George Suarez

CSE 310

Homework 5

1. What are the four possible output values of a 1-bit binary variable in Verilog?

* Logic 0
* Logic 1
* Unknown Logic Number x
* Higher Impendence Number z

1. Which of the statements about Verilog is/are true?

* In Verilog, one module can instantiate other modules, and can have multiple instances of another module.
* *always @ \** can be used in describing Combinational logic.
* *always @ \** can be used in describing memory made of flip-flops and registers.
* *always @ \** can be used preceding a *case* statement.

1. 2x1 Mux Module

module mux\_2to1( input wire din\_0, input wire din\_1, input wire S,

input wire E, output wire out);

wire and1, and2, not1;

assign not1 = !S;

assign and1 = n1 & din\_0 & E;

assign and2 = din\_1 & S & E;

assign out = a1 | a2;

endmodule

4x1 Mux Module

module mux\_4to1( input wire [3:0] I, input wire [1:0] S, input wire E,

output wire y);

//put internal wires here

wire w1, w2;

//create instances of mux\_2to1 for connections

mux\_2to1 ( w1, I[0], I[1], S[0], E );

mux\_2to1 ( w2, I[2], I[3], S[1], E );

mux\_2to1 ( y, w1, w2, S[1], E );

endmodule //End Of Module mux\_4to1

1. The following Verilog code is a behavioral model of a 4-to-1-line multiplexer and the stimulus. Fill in the missing code underlined.

//Design module of 4-to-1-line MUX

module mux4\_1\_bh (I, select, y); //I is data input, select is selection lines, y is output

input [3:0] I;

input [1:0] select;

output reg y; //y is of data type reg

always @ ( select[0] or select[1])

case ( select )

2'b00: y = I[0];

2'b01: y = I[1];

2'b10: y = I[2];

2'b11: y = I[3];

endcase

endmodule

//stimulus

module test\_mux ;

input [3: 0] D;

input [1:0] S;

output Y;

//instantiate mux4\_1\_bh

mux4\_1\_bh uut ( .I(D), .S(select), .y(Y) );

//start simulation, provide stimulus to the module under test

initial

begin

D = 4'b0101;

S = 2'b00;

repeat (3)

initial

begin

#10 S = S + 1; //increase S by 1

end

initial

begin

//monitor D, S, Y

$monitor(“%b \t %b \t %b, D, S, Y);

end

endmodule

1. Textbook 7.11: Obtain the 15-bit Hamming code word for the 11-bit data word 11001001010.

Position: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Notation:

Data: \_ \_ 1 \_ 1 0 0 \_ 1 0 0 1 0 1 0

Parity Data:

= 0

Hamming Code: 101110011001010

1. Textbook 7.10: Given the 8-bit data word 01011011, generate the 13-bit composite word for the Hamming code that corrects single errors and detects double errors.

Position: 1 2 3 4 5 6 7 8 9 10 11 12 13

Notation:

Data: \_ \_ 0 \_ 1 0 1 \_ 1 0 1 1 \_

Parity:

Hamming Code: 0001101110111