WILFRID LAURIER UNIVERSITY

LAURIER

Inspiring Lives.

Course Project Milestone-3 Instructions

CP216 - Introduction to Microprocessors
Department of Physics and Computer Science, Faculty of Science, Waterloo
Spring 2025

Description:
This term project will conclude with you implementing a CPU simulator with a memory hierarchy simulation for some executable instruction sets of your choice. This project is to help you learn the implementation of the microprocessor execution cycles and memory accesses.

You are required to complete the project in **groups of Three/Four** so that the project expectations are held to a higher standard than the work of individuals. To complete the project, you will need to create and submit **Three milestone deliverables:**

Milestone 1: ARMv7 Simulator – Due Week 8
Peer Review of the Simulators – Due Week 8
Milestone 2: Memory Hierarchy Simulation – Due Week 12

Milestone-2 Background:
Processors are generally able to perform operations on registers faster than the access time of large capacity main memory (DRAM). Alhough SRAM memory is fast, it is not economical to provide all the main memory with SRAM. Introducing a small SRAM memory, called a **cache**, between the main memory and the processor can alleviate the problem.

Milestone-2 Requirements:
In this project milestone, you will become familiar with how caches work and how they impact the computer performance. To achieve these goals, you will build a cache simulator and validate its correctness. Then, you extend the ARM simulator implemented in Milestone 1 by adding support for a memory hierarchy.

The memory hierarchy has the following specifications:

- **Level 1:** L1 instruction cache and L1 data cache
  - Type: Write back Direct mapping or Write-back Fully associative
  - Size: 1KB each
  - Block Size: configurable: 4, 8, 16 or 32 bytes

- **Level 2:** Unified Cache of 16KB
  - Type: Write back Direct mapping cache
  - Block Size: configurable: 16, 32 or 64 bytes

You will also need to generate a set of benchmark program binary files. For each, you are required to find the best configuration for the memory hierarchy to achieve the best performance. To achieve this, you have to perform a series of experiments and log the collected data in a table. The cost function you are trying to minimize for each benchmark program is:

Cost = 0.5 L1 cache misses count + L2 cache misses count + write back operations count

Your final update simulator design must:

1. Use one of the cache configurations for the L1 and L2 memory levels.
2. Read ARM machine code file. The file is just a binary file containing the program instruction machine code. The first 4 bytes (32-bit word) in the file belong to the first instruction; the 2nd 4 bytes belong to the 2nd instruction, etc. The first instruction is assumed to be at location 0x00 in the main memory.
3. Decode each machine code word, then translate it into a true ARM instruction.
4. Perfom the necessary cache memory allocation.
5. Count the L1 cache misses, L2 cache misses and write back operations.
6. Calculate the cost and log the value in a file.
7. Repeat the process with another cache configuration and record the new cost.
8. Repeat the process by reading another binary file **(use at least 2 different binary program files)** until you find the optimal cache memory configuration.

Milestone-2 Submission Guidelines:

Please, do not combine the submitted video, poster and codes in one zip folder, just submit attach them separately into the dropbox.

1. **Presentation/Demo Video in MP4:** Each Group must submit a (10 mins) video showing the project simulations and the test cases. All team members should participate in the video. Your video should include a complete simulation demo of your simulator and build different test cases by applying different binary files. It should also highlight the collected data and trends to derive your conclusions about the best optimal cache memory configuration.

2. **Manual Poster (ONE Landscape Page in PDF):** Your poster should present the data collected from the experiments and your analysis. The analysis may involve plotting the collected data and outlining the conclusions that can be extracted from the graphed data. Your conclusions should highlight the best configuration for the cache memory.

3. **Upload the project source files in one zip folder (Don't include the video and poster in that folder):** Simulator code, test-case binary files

Some Remarks:

- All deliverables must be submitted to their associated Dropbox on MyLearningSpace by the due date. Each **Group** is responsible for ensuring its files are successfully uploaded on time in the required format(s).

- You can use Generative AI tools to generate logic and testing codes or binary files for the test cases. Please highlight in the poster/video which parts of your code are generated by the AI tool and which are not. Also, you can find some open-source online implementations on GitHub. If you use any of these open-source codes, please cite these repositories in your poster and include them in your reference list.