Lab 1: 1-bit ALU and 4-bit ALU

Due: 18:30, September 27, 2016

Objective

- To be familiar with structural modeling, data flow modeling, behavioral modeling, and module instantiation.

Action Items

1. Write a Verilog module that models a 1-bit ALU with and, or, xor and addition functions using gate primitives (structural modeling) and the file mux4_to_1.v, and test your module using the testbench file lab1_1_t.v.

   You have to use the following template for your design.

```
module lab1_1(a, b, c, aluctr, d, e);
input a, b, c;
input [1:0] aluctr;
output d, e;
   // add your design here
Endmodule
```

aluctr[1]	aluctr[0]	behavior
0	0	{e, d} = a + b + c

0	1	d = a and b
		e = 0

1	0	d = a or b
		e = 0

1	1	d = a xor b
		e = 0

2. Re-write the module using continuous assignments (data flow modeling) and re-test your module using the testbench file lab1_2_t.v.

You have to use the following template for your design.

```
module lab1_2(a, b, c, aluctr, d, e);
input a, b, c;
input [1:0] aluctr;
output d, e;
   // add your design here
endmodule
```

3. Re-write the module using behavioral modeling and re-test your module using the testbench file lab1_3_t.v.

   You have to use the following template for your design.

```
module lab1_3(a, b, c, aluctr, d, e);
input a, b, c;
input [1:0] aluctr;
output d, e;
   // add your design here
endmodule
```

4. Write a Verilog module that models a 4-bit ALU and test your module using the testbench file lab1_4_t.v. The 4-bit ALU must be implemented by using four 1-bit ALUs (i.e., instances of previous modules) and necessary interconnects. The addition function is designed for unsigned addition.

   You have to use the following template for your design.

```
module lab1_4(a, b, c, aluctr, d, e);
input [3:0] a,b;
input [1:0] aluctr;
input c;
output [3:0] d;
output e;
   // add your design here
endmodule
```

✓ Attention

1. When you are doing the simulation of lab1_4, you have to change your runtime to 10000ns in “Simulation Settings” before you run simulation.
2. You can add *$monitor* in your testbench to show all the informations of your inputs and outputs when you run simulation.