25 Additional Verilog Interview Questions and Answers

1. What is the purpose of the `default_nettype none` directive in Verilog?

Answer: It disables implicit declaration of nets, helping to catch undeclared identifiers and typos.

2. How do you model tri-state logic in Verilog?

Answer: Using 'assign' with 'z' values or conditional assignment to '1'bz' to represent high impedance.

3. What is the difference between blocking and non-blocking assignments in Verilog testbenches?

Answer: Blocking is often used for procedural code in testbenches, while non-blocking is preferred in RTL to mimic hardware behavior.

4. What is a race condition in simulation and how do you avoid it?

Answer: Occurs when order of execution affects results. Avoid by consistent use of blocking/non-blocking and proper sensitivity lists.

5. How can you model a shift register in Verilog?

Answer: Using always blocks with shift operations like 'data <= {data[6:0], serial_in};'.

6. What is the function of $finish and $stop in Verilog?

Answer: $finish ends the simulation completely; $stop pauses it for debugging.

7. How are constants defined in Verilog?

Answer: Using `define, parameter, or localparam.

8. What is a generate block and where is it used?

Answer: Used for creating multiple instances or conditional instantiation during elaboration.

9. What is a blocking assignment best used for?

Answer: For combinational logic within always blocks.

10. How do you perform synthesis-only operations?

Answer: Avoid system tasks and use constructs supported by synthesis tools.

11. What is the difference between 'initial' and 'always' blocks?

Answer: 'initial' runs once at simulation start; 'always' runs whenever triggered by sensitivity list.

12. How are delays specified in Verilog?

Answer: Using # followed by a time unit, e.g., #10; often only used in testbenches.

13. What is meant by 'posedge clk or negedge rst' in an always block?

Answer: It triggers on rising clock edge or falling reset edge, typical for flip-flops.

14. What is the role of $random in Verilog?

Answer: Generates pseudo-random numbers, used for testbench stimulus.

15. Can you nest modules in Verilog?

Answer: Yes, modules can instantiate other modules.

16. What is the difference between logic [7:0] and reg [7:0] in SystemVerilog?

Answer: 'logic' is a 4-state data type replacing 'reg' in SystemVerilog.

17. How do you ensure testbench coverage?

Answer: Use functional coverage metrics and constrained random testing.

18. What are packed and unpacked arrays in Verilog/SystemVerilog?

Answer: Packed arrays are stored contiguously like vectors; unpacked arrays are like memory arrays.

19. How do you handle asynchronous reset in Verilog?

Answer: Include it in the sensitivity list and reset signals inside the always block.

20. What are synthesizable and non-synthesizable constructs?

Answer: Synthesizable constructs can be mapped to hardware; non-synthesizable are simulation-only (e.g., $display).

21. What is a net in Verilog?

Answer: A net (like wire) connects structural elements and reflects value changes immediately.

22. What is a procedural assignment?

Answer: Assignment inside initial or always blocks, e.g., using reg or integer types.

23. What are attributes in Verilog?

Answer: Compiler directives used to control synthesis behavior or annotate code.

24. How are enums declared in SystemVerilog?

Answer: Using 'enum {STATE1, STATE2};' helpful for FSMs.

25. What are logic levels in Verilog?

Answer: 0, 1, x (unknown), and z (high impedance) are the four logic levels.