ECE571: Advanced Microprocessor Design – Final Project
Fall 2022

Due: Friday, 16 December 2022 5PM (Last day of Classes)

Overview:

- Design a project that explores the power, energy, and/or performance of a modern computing system. This is very open-ended, but some guidelines are below.

Guidelines:

- You may work either alone or in groups of two or three. If you work in a group your end project will have higher expectations.
- You may use any system you like for this project. It can be one of the systems used in class (Haswell-EP, ryzen, Ampere), a personal system you have access to, or I can set up accounts on some of my other server or embedded boards.
- Feel free to use more exotic systems or operating systems. It does not have to be x86/ARM, nor does it have to run Linux.

Part 1: Topic Selection (due 10 November 2022) (5pts)

Each group should send a brief e-mail describing your project topic and listing group members.

Part 2: Related work / Progress Report (due 22 November 2022) (10pts)

Give a brief status report on your project. Re-iterate the topic, how preparations are going, whether you have access to all the hardware you need, and any other status you might have. If things are not going well, there is still time to modify the topic.

Also provide some "related work" for your project. Do a literature search and find examples of other people who have done similar research. It is OK if you find a lot of related work, or even if someone has done the exact same thing before. This is a class project (not a PhD thesis) so reproducing something that has been done before is perfectly fine.

A quick way of finding related work is using Google, but don't limit yourself to papers turned up that way. Sometimes you might find an interesting paper via Google, but you will not be able to find a free copy online (just a site asking you to pay money). If it's a journal like the IEEE or ACM you can still get copies of the articles for free. If you're on an on-campus internet connection (ending in .maine.edu) you can go to the UMaine library homepage, search for IEEE or ACM there, and it will give you a link to click through to get to those sites (IEEE explore or similar) and you will then be able to find and download those papers for free.

I'd prefer if the references you find are books or academic papers, but if you find a few good blog or website references that's probably OK.

What I would like to receive:

- A file containing the status update, as well as a few paragraphs about the related work. If you are working alone, I'd like to see at least three. If you are working as a group, I'd like to see at least six. (More is fine).

- A list of references with the work cited.
- Indicate if you are willing to present Tuesday instead of Thursday (there will be a small bonus for going early).
- You can submit the status update by e-mail. Only one submission is needed per group.

An short example of roughly what I expect:

> Our project is a low-powered video game. We have some of the code working and we've obtained a Raspberry Pi. We'll need to borrow a WattsUpPro to conduct power measurements.
>
> Related Work:
>
> Our research involves making an open-source low-powered video game that will run on an embedded Raspberry Pi board. Weaver[1] wrote a cross-platform assembly language game for the ARM platform but unlike us he did not characterize the power consumption while running. Mallow and Snap[2] look at optimizing a Ray-Tracing program on the Tegra2 ARM system. This is similar to our project, only they looked solely at ray-tracing applications and not video games.
>
> [1] V. Weaver. "Tom Bombem: An ARM Implementation of the Classic DOS game." Proc. of the 4th Conference on Useless Video Games, p 10-18, May 1996.
>
> [2] M. Mallow and G. Snap. "Optimizing Energy Consumption on the Tegra2." Journal of Embedded Programming, p11-19, Vol 1 Issue 15, June 2010.

Part 3: In-class Presentations, 6 and 8 December 2022 (40pts)

- You will have roughly 10 minutes to present. Plan for 8 minutes describing your project and allowing 2 minutes for questions.
- Give a summary of what you did and why. Show any results you obtained. Describe any future work that needs to be done.
- You may present slides using the projector if you want, but that's not strictly necessary.

Part 4: Project Writeup, Due 16 December 2022 (45pts)

This will be a short paper (6-8 pages) that will be in the style of an a short academic conference/journal/workshop paper and should have the following sections:

1. Introduction – describe your project and provide some background on what you are looking at
2. Related Work (what you submitted in Part 2 possibly extended a little to fit the flow of the paper)
3. Experimental Setup – list everything you did to set up your project. List benchmarks used, compiler options, hardware used, software versions, etc. It is best to provide too much than too little. Charts and diagrams are fine too.

4. Results / Analysis – describe what you found, feel free to include graphs and tables
5. Conclusion / Future Work
6. Bibliography / References Cited

Other things to include:

1. If you worked in a group, please add a section after the conclusion describing how the work was distributed among the group members
2. If you wrote source code, feel free to include it if possible. You can just include it as a separate file with your paper submission

You can e-mail your final report to me. pdf or word document is fine, the code should be attached too.

Project Ideas:

- Power/Energy overhead of architectural features
  - Power/Energy slowdown caused by Meltdown and Spectre fixes.
  - Power/Energy overhead from prefetching (can turn off prefetching on some systems, like core2, in theory cortex a9, and many recent intel chips)
  - Power/Energy overhead from branch prediction (can turn off branch predictors on MIPS chips, not sure about ARM)
  - Power/Energy overhead from caches (find a system you can disable the cache? The ARM systems)
  - Power/Energy overhead from virtual memory
  - Power/Energy implications from frequency scaling
  - Power/Energy implications from GPUs
  - Power/Energy implications from network devices (ethernet, wireless, bluetooth, etc)
  - Power/Energy implications from vector instructions (SSE)
  - Power/Energy implications of multi-threading or multi-processing
- Thermal Issues
  - Visualize the temperatures inside your laptop/desktop/embedded board. Some machines (such as Mac laptops) have 20+ temperature sensors.
  - If your laptop/desktop/Pi3 overheating? Investigate ways to keep this from happening. Is the performance increase worth the extra power usage?
- Hardware Performance Counters
  - Write some performance counter validation tests, see if you can match "expected" results
  - Estimate power/energy using performance counters
  - Validate RAPL energy measurements on real hardware.

- Operating Systems
  - See how much the recent SPECTRE security fixes have slowed down modern machines. On Linux you can possibly turn these off.
  - Power/Energy comparison of same task under various operating systems (Linux, OSX, Windows, FreeBSD, etc.)
  - Make DVFS frequency scaling decisions based on hardware counter results
  - Make power/energy/performance characteristics of various Linux kernel versions
- Architectural Comparisons
  - Power/Energy vs performance on various ARM processors
  - Power/Energy vs performance on ARM vs x86
  - Power/Energy of other embedded system or DSP boards
- Application Investigation
  - Pick a favorite application type and compare the power/ performance of various implementations
  - Investigate the power/performance of a benchmark when varying compiler options
  - Pick a poorly behaving benchmark (power or performance wise), find the cause of poor performance, and improve it.
- Simulation
  - Get the tools for and attempt one of the JILP architecture competitions (it does not have to be the current one).
    - Branch predictor competition `http://www.jilp.org/cbp2016/`
    - Cache Replacement competition 2017
    - Value prediction competition 2018
  - Explore power/performance of an architectural feature using a simulator or DBI tool.
- Many other topics are open for investigation, feel free to suggest something. See Below for a list of previous topics for inspiration.

Topics from Previous Years

- 2020
  - Performance measurement of a laptop
  - Intel and GPU RAPL measurements
  - Raspberry Pi power and perf comparison
  - Performance comparison of 3 different art programs
- 2019
  - Performance comparison of Jetson TX1 to TX2

  - Prefetcher performance in glibc
  - Power measurement with glibc scaling
  - Web browser power comparison
- 2018
  - Raspberry Pi 3B+ benchmarks
  - Cache aware programming
  - Laptop power comparison: Linux/Windows/FreeBSD
  - Pi power comparison: Linux vs vmwOS
  - Performance measurements of old hardware
- 2017
  - Cache behavior of Raspberry Pi
  - TLB behavior of different filesystems
  - Effects of frequency scaling on power
  - RAM frequency power implications
  - Matrix calculation size vs performance
- 2016
  - JILP Branch prediction competition
  - Power comparison of Custom ECE598 OS vs Raspbian
  - ARM/x86 Video Game power consumption comparison
  - Performance of DNA analysis code
  - Raspberry Pi GPU power usage
  - Power usage of cache on DSP
  - Measuring Raspberry Pi power with Teensy board
  - Power benefits of embedded processor
  - RAPL power measurement validation
  - Raspberry Pi/x86 Power usage comparison
  - Power performance of multiple generations of x86 machines
- 2014
  - Correlating code behavior with power usage
  - Hard drive power analysis
  - Windows vs Linux Power Usage Comparison
  - Reducing current consumption of a research project
  - PIC 8/16/32-bit power comparison

  - GPU/CPU power comparison
  -

- 2013
  - Fuzzy Dynamic Frequency Scaling
  - Power prediction of Raspberry Pi using perf counters
  - Power consumption of Pi-calculation algorithms
  - Power overhead of prefetching on ARM Pandaboard
  - Web-browser power usage comparison