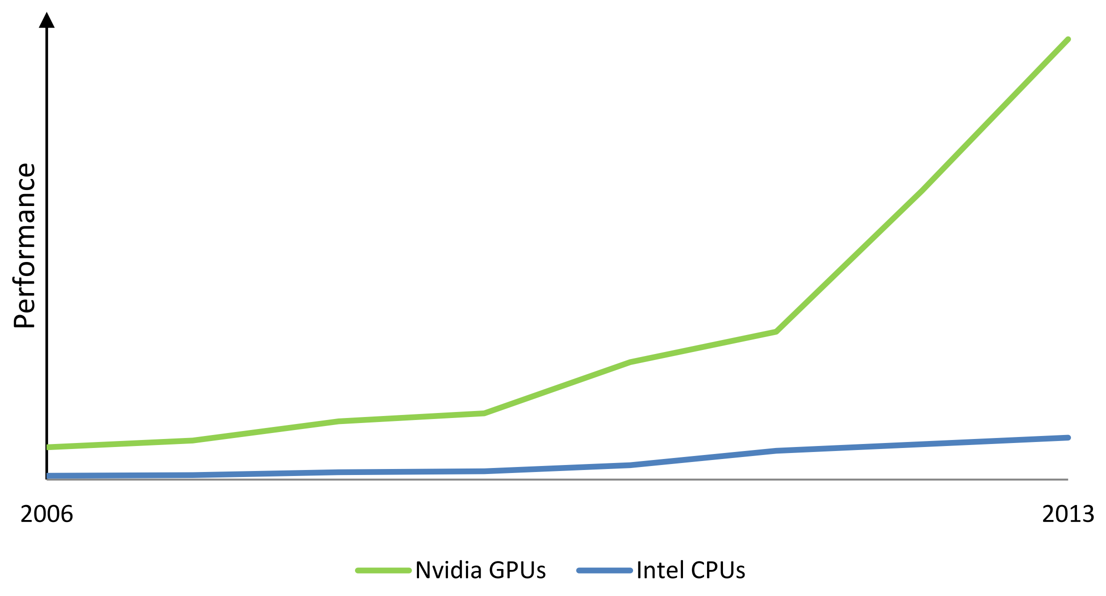
HIGH PERFORMANCE CONSULTING COURSE OFFERINGS

LEARN TO TAKE ADVANTAGE OF POWERFUL GPU BASED ACCELERATOR TECHNOLOGY TODAY



CONTENTS

ACRONYMS AND TERMINOLOGY

- CUDA – A GPU Computing framework
- OpenCL – Open Compute Language, a GPU Computing framework

GPU COMPUTING INTRODUCTION COURSE

Gives engineers and technical managers a solid introduction to today’s key technologies in the GPU Computing space, highlights where the technology trends are heading, and provides valuable orientation in current hardware and software.

The first half generally offers best value for technical managers and system designers, while the second half is more directed towards software developers.

CONTENT

- GPU technology advantages
- GPU Computing concepts
- Orientation in how, when, and where GPUs are being applied today
- GPU Computing application demonstration
- GPU hardware solutions (embedded/COTS/HPC)
- Hardware and software trends in GPU Computing space
- GPU Computing hardware and software model
- GPU Computing characteristics by code examples

GOAL

- To understand the basics and benefits of GPU computing technology

LEVEL: BASIC

DURATION: HALF-DAY

INTENDED AUDIENCE

- Software developers
- Technical managers
- System designers

DELIVERABLES

- Lecture slides

GPU COMPUTING DEVELOPER FOUNDATION COURSE

Gives software developers and engineers a solid foundation in implementing the best technologies available today for the GPU Computing technology space. Theory and concepts of GPU Computing are interleaved with hands-on coding exercises. Principles of problem-based learning and 'learning by doing' are employed. The course is based on one of the two dominant GPU Computing frameworks; either CUDA or OpenCL. In addition, problems and algorithms from the developers' area of interest may be incorporated into the existing code samples.

CONTENT:

- GPU Computing overview
- Hardware and software trends in the GPU Computing space
- CUDA/OpenCL: platform and execution model
- Programming model
- Hardware & Software model
- Code examples that expose GPU hardware and software characteristics
- Code examples from developers area of interest (optional)
- Common GPU Computing architectures in detail

GOAL

- The programmer will have built a foundation for writing basic CUDA/OpenCL-based applications
- The developer will achieve a strong orientation in existing GPU hardware platforms and its capabilities
- The developer will become confident to get started on their own in writing GPU computing applications

LEVEL: INTERMEDIATE

DURATION: 2 DAYS

INTENDED AUDIENCE

- Software developers and engineers

PREREQUISITES

- Intermediate C/C++ programming experience

DELIVERABLES

- Lecture slides
- Course sample code

COMPLEMENTARY TOPICS

The following complementary topics are add-on packages that may be added on to an existing course package or taken individually. The course levels range from intermediate to advanced.

HIGH LEVEL API PROGRAMMING

- Thrust
- OpenACC

MOBILE COMPUTING

- Embedded hardware
- Software aspects
- Hands-on computing examples

ADVANCED CUDA/OPENCL OPTIMIZATION STRATEGIES

- Fusing kernels
- Dynamic parallelism
- Memory bandwidth utilization strategies
- Instruction Level Parallelism
- Warp shuffle
- Optimizations via the texture cache

MISCELLANEOUS

- Data visualization through CUDA/OpenGL/DirectX interoperability
- Matlab application acceleration through CUDA

SPARSE MATRIX COMPUTATIONS USING CPUS AND GPUS

Teaches software developers, engineers and scientists how to perform sparse iterative methods on multi- and many-core CPUs as well as GPU based platforms. The course contains various methods for solving large sparse linear problems including state-of-art methods such as advanced preconditioners and multi-grid methods.

The computations are performed with PARALUTION, a software for iterative methods on various parallel systems including multi-core CPUs, GPU and the Xeon Phi (MIC). The library provides full portability of code across different hardware.

CONTENT:

- Short introduction to GPU Computing
- Iterative methods
  - Kyrlov subspace methods: CG, BiCGStab, GMRES, IDR
  - Preconditioners: Splitting, ILU, Approximate Inverse
  - Multigrid methods: Algebraic/Geometric
- Sparse matrix computations on GPUs
  - Vector-vector routines
  - Sparse matrix-vector multiplications
  - Matrix formats
  - Iterative methods
  - Performance Analysis
- Integration to existing software packages
  - PARALUTION Plug-ins
  - Advanced integration techniques
- Various GPGPU software overview
- Portability of the code on various hardware

GOAL

- To understand the modern methods used for solving large and sparse linear problems
- To obtain practical knowledge of how to perform and accelerate these methods on GPUs
- To learn how to integrate a sparse solver into existing software packages

LEVEL: INTERMEDIATE

DURATION: 1 DAY

INTENDED AUDIENCE

- Software developers and engineers

PREREQUISITES

- Intermediate C/C++ programming experience

DELIVERABLES

- Lecture slides
- Course sample code
- PARALUTION user manual