EE P 524 A Au 22: Applied High-Performance Gpu Computing

TITLE: Applied High-Performance GPU Computing (GPUCompute)

Instructor: Colin Reinhardt, *Affiliate Assistant Professor, UW ECE*

Overview:

- Learn the real-world skills, undocumented “black art” techniques, and theory you will need to develop optimal parallelized GPU algorithms and software.
- Study underlying GPU processor microarchitectures & memory subsystems that govern performance, SIMD/SIMT/SPMD.
- GPU kernel design and coding, debugging, profiling, performance analysis, host- and device-side memory and cache optimization
- Use the NVIDIA CUDA platform and Turing T4 GPUs.

Schedule:

- **Week 1 (10/3)** : Course Intro & logistics, overview of parallel GPU computing and NVIDIA CUDA platform. Parallelism concepts.
- **Week 2 (10/10)** : multidimensional grids. CUDA device code. GPU HW architecture 101.
  - Homework-1 DUE : 17 OCT, 5:59 PM
- **Week 3 (10/17)** : kernel languages. Profiling and HW Arch 201 and iterative optimization. Analysis of algorithms. Roofline model.
  - Homework-2 DUE : 24 OCT, 5:59 PM
- **Week 4 (10/24)** : GPU HW Arch 201: memory hierarchy. Parallel performance theory. GEMM kernels.
  - Homework-3 DUE : 31 OCT, 5:59 PM
- **Week 5 (10/31)** : Introduction to parallel patterns. Advanced kernel language features and kernel performance.
- **Week 6 (11/7)** : 2D/3D Convolution and stencil patterns.
  - Homework-4 DUE : 14 NOV, 5:59 PM
- **Week 7 (11/14)** : GPU Numerical precision. Parallel histogram. atomic operations.
  - Final Project : *Proposal* DUE : 21 NOV, 5:59 PM
- **Week 8 (11/21)** : Parallel reduction pattern
  - Final Project : *Design* DUE : 28 NOV, 5:59 PM
- **Week 9 (11/28)** : Parallel prefix sum/scan patterns.
- **Week 10 (12/5)** :Parallel merge and sort.
  - Final Project : *Final Report* DUE : Friday 16 DEC, 11:59 PM

Course Overview:

There will be 10 lectures given via Zoom and supplemented with additional video recordings. Hand-on programming exercises will be part of the weekly lecture. Also there will be 4 homework assignments which will consist of (a) readings, (b) coding (c) theoretical/mathematical problems.

The remainder of the quarter will be focused on a final project which will be comprised of a fairly significant GPU code design and implementation on a topic approved by the course instructor, utilizing and applying techniques learned in the class.

Grading:

- Homeworks (4) 55% : (11 / 12 / 12 / 20 pts)
- Final Project 45%

Late Homework Policy

Lose 1 pt if it's late, and another pt each additional 24hrs late until 5 days (at -6 pts). Then you lose 2 more at 10 days late and 1 more per additional 5 days until you hit zero pts.

Of course if you have a personal emergency or a doctor's note (or some other legitimate extenuating circumstance) you will be granted an extension.

Course Materials:

Primary textbook:

Kirk and Hwu, *Programming Massively Parallel Processors*, 3rd Ed. Morgan Kaufmann, 2017

- Available through UW Libraries as online ebook.
- Get the O'Reilly App and create account using @uw.edu student SSO

Additional course reading materials and a list of supplementary reading materials will be posted on the course website. Nearly all materials will be available in online electronic formats, either freely available public literature or through UW Library (www.lib.washington.eduLinks to an external site.)

Course Prerequisites

- Proficiency programming with C and/or C++ programming and using integrated software development environments (Visual Studio, Eclipse) for building and debugging
  - The standard template library (STL) will be used.
- Familiarity with vector calculus and partial differential equations (PDEs); physical foundations and formulation of PDEs, Maxwell's Eqns, the wave equation, and the dispersion equation.

- Comfort with applied matrix analysis and linear algebra and numerical analysis, eigensystems, eigenvalue problems and solutions, particularly the basic vector and matrix operations.

Course Policies

You may collaborate and discuss homework assignments and project design and implementation with your fellow classmates, professor, TA and others. However, the work you submit must be your own, and you must write your own code(s). *Directly copying code without explicit attribution and/or plagiarizing is not allowed.*