Verilog HDL:
Value Change Dump File

Pravin Zode

Outline

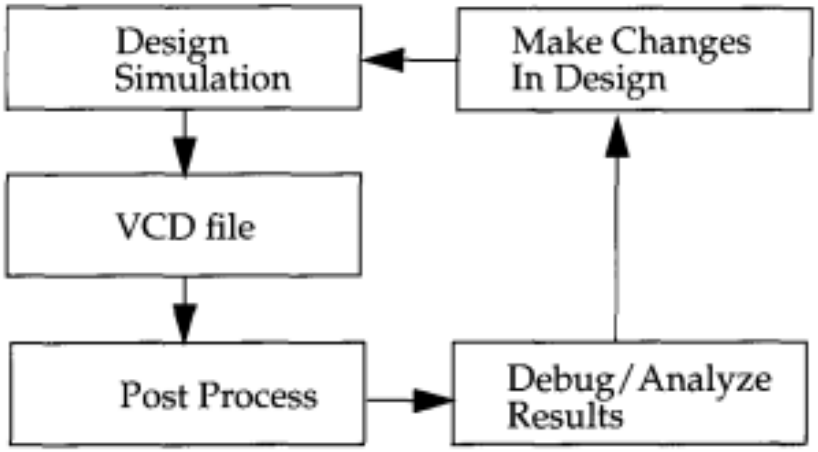
- VCD File use
- System Task for VCD file

Introduction

- Verilog provides two powerful features for simulation and debugging
  - **Value Change Dump (VCD):** Captures signal transitions for waveform visualization
  - **Programming Language Interface (PLI):** Allows interaction between Verilog and external programs (C, SystemVerilog, etc.)

VCD (Value Change Dump)

- VCD file is an ASCII file that records simulation information, including time, scope, signal definitions, and signal value changes
- Used for debugging, analysis, and waveform visualization after a simulation run.
- Helps designers understand signal transitions over time.



System Tasks for VCD Handling

- **$dumpvars**(level, module_instance);
  - Dumps all signals in the specified module instance
  - level = 0 dumps all signals, while higher levels control hierarchy depth
- **$dumpon;**
  - Starts dumping signal values into the VCD file
- **$dumpoff;**
  - Stops the dumping process temporarily

Specifying File

- If no file is specified, the simulator assigns a default name
- The $dumpfile task sets a custom file name for dumping

```
initial
    $dumpfile("myfile.dmp"); // Simulation info saved in "myfile.dmp"
```

Dumping Signal Values in a Module

- $dumpvars; (without arguments) dumps all signals in the design
- $dumpvars(level, instance); controls the hierarchy depth

```
initial
    $dumpvars;  // Dump all signals in the design

initial
    $dumpvars(1, top);  // Dump signals in module `top`, one level deep
```

Example : Simple VCD file

```
1   module simple_test;
2       reg a, b;
3       wire sum;
4
5       assign sum = a ^ b;
6
7       initial begin
8           a = 0; b = 1;
9           #10 a = 1; b = 0;
10          #10 a = 1; b = 1;
11          #10 a = 0; b = 0;
12          #100 $finish;
13      end
14
15      initial begin
16          $dumpfile("simple.vcd"); // Set VCD filename
17          $dumpvars; // Dump all signals in the design
18      end
19  endmodule
```

Example : Selective Signal Dumping

```
1  module test_module;
2      reg clk, reset, enable;
3      wire out;
4
5      assign out = clk & enable;
6
7      initial begin
8          clk = 0; reset = 1; enable = 0;
9          #5 reset = 0; enable = 1;
10         #10 enable = 0;
11         #50 $finish;
12     end
13
14     always #5 clk = ~clk;
15
16     initial begin
17         $dumpfile("selective.vcd");
18         $dumpvars(0, test_module.clk, test_module.reset); // Dump only clk and reset
19     end
20 endmodule
```

Example : Dumping Up to One Level of Hierarchy

```
1  module sub_module;
2      reg x, y;
3  endmodule
4
5  module top;
6      reg clk;
7      sub_module sm1();
8
9      always #5 clk = ~clk;
10
11     initial begin
12         $dumpfile("one_level.vcd");
13         $dumpvars(1, top); // Dump signals only in top, not in submodules
14     end
15 endmodule
```

Example : Dumping Signals Up to Two Levels

```
1   module sub;
2       reg x, y;
3   endmodule
4
5   module mid;
6       sub sm();
7   endmodule
8
9   module top;
10      mid m();
11
12      initial begin
13          $dumpfile("two_levels.vcd");
14          $dumpvars(2, top); // Dump signals in top and mid, but not in sub
15      end
16  endmodule
```

Example : Controlling the Dumping Process

```
1   module counter_module;
2       reg clk, reset;
3       reg [3:0] count;
4
5       always #10 clk = ~clk;
6
7       always @(posedge clk or posedge reset) begin
8           if (reset)
9               count <= 0;
10          else
11              count <= count + 1;
12      end
13
14      initial begin
15          clk = 0; reset = 1;
16          #5 reset = 0;
17          #200 $finish;
18      end
19
20      initial begin
21          $dumpfile("control_dump.vcd");
22          $dumpvars(0, counter_module);
23          #50 $dumpoff;  // Stop dumping at time = 50
24          #100 $dumpon;  // Resume dumping at time = 150
25      end
26  endmodule
```

Example : Counter ($dumpon and $dumpoff)

```
1  module dump_control;
2      reg clk, enable;
3
4      always #5 clk = ~clk;
5
6      initial begin
7          clk = 0; enable = 1;
8          #20 enable = 0;
9          #50 enable = 1;
10         #200 $finish;
11     end
12
13     initial begin
14         $dumpfile("dump_on_off.vcd");
15         $dumpvars(0, dump_control);
16         #50 $dumpoff;  // Stop dumping at time = 50
17         #100 $dumpon;  // Resume dumping at time = 100
18     end
19 endmodule
```

Example : Counter ($dumpon and $dumpoff)

```
1   module counter_vcd;
2       reg clk, reset;
3       reg [3:0] count;
4       initial begin
5           $dumpfile("counter.vcd");  // Specify VCD file name
6           $dumpvars(1, counter_vcd); // Dump signals at 1 level of hierarchy
7
8           clk = 0; reset = 1;
9           #5 reset = 0;  // Release reset after 5 time units
10      end
11      always #10 clk = ~clk; // Toggle clock every 10 time units
12      always @(posedge clk or posedge reset) begin
13          if (reset)
14              count <= 0;
15          else
16              count <= count + 1;
17      end
18      initial begin
19          $dumpon;       // Start dumping signals
20          #100 $dumpoff; // Stop dumping after 100 time units
21          #50 $finish;   // End simulation
22      end
23  endmodule
```

Example : Creating a Checkpoint Using $dumpall

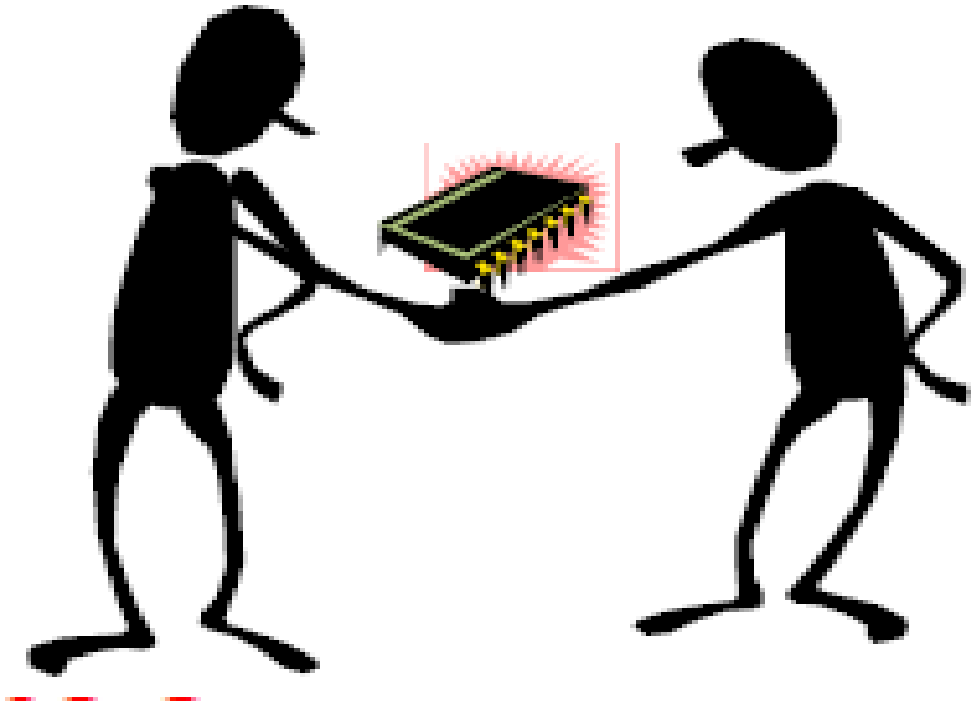
```
1   module sub_block;
2       reg data;
3   endmodule
4
5   module main_block;
6       sub_block s1();
7       reg clk;
8
9       always #10 clk = ~clk;
10
11      initial begin
12          $dumpfile("checkpoint.vcd");
13          $dumpvars(2, main_block);
14          #75 $dumpall; // Create a checkpoint at time = 75
15      end
16  endmodule
```

Example : Multiple Dumpfiles

```
1   module sub_module;
2       reg x, y;
3   endmodule
4
5   module top;
6       reg clk;
7       sub_module sm1();
8
9       always #10 clk = ~clk;
10
11      initial begin
12          $dumpfile("top_level.vcd");
13          $dumpvars(0, top);
14      end
15
16      initial begin
17          $dumpfile("sub_level.vcd");
18          $dumpvars(0, top.sm1); // Dump signals only from sub_module
19      end
20  endmodule
```

Summary

Task	Description
$dumpfile("filename");	Specifies the VCD file name
$dumpvars;	Dumps all signals in the design
$dumpvars(level, module_instance);	Dumps signals up to level hierarchy
$dumpon;	Starts signal dumping
$dumpoff;	Stops signal dumping
$dumpall;	Captures the current values of all signals



Thank you !

Happy Learning