CISC-221 Computer Architecture Lab 6
Enhancing Performance and Code Optimization
Winter Term 2018

Instructor: Dave Dove (dove@cs.queensu.ca).

1 Introduction

In this lab you will optimize a short C program. You will analyze the assembly language version of both the original and optimized programs and estimate the number of instruction executions of each. You will then run a statistical analysis of each program and use the output to calculate the performance increase exhibited by your optimized program. You will also compare your estimate of instruction executions to the value output from the statistical analysis program (rpistat). We will discuss the rpistat program in class before the lab.

2 Logistics

You may work individually or in pairs for this assignment. All submissions for this lab are electronic. Clarifications and corrections will be posted to the course website.

If you are working as a pair, only one submission is required for two students but make sure that both your names are included in the submission. Either student may make the submission.

You must upload your submission to the Lab6 Submission item on the course website by the due date and time.

4 Helpful Details

References:
Course text: Chapter 5.

5 What you need to do:

Log in to the raspberry pi assigned to your group for this work (pi0 – pi31.caslab.queensu.ca) All code related work must be done on the ARM architecture of the raspberry pi.

1. Download the required files from the Lab page of the course website to a windows machine or mac so that you can edit the included word and assembly language files using Microsoft Office.
2. Create a new directory on your Z drive for this lab that is available to your pi.
3. Copy the lab6start.c file to that directory.
4. Compile (on a pi) the original version of the function file using the command: gcc –O0 –o lab6start lab6start.c Notice the option –O0 is – capital o zero. This will (almost) inhibit all compiler optimizations.

5. Review the lab6start.s assembly language program provided. Comments corresponding to the statements in the C code for this program have already been provided.
6. Now that you can relate the assembly language code to the C program, calculate the number of instructions that will be executed by calling the function once, then extend your calculation to the number of instructions executed by the function during the entire C program. Use the Instruction Execution Estimation Worksheet (provided on the web site) to do these calculations.
7. Run the command: rpistat lab6start . This will create a file: rpistat.txt. Rename the file to lab6start.txt. Copy the Cycles, Instructions **(use the number in the square brackets)** and the CPI to the Function Summary worksheet provided on the course web site (*Before* section). The number of instructions should be close (within 5 – 10%) of your estimate.
8. Create a copy of the lab6start file and name it lab6fini.c Optimize the lab6 function. Use one or two specific, simple optimizations as covered in the course text. Do not do a major rewrite eliminating loops and removing variables. Do not change the semantics of the code. The object is to see the effect of simple optimizations that a compiler routinely will make. Do not change any other code in the lab6fini.c file outside of the lab6 function.
9. Compile your optimized program using the command: gcc –O0 –S lab6fini.c to create an assembly language version of your optimized program.
10. Repeat steps 4 through 7 inclusive using your optimized version of the function. After running rpistat on your optimized program, rename the file to lab6fini.txt. Copy the Cycles, Instructions **(use the number in the square brackets)** and CPI figures to the Function Summary worksheet (*After* section).
11. Complete the Function Summary Report.
12. Assembly the following documentation and upload it to the Lab6 dropbox on the course website. Your upload should include:

    a) The lab6fini.c code that is your optimized C program.

    b) The commented versions of your lab6fini.s assembly language program commented in the style of the commented lab6start.s program provided. (only include the lab6 function section of the .s file and include only the C statements from the lab6fini.c program as comments.

    c) The completed Instruction Execution Estimation Worksheet for t his function.

    d) The completed Function Summary Report.