CSE4372/5392 (Spring 2025)
Lab #8

This lab is due by April 15, with a 10% penalty per week day for being late.

In this lab, you will add support for reading and writing memory.

The following steps will guide you through this process:

1. Start with the project from Lab 7.

2. Add these signals to the rv32_mem_top port list:

```
 // memory interface
 output [31:2] memif_addr,
 input [31:0] memif_rdata,
 output memif_we,
 output [3:0] memif_be,
 output [31:0] memif_wdata,

// io interface
output [31:2] io_addr,
input [31:0] io_rdata,
output io_we,
output [3:0] io_be,
output [31:0] io_wdata,
```

3. Connect the memory interface signals to the data port of the memory module (from Lab 4) in the project top.

4. Design an I/O module that contains at least the free push button and some LEDs in the I/O space. For reads, register the output as in the memory module. Connect the i/o interface signals of mem_top to the I/O module in the project top.

5. Route the ALU data from the EX stage to both the memif_addr and io_addr outputs.

6. Generate the memif_we, io_we, memif_be, and io_be control signals to drive the memory and i/o interface outputs. When a store operation occurs, io_we is active when A31 is high and memif_we is active when A31 is low. The byte enable signals can be identical for memory and i/o operations.

7. Route the rs2 data from the ID stage to both memif_wdata and io_wdata.

8. Make both memif_rdata and io_rdata available for use in the WB stage (they will already be registered by the dual port memory and i/o module).

9. Modify the wb_enable signal generation in the ID stage to also assert with load instructions. Generate a we signal in the ID stage and register through the stages to inform the MEM stage that a write to memory or I/O will occur.

10. Generate a control signal that informs the WB stage of the source for register writes -- a register value in alu_out (existing from Lab 5), or a memory value from memif_rdata, or an i/o value from io_rdata.

11. Demonstrate operation of load and store instructions from/to memory and i/o through the pipeline showing the expected register results to the TA.

When you write your test program, you must place NOPs between a load instruction and a subsequent instruction using the same register (this will be fixed in Lab 9).

12. Note the I/O part is required for CSE 5392 students and optional for everyone else.

13. Send your Verilog source .v or .sv files (don't send the entire project) to the TA for credit.