Department of Computer Science and Engineering, Texas A&M University

CSCE 312: Computer Organization: Spring 2024

Final Project – Demo and Report guidelines

1. Instruction to demo

1.1 ISA description

Describe your ISA design with your own figures or tables. Refer to the chapter 4.1 "The Y86 Instruction Set Architecture". You should at least show your own version of Figure 4.1, 4.2, 4.3, and 4.4.

1.2 Test programs

- You prepare at least three well known programs such as matrix addition, sorting algorithm, or program that checks palindrome. Your programs must show that they cover all instructions provided by your processor and make changes all the processor states. Source files should have enough comments to explain the code lines.
- Explain program logic using C program and assembly code.
- Demonstrate memory file generation for the Logisim.
- For each program, present expected changes of the processor's states using well formatted file.

1.3 Program run on your processor

- Load memory files to instruction memory and data memory. Then, set the tick frequency to 2 KHz. When the Logisim gets ready to execute the loaded program, you enable ticks. Once the processor starts execution, you cannot touch the Logisim until it falls into halt state.
- After the processor falls into halt state, you stop ticks and prove that the changes of processor state are identical to your expected changes.
- Repeat above two steps with the other two programs.

1.4 Timing of processor stages

- Present the timing diagram of your processor. The diagram at least includes outputs of each stage. If you implement Y86, the following signals must be included - icode, ifun, rA, rB, valC, valP, valA, valB, valE, valM, (new) PC.
- Use well drawn figure to present your timing diagram. Suggested to use any drawing tools.
- Run one program and verify that your processor follows the timing diagram. You click the clock generator manually and show cycle by cycle changes of the signals.
- Use couple of instructions until all timings are verified.

1.5 Design

- For each unit, you explain its basic functionalities. You should show design files of each unit such as input selection tables, FSM, etc. You at least show input selection tables of each unit. Your circuit must follow the input selection tables.

2. Report

Your report should contain the full description of our circuit, where you will explain all the stages of your implementation. Attach necessary screenshots of your circuit when necessary. You should add the timing diagram and details about all of the test programs. You need to explain that the test programs cover all the instructions. You also should show that the output you obtained from simulation is the same as the expected output with yis binary.

3. Evaluation

2.1 Minimum scope of implementation.

You must implement at least fetch unit. That is, your processor fetches instructions and fall into HALT state at the end. If it is not satisfied, other evaluation will not be performed. This is the minimum work for the project to get credits.

2.2 ISA description

Refer to the chapter 4.1 "The Y86 Instruction Set Architecture". You should at least show your own version of Figure 4.1, 4.2, 4.3, and 4.4.

2.3 Test programs and automated memory file generation.

Prepared programs are evaluated to see if they cover all instructions provided by your processor and make changes all the processor states. Another evaluation point is to see how memory files are is effectively generated. If you manually make memory files for Logisim, you will lose credits.

2.4 Processor Functionality

Once your processor has at least fetch unit, the functionality is evaluated using the three programs and the expected processor state changes. This is the most important evaluation, so you get the biggest credits from this check. Even if you have partially done processor, you can come up with a strategy to show that your processor is functioning.
Ex 1.) Only Fetch Unit - show PC changes and produced values from fetch unit and compare them with manually computed values.
Ex 2.) Fetch + Decode/writeback – show PC and register changes.
Ex 3.) Fetch + decode/writeback + execute – show PC, register, and CC changes.
Ex 4.) Fetch + decode/writeback + execute + memory – show PC, register, CC, and memory changes.
Note that even though your processor is partially done, you must use programs that are designed to test complete processor.

2.5 Design verification

After functionality check, your processor design is verified with your design files. The check point here is to see if your design files and actual implementation are identical. For example, you must modify input selection table from tutorials for your own processor. If there is a mismatch you will lose credits. You must have input selection table for each input control modules (grey boxes in figures from tutorials). If you do not have it, you will lose credits. Visually well-organized circuits and wires are also parts of evaluations.

2.6 Etc.

During demo, TA will ask you few questions about your processor design and evaluate how well you understand your own design. If you just get some circuits from on/offline sources but cannot explain well, you will get huge deduction.

4. Submissions

- Three program sets (C/assembly/binary).
- Logisim circuit file(s) of your processor.
- Timing diagram of signals from each stage.
- Input selection tables of control modules of each stage and other design files.
- Project report
- Individual one-page report per team-member (like what you submitted for lab4).