. In below Verilog code what value of “a” is displayed

Always @(posedge clock) begin

a=0;

a<=1;

$strobe(a);

End

1. 0
2. **1**
3. Either 0 or 1 depends upon simulation implementation
4. None
5. What is the value of “a” in given Verilog code. Choose most suitable answer from the options.

always @(clk) begin

a = 0;

a <= 1;

end

A. a=1

B. a=0

C. Both of above

**D.A=0 in this simulation and a=1 would be in next simulation**

1. Analyze the code and find out what logic blocks will generate.

module lower (clk, in1,in2, out2);

Input clk,in1,in2;

Output out2;

Reg tmp;

always @(posedge clk)

begin

tmp<=(in1 ^ in2);

tmp<=(in1 & in2);

tmp<=(in1 | in2);

end

assign out2=tmp;

endmodule

1. one d flip-flop and one x-or gate
2. three d-flip-flop and one or gate
3. **one d-flip-flop and one or gate**
4. three d-flip-flop and one and gate
5. Ambiguous condition arises when any operand bit is
6. 0;
7. 0 or 1;
8. **X or Z;**
9. None;
10. Output of “==” can be
11. X;
12. 1;
13. 0;
14. All of these;
15. a = #5 b; means:
16. Wait five time units before doing the action for "a = b;"
17. **The value of "b "is calculated and stored in an internal temp register. After five time units, assign this stored value to a.**
18. The value of "a " stored in an internal temp register after five time units.
19. None
20. Operators are arranged as per the highest precedence, which one is correct?
21. Multiplication>Addition>Conditional>Unary;
22. Addition>Multiplication>Unary>Conditional;
23. **Unary>Multiplication>Addition>Conditional;**
24. Conditional>Multiplication>Addition>Unary;
25. 1;
26. 2;
27. **3**;
28. 4;
29. Analyse below code segment to choose correct answer

always @(irq)

begin

{in2,in1,in1} = 3'b0;

casez (irq)

3'b1?? : in2 = 1'b1;

3'b?1? : in1 = 1'b1;

3'b??1 : in0 = 1'b1;

endcase

end

1. Unknown case
2. This is full case
3. This is parallel case
4. **Case which is not parallel**
5. Analyse the code below and choose the right options

reg clock;

initial

begin

clock = 1'b0;

forever #10 clock = ~clock;

end

1. The forever loop will execute the statement infinitelywithout advancing simulation time.
2. **Forever loop will execute the statement infinitely and rest of the design is executed in simulation time.**
3. Timing control construct is optional in forever.
4. None
5. module case4(output integer y, input [1:0] sel)

always@(sel)

casez(sel)

2’b00:y=1;

2’bx0:y=3;

2’b1x:y=4;

2’bz0:y=5;

2’b1z:y=2;

2’b1?:y=6;

default:y=7;

endcase

endmodule

The value of y if sel is 1z

A.6

**B.2**

C.4

D.7

1. module(output reg y);

reg t;

reg s;

reg x;

always @(posedge t or posedge s)

if (s)

y=0;

else

y=x;

initial

begin

t=0;

s=1;

#15 s=0;

x=1;

#40 s=1;

#10 s=0;

$finish;

$monitor ($time,”y=%d”,y);

end

always t=~t;

endmodule

What is output of above program?

1. 0 y= 0

20 y=1

40 y=0

1. **0 y= 0**

**20 y=1**

**55 y=0**

1. 0 y= 0

15 y=1

40 y=0

1. 0 y = 0

Initial

begin

x=0;

y=0;

#0 x=1;

#0 y=1;

$strobe(“ %d %d”,x,y);

end

What is output of above program?

1. 0 0
2. 0 1
3. 1 0
4. **1 1**
5. Which of the following is not correct timescale in verilog?
6. 1ns/10ps
7. 100ns/1ns
8. 10ns/100ps
9. **110ns/100ps**
10. If sequential UDPs store state, the output termial must also be declared as a \_\_\_.
11. **reg.**
12. net
13. wire
14. None of this.
15. Which type of value UDPs do not handle?
16. 1.
17. 0.
18. X.
19. **Z**
20. The \_\_\_\_ terminal must always appear first in the terminal list. Multiple \_\_\_\_ terminals are not allowed.
21. Output, Input.
22. Input, Output.
23. Input.
24. **Output.**
25. In shorthand notation what is the value of (?) in given example?

a b : out;

1. ? : 1;
2. **0, 1, X**.
3. 0, 1, Z.
4. 0,1.
5. X, Z.