16-bit ALU

DDCO Assignment 1

August 24, 2017

The ALU (Arithmetic and Logic Unit) is the heart of a CPU since it performs the arithmetic and logic operations which are the key computations. Your task in this assignment is to design and simulate a 16-bit[1] ALU which performs the following core operations:

- **Arithmetic** Addition and subtraction (two's complement)
- **Logic** And and or operations

Each operation would receive as input two 16-bit operands and the output would be a 16-bit result (along with carry and overflow). Which of the four operations is to be performed would be indicated by a two bit operation input. So the ALU module inputs are:

- **i0[15:0]** 16-bit operand 0
- **i1[15:0]** 16-bit operand 1
- **op[1:0]** 2-bit operation code

Operation code meaning:

op[1:0]	operation
00	add
01	subtract
10	and
11	or

The ALU module outputs should be:

- **o[15:0]** 16-bit operation result
- **carry** Carry out of MSB
- **overflow** Arithmetic overflow

During logic operations the values of the carry and overflow outputs are irrelevant.

[1] Instead of a 16-bit ALU, it is reasonably straightforward to make a 32-bit or even 64-bit ALU. But in order to keep wiring requirements manageable, we stick to 16-bits.

1 Design

Once you have designed the ALU logic, it needs to be specified using the Verilog syntax discussed in class in the file alu.v. The entire ALU should be composed of purely combinational logic elements that have been discussed in class. Also, only the Verilog module definition and instantiation syntax discussed in class should be used.

2 Simulate

Once the ALU is designed, it needs to be simulated to verify proper functionality of its various operations. In order to do so, appropriate inputs need to be applied to the ALU. The inputs are specified as test vectors, four of which are specified as samples on lines 14 to 17 of tb_alu.v. The samples test only the arithmetic functionality. Additional test vectors need to be added not only for logic but arithmetic operations as well. Be sure to change value of TESTVECS on line 2 whenever the number of test vectors is modified.

To simulate the ALU with the test vectors, run the commands:

```
iverilog -o tb_alu lib.v alu.v tb_alu.v
vvp tb_alu
```

Above commands should produce the tb_alu.vcd file. To view the waveforms, run the command:

```
gtkwave tb_alu.vcd
```

For more gtkwave info see the README file in a0.zip sent earlier.