CPSC 121: Models of Computation
Lab #9: A Working Computer

Objectives

In this lab, we revisit the Y86 processor, which you saw components of in labs 4 and 6. Recall that in lab 4, you learned about the ALU and in lab 6, you looked at the RAM. Our goal with this lab is for you to appreciate that a computer is a complex sequential circuit that you now have the tools and knowledge to analyse.

This lab will also expose you to *machine code* which are instructions executed directly by the computer's central processing unit (CPU). Our goal here is for you to realize that an appropriate string of binary numbers actually can be used to *program* a circuit like this one. This processor will appear in more detail in CPSC 313 – enjoy!

1 Prelab

Download and print out the file `playcpu.pdf` from the course website. Go through one complete clock cycle with the entire computer (all four stages). Two clock cycles have been done for you already as examples. **TODO (pre-lab): Starting with the Fetch/Decode stage and following the instructions provided, correctly fill in the next column for each of the four stages. It might make your work easier if you print the document single-sided.**

2 A Paper Computer

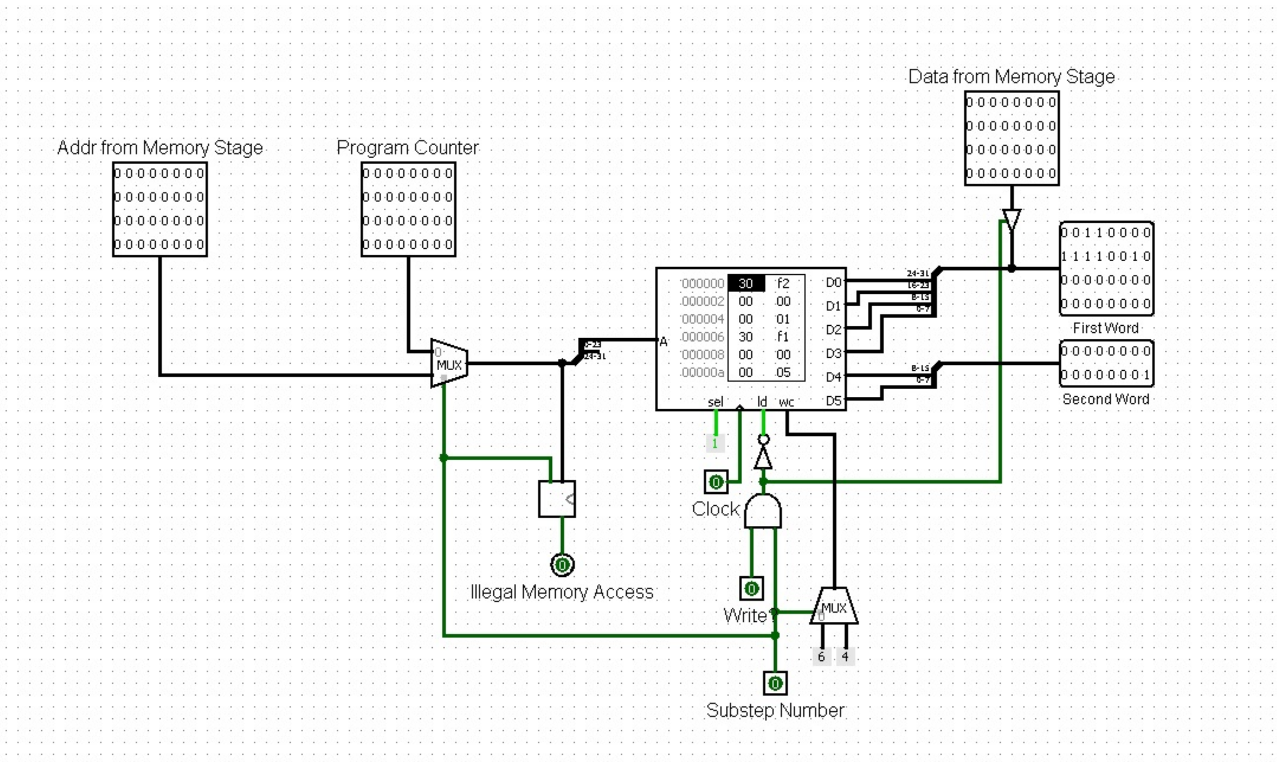
Team up with another group to form a team of four. The four of you will be running the rest of the program you started in the prelab, with each of you taking on the role of a different part of the computer. Halfway through the exercise, you will switch roles as follows (the table is shaded to indicate where you should switch): Memory and Registers becomes Fetch and Decode; Fetch and Decode becomes Execute; Execute becomes ALU and Decide; ALU and Decide becomes Memory and Registers.

- Memory/Registers: Your job is to handle two types of computer memory: its RAM (just called "memory") and its small component of fast memory (called "registers")
- Fetch/Decode: Your job is to take instructions out of memory and parse them for the computer to use
- Execute: Your job is to execute instructions using the ALU and then tell the computer what instruction to execute next
- ALU: Your job is to do the arithmetic and logic operations for the Execute stage

TODO: Pick your roles and run the computer program on paper. Check with your TA regularly to make sure you haven't made a mistake. What does the program do?

3 A Mystery Program

1. Download the files `Y86-cpu.circ` and `simple-loop.img` from the course website and open up `Y86-cpu.circ` in Logisim
2. Find the 16MB RAM module and click on it using the poke tool. Double click on the magnifying glass in the middle to open the module.
3. Find the RAM module and right-click on it. Select "Load image" and load in `simple-loop.img`. Your screen should now look like this:



4. Now return to the main screen by pressing Control-Left arrow. *Note: There is a bar in the bottom left hand corner of your Logisim screen that allows you to zoom in and out of the circuit.*
5. To clock the computer, press Control-T. *Note: Two clock cycles equal one line of instruction. If you reset the clock using Reset Simulation or Ctrl-R, you will have to reload the image into the RAM module.*

TODO: Run the program in Logisim. At what line in the program does it have a loop?

Note: `%eax`, `%ecx`, `%edx`, `%ebx`, `%esp`, `%ebp`, `%esi`, and `%edi` are the register values. The other inputs and outputs to the modules you will be interested in looking at are named similarly to the paper computer from the first part of lab.

Go to the "Simulate" menu and select "Logging" to log the following values over time: **Program Counter, iCd, iFn, bch, valE, eax, ecx** and **edx**. For each of the eight values, click "Change Radix" twice so they are all set to 16. Press Ctrl-K to begin clocking your computer, which will then fill in the table for these values. (To stop clocking the computer, press Ctrl-K again.)

TODO: What are these eight values doing over time? What do you see?

TODO: Pick one of the following modules in Logisim and look at in detail: Decode, Branch? (inside Execute), or PC Update. What is this module doing? What are its inputs and outputs? How does it relate to the paper computer we simulated at the start of class?

4 Further Analysis

TODO (further analysis): Why have we broken up the computer into different stages? What are each of the following stages doing when the current instruction is being processed by a different stage?

5 End of Lab Survey

TODO: To help us improve these labs both this term and for future offerings, complete the survey at http://www.tinyurl.com/cs121labs.

TODO: In addition to the End of Lab Survey, you will receive one bonus mark for participating in the End of Term lab curriculum survey, available at: http://tinyurl.com/CS121LabSurvey.

A Marking scheme

All labs are out of ten marks, with two marks for prelabs, and eight marks for in-lab work. In more detail:

- Two marks - Prelab questions
- Five marks - In-lab questions (two marks for the paper computer, one mark for finding the line where the program loops, one mark for logging the eight values and one mark for looking at a module in detail)
- Two marks - Further analysis question
- One mark - End of lab survey, with one bonus mark for the other survey

TAs may at their discretion award one bonus mark, such as for completing the challenge problem. (In this lab, it is possible to get 12 out of 10.)

B Challenge Problem

TODO (challenge): Look back on the paper computer you completed at the start of lab and write out an instruction (in hexadecimal) that takes the value from register 5, subtracts it from the value in register 6 and stores the resulting value into register 6.