GPU Programming with CUDA (C and PGI CUDA Fortran) and the PGI Accelerator Programming Model

Michael Wolfe
Michael.Wolfe@pgroup.com
http://www.pgroup.com

March 2011

Part 4: Wrap-up

4-1

The Portland Group®

Accelerators are the Future of HPC

- HPC can't support its own designs
  - commodity parts, software as well as hardware
- Accelerators allow break from ISA compatibility
- Accelerators allow strong scaling
- GPUs are the game today
  - designs are essentially free
- You must program to the new model
- Downside
  - future of commodity may be in mobile
  - commodity cpus and accelerators may not solve HPC problems

4-2

The Portland Group®

Programming Accelerators

- Goals
  - productivity
  - performance
  - portability
- Will it run fast on tomorrow's accelerators?

4-3

The Portland Group®

Options

- Low level
  - OpenCL, CUDA
  - full control
  - low on productivity, performance portability
  - high on performance
  - language is portable, even if programs are not

4-4

The Portland Group®

Options

- Libraries
  - Magma, etc.
  - programming to the library
  - essentially a limited-vocabulary language
  - high on portability, productivity, performance
    - *if* your program fits the vocabulary

4-5

The Portland Group®

Options

- Class library
  - TBB, Ct, (Rapidmind), Thrust
  - A type system and implementation
  - Advantage: some information instantiated at compile time
  - Other advantages / disadvantages are the same as library approach

4-6

The Portland Group®

Options

- High level, PGI Accelerator model, (eventually OpenMP)
  - High on productivity, portability
  - Performance is improving over time
  - Open question: how portable is the model?

4-7

The Portland Group®

How to Reach a Petaflop

- 10^6 = megaflop
- 10^9 = gigaflop
- 10^{12} = teraflop
- 10^{15} = petaflop
  - Jaguar
    - 18,688 dual-socket six-core nodes
      2.6GHz, 4-8 GFlops/core, 224,256 cores
    - (.224 x 10^6 cores) x (2.6 x 10^9 GHz) x (4 results)
      = 2.32 x 10^{15} results/cycle = 2.32 Petaflops (double precision)
    - Top500 Rmax = 1.759 PFlops, Rpeak = 2.331 PFlops

4-8

The Portland Group®

How to Reach an Exaflop

- 10^6 = megaflop
- 10^9 = gigaflop
- 10^{12} = teraflop
- 10^{15} = petaflop
- 10^{18} = exaflop = 10^9 x 10^9
  - one billion gigaflop cores = one exaflop (MPI^{max})
    - 16-cores in 2011, 32-2013, 64-2015, 128-2017, 256 in 2019
    - 1 million quad-socket 256-core nodes at 1GHz
    - 50X nodes, 2X sockets/node, 40X cores/socket relative to Jaguar
    - at 4 results/GHz, reduce by ¼, higher clock reduces as well
  - one million teraflop cores = one exaflop
    - 1,000 ops / cycle (1GHz)

How to Reach an Exaflop

- Maybe O(100,000) 10-teraflop nodes
  - wide SIMD, multithreading, latency tolerant
  - 1GHz clock = 10,000 operations/cycle (5,000 mul+add)

Jaguar	Proposed
224,256 cores	O(100,000) units
37,376 sockets	100,000 sockets
18,688 nodes	O(100,000) nodes
2.5GHz clock	1GHz clock
4-8 GFlops/core	O(1,000) GFlops/unit
24-48 GFlops/socket	1,000 GFlops/socket
48-96 GFlops/node	O(1,000) GFlops/node

Additional Information

- PGI Accelerator Programming Model
  - x86+NVIDIA
  - PGI Fortran and C
  - Linux, Windows, OSX
  - www.pgroup.com/accelerate for documentation, FAQ, articles
- PGI CUDA Fortran
  - X86 + NVIDIA
  - PGI Fortran
  - Linux, Windows, OSX
  - www.pgroup.com/cudafortran for documentation, FAQ, articles
- Common Compiler Feedback Format (CCFF)
  - integrated into all PGI compilers and pgprof
  - www.pgroup.com/CCFF for additional information

Copyright Notice