AMD
XILINX

Accelerating Applications with the Vitis Unified Software Environment

AI-ACCEL (v1.0)

Course Specification

Course Description

Learn how to develop, debug, and profile new or existing C/C++ and RTL applications in the Vitis™ unified software environment targeting both data center (DC) and embedded applications.

The emphasis of this course is on:

- Building a software application using the OpenCL™ API to run hardware kernels on Alveo™ accelerator cards
- Building a software application using the OpenCL API and the Linux-based Xilinx runtime (XRT) to schedule the hardware kernels and control data movement on an embedded processor platform
- Demonstrating the Vitis environment GUI flow and makefile flow for both DC and embedded applications
- Describing the Vitis platform execution model and XRT
- Describing kernel development using C/C++ and RTL
- Utilizing the Vitis analyzer tool to analyze reports
- Explaining the design methodology to optimize a design

What's New for 2022.1

- Amdahl's Law: New module and lab
- All labs have been updated to the latest software versions

Level – AI 3

Course Details

- 3 days ILT
  - 24 lectures
  - 11 labs

Course Part Number – AI-ACCEL

Who Should Attend? – Anyone who needs to accelerate their software applications using FPGAs, SoCs (such as Zynq®-7000 SoCs, Zynq UltraScale+™ MPSoCs), and Versal® ACAPs

Prerequisites

- Basic knowledge of Xilinx FPGA architecture
- Comfort with the C/C++ programming language
- Software development flow

Software Tools

- Vitis unified software environment 2022.1

Hardware

- Architecture: Xilinx Alveo accelerator cards, SoCs, and ACAPs
- Demo board: Zynq UltraScale+ MPSoC ZCU104 board

After completing this comprehensive training, you will have the necessary skills to:

- Describe how the FPGA architecture lends itself to parallel computing
- Explain how the Vitis unified software environment helps software developers to focus on applications
- Describe the Vitis (OpenCL API) execution model and XRT native APIs
- Analyze the OpenCL API memory model
- Create kernels from C, C++, or RTL IP using the RTL Kernel Wizard
- Apply host code optimization and kernel optimization techniques
- Move data efficiently between kernel and global memory
- Profile the design using the Vitis analyzer tool

Course Outline

Day 1

Vitis Tool Flow

- **Introduction to the Vitis Unified Software Platform**
  Explains how software/hardware engineers and application developers can benefit from the Vitis unified software environment and OpenCL framework. {Lecture}
- **Amdahl's Law**
  Provides insights into selecting functions for acceleration by illustrating its impact on system performance. {Lecture, Lab}
- **Vitis IDE Tool Overview**
  Describes the elements of the development flow, such as software emulation, hardware emulation, and system run as well as debugging support for the host code and kernel code. {Lecture, Labs}
- **Vitis Command Line Flow**
  Introduces the Vitis environment makefile flow where the user manages the compilation of host code and kernels. {Lecture, Labs}

Basics of Hardware Acceleration

- **Introduction to Hardware Acceleration**
  Outlines the fundamental aspects of FPGAs, SoCs, and ACAPs that are required to guide the Vitis tool to the best computational architecture for any algorithm. {Lecture}

Alveo Data Center Accelerator Cards

- **Alveo Data Center Accelerator Cards Overview**
  Describes the Alveo Data Center accelerator cards and lists the advantages of these cards and the available software solutions stack. {Lecture}
- **Getting Started with Alveo Data Center Accelerator Cards**
  Describes the hardware and software installation procedures for the Alveo Data Center accelerator cards. {Lecture}

Day 2

Vitis Execution Model and XRT

- **Vitis Execution Model and XRT**
  Describes the XRT and the OpenCL APIs used for setting up the platform, executing the target device, and post-processing. {Lecture, Lab}
- **Xilinx Runtime Library (XRT) Native APIs**
  Describes the XRT native APIs used for opening a device, loading XCLBIN, creating buffers, executing a kernel, and controlling a graph. {Lecture, Lab}
- **Synchronization**
  Describes OpenCL synchronization techniques such as events, barriers, blocking write/read, and the benefit of using out-of-order execution. {Lecture, Lab}
- **Xilinx Card Utilities**
  Describes the various tool utilities available with the Xilinx Runtime (XRT), such as the Xilinx board, board management, and xclbin utilities. The various commands and usage of these utilities are also covered. {Lecture}

NDRange (Optional)

- **Introduction to NDRanges**
  Explains the basics of NDRange (N dimensional range) and the OpenCL execution model that defines how kernels execute with the NDRange definition. {Lecture}

- **Working with NDRanges**

  Outlines the host code and kernel code changes with respect to NDRange. Also explains how NDRange works and the best way to represent the work-group size for the FPGA architecture. {Lecture}

Design Analysis

- **Profiling**

  Describes the different reports generated by the tool and how to view the reports that help to optimize data transfer and kernel optimization using the Vitis analyzer tool. {Lecture}
- **Debugging**

  Explains the support for debugging host code and kernel code as well as tips to debug the system. {Lecture}

Kernel Development

- **Introduction to C/C++ based Kernels**

  Describes the trade-offs between C/C++, OpenCL, and RTL applications and the benefits of C-based kernels. {Lecture, Lab}

Day 3

Kernel Development

- **Using the RTL Kernel Wizard to Reuse Existing IP as Accelerators**

  Describes how the Vitis unified software development provides RTL kernel developers with a framework to integrate their hardware functions into an application running on a host PC connected to an FPGA via a PCIe® interface. {Lecture, Lab}

Optimization Methodology Guide

- **Optimization Methodology**

  Describes the recommended flow for optimizing an application in the Vitis unified software development environment. {Lecture}
- **C/C++ based Kernel Optimization**

  Reviews different techniques such as loop unrolling, pipelining, and DATAFLOW. {Lecture}
- **Host Code Optimization**

  Describes various optimization techniques, such as reducing the overhead of kernel enqueing and optimizing the data transfer between kernels and global memory. {Lecture}
- **Optimizing the Performance of the Design**

  Illustrates various optimization techniques, such as optimizing the host code and data transfer between kernels and global memory, to improve kernel performance. {Lab}

Libraries

- **Vitis Accelerated Libraries**

  Describes the Vitis accelerated libraries that are available for domain-specific and common libraries. These libraries are open-source, performance-optimized libraries that offer out-of-the-box acceleration. {Lecture}

Platform Creation

- **Creating a Vitis Embedded Acceleration Platform (Edge)**

  Describes the Vitis embedded acceleration platform, which provides product developers an environment for creating embedded software and accelerated applications on heterogeneous platforms based on FPGAs, Zynq SoCs, and Alveo data center cards. {Lecture}

Appendix

Alveo Data Center Accelerator Cards

- **Alveo Accelerator Card Ecosystem Partner Solutions (Optional)**

  Outlines the partner solutions available in the cloud and on premises for Alveo Data Center accelerator cards. {Lecture}
- **Xilinx Real-Time Video Server Appliance (Optional)**

  Describes the Xilinx Real-Time Video Server appliance reference architectures, the optimized software solution stack for video applications, and various features offered by Alveo card live transcoding. {Lecture}