SystemVerilog Assignment

Name: Shefin Muhammed MS

Systemverilog Lab3 : Interfaces & Basic assertions

1. Interfaces & Basic assertions

1.1. and if

Code:

```
1
2  // Interface Definition
3
4  interface and2_if;
5    logic a;
6    logic b;
7    logic y;
8
9    // Modports for direction control
10   modport dut (input a, b, output y);
11   modport tb  (output a, b, input y);
12 endinterface
13
14 // DUT Module
15
16 module and2 (and2_if.dut intf);
17   assign intf.y = intf.a & intf.b;
18 endmodule
19
20
21 // Testbench (Stimulus Generator)
22
23 module tb (and2_if.tb intf);
24   initial begin
25     $display("Time | a | b | y");
26     $monitor("%4t | %b | %b | %b", $time, intf.a, intf.b, intf.y);
27     #1 intf.a = 0; intf.b = 0;
28     #1 intf.a = 0; intf.b = 1;
29     #1 intf.a = 1; intf.b = 0;
30     #1 intf.a = 1; intf.b = 1;
31     #1 $finish;
32   end
33 endmodule
34
35
36 // Top-level Module
37
38 module tb_top;
39   and2_if dut_if();         // Create interface instance
40   and2    dut_inst(dut_if); // Connect DUT
41   tb      tb_inst(dut_if);  // Connect Testbench
42 endmodule
```

Output:

```
# KERNEL: Time | a | b | y
# KERNEL:    0 | x | x | x
# KERNEL:    1 | 0 | 0 | 0
# KERNEL:    2 | 0 | 1 | 0
# KERNEL:    3 | 1 | 0 | 0
# KERNEL:    4 | 1 | 1 | 1
```

1.2. diff if

Code:

```
// Code your design here
interface dff_if (input clk);
  logic d;
  logic q;

  clocking cb @(posedge clk);
    default input #1ns output #1ns;
    input q;
    output d;
  endclocking : cb

  modport dut (input d, input clk, output q);
  modport tb (clocking cb, input q, output d, input clk);
endinterface : dff_if

module dff (dff_if.dut intf);
  always @(posedge intf.clk) begin
    intf.q <= intf.d;
  end
endmodule

module top;
  bit clk = 0;

  initial
    forever #2 clk = ~clk;

  dff_if dut_if(clk);
  dff    dut  (dut_if);
  tb     tb_inst (dut_if);

endmodule

program tb(dff_if.tb intf);

  initial begin
    fork
      drv();
      mon();
    join
  end

  task drv;
    repeat(10) begin
      @(intf.cb);
      intf.d <= $random();
      $display("drv: d = %b", intf.d);
    end
  endtask

  task mon;
    repeat(10) begin
      @(intf.cb);
      $display("mon: q = %b", intf.q);
    end
```

```
56    endtask
57
58  endprogram
```

Output:

```
# KERNEL: drv: d = x
# KERNEL: mon: q = x
# KERNEL: drv: d = 0
# KERNEL: mon: q = 0
# KERNEL: drv: d = 1
# KERNEL: mon: q = 1
# KERNEL: drv: d = 1
# KERNEL: mon: q = 1
# KERNEL: drv: d = 1
# KERNEL: mon: q = 1
# KERNEL: drv: d = 1
# KERNEL: mon: q = 1
# KERNEL: drv: d = 1
# KERNEL: mon: q = 1
# KERNEL: drv: d = 1
# KERNEL: mon: q = 1
# KERNEL: drv: d = 1
# KERNEL: mon: q = 1
# KERNEL: drv: d = 0
# KERNEL: mon: q = 0
# KERNEL: drv: d = 1
# KERNEL: mon: q = 1
```