Developing and Optimizing Applications Using the OpenCL Framework for FPGAs

Course Description

Learn how to develop new applications written in OpenCL, C/C++, and RTL in the SDAccel™ development environment for use on Xilinx FPGAs. Porting existing applications is also covered.
This course also demonstrates how to debug and profile OpenCL code using the SDAccel development environment. In addition, you will also learn how to maximize performance and efficiently utilize FPGA resources.

Level: SDx 2

Training Duration: 2 days

Who Should Attend?

Software and hardware developers who want to develop OpenCL, C/C++, and RTL applications in the SDAccel development environment.

Prerequisites:

- Basic knowledge of C/C++

Software Tools:

- SDAccel development environment and common build tools

Skills Gained: After completing this training, you will be able to:

- Identify parallel computing applications suitable for accelerating on FPGAs
- Discover how the FPGA architecture lends itself to parallel computing
- Write OpenCL programs for FPGAs
- Examine the OpenCL execution model
- Analyze the OpenCL memory model
- Profile and debug OpenCL code using the SDAccel development environment
- Discover how to maximize performance in FPGA fabric
- Efficiently utilize FPGA memory resources
- Utilize the SDAccel development environment
- Rapidly develop FPGA applications using OpenCL
- Port programs written in OpenCL for CPUs or GPUs to Xilinx FPGAs

Course Outline

1. Introduction to OpenCL

2. Comparison of CPU, GPU, and FPGA Architectures

3. OpenCL Support for Xilinx FPGAs

4. FPGA Hardware Details

5. Introduction to the OpenCL API

6. OpenCL Execution Model

7. Memory Hierarchy

8. Profiling and Debugging

9. Optimization

10. Using the SDAccel Development Environment: Coding, Compiling, Emulating, Profiling, and Debugging

11. Using Existing C/C++ Code as Kernels in OpenCL

12. RTL IP as Kernels in OpenCL

Lab Description

Lab 1: Creating an OpenCL Program from Scratch

Creating an OpenCL Program from Scratch – Provides an overview of OpenCL API, memory transfers, and kernel enqueuer operations.

Lab 2: Vector Addition

Vector Addition – Learn how to execute parallel kernels.

Lab 3: Pi by Monte Carlo Processes

Pi by Monte Carlo Processes – Implement the Pi by Monte Carlo processes.

Lab 4: Maximizing Performance

Maximizing Performance – Use vector data types and increase bandwidth.

Lab 5: Optimizing Kernels

Optimizing Kernels – Use Loop Unrolling and Loop Pipelining.

Lab 6: Profiling and Debugging Using the SDAccel Development Environment GUI

Profiling and Debugging Using the SDAccel Development Environment GUI – Learn how to use interactive programming tools to improve performance and squash bugs.

Lab 7: Optimizing C/C++ Code for OpenCL

Optimizing C/C++ Code for OpenCL – Convert existing C/C++ code into a kernel that can be used by OpenCL

Lab 8: Using an RTL Kernel

Using an RTL Kernel – Learn how to use existing, highly optimized IP in a new OpenCL application.