L16: Libraries, OpenCL and OpenAcc

Administrative

- Remaining Lectures
  - Monday, April 15: CUDA 5 Features (small exercise)
  - Wednesday, April 17: Parallel Architectures and Getting to Exascale
- Final projects
  - Poster session, April 24 (dry run April 22)
  - Final report, May 1

Final Project Presentation

- Dry run on April 22, Presentation on April 24
  - Easels, tape and poster board provided
  - Tape a set of Powerpoint slides to a standard 2'x3' poster, or bring your own poster.
- Final Report on Projects due May 1
  - Submit code
  - And written document, roughly 10 pages, based on earlier submission.
  - In addition to original proposal, include
    - Project Plan and How Decomposed (from DR)
    - Description of CUDA implementation
    - Performance Measurement
    - Related Work (from DR)

Sources for Today's Lecture

References:

OpenAcc Home page:

http://www.openacc-standard.org/

OpenCL Home page:

http://www.khronos.org/opencl/

Overview of OpenCL:

http://www.khronos.org/assets/uploads/developers/library/overview/opencl-overview.pdf

http://www1.coe.neu.edu/~pmistry/perhaad/Research_files/icpe-opencl-tutorial.pdf

2012 Tutorial

Additional Reference (new book):

Heterogeneous Computing with OpenCL, B. Gaster, L. Howes, D. Kaeli, P. Mistry, D. Schaa, Morgan Kaufmann, 2012.

Many Different Paths to Parallel Programming

- In this course, we have focused on CUDA
- But there are other programming technologies, some of which we will discuss today
- Range of solutions:
  - Libraries
  - Higher Levels of Abstraction
    - OpenACC (today)
    - PyCUDA, Copperhead, etc.
  - Non-proprietary, but similar solutions
    - Specifically, OpenCL

A Few Words About Libraries

- Take a look at what you find in /usr/local/cuda-5.0/lib
  - CUBLAS: linear algebra
  - CUSPARSE: sparse linear algebra
  - CUFFT: signal processing (FFTs)
- More in https://developer.nvidia.com/gpu-accelerated-libraries
- Also, https://developer.nvidia.com/thrust
  - Thrust:
    - STL-like templated interfaces to several algorithms and data structures designed for high performance heterogeneous parallel computing

Advantages of Libraries

- Some common algorithms can be encapsulated in a library
  - Library is developed by performance expert
  - Can be used by average developer
  - Accelerates development process
- Often programmer does not even need to think about parallelism
  - Parallelism is embedded in the library
- Disadvantages of libraries
  - Difficult to customize to specific contexts
  - Complex and difficult to understand

What is OpenAcc?

- High-level directives can be added to C/C++ or Fortran programs
  - Mark loops or blocks of statements to be offloaded to an attached accelerator
  - Portable across host, OS and accelerator
  - Similar in spirit to OpenMP
- Example constructs
  - #pragma acc parallel [clause[,]…] *newline*
    block of code or loop
  - *Example clauses:*
    async[], copy_in(), private(), copy_out(), …
- Announced at SC11, November 2011
- A partnership between Nvidia, Cray, PGI, and CAPS

Why Openacc?

- Advantages:
  - Ability to add GPU code to existing program with very low effort, similar to OpenMP vs. Pthreads
  - Compiler deals with complexity of index expressions, data movement, synchronization
  - Has the potential to be portable and non-proprietary if adopted by other vendors
  - Real gains are being achieved on real applications
- Disadvantages:
  - Performance may suffer, possibly a lot
  - Cannot express everything that can be expressed in CUDA
  - Still not widely adopted by the community, but it is new so this may change

What is OpenCL?

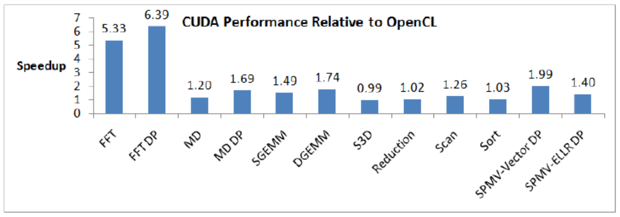
- Open source standard specification for developing heterogeneous parallel applications
  - i.e., parallel codes that use a mix of different functional units
  - Goal is to unify how parallelism is expressed, how to offload computation to accelerators like GPUs, and port code from one platform to another
- Advantages over CUDA
  - Not proprietary
  - Not specific to Nvidia GPUs
- Disadvantages (my list)
  - Portable, but not necessarily performance portable
  - CUDA is customized to Nvidia hardware execution model, so can be made faster
  - A bit low level in terms of programmer abstractions

OpenCL short presentation

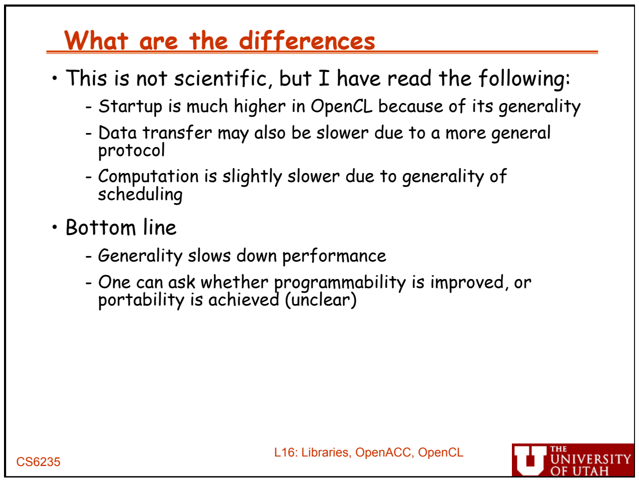
- Taken from

http://www.khronos.org/assets/uploads/developers/library/overview/opencl-overview.pdf
http://www1.coe.neu.edu/~pmistry/perhaad/Research_files/icpe-opencl-tutorial.pdf

A recent comparison, from Vetter, "Programming Scalable Heterogeneous Systems Productively", SOS15, March 2011.



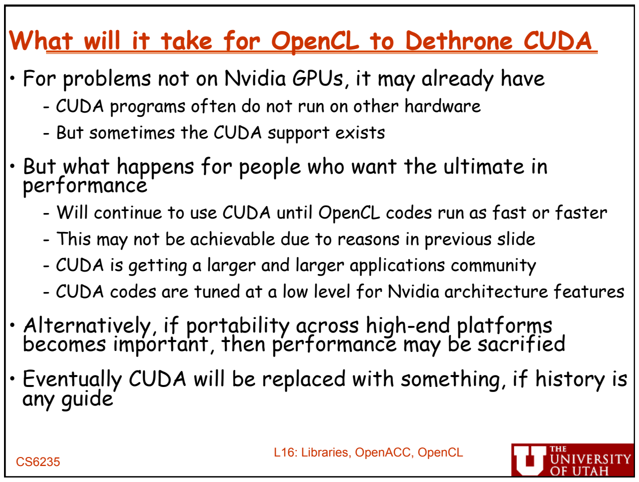
See also: Anthony Danalis, Gabriel Marin, Collin McCurdy, Jeremy S. Meredith, Philip C. Roth, Kyle Spafford, Vinod Tipparaju, and Jeffrey S. Vetter. 2010. The Scalable Heterogeneous Computing (SHOC) benchmark suite. In *Proceedings of the 3rd Workshop on General-Purpose Computation on Graphics Processing Units (GPGPU '10). ACM, New York, NY, USA, 63-74.*



What are the differences

- This is not scientific, but I have read the following:
  - Startup is much higher in OpenCL because of its generality
  - Data transfer may also be slower due to a more general protocol
  - Computation is slightly slower due to generality of scheduling
- Bottom line
  - Generality slows down performance
  - One can ask whether programmability is improved, or portability is achieved (unclear)

CS6235 L16: Libraries, OpenACC, OpenCL



What will it take for OpenCL to Dethrone CUDA

- For problems not on Nvidia GPUs, it may already have
  - CUDA programs often do not run on other hardware
  - But sometimes the CUDA support exists
- But what happens for people who want the ultimate in performance
  - Will continue to use CUDA until OpenCL codes run as fast or faster
  - This may not be achievable due to reasons in previous slide
  - CUDA is getting a larger and larger applications community
  - CUDA codes are tuned at a low level for Nvidia architecture features
- Alternatively, if portability across high-end platforms becomes important, then performance may be sacrified
- Eventually CUDA will be replaced with something, if history is any guide

CS6235 L16: Libraries, OpenACC, OpenCL