Nang Thiri Wutyi
20113
9/23/2024

Verilog Lab Week #1

1. "Hello World" in Verilog

```
//------------------------------------------------------

// This is my first Verilog Program

// Design Name : hello_world

// File Name : hello_world.v

// Function : This program will print 'hello world'

//------------------------------------------------------

module helloWorld;

 initial begin

                $display ("Hello World!!!");

                #10 $finish;

 end

endmodule // End of Module helloWorld
```

Results

```
[2024-09-24 06:35:47 UTC] iverilog '-Wall' '-g2012' design.sv testbench.sv  && unbuffer vvp a.out
Hello World!!!
design.sv:10: $finish called at 10 (1s)
Finding VCD file...
No *.vcd file found. EPWave will not open. Did you use '$dumpfile("dump.vcd"); $dumpvars;'?
Done
```

2. The digital circuit schematics based on the module gate

Main module "gate"

```
module gates(
   output wire out0, // Declare outputs
   output wire out1,
```

```
    output wire out2,
    input wire in1,   // Declare inputs
    input wire in2,
    input wire in3,
    input wire in4
);

    // Logic operations
    not U1(out0, in1);
    and U2(out1, in1, in2, in3, in4);
    xor U3(out2, in1, in2, in3);

endmodule
```

Testbench for gate

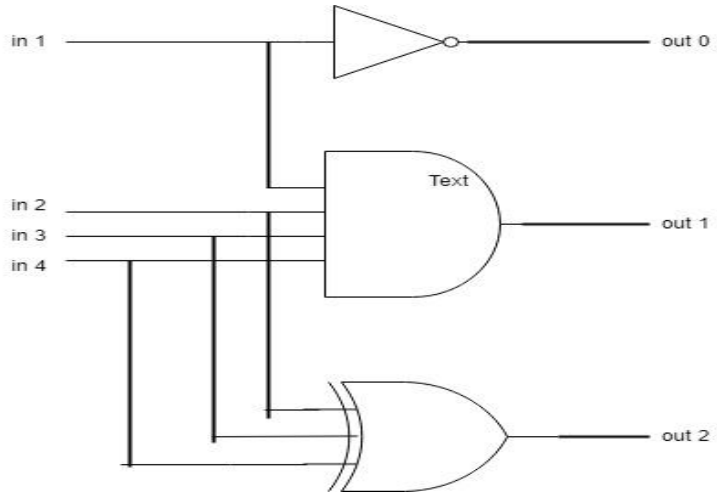
```
// Testbench for the "gates" module
module testbench();

    // Inputs to the gates module (reg type for testbench)
    reg in1, in2, in3, in4;

    // Outputs from the gates module (wire type)
    wire out0, out1, out2;

    // Instantiate the "gates" module
    gates uut (
        .out0(out0),
        .out1(out1),
        .out2(out2),
        .in1(in1),
        .in2(in2),
        .in3(in3),
        .in4(in4)
    );

    // Test logic
    initial begin
        // Display the header
        $display("in1 in2 in3 in4 | out0 out1 out2");
        $display("-----------------------------");
```

```
        // Apply different input combinations and display results
        in1 = 0; in2 = 0; in3 = 0; in4 = 0; #10;
        $display("%b   %b   %b   %b | %b   %b   %b", in1, in2, in3, in4, out0, out1, out2);

        in1 = 0; in2 = 1; in3 = 1; in4 = 0; #10;
        $display("%b   %b   %b   %b | %b   %b   %b", in1, in2, in3, in4, out0, out1, out2);

        in1 = 1; in2 = 1; in3 = 0; in4 = 1; #10;
        $display("%b   %b   %b   %b | %b   %b   %b", in1, in2, in3, in4, out0, out1, out2);

        in1 = 1; in2 = 1; in3 = 1; in4 = 1; #10;
        $display("%b   %b   %b   %b | %b   %b   %b", in1, in2, in3, in4, out0, out1, out2);

        $finish;
    end
endmodule
```

Results

```
[2024-09-24 17:33:14 UTC] iverilog '-Wall' '-g2012' design.sv testbench.sv  && unbuffer vvp a.out
in1 in2 in3 in4 | out0 out1 out2
-----------------------------
0   0   0   0   |  1    0    0
0   1   1   0   |  1    0    0
1   1   0   1   |  0    0    0
1   1   1   1   |  0    1    1
testbench.sv:40: $finish called at 40 (1s)
Done
```

Digital circuit schematics



3. **Main Module for oneBitFA1 (Continuous Assignment)**

```
module oneBitFA1(
    input wire a,   // Input bit a
    input wire b,   // Input bit b
    input wire ci,  // Input carry-in
    output wire co, // Output carry-out
    output wire sum // Output sum
);
    // Continuous assignment to calculate sum and carry-out
    assign {co, sum} = a + b + ci;

endmodule
```

Testbench for oneBitFA1

```
module tb_oneBitFA1;
    // Inputs
    reg a, b, ci;
    // Outputs
    wire co, sum;

    // Instantiate the full adder module
    oneBitFA1 uut (
        .a(a),
        .b(b),
        .ci(ci),
        .co(co),
        .sum(sum)
    );

    // Initial block to apply inputs
    initial begin
        // Apply different combinations of inputs
        a = 0; b = 0; ci = 0;
        #10 a = 0; b = 0; ci = 1;
        #10 a = 0; b = 1; ci = 0;
        #10 a = 0; b = 1; ci = 1;
```

```
        #10 a = 1; b = 0; ci = 0;
        #10 a = 1; b = 0; ci = 1;
        #10 a = 1; b = 1; ci = 0;
        #10 a = 1; b = 1; ci = 1;
        #10 $finish;  // End the simulation
    end

    // Monitor to display the results
    initial begin
        $monitor("At time %0d: a = %b, b = %b, ci = %b | sum = %b, co = %b", $time, a, b,
ci, sum, co);
    end
endmodule
```

Results

```
[2024-09-24 07:39:03 UTC] iverilog '-Wall' '-g2012' design.sv testbench.sv  && unbuffer vvp a.out
At time 0: a = 0, b = 0, ci = 0 | sum = 0, co = 0
At time 10: a = 0, b = 0, ci = 1 | sum = 1, co = 0
At time 20: a = 0, b = 1, ci = 0 | sum = 1, co = 0
At time 30: a = 0, b = 1, ci = 1 | sum = 0, co = 1
At time 40: a = 1, b = 0, ci = 0 | sum = 1, co = 0
At time 50: a = 1, b = 0, ci = 1 | sum = 0, co = 1
At time 60: a = 1, b = 1, ci = 0 | sum = 0, co = 1
At time 70: a = 1, b = 1, ci = 1 | sum = 1, co = 1
testbench.sv:27: $finish called at 80 (1s)
```

Main Module for oneBitFA2 (1-bit Full Adder using Always Block)

```
module oneBitFA2 (
    input wire a_i,
    input wire b_i,
    input wire ci_i,
    output reg sum_o,
    output reg co_o
);

    // Always block for sum and carry-out
    always @(*) begin
        {co_o, sum_o} = a_i + b_i + ci_i;
    end
endmodule
```

Testbench for oneBitFA2

```
module tb_oneBitFA2;

    // Declare testbench inputs and outputs
    reg a_i, b_i, ci_i;
    wire sum_o, co_o;

    // Instantiate the oneBitFA2 module
    oneBitFA2 uut (
        .a_i(a_i),
        .b_i(b_i),
        .ci_i(ci_i),
        .sum_o(sum_o),
        .co_o(co_o)
    );

    // Testbench procedure
    initial begin
        $display("Testing 1-bit full adder using always block (oneBitFA2)...");
        $monitor("a_i=%b, b_i=%b, ci_i=%b -> sum_o=%b, co_o=%b", a_i, b_i, ci_i,
sum_o, co_o);

        // Test case 1
        a_i = 0; b_i = 0; ci_i = 0;
        #10;

        // Test case 2
        a_i = 0; b_i = 0; ci_i = 1;
        #10;

        // Test case 3
        a_i = 0; b_i = 1; ci_i = 0;
        #10;

        // Test case 4
        a_i = 0; b_i = 1; ci_i = 1;
        #10;

        // Test case 5
        a_i = 1; b_i = 0; ci_i = 0;
        #10;
```

```
        // Test case 6
        a_i = 1; b_i = 0; ci_i = 1;
        #10;

        // Test case 7
        a_i = 1; b_i = 1; ci_i = 0;
        #10;

        // Test case 8
        a_i = 1; b_i = 1; ci_i = 1;
        #10;

        // End simulation
        $finish;
    end
endmodule
```

Results

```
[2024-09-24 07:44:53 UTC] iverilog '-Wall' '-g2012' design.sv testbench.sv  && unbuffer vvp a.out
At time 0: a = 0, b = 0, ci = 0 | sum = 0, co = 0
At time 10: a = 0, b = 0, ci = 1 | sum = 1, co = 0
At time 20: a = 0, b = 1, ci = 0 | sum = 1, co = 0
At time 30: a = 0, b = 1, ci = 1 | sum = 0, co = 1
At time 40: a = 1, b = 0, ci = 0 | sum = 1, co = 0
At time 50: a = 1, b = 0, ci = 1 | sum = 0, co = 1
At time 60: a = 1, b = 1, ci = 0 | sum = 0, co = 1
At time 70: a = 1, b = 1, ci = 1 | sum = 1, co = 1
testbench.sv:27: $finish called at 80 (1s)
Done
```