

CS 350c, Spring 2017, Laboratory 3

A y86 Implementation Simulator

Assigned: Tuesday, April 18, 2017

Due: Thursday, May 4, 2017, by 9:30 am

WARNING! Late labs not accepted!

1 Introduction

In this lab, you will learn about one possible y86 implementation. This will involve filling in some missing pieces of a C-language program that attempts to model the y86 design shown in Figure 4.41 of the class textbook, "Computer Systems: A Programmer's Perspective" by Bryant & O'Hallaron (3rd edition).

This laboratory is designed to help you more thoroughly understand what a simple pipeline involves. You are asked to fill in some pieces we have deleted from our C-language version of the PIPE- version of the y86 implementation.

2 Logistics

You are expected to work on this lab alone. However, you may communicate with others concerning your understanding of C code and the various tools, e.g., the compiler, assembler, linker, loader, and other systems issues. And, you may discuss the implementation specification of the PIPE- version of the y86 microprocessor. And, of course, you may discuss the tools that you use – including anything provided to you for our class. The results that you submit in response to this laboratory must be created and provided by you alone.

Any clarifications and revisions to the laboratory will be posted on the top-level course web page.

3 Handout Instructions

You will find the file `y86-pipeline.tar` referenced on the homework page of the class website. You may download this file so you can use its contents. There may be changes or updates, so please be sure to download the latest version – check the top-level class web page for any update or correction announcements.

The primary thing you will find is a C-language program that attempts to capture the essence of what is shown in Figure 4.41 (on page 424 of the 3rd edition of *Computer Systems, A Programmers Perspective*). Originally, this file contained a complete, working version of a simulator that was modeled after Figure 4.41. However, we have removed several sections of our (model) code, and left instructions for you to implement the missing functionality.

The format for our y86 simulator input file is designed to be simple. Note, we use exactly the same input format used in Laboratory 2. Each line of an input file must contain two numbers: a natural-number memory address (0..65535) and a natural number byte value (0..255). The format for each byte is:

```
(  
  ( <address_0> . <byte_0> )  
  ( <address_1> . <byte_1> )  
  ...  
  ( <address_N> . <byte_N> )  
)
```

By default, our PIPE- y86 implementation simulator initializes the program counter to the first address found in the address-value pairs; i.e., `address_0`. This is handled for you. Therefore, your programs may start at any address; the first address listed will be assumed to be the starting (program counter) address.

4 Evaluation

Your lab will be evaluated on how well your updated y86 PIPE- simulator works and whether it get the correct answers when run with our test programs.

Hints

First, get a very simple program (e.g., load a constant, add two numbers, etc.) to work correctly. You should use your own assembler to generate suitable programs. Once a very simple program is working, you can incrementally increase the complexity of the program(s) you create, and then try all of your programs on your evolving pipeline simulator. Remember, you have to make the pipelined simulator function correctly, so first concentrate on Parts 1–4, and finally on Part 5 (see the `README` in the laboratory tar file).

You may talk with your colleagues, your instructor, and your lab assistant for ideas to implement useful debugging aids and/or other tools. Remember, you are trying to get a pipelined implementation to work, and this will be a good test of your understanding of how the PIPE- implementation actually functions.

Hand In Instructions

Please follow the instructions below for turning in your work.

- Make sure you have included your identifying information in your submission. In this laboratory, you will provide the entire modified simulator. Remember to confine your changes to the areas where solutions are requested.
- To handin your solution to Laboratory 3, please submit it as an ASCII file. Submit your updated `y86-pipeline-lab.c` file to the Canvas system. Additional submission instructions may appear on the Homework and Laboratory webpage close to due date.
- WARNING! Late labs not accepted!