**Battleship Game**

**Testbench of the Code:**

`timescale 1ns / 1ps

module tb\_top;

// Inputs

reg clk;

reg [3:0] sw;

reg [3:0] btn;

// Outputs

wire [7:0] led;

wire [7:0] seven0;

wire [7:0] seven1;

wire [7:0] seven2;

wire [7:0] seven3;

// Instantiate the Unit Under Test (UUT)

top uut (

.clk(clk),

.sw(sw),

.btn(btn),

.led(led),

.seven0(seven0),

.seven1(seven1),

.seven2(seven2),

.seven3(seven3)

);

// Clock generation

initial begin

clk = 0;

forever #5 clk = ~clk; // 100 MHz clock (period = 10 ns)

end

// Stimulus generation

initial begin

// Initialize inputs

sw = 4'b0000;

btn = 4'b0000;

// Wait for global reset to finish

#50;

// Apply test cases

// reset button

sw = 4'b0000; btn = 4'b0100; #500;

btn = 4'b0000;

#5000;

// start button

sw = 4'b0000; btn = 4'b0010; #500;

btn = 4'b0000;

// waiting for 1 second

#11300;

// A input section //

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// A input 1

sw = 4'b0011; btn = 4'b1000; #500;

btn = 4'b0000;

#5000;

// A input 2

sw = 4'b0101; btn = 4'b1000; #500;

btn = 4'b0000;

#5000;

// A input 3

sw = 4'b0111; btn = 4'b1000; #500;

btn = 4'b0000;

#5000;

// A input 4 but matching with 2

sw = 4'b0101; btn = 4'b1000; #500;

btn = 4'b0000;

#11300;

// A input 5

sw = 4'b1101; btn = 4'b1000; #500;

btn = 4'b0000;

#11400;

// B input section //

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// B input 1

sw = 4'b0010; btn = 4'b0001; #500;

btn = 4'b0000;

#5000;

// B input 2

sw = 4'b0101; btn = 4'b0001; #500;

btn = 4'b0000;

#5000;

// B input 3

sw = 4'b1111; btn = 4'b0001; #500;

btn = 4'b0000;

#5000;

// B input 4

sw = 4'b0101; btn = 4'b0001; #500;

btn = 4'b0000;

#11300;

//B input 5

sw = 4'b1101; btn = 4'b0001; #500;

btn = 4'b0000;

#5000;

// sw = 4'b1101; btn = 4'b0001; #500;

// btn = 4'b0000;

#11300;

// A shoot section

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// A shoots B ship 1

sw = 4'b0010; btn = 4'b1000; #500;

btn = 4'b0000;

#11300;

// B shoots empty space

sw = 4'b0001; btn = 4'b0001; #500;

btn = 4'b0000;

#11300;

// A shoots B ship 2

sw = 4'b0101; btn = 4'b1000; #500;

btn = 4'b0000;

#11300;

// B shoots A ship 1

sw = 4'b0011; btn = 4'b0001; #500;

btn = 4'b0000;

#11300;

// A shoots B ship 3

sw = 4'b1111; btn = 4'b1000; #500;

btn = 4'b0000;

#11300;

// B shoots A ship 2

sw = 4'b0101; btn = 4'b0001; #500;

btn = 4'b0000;

#11300;

//A shoots B ship 4 and wins

sw = 4'b1101; btn = 4'b1000; #500;

btn = 4'b0000;

#11300;

#20000;

// sw = 4'b1010; btn = 4'b1000; #500;

// btn = 4'b0000;

// #5000;

// sw = 4'b0101; btn = 4'b1000; #500;

// btn = 4'b0000;

// #5000;

// Return to default

sw = 4'b0000; btn = 4'b0000; #50;

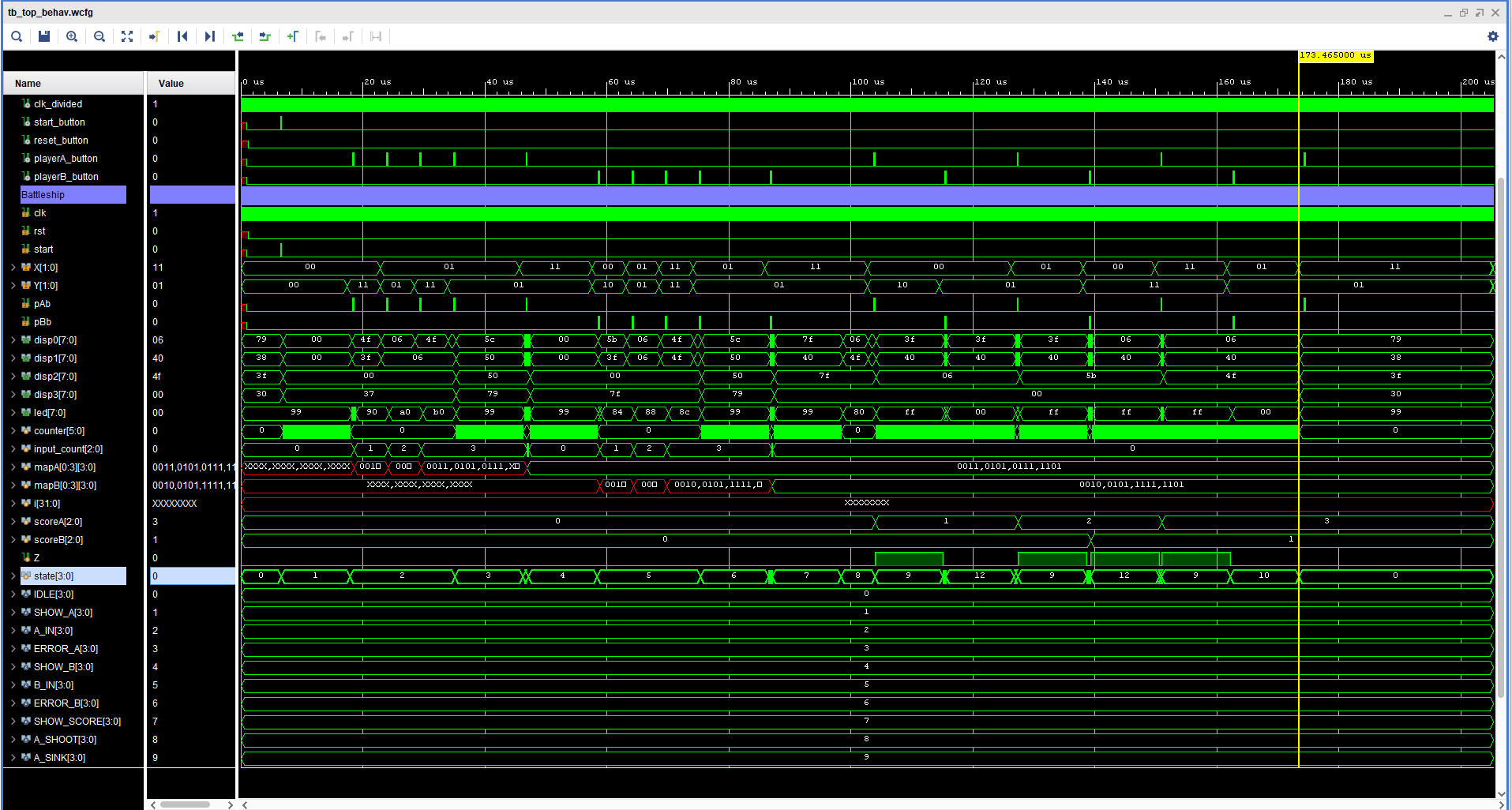
// Finish simulation

$stop;

end

endmodule

**Waveform Diagram:**



Explanation of the waveform:

I have tested the functionality of the documents. The following is the flow of the

* A start button is pressed. Which starts the state machine and starts taking input.
* A takes four unique inputs. The error functionality is also tested on the test bench.
* The 2nd and 4th input are same which prompts error screen and asks to enter A again until four inputs are completed.
* The same functionality is also tested on B inputs.
* After 4 inputs are inserted, they are stored in register memory and ready to be compared.
* Then, shooting battle commences and A and B players take turn. A player takes the first turn.
* In this testbench. A shoots B ship 1 which increases scoreA.
* 2nd B first shot misses and scoreB remains zero.
* Then A and B shoots each other.
* Until A shoots all 4 ships and A wins.