CDA 5106 Advanced Computer Architecture I (Fall 2014)
Final Project

Important dates: October 17, 2014, (One-page proposal)
November 12, 17, 19, 24, and December 1, 2014 (tentative presentation dates)
December 1st, 2014, 11:59pm (Source code & project report due)

Description
In this project, you will use a computer architecture simulator (e.g., Simplescalar) to explore some cutting-edge research ideas.

Policy
In this project, the students registered for CDA5106 need to work in a group of 3 students. You need to specify the contributions of each group member so that appropriate grade can be assigned accordingly. I will assign the presentation dates, but generally those with simpler projects and/or more group members will present earlier rather than later.

Tasks
In this project, you will use a computer architecture simulator carry out some research activities in computer architecture. You first need to identify a research topic and write up a research proposal. Then, you need to discuss your proposal with the instructor. Your proposal should contain the names of group members, the title of your research project, and a detailed plan to carry out the work. Based on your anticipated progress, you will select a date to present your work to the class. At the time of your presentation, your simulator needs *not* be complete as long as you can produce some preliminary results. If you choose to implement some ideas from a paper, you should NOT copy the original results from the paper. Instead, your results must come from your own simulation. The presentation is about 20 minutes for each group and 5 minutes for questions.

Your final project report should follow a formal paper format, i.e., it should contain the following sections: abstract, introduction, related work, one or more technical sections, simulation methodology, simulation results, conclusion, and references. *You turn in your source code only if it works.*

Grading
Proposal: 10%
Presentation: 30%
Research: 30% (i.e., the code implementing the research ideas)
Final research report: 30% (you must use your own results.)

You are also responsible to come up with 1 question on the topic you work on. Some of the assembled questions will be used as final exam questions.

Late submissions will not be accepted.

Examples of research ideas

- Major references for ideas: International Symposium on Computer Architecture(ISCA), International Symposium on Microarchitecture (MICRO), International Conference on High Performance Computer Architecture (HPCA), Architectural Support for Programming Languages and Operating Systems (ASPLOS), etc. Also, check the computer architecture website/online publications for more references.
- Microarchitecture for instruction flow, register data flow, or memory data flow
    - Branch prediction (at most 2 groups on this topic. If you would like to work on branch prediction, talk to me first.)
    - Cache design
    - Value prediction
    - Prefetching techniques
    - Etc.
- Microarchitecture for throughput computing
    - Multithreaded architecture
    - Multi-core architecture
- Complexity-effective, scalable design such as checkpoint recovery and processing, hierarchical issue queue, clustered architecture, etc.
- Memory hierarchy design
    - Cache design for single-core processors
    - Cache design for multi-core processors, especially how to share the cache among multiple cores in multi-core processors
- Low power or Energy-efficient architectures
- Architecture support for reliability, availability, and security
- Novel GPU Architectures
- Your own ideas

Simulation Environments

- Simplescalar – if you are studying an architectural change to a single core, using Simplescalar may be the simplest way to proceed. You MUST use sim-outorder and report on the differences you see in simulated execution time (NOT the wall clock time of the simulator). You may also use sim-cache (if doing cache studies) or sim-bpred (if doing branch predictor studies) to more quickly search the parameter space and understand the benefits and limitations of your approach. But you must still always show the effects on execution time using sim-outorder.
- Simplescalar + Wattch – if you are studying energy efficiency or have some idea that affects power and not just performance, you may want to use Simplescalar+Wattch. Wattch is a tool originally from Princeton/Harvard that is now incorporated into Simplescalar. It has its own separate obtuse almost incomprehensible config file and output format, but it will give you power numbers from the same simulation run that you get performance numbers from. If you are adding an architectural entity you will need to add a Wattch power model (or analytically model your power based on simulation

statistics). If you are modifying an existing structure (e.g. banking a register file) you may need to modify the existing Wattch power model for that component.

- SESC – a multicore simulator from the University of Illinois. It can be a pain to configure, but it does allow you to run the same application across multiple cores for a set of commonly used parallel applications (the SPLASH benchmarks).
- GPU simulators – there are several of these around. GPGPU-Sim is one that has been used before. The key for this course is to find one that you can modify that *architecture* of the GPU for your project and not just find a simulator of a particular card that will emulate CUDA or other code that you write.
- DRAMsim – from the University of Maryland. Models the physical SDRAM memory system (channels, ranks, banks, rows, columns) in excruciating detail.

Past Project Examples

Ke Chen and Xinruo Zhang. "Predicting and Eliminating Dead Blocks to Increase Cache Efficiency".

Jonathan Pecoraro and Sarah Loewy. "Procedure-based Prefetching".

Hamid Izadinia and Fereshteh Sadeghi. "Improving Prefetchers using Adaptive Pattern Prediction".

Ajay Hardikar and Surbhi Bhardwaj. "Dynamic Branch Prediction Using Perceptrons".

Yazan Jadaa and Zac Chenaille. "Low Power Caches with Better Performance".

Thomas Godfery and Saleem Sahawneh. "A Comparison of Energy-saving Cache Techniques".

Matthew Fontaine and David Adams. "Hardware Profiling with Data Compression".

Steven Feldman, Jason Hochreiter, and Brendan Lynch. "Study of Heterogeneous Cores and Their Effect on Energy and Processing Efficiency".

Jun Ye and Shiyuan Wang. "The Tradeoff Between Performance and Fairness of Dynamic Cache Partitioning in Multicore Processors".

Raghu Avula and Chandan Reddy Gurala. "Pseudo Set-Associative Caches: Power and Performance."

Jose Sanchez and Frank Plochan. "Cache Prefetching with Spatio-Temporal Memory Streaming (STeMS)".

Jun Ding, and Ying Wang. "Data Value Prediction".

Anisha Reddy and Taranjeet Singh Bhatia. "Increasing cache efficiency by predicting and eliminating dead blocks".

Yuan Chang and Tangke Cong. "Energy Efficient dynamic DRAM and cache size controller".

Erfan Davami and Rahmatollah Beheshti. "SOM-based Memory Prefetching Algorithm".

Mark Diehr, Paul Timmons, and Charles Snyder. “Data Prefetching: A Hybrid Approach”.

Emily Sassano, Brian Lynch, and Joel Onema. “Stride-detecting Prefetchers vs. Stream Prefetchers”.

Thomas Carroll and Keenan Simmons. “Branch Prediction using Confidence Estimators for Single Core Processors”.

Steven Braeger and Nick Arnold. “Branch Prediction for Stream Processors”.

Lisa Soros, Anthony Wehrer, and Pierre LaBorde. “MESI vs. MOESI in Multicore Processors”.

Justin Refi, Henry Heck, and Claudio Romano. “Predictive Cache Reallocation in Multicore Systems”.

Andrew Yee, Deli Zhang, and Yazen Ghannam. “Improving the Energy Efficiency of Modern Processors”.