LAB REPORT
CSE4010 Computer Architecture
Instructor: Lawrence Orijuela

Name:_____Cameron Hartling_____ SCORE: ___/30

Student ID: _____007735059_____ DUE: __9/19/2024__

LAB 1

Report

Verilog is a hardware description language that is used for electronic design. It is used for simulation, timing analysis, test analysis, and logic synthesis.

A module in Verilog is a block of code that implements a certain functionality. A module acts similarly to a class in C++. A testbench in Verilog is used for the verification of the digital hardware design. It is used to mainly verify the timing and functionality of the design.

In this lab I have installed Verilog and VS Code and I have compiled and run two different Verilog testbenches and their associated modules. The module in part A gets an input wire A and sets wire B equal to wire A and wire C equal to the inverse of wire A. The testbench then alternates the value/current of wire A every 20 nanosecond for 80 nanoseconds. The waveform shows that the program runs correctly. The module in part B gets two input wires W and X and sets wire Y equal to the opposite of wire X and wire Z equal to the opposite of wire Z. The testbench then alternates wires X every 40 nanoseconds and wire W every 40 nanoseconds starting at 20 nanoseconds. When wire W changes there are no changes in the output wires and when wire X changes, both the output wires change. This shows that the module is functioning correctly.

Source Code for Parts A and B

Part A

wireTest.v

```
// Create a module called wireTest with wires named A, B, and C
module wireTest (A,B,C);

    input A;  // Wire A is an input wire
    output B; // Wire B is an output wire
    output C; // Wire C is an output wire

    assign B = A;  // Wire B has the same value/charge of A
    assign C = !A; // Wire C has the opposite value/charge of A

endmodule // End the module called wireTest
```

wireTest_tb.v

```
`timescale 1ns / 1ns   // Sets time interval of wireTest
`include "wireTest.v"  // Links wireTest.v to this file

module wireTest_tb; // Create module called wireTest_tb

reg A;  // Input wire called A
wire B; // Output wire called B
wire C; // Output wire called C

wireTest uut(A, B, C); // Instantiate wireTest as uut

initial begin // Starts clock

    $dumpfile("wireTest_tb.vcd"); // Sets file for output of the test
```

```
    $dumpvars(0, wireTest_tb);

    A = 0; // Set wire A to 0
    #20 // Wait 20 nanoseconds

    A = 1; // Set wire A to 1
    #20 // Wait 20 nanoseconds

    A = 0; // Set wire A to 0
    #20 // Wait 20 nanoseconds

    A = 1; // Set wire A to 1
    #20 // Wait 20 nanoseconds

    $display("Wire Test Complete!"); // Print the string

end // Ends clock

endmodule // Ends wireTest_tb module
```

Part B

wireTest2.v

```
// Create a module called wireTest2 with wires named A, B, and C
module wireTest2 (W, X, Y, Z);
    // Input wires
    input W;
    input X;
    // Output wires
    output Y;
    output Z;
```

```
    assign Y = !X;  // Wire Y has the same value/charge of X
    assign Z = !Y; // Wire Z has the opposite value/charge of Y

endmodule // End the module called wireTest2
```

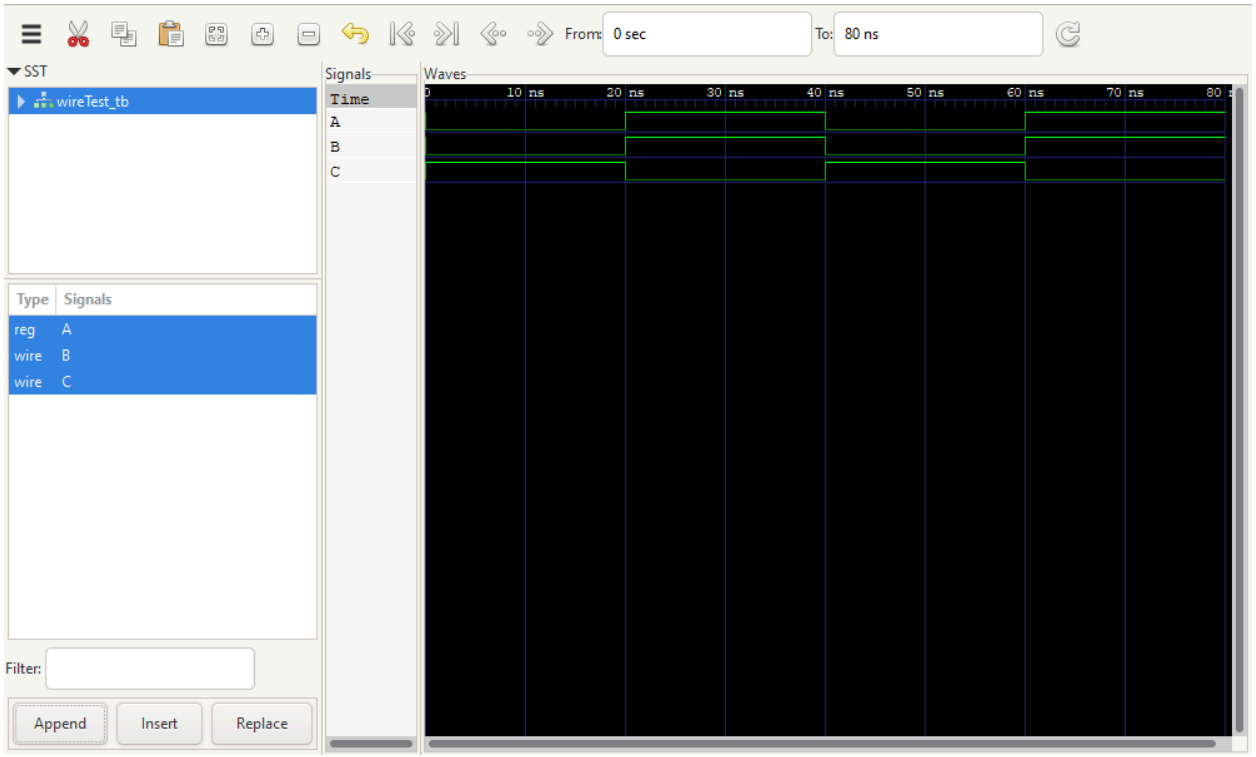
wireTest2_tb.v

```
`timescale 1ns / 1ns   // Sets time interval of wireTest
`include "wireTest2.v"  // Links wireTest2.v to this file

module wireTest2_tb; // Create module called wireTest2_tb

reg W;  // Input wire called W
reg X;  // Input wire called X
wire Y; // Output wire called Y
wire Z; // Output wire called Z

wireTest2 uut(W, X, Y, Z); // Instantiate wireTest as uut

initial begin // Starts clock

    $dumpfile("wireTest2_tb.vcd"); // Sets file for output of the test
    $dumpvars(0, wireTest2_tb);

    W = 0; // Set wire W to 0
    X = 0; // Set wire X to 0
    #20 // Wait 20 nanoseconds

    W = 1; // Set wire W to 1
```

```
    #20 // Wait 20 nanoseconds

    X = 1; // Set wire X to 1
    #20 // Wait 20 nanoseconds

    W = 0; // Set wire W to 0
    #20 // Wait 20 nanoseconds

    X = 0; // Set wire X to 0

    $display("Wire Test Complete!"); // Print the string

end // Ends clock

endmodule // Ends wireTest2_tb module
```

Screenshots for Parts A and B

Part A



Part B