Computer Architecture HW2.

(Jake) YoonTae Jung

1) Test-benches

a) Decoder testbench

```
32  module testDecoder;
33  reg addr0, addr1;
34  reg enable;
35  wire out0,out1,out2,out3;
36  //behavioralDecoder decoder (out0,out1,out2,out3,addr0,addr1,enable);
37  structuralDecoder decoder (out0,out1,out2,out3,addr0,addr1,enable); // Swap after testing
38
39  initial begin
40  $display("En A0 A1| O0 O1 O2 O3 | Expected Output");
41  enable=0;addr0=0;addr1=0; #1000
42  $display("%b  %b  %b |  %b  %b  %b  %b | All false", enable, addr0, addr1, out0, out1, out2, out3);
43  enable=0;addr0=1;addr1=0; #1000
44  $display("%b  %b  %b |  %b  %b  %b  %b | All false", enable, addr0, addr1, out0, out1, out2, out3);
45  enable=0;addr0=0;addr1=1; #1000
46  $display("%b  %b  %b |  %b  %b  %b  %b | All false", enable, addr0, addr1, out0, out1, out2, out3);
47  enable=0;addr0=1;addr1=1; #1000
48  $display("%b  %b  %b |  %b  %b  %b  %b | All false", enable, addr0, addr1, out0, out1, out2, out3);
49  enable=1;addr0=0;addr1=0; #1000
50  $display("%b  %b  %b |  %b  %b  %b  %b | O0 Only", enable, addr0, addr1, out0, out1, out2, out3);
51  enable=1;addr0=1;addr1=0; #1000
52  $display("%b  %b  %b |  %b  %b  %b  %b | O1 Only", enable, addr0, addr1, out0, out1, out2, out3);
53  enable=1;addr0=0;addr1=1; #1000
54  $display("%b  %b  %b |  %b  %b  %b  %b | O2 Only", enable, addr0, addr1, out0, out1, out2, out3);
55  enable=1;addr0=1;addr1=1; #1000
56  $display("%b  %b  %b |  %b  %b  %b  %b | O3 Only", enable, addr0, addr1, out0, out1, out2, out3);
57  end
58  endmodule
59
```

Test bench of decoder was offered by the professor and it shows the truth table by changing the enable and inputs.

b) Multiplexer test-bench

```
32  module testMultiplexer;
33  reg in0, in1, in2, in3;
34  reg addr0, addr1;
35  wire out;
36  //behavioralMultiplexer mux(out, addr0,addr1, in0,in1,in2,in3);
37  structuralMultiplexer mux(out, addr0,addr1, in0,in1,in2,in3);
38
39  initial begin
40  $display("A0 A1| In0  In1  In2  In3 | Expected Output | Output");
41  addr0=0;addr1=0; in0=1;in1=0;in2=0;in3=0;  #1000
42  $display("%b  %b|  %b  %b  %b  %b |      In0      |  %b", addr0, addr1, in0, in1, in2, in3, out);
43  addr0=1;addr1=0; in0=0;in1=1;in2=0;in3=0;  #1000
44  $display("%b  %b|  %b  %b  %b  %b |      In1      |  %b", addr0, addr1, in0, in1, in2, in3, out);
45  addr0=0;addr1=1; in0=0;in1=0;in2=1;in3=0;  #1000
46  $display("%b  %b|  %b  %b  %b  %b |      In2      |  %b", addr0, addr1, in0, in1, in2, in3, out);
47  addr0=1;addr1=1; in0=0;in1=0;in2=0;in3=1;  #1000
48  $display("%b  %b|  %b  %b  %b  %b |      In3      |  %b", addr0, addr1, in0, in1, in2, in3, out);
49  end
50  endmodule
51
```

The 4bit-Mux selects one input among 4 inputs by using selectors. There are 4 cases of 2 bit selectors and each of the case represents one output.

When (addr0, addr1)=(0,0) the output becomes In0

(addr0,addr1)=(1,0) → output: In1 , (addr0, addr1)=(0,1) →output: In2, (addr0, addr1)=(1,1) → In3

To show this I mad expected output to be HIGH and the others to LOW. Therefore only one input is HIGH and the others are LOW in the truth table.

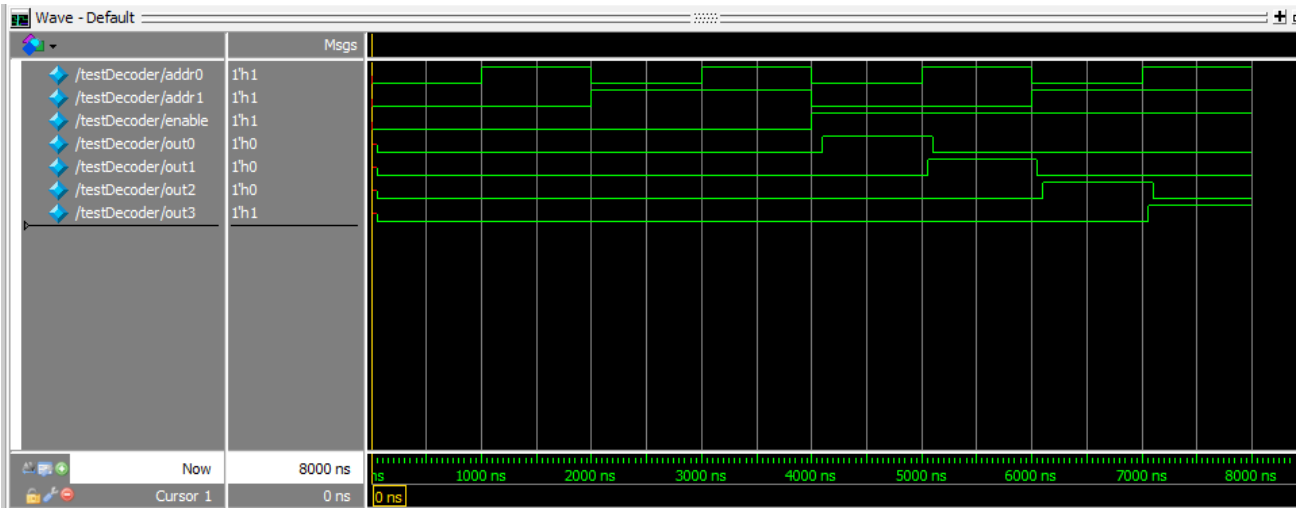
c) FullAdder test-bench

```
26  module testFullAdder;
27  reg a, b, carryin;
28  wire sum, carryout;
29  //behavioralFullAdder adder (sum, carryout, a, b, carryin);
30  structuralFullAdder adder(sum, carryout, a, b, carryin);
31
32  initial begin
33  $display("A  B Carryin| Carryout Sum");
34  a=0;b=0; carryin=0; #1000
35  $display("%b  %b   %b    |     %b     %b", a, b, carryin, carryout, sum);
36  a=0;b=1; carryin=0; #1000
37  $display("%b  %b   %b    |     %b     %b", a, b, carryin, carryout, sum);
38  a=1;b=0; carryin=0; #1000
39  $display("%b  %b   %b    |     %b     %b", a, b, carryin, carryout, sum);
40  a=1;b=1; carryin=0; #1000
41  $display("%b  %b   %b    |     %b     %b", a, b, carryin, carryout, sum);
42  a=0;b=0; carryin=1; #1000
43  $display("%b  %b   %b    |     %b     %b", a, b, carryin, carryout, sum);
44  a=0;b=1; carryin=1; #1000
45  $display("%b  %b   %b    |     %b     %b", a, b, carryin, carryout, sum);
46  a=1;b=0; carryin=1; #1000
47  $display("%b  %b   %b    |     %b     %b", a, b, carryin, carryout, sum);
48  a=1;b=1; carryin=1; #1000
49  $display("%b  %b   %b    |     %b     %b", a, b, carryin, carryout, sum);
50
51  end
52  endmodule
53
```

The structuralFullAdder get inputs a, b, carryin and outputs sum and carryout. To show the truth table I designed the all cases of 8 and showed all inputs and outputs from the FullAdder.

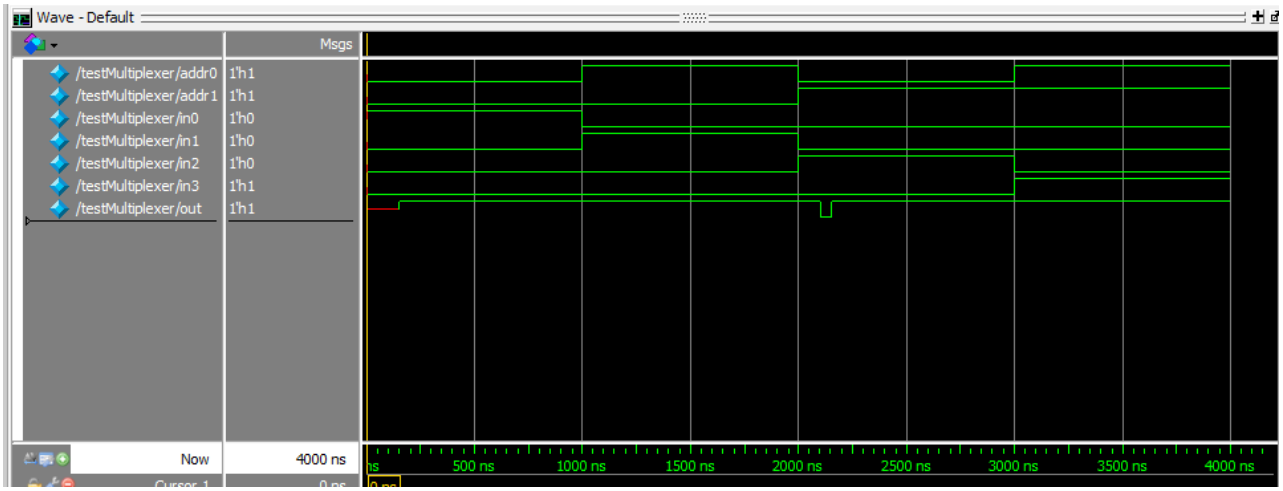
2) Waveform(propagation delay)

a) Decoder waveform



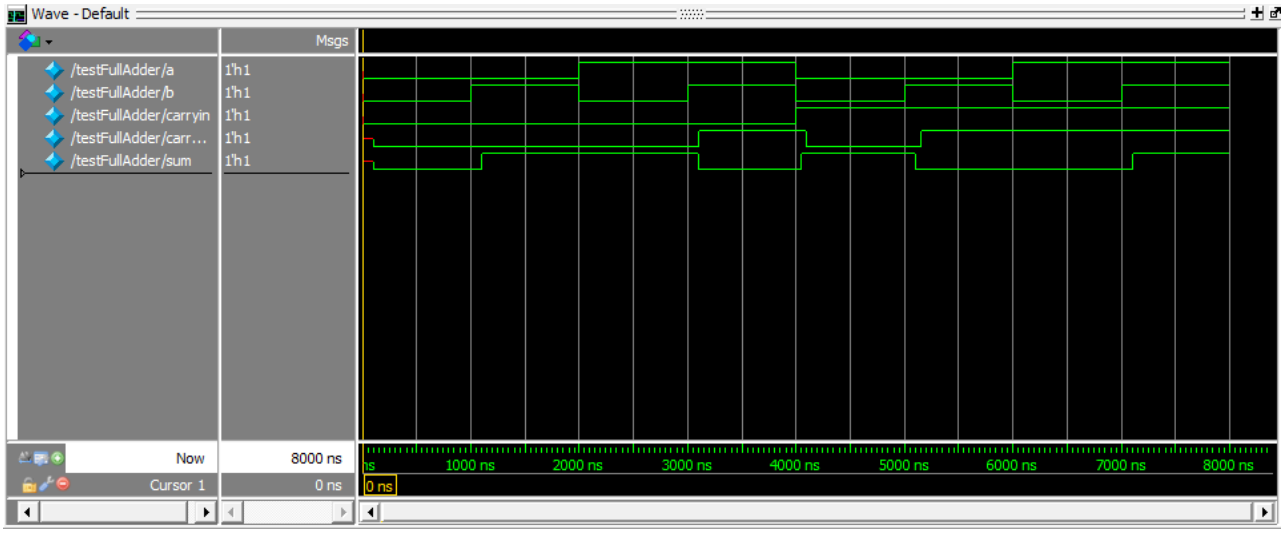
When the enable is LOW the outputs don't change and when the enable is high the outputs start to change. The outputs react to each combinations of addr0 and addr1. At the combination of (addr0, addr1)=(0, 0) after 50ms of propagation delay out0 went up to high. Other combinations of addresses also made the output to change after the propagation delays.

b) Mux waveform



The waveforms match with the truth tables of mux. However there exists a propagation delay in the gates, so the output changes little bit later after the address changes. When we check the first output of (addr0, addr1) = (0, 0) it goes up after the delay and also after 2000ns when the addr1 changes from 0 to 1 the output show glitch because there is a mismatch of because there is additional inverter attached to addr0.

c) FullAdder waveform



The waveform matches with the truth table and because of the propagation delays in the gates. The carryout and the sum comes out after delays.