CRAY
THE SUPERCOMPUTER COMPANY

Programming GPU Devices Using OpenACC Directives on the Cray XK6 Platform

Luiz DeRose, Alistair Hart, Heidi Poxon, & Adrian Tate
Cray Inc.

Patrick Wohlschlegel
Allinea

Agenda – Day 1 (March 6, 2012)

09:00 - 09:15 Welcome / Introductions (Luiz DeRose)

09:15 - 09:30 Overview of the Cray XK system (Luiz DeRose)

09:30 - 10:30 Steps to create a hybrid code (Heidi Poxon)

10:30 - 11:00 Break

11:00 - 12:00 OpenACC (Alistair Hart)

- Execution and memory models
- OpenACC Directives
- CUDA Interoperability
- CCE Support status

12:00 - 13:00 Lunch

13:00 - 13:30 User experiences talks on the Cray XK6 system

- Tim Ewart, (U Geneva)

13:30 - 14:30 Use case examples (Alistair Hart)

14:30 - 15:00 Break

15:00 - 15:30 How to build/run existing CUDA and OpenCL on the Cray XK6 (Alistair Hart)

15:30 - 17:30 Lab (Cray / CSCS)

Agenda – Day 2 (March 7, 2012)

09:00 - 09:45 Performance Tools for the Cray XK (Heidi Poxon)

09:45 - 10:30 DDT debugger for the XK6 (Patrick Wohlschlegel)

10:30 - 11:00 Break

11:00 - 11:30 Cray libsci_acc (Adrian Tate)

11:30 - 12:00 OpenACC future (Luiz DeRose)

- **New features (functionality and performance)**
- **Standardization**
- **Support for other architectures**

12:00 - 13:00 Lunch

13:00 - 14:30 User experiences talks on the Cray XK6 system

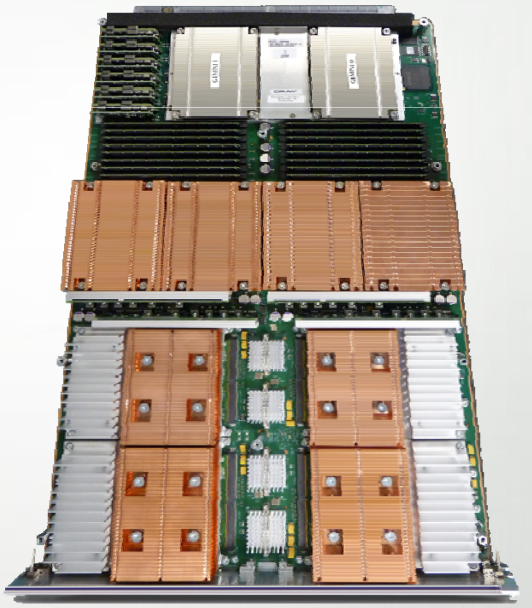
- **Xavier Lapillonne (MeteoSwiss)**
- **Joachim Stadel (U. Zurich)**
- **Matthias Christen (USI)**

14:30 - 15:00 Break

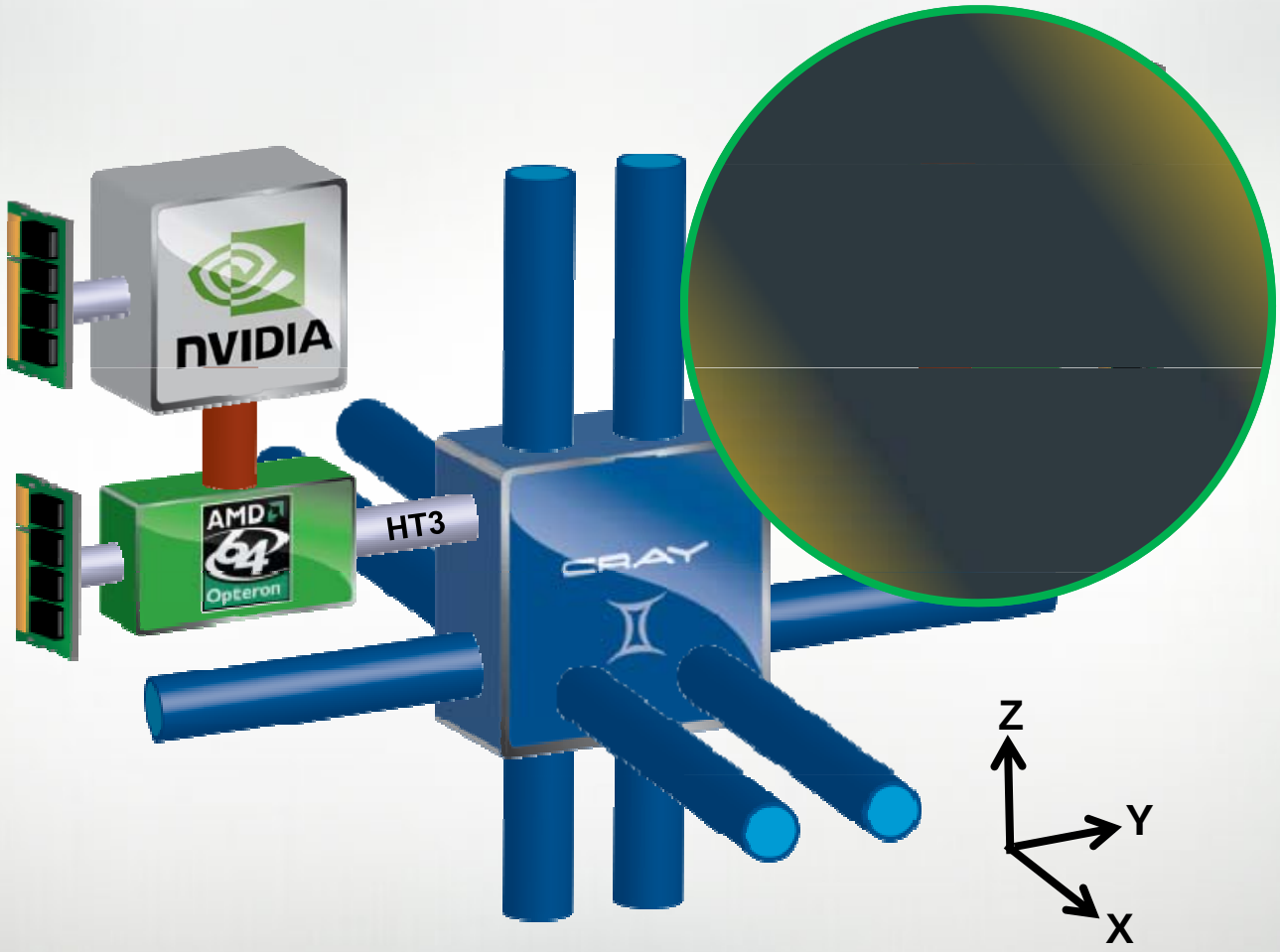
15:00 - 17:30 Lab (Cray / CSCS / Allinea)

Cray XK6 Overview

- The Cray XK6
  - Next generation NVIDIA Fermi X2090 GPU
    - 20% better performance than 2070
    - compute: 448→512 cores; 1.15→1.30 GHz clock
    - memory: 6GB; 138 GB/s bandwidth
  - AMD Interlagos CPU
  - Cray Gemini interconnect
    - high bandwidth/low latency scalability
  - Fully compatible with Cray XE6 product line
  - Fully upgradeable from Cray XT/XE systems



Cray XK6 Compute Node



Cray Vision for Accelerated Computing

- Most important **hurdle for widespread adoption of accelerated computing is programming difficulty**
  - Need a single programming model that is **portable across machine types**
    - Portable expression of heterogeneity and multi-level parallelism
    - Programming model and optimization should not be significantly different for "accelerated" nodes and multi-core x86 processors
    - **Allow users to maintain a single code base**

- Cray's approach to Accelerator Programming is to provide an **ease of use** tightly coupled **high level programming environment** with compilers, libraries, and tools that will **hide the complexity** of the system
  - Focus on integration and differentiation
  - Target **ease of use** with extended **functionality** and increased **automation**

- **Ease of use** is possible with
  - Compiler making it **feasible for users** to write applications in **Fortran, C, C++**
  - Tools to help users port and optimize for accelerators
  - Auto-tuned scientific libraries

Unified X86/GPU Programming Environment

- The Cray XK6 includes the first-generation of the Cray Unified X86/GPU Programming Environment

- Why is Cray putting so much effort into this?
  - **It is hard to get good performance from hybrid systems**
  - Opens up GPU computing to a larger user base
  - A **good Programming Environment narrows the gap** between observed and achievable performance

- The Cray XK6 PE supports three classes of users:
  1. “Hardcore" GPU programmers with existing CUDA ports
  2. Users with parallel codes, ideally with some OpenMP experience, but less GPU knowledge
  3. Users with serial codes looking for portable parallel performance with and without GPUs

Programming for a Node with Accelerator

- Fortran, C, and C++ compilers
  - **Directives to drive compiler optimization**
    - Compiler does the “heavy lifting” to split off the work destined for the accelerator and perform the necessary data transfers
    - Compiler optimizations to take advantage of accelerator and multi-core X86 hardware appropriately
  - Advanced users can mix CUDA functions with compiler-generated accelerator code
  - Debugger support with DDT
- Cray Reveal, built upon an internal compiler database containing a representation of the application (the CCE Program Library)
  - Source code browsing tool that provides interface between the user, the compiler, and the performance analysis tool
    - **Scoping tool** to help users port and optimize applications
    - **Performance measurement and analysis** information for porting and optimization
- Scientific Libraries support
  - Adaptive Auto-tuned libraries (using Cray Auto-Tuning Framework)

Accelerator Programming

- **Why a new model?** There are already many ways to program:
  - CUDA and OpenCL
    - All are quite low-level and closely coupled to the GPU
  - PGI CUDA Fortran
    - Still CUDA just in a better base language
  - PGI accelerator directives, CAPS HMPP
    - First steps in the right direction – Needed standardization

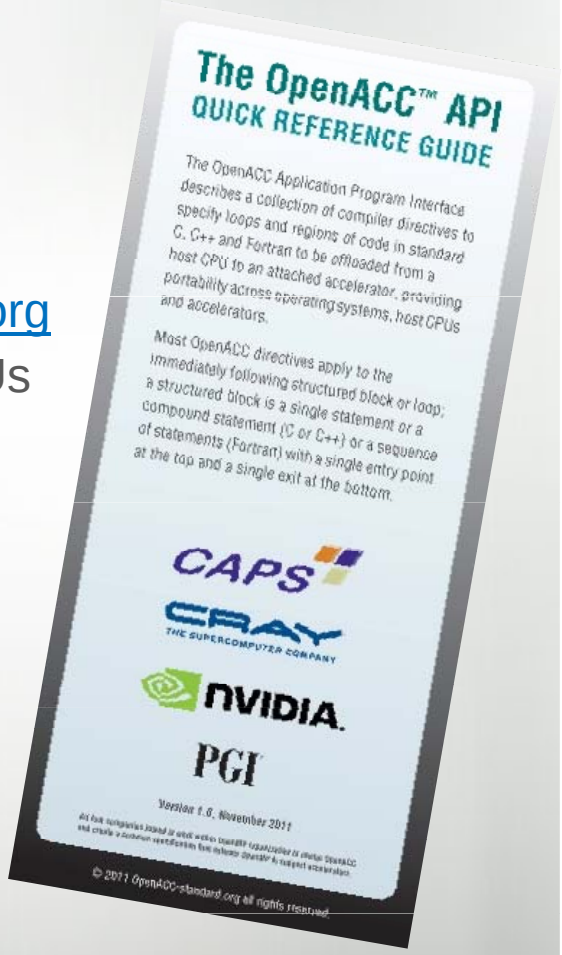
- User needs to write specialized kernels:
  - Hard to write and debug
  - Hard to optimize for specific GPU
  - Hard to update (porting/functionality)

- **Directives provide high-level approach**
  - **Simple programming model for heterogeneous systems**
  - **Based on original source code**
    - **Easier to maintain/port/extend code**
    - The same source code can be compiled for multicore CPU
  - Possible performance sacrifice
    - A small performance gap is acceptable (do you still hand-code in assembler?)
    - Goal is to provide at least 90% of the performance obtained with hand coded CUDA
      - Already seeing this in many cases, more tuning ongoing

OpenACC®
DIRECTIVES FOR ACCELERATORS

CRAY THE SUPERCOMPUTER COMPANY

- A common directive programming model for today's GPUs
  - Announced at SC11 conference
  - Offers portability between compilers
    - Drawn up by: NVIDIA, Cray, PGI
  - Works for Fortran, C, C++
    - Standard available at www.OpenACC-standard.org
    - Initially implementations targeted at NVIDIA GPUs
- Current version: 1.0 (November 2011)
- Compiler support:
  - Cray CCE: partial now, complete in 2012
  - PGI Accelerator: released product in 2012
  - CAPS: released product in Q1 2012



CAPS CRAY THE SUPERCOMPUTER COMPANY NVIDIA The Portland Group

OpenACC Accelerator Directives

- Compiler directives provide a simple programming model for heterogeneous systems
  - Can compile in the presence or absence of an accelerator
- An **open standard** is the most attractive for developers
  - Portability; multiple compilers for debugging; permanence
  - Helps programmer tools proliferation
  - Provides faster time to adoption of accelerators
- Proposed to the OpenMP Language Committee
  - Subcommittee of OpenMP ARB, aiming for OpenMP 4.0
    - Includes most major vendors
    - Co-chaired by Cray

Likely Application Migration Paths