AMD
together we advance_

Embedded Heterogeneous Design

EMBD-HET (v1.0)

Course Specification

Course Description

This course covers the AMD Versal™ architecture and illustrates the tool flow for developing HLS and AI Engine components as well as integrating an entire system project to design an embedded heterogeneous system using the v++ tools and AMD Vitis™ Unified IDE.

The emphasis of this course is on:

- Describing an embedded heterogeneous system design
- Illustrating the AMD Versal adaptive SoC architecture, NoC, and AI Engine
- Describing an AMD Versal design tool flow
- Developing HLS and AIE components using the AMD Vitis tool
- Utilizing the v++ command line tools for component compilation, linking, and packaging to run emulation
- Demonstrating the system design flow for a heterogeneous embedded system using the AMD Vitis Unified IDE

Level – Embedded Software 4

Course Details

- 2 day ILT /29

Course Part Number – EMBD-HET

Who Should Attend? – Software and hardware developers, system architects, and anyone who needs to accelerate their software applications using AMD devices

Prerequisites

- Comfort with the C/C++ programming language
- Software development flow
- AMD Vitis tool flow

Software Tools

- AMD Vitis Unified IDE 2023.2

Hardware

- Architecture: AMD Versal adaptive SoCs

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

- Identify the resources available in AMD adaptive SoC architectures
- Describe the processes for developing the components for each type of resource
- Describe the optimization and debug methodologies for each component
- Develop components and link and package a system design using the v++ tools
- Assemble a complete system and run hardware emulation using the AMD Vitis Unified IDE

Course Outline

Day 1

- **Introduction to Embedded Heterogeneous Design**
  Defines what an embedded heterogeneous design is and identifies the AMD SoCs that support these types of designs. Also describes the tools required to develop embedded heterogeneous designs with AMD Versal devices. {Lecture}
- **Versal Adaptive SoC - Architecture Overview**
  Provides a high-level overview of the Versal architecture, illustrating the various engines available in the Versal architecture. {Lecture}
- **Versal Adaptive SoC - NoC Architecture**
  Provides a deep dive into the sub-blocks of the NoC and how they are used. Describes how the NoC is accessed from the programmable logic. {Lecture}
- **Versal AI Engine Architecture**
  Introduces the architecture of the AI Engine and its components. {Lecture}
- **Versal AI Engine Memory and Data Movement**
  Describes the memory module architecture for the AI Engine and how memory can be accessed by the AI Engines in the AI Engine arrays. {Lecture}
- **Versal Adaptive SoC - Application Partitioning 1**
  Covers what application partitioning is and how an application can be accelerated by using the various compute domains in the Versal device. Also describes how different models of computation (sequential, concurrent, and functional) can be mapped to the Versal adaptive SoC. {Lecture}
- **Driving the AMD Vitis Unified IDE**
  Introduces the terminology and features of the Vitis Unified IDE and talks about the basic behaviors required to drive the Vitis Unified IDE to generate a C/C++ application. {Lecture}
- **Tool Flow for Heterogeneous Systems**
  Maps the various compute domains in the Versal architecture to the tools required and describes how to target them for final image assembly. {Lecture, Lab}
- **Introduction to Vitis High-Level Synthesis Components**
  Provides an overview of high-level synthesis (HLS), the Vitis Unified IDE for HLS development flow, and the verification advantage. {Lecture, Lab}
- **Vitis HLS Methodology and Optimization Techniques**
  Describes the different methodologies of Vitis HLS development and various optimization techniques for improving performance. {Lecture, Lab}

Day 2

- **AI Engine Programming: Kernels and Graphs**
  Describes AI Engine kernels and adaptive data flow (ADF) graphs along with their programming flows. {Lecture, Lab}
- **Analyzing AI Engine Design Reports Using the Vitis Unified IDE**
  Describes the different reports generated by the tool and how to view the reports that help to optimize and debug AI Engine kernels using the Analysis view in the Vitis Unified IDE. {Lecture}
- **Versal AI Engine Application Debug and Trace**
  Shows how to debug an AI Engine application running on the Linux® OS and how to debug via hardware emulation that allows simulation of the application. {Lecture}
- **Development Using the v++ Command Line Tools**
  Illustrates the v++ command line tool flow for compiling AI Engine designs and HLS kernels and linking them for use on a target platform. Package a design to run software/hardware emulation is also covered. {Lecture, Lab}

- **Custom Platform Development**
  Describes the custom platform creation process using the AMD Vivado™ IP integrator, RTL, HLS, and Vitis environment. {Lecture, Lab}
- **Embedded Heterogeneous System Design Flow**
  Demonstrates the Vitis compiler flow to integrate a compiled AI Engine design graph (`libadf.a`) with additional kernels implemented in the PL region of the device (including HLS and RTL kernels) and linking them for use on a target platform. These compiled hardware functions can then be called from a host program running in the Arm® processor in the Versal device or on an external x86 processor. {Lecture, Lab}