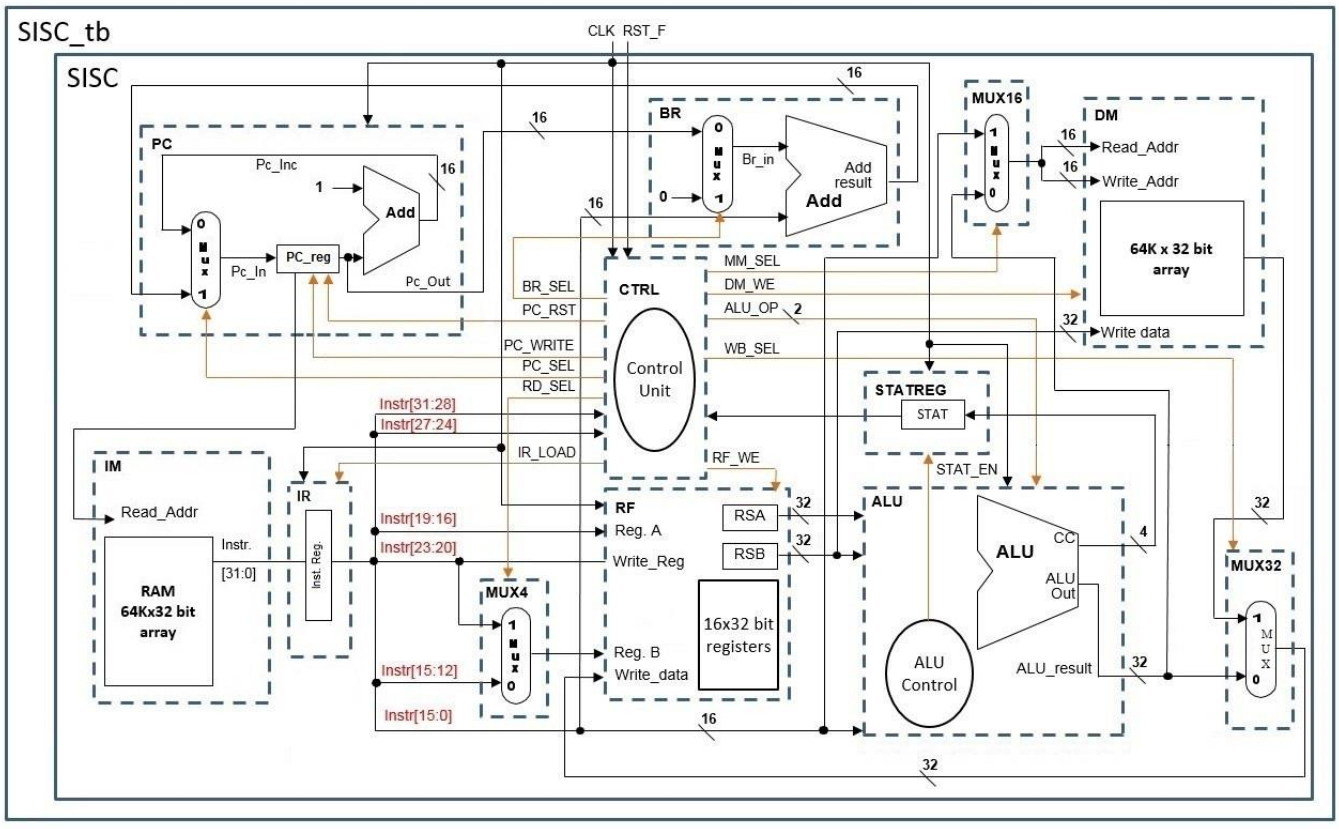
ECE:3350 Spring 2019 -
Computer Architecture and Organization
Simple Instruction Set Computer (SISC) Project

Part 3: Due Monday, April 8, @ 8:30 am

This part has two distinct goals: **1)** Complete the datapath by adding the data memory and modifying the control unit to support load and store instructions (as shown below), and **2)** Modify the datapath to support the SWAP instruction, and write two machine-language programs to test your design. More details are given below.



You are given:

- Your own solution to Part 2. **Again, you will not be able to succeed in Part 3 without first completing Part 2.**
- An imem.data file containing a program designed to test all of the instructions from Parts 1 and 2 and the LDX, LDA, STX and STA instructions of part 3. This is so that you can be sure your modifications for Part 3 have not broken your previous functionality.
- Three example data memory files (datamemory.data, sort_data.data and mult_data.data).

- The Verilog files dm.v and mux16.v (complete with descriptions), which you are not to modify.
- The diagram shown above.

You are required to:

- Modify the ctrl and sisc modules to include the new control lines and modules for the load and store instructions. Note that your sisc_tb file should not need to change from Part 2.
- Simulate your design using the supplied imem.data file and verify that the load and store instructions are working

Now modify the datapath to support the SWP instruction:

- Follow these guidelines while making your changes to the datapath:
  - You may create any new control lines or modules you deem necessary.
  - You may not remove or modify any provided file except the multiplexers.
  - Update the imem.data file to test the SWP instruction.
  - Your final solution must still correctly execute all of the previous instructions.
- Update the FSM state diagram and datapath diagram to reflect your modifications, for both the load and store instructions and the swap instruction.
- Write two machine language programs:
  - Sort a list of N signed, 32-bit integers stored in main memory using the bubble sort algorithm. This program must use your newly implemented SWAP instruction at least once. The number of integers in the list, N, will be stored in memory location 0, with the integers stored at addresses 1 through N. After execution of your program, the integers should be in ascending order (least integer stored at address 1, and greatest stored at address N). An example list is provided in sort_data.data. **Save this instruction memory file as sort_instr.data.**
  - Multiply two unsigned, 32-bit integers stored in memory and write the resulting 64 bit product back to memory. The two integers will be stored in memory locations 0 and 1, and the product should be written to locations 2 and 3, such that the most significant bits are at address 2. An example data file is provided as mult_data.data. **Save this instruction memory file as mult_instr.data.**
- Compress your project folder, along with the 'work' directory, the three instruction memory files, the data memory files, updated state diagram, and all .v files into a .zip file named "Part3.zip".

Details/Notes:

- Part 3 contains two significant tasks: the addition of load and store instructions, and the modifications to implement SWAP. **I highly suggest that, once you finish the changes necessary to support load and store instructions, you copy your project folder somewhere safe**, so that even if you don't complete the SWAP implementation, you will be guaranteed to get the points for the load and store section. Additionally, if you are stuck on the SWAP

modifications and need to restart them, you can begin from the saved copy and won't need to re-implement load and store as well.

- Note that for the SWP instruction, you are required to write two values back to the register file in one instruction execution cycle. This is possible if you write one value during the mem cycle and one during the writeback cycle.

Submission Overview:

- Your .zip file should be named "Part3.zip" and contain the following:
  - All the .v and .data files
  - The 'work' directory.
  - Your updated FSM state diagram.
  - An updated datapath diagram.

Rubric:

FSM and Datapath Diagrams	10 pts
Load and Store Implementation	15 pts
Swap Implementation	15 pts
Bubble Sort	10 pts
Multiplication	10 pts
AllInstr.data	10 pts
Total	**70 pts**