Lab 2: Implementation and verification of combination logic

The deadline for this lab is Sunday, February 20, 11:59PM. **Late submissions are not accepted**.

Objective

The purpose of this laboratory exercise is to let you start designing your own simple combinational logic in Verilog. The designs that you will implement in this lab will be re-used in the future laboratory exercises. After completion of this lab you should:

- get your hands dirty by designing some of the combinational logic that we discussed in the lecture;
- be more confident with the toolflow;
- design hardware modules using behavioral description;

Lab Instructions

Part 1a: Implement 8-bit 10-to-1 Multiplexer in Verilog.
As we discussed in the lecture, multiplexer is a combinational logic that based on the input *select* signal, choses one of the inputs and routes it to the single output port.

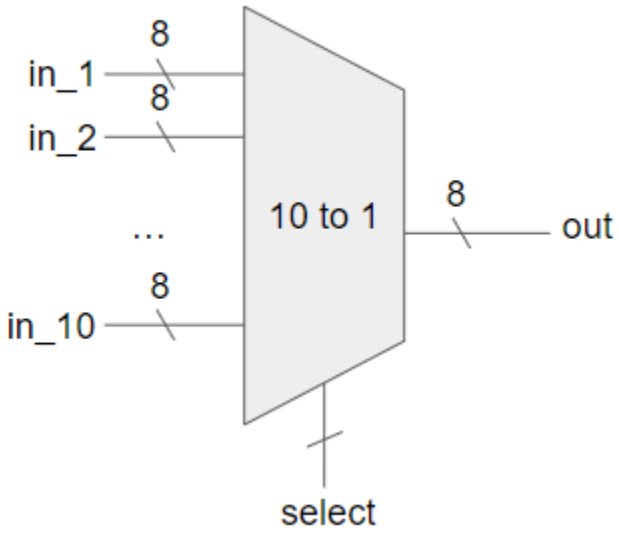

Figure 1 - 8 bit 10-to-1 Multiplexer

Part 1b: Verify the mux.
Write a C++ testbench and build the simulator using Verilator to test the mux with 100000 random tests. Dump the VCD waveform and use GTKWave if debugging is needed.

Part 2a: Implement a simple 8-bit Arithmetic Logic Unit (ALU) in Verilog.
Below is the port level schematic of the ALU that we will be designing.

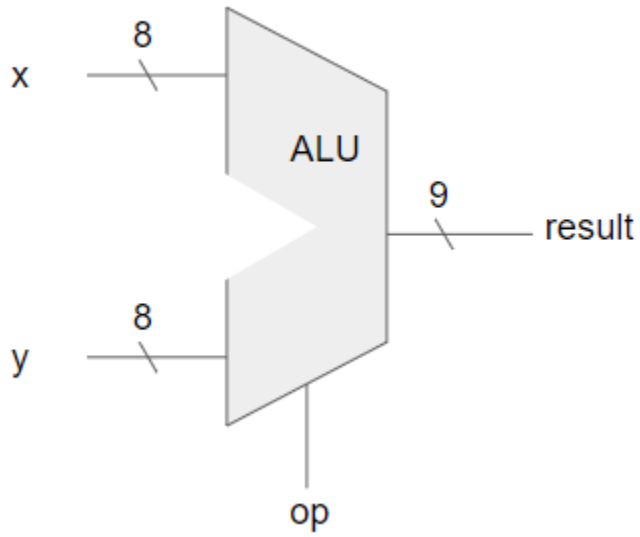

Figure 2 - 8 bit ALU

This ALU accepts three inputs: x, y and op. Based on the op signal the ALU will either add or subtract the x and y. If op == 0, result = x+y, otherwise result = x-y.

Part 2b: Verify the ALU.
Write a C++ testbench and build the simulator using Verilator to test the ALU. Create a testing infrastructure that will test the ALU for all possible values of x, y and y. Dump the VCD waveform and use GTKWave if debugging is needed.

Submission

For successful completion of this laboratory exercise you will need to do the following:

1) Sign-off all parts of the lab. In other words, you will have to demonstrate to me that all parts of this lab have been completed. The demonstration can be done live one-on-one during the office hours. Or alternatively, you could record a video, upload it to YouTube and share a link with me.
2) The code must be submitted into a private GitHub repository. I should be added as a collaborator to view your code. You can use the same repo that you used previously.

The demonstration should be done before the deadline. If demonstration will be done via recorded video, upload the video to any platform and share the link with me before the deadline.

Good luck!