C++ AMP, Part 1 of 2: Parallelism and Management

page 1

Meet the expert: John Stratton, Ph.D., is a senior architect at Multicoreware Inc. and a visiting lecturer at the University of Illinois at Urbana-Champaign. John has been at the forefront of research and education in heterogeneous computing, reaching hundreds of students through the Virtual School of Computational Science and Engineering's courses on heterogeneous computing and optimization for scientific applications. John writes papers and articles for leading academic conferences and journals as well as broad-reaching publications such as IEEE Computer. He is also an active participant and presenter at several industry and technology groups and events across the country.

Prerequisites: This course assumes that you have a good understanding of core C++ concepts, included classes, objects, containers, and iterators. You should also be familiar with Visual Studio 2012 for Visual C++ development, including compilation, testing, and debugging. Although not required or expected, you may get more out of some parts of the course if you are familiar with multithreaded programming, Visual Studio 2012's debugging capabilities for multiple threads, and basic computer architecture concepts.

Runtime: 02:04:59

Course description: The course will start with an overview of how you can analyze code in terms of kinds of work, and identify where C++AMP might be useful. We'll talk about why C++AMP is useful for taking full advantage of modern computing systems, and how you can begin to use C++AMP constructs in your existing code. We'll also compare C++AMP to other popular heterogeneous computing models, so that you can see the tradeoffs associated with each. Then we're going to cover the abstractions available in C++AMP for managing the accelerator devices themselves. Once an application has a handle on the device it wants to use for computation, it will also need to express data collections and data indexes, possibly in multiple dimensions.

Course outline:

Code Parallelism
- Introduction
- Types of Work: Sequential
- Types of Work: Task Parallel
- Types of Work: Data Parallel
- Demo: Types of Work
- Summary

AMP Data Parallelism
- Introduction
- System Architecture Model
- Basic C++ AMP Usage
- Demo: C++ AMP and Features
- Task vs. Data Parallelism
- Summary

Heterogeneous Computing
- Introduction
- Heterogeneous Computing Systems
- Benefits and Uses of Parallelism
- What is C++ AMP Good For?
- Demo: Utilizing the GPU
- Summary

Alternate Models
- Introduction
- CUDA Overview and Tradeoffs
- Demo: CUDA
- OpenCL Overview and Tradeoffs
- Demo: OpenCL
- Summary

Accelerator Management
- Introduction
- Managing Accelerators
- Demo: Accelerators
- Managing Accelerator Contexts
- Demo: Accelerators and C++AMP
- Summary

Data Management
- Introduction
- Multidimensional Ranges
- Multidimensional Indexes
- Accelerator Data Management
- Demo: Data Management
- Demo: Effects
- Array Views
- Demo: Array Views
- Summary

Accelerator Programs
- Introduction
- Launching Accelerator Programs
- restrict(amp)
- Demo: parallel_for_each
- Demo: restrict(amp)
- Demo: CPU Variables
- Summary