

CSCE 231/2303: Computer Organization and Assembly Language Programming Spring 2025

Project 2: Cache Performance

Introduction

Processors can perform operations on registers faster than the access time of large capacity main memory (DRAM). Though SRAM memory is fast, providing all the main memory with SRAM is not economical. 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. Also, you will study the effects of the cache parameters on its performance. For that, you will build a set-associative cache simulator and validate its correctness; then you will use the simulator to run some experiments and analyze the results.

Basic Cache Simulator

For this project, you will build a cache simulator using the C/C++ language to simulate n-way set-associative cache with the following characteristics:

- Cache size: 64 Kbytes (fixed).
- Cache line size (variable): 16, 32, 64, and 128 bytes.
- Number of ways (variable): 1, 2, 4, 8 and 16
- Computer memory address space: 64 Mbytes.

You are given a skeleton for the simulator in C/C++. The skeleton has a set of memory reference generators that you will use for simulation. The function names are: `memGen1()`, `memGen2()`, `memGen3()`, `memGen4()`, `memGen5()` and `memGen6()`.

You must use these functions to generate memory addresses used during simulation. At least, you must generate 1,000,000 memory references in each experiment. For each experiment, you have to measure the hit and miss ratios.

Experiments

For each generator, perform the following experiments:

1. Fix the number of sets to 4 and vary the line size. Plot the hit ratio against the line size.
2. Fix the line size to 64 bytes and vary the number of sets (ways). Plot the hit ratio against the number of ways.

In the two experiments, comment on the results and draw some conclusions.

Deliverables

- The cache simulator source code (see the guidelines below)
- A report (at least five pages in addition to the cover) to present the 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 schedule an appointment for the demo on July 19th. More information to follow.

Grading

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

Guidelines

- Work in a group (size = 2).
- Use GH for development. All the deliverables must be in the GH repo. Ensure it is well-structured and includes enough documentation to make it useful.
- Before the demo time, you need to submit the URL to your GH repo.
- You must provide the data or source files used for validating the simulator (as a part of your submission).
- The source code must be well-commented. Also, the files must be appropriately named.
- The archive containing the source code must include a `readme.txt` file that lists all the files and a brief description.
- The report must be in **PDF** format.
- Bring a hard copy of your report to present in the interview.
- 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.
- The graph(s) presented in the report must be well-labeled. A color printout of the graph(s) is recommended (you don't have to color print the rest of the report. It won't impress me)
- Deadline: May 21st.