1. How can I abort the execution of a task or a block of code?

In Verilog, you can abort a task or block of code using the $stop or $finish system tasks.

* $stop: It halts simulation, but it allows you to continue execution after inspecting the state using a debugger.
* $finish: Ends the simulation entirely.

1. What does it mean to “short-circuit” the evaluation of an expression?

Short-circuit evaluation refers to the technique used in logical expressions where the evaluation stops as soon as the result is determined.

For example:

In AND (&&) and OR (||) operations, if the result can be determined by the first operand, the second operand is not evaluated. In A && B, if A is 0, Verilog doesn’t evaluate B because the result is already determined to be false.

1. What is the difference between a constant part-select and an indexed part-select of a vectored net?

* Constant part-select: The indices used to select a range of bits are constant literals. These indices don’t change during simulation or synthesis.

Example: wire [7:0] data;

assign out = data[3:0]; // Constant part-select

* Indexed part-select: The indices are variables, often determined dynamically during simulation or at runtime. This allows the part-select range to be computed at runtime.

Example: wire [7:0] data;

integer i;

assign out = data[i+:4]; // Indexed part-select, size 4, starts at i

1. Illustrate how memory indirection is achieved in Verilog.

Memory indirection in Verilog can be done using a memory (array) and referencing it via an index or pointer-like mechanism. Verilog arrays allow dynamic indexing, which can simulate indirect addressing.

Example:

reg [7:0] mem [0:255]; // Memory array with 256 locations

integer idx;

always @\* begin

idx = some\_value; // Indirection via variable index

out = mem[idx]; // Access memory indirectly using idx

end

1. What is the logic synthesized when a non-constant is used as an index in a bit-select?

When a non-constant expression is used as an index in a bit-select, the synthesis tool typically synthesizes logic that computes the index dynamically. If the expression is a variable or a signal, the synthesis will likely create multiplexers (MUXes) or arithmetic logic to compute the correct select position during runtime.

1. How are string operands stored as constant numbers in a reg variable?

In Verilog, strings can be represented as an array of characters. When you store a string in a reg variable, the string is represented by a sequence of ASCII values (each character as a byte).

Example:

reg [7:0] my\_string [0:9];

initial begin

my\_string[0] = "H";

my\_string[1] = "e";

my\_string[2] = "l";

my\_string[3] = "l";

my\_string[4] = "o";

end

1. How can I typecast an expression to control its sign?

In Verilog, you can use $signed and $unsigned system functions to typecast an expression to control its signed or unsigned interpretation.

Example:

reg [7:0] unsigned\_value;

reg signed [7:0] signed\_value;

assign signed\_value = $signed(unsigned\_value);

1. What are the pros and cons of using hierarchical names to refer to Verilog objects?

* Pros:
  + Hierarchical names can make it clear where a signal or module resides in the design hierarchy.
  + They help to prevent naming conflicts by providing uniqueness within the design.
* Cons:
  + Hierarchical names can make the code longer and harder to manage if the design becomes very deep.
  + They may reduce portability because they tie the design to a specific hierarchy.

1. Explain fork-join in Verilog with an example.

fork-join is used for parallel execution of multiple blocks of code in Verilog. It allows concurrent execution of statements, waiting for all branches to finish before proceeding.

Example:

initial begin

fork

$display("Task 1");

#5 $display("Task 2");

#10 $display("Task 3");

join

end

In this example, tasks will be executed concurrently. The join ensures that the simulation doesn't proceed until all tasks are finished.

1. Can I return from a function without having it disabled?

No, you cannot return from a Verilog function until the function's code reaches its end or encounters a disable statement. However, you can use return to immediately exit the function.

1. What is strobing? How do I selectively strobe a net?

Strobing refers to the act of temporarily enabling or "validating" a net at specific times, often using a clock or other control signal. In Verilog, strobing is commonly done with posedge or negedge edge-sensitive triggers.

1. How can I selectively enable or disable monitoring?

You can use disable statements or conditional assignments to selectively enable or disable monitoring in Verilog.

Example:

always @(posedge clk) begin

if (enable\_monitoring) begin

$display("Monitoring enabled");

end

end

1. How can I specify an argument on the Verilog simulator’s command line?

You can specify arguments on the simulator’s command line using the + syntax. For example, for the ModelSim simulator: vsim -do "run -all" +arg\_name=value

1. Can the `define be used for text substitution through variable instead of literal substitution only?

No, define is used for literal text substitution. It cannot be used with variables. It replaces the text literally at compile time, so it can only work with fixed values.