CSCE 231/2303: Computer Organization and Assembly Language Programming
Summer 2020

Project 3: Cache Performance

Introduction

Processors are generally able to perform operations on registers faster than the access time of large capacity main memory (DRAM). Though SRAM memory is fast, it is not economical to provide all the main memory with SRAM. Introducing a small SRAM memory, called cache, between the main memory and the processor, can alleviate the problem.

In this project, you will become familiar with how caches work. For that, you will build a direct-mapped cache simulator and validate its correctness.

Basic Cache Simulator

For this project you will build a cache simulator, using the C/C++ language, to simulate a direct mapped cache with the following characteristics:

- Cache size (variable): from 8Kbytes to 64Kbytes.
- Cache line size (variable): from 8 bytes to 128 bytes.
- Main memory address space: 64 Mbytes.

You are given a skeleton for the simulator in C/C++. The skeleton has set of memory reference generators that you will use for simulation. The function names are: `memGen1(), memGen2(), memGen3(), memGen4(), memGen5()` and `memGen6()`. These functions must be used to generate memory addresses used during simulation. At least 1,000,000 memory references must be generated in each experiment. For each experiment, you have to measure the hit and miss ratios.

Note: You are given a skeleton which has, almost, everything for simulation. Probably you will end up writing few lines of code to get it done. The project is about experimenting with the simulator to understand the cache behavior and how its performance is influenced by the parameters. Data collection, presentation and analysis are what determine your project grade.

Experiments

1) For each generator measure and graph the miss ratio for line sizes: 4, 8, 16, 32, 64 and 128 bytes ($ size is fixed at 64Kbytes)
2) For each generator measure and graph the miss ratio for cache sizes: 8, 16, 32 and 64 Kbytes.
3) Analyze the plotted graphs and draw conclusions.

Deliverables

- The cache simulator source code (see the guidelines below)
- A report (at least 5 pages in addition to the cover) to present the experiments collected data and your analysis. The analysis involves plotting the collected data and outlining the conclusions that can be extracted from the graphed data.
- You must submit your report and source code as well as scheduling an appointment for the demo before the final exam time slot.

Grading

- Simulator implementation with test cases used to verify its functionality [30%]
- Experiments execution, data collection and data presentation [40%]
- Data analysis and conclusions [30%]

Guidelines

- Work in a group of 2 students (same as project 1).
- Deadline for the report and source code submission, as well as the demo: the final exam slot starting time.
- Before the demo time, you need to submit the source code as well as the report through BB.
- You must provide the data or source files used for validating the simulator.
- The source code must be well-commented. Also, the files must be named properly.
- The archive that contains the source code must include a readme.txt file that lists all the files and a brief description for each one of them.
- The report must be in **PDF** format.
- The report must use 1.15 line spacing and font size 11 for the text.
- The report cover page must show the course name, course number, the semester, and the group members' names and IDs.
- The graph(s) presented in the report must be well-labeled and colored.