AMD XILINX

Designing with Versal AI Engine: Architecture and Design Flow (1)

ACAP-AIE1 (v1.0)

Course Specification

Course Description

This course describes the Versal® AI Engine architecture, how to program the AI Engines (single kernel programming and multiple kernel programming using data flow graphs), the data communications between the PL and AI Engines, and how to analyze the kernel program using various debugger features.

The emphasis of this course is on:

- Illustrating the AI Engine architecture
- Designing single AI Engine kernels using the Vitis™ unified software platform
- Designing multiple AI kernels using data flow graphs with the Vitis IDE
- Reviewing the data movement between AI Engines, between AI Engines via memory and DMA, and between AI Engines to programmable logic (PL)
- Analyzing and debugging kernel performance

What's New for 2022.1

- Added information on AI Engine support for broadcast windows
- Added information on external traffic generators
- Introduced the upcoming AI Engine-ML architecture
- Most modules: Updated the content based on the enhanced programming model
- Labs have been updated with the enhanced programming model
- All the labs have been updated to the latest software versions

Level – ACAP 2

Course Details

- 2 days ILT
- 13 lectures
- 4 labs

Price –

Course Part Number – ACAP-AIE1

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

Prerequisites

- Comfort with the C/C++ programming language
- Software development flow
- Vitis software for application acceleration development flow

Software Tools

- Vitis unified software platform 2022.1

Hardware

- Architecture: Xilinx Versal ACAPs

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

- Describe the Versal ACAP architecture at a high level
- Describe the various engines in the Versal ACAP device and the motivation behind the AI Engine
- Describe the architecture of the AI Engine
- Describe the memory access structure for the AI Engine
- Describe the full application acceleration flow with the Vitis tool
- Enumerate the toolchain for Versal AI Engine programming
- Explain what intrinsic functions and AI Engine APIs are
- Program a single AI Engine kernel using the Vitis IDE tool
- Program multiple AI Engine kernels using Adaptive Data Flow (ADF) graphs
- Use Vitis Model Composer for AI Engine kernel development and modeling of a heterogeneous device

Course Outline

Day 1

Versal ACAP Architecture

- **Overview of the Versal ACAP Architecture**

  Provides an overview of the Versal architecture at a high level and describes the various engines in the Versal ACAP, such as the Scalar Engines, Adaptable Engines, and Intelligent Engines. Also describes how the AI Engine in the Versal ACAP meets many dynamic market needs. {Lecture}

Versal AI Engine Architecture

- **Introduction to the Versal AI Engine Architecture**

  Introduces the architecture of the AI Engine and describes the AI Engine interfaces that are available, including the memory, lock, core debug, cascaded stream, and AXI-Stream interfaces. {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}

Vitis Tool Flow

- **Versal ACAP Tool Flow**

  Reviews the Vitis tool flow for the AI Engine and demonstrates the full application acceleration flow for the Vitis platform. {Lecture, Lab}

Design Analysis

- **Application Partitioning on Versal ACAPs 1**

  Covers what application partitioning is and how an application can be accelerated by using various compute engines in the Versal ACAP. Also describes how different models of computation (sequential, concurrent, and functional) can be mapped to the Versal ACAP. {Lecture}

The Programming Model

- **Scalar and Vector Data Types**

  Provides an AI Engine functional overview and identifies the supported vector data types and high-width registers for allowing single-instruction multiple-data (SIMD) instructions. {Lecture}

- **AI Engine APIs and Intrinsic Functions**

  Describes what intrinsic functions are, the three types of vector management operations using intrinsic functions and AI Engine APIs (load and store, element conversion, and lane insertion/extraction), multiplication functions, and application-specific functions. {Lecture}

Day 2

Design Analysis

- **Vitis Analyzer**

  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 Vitis analyzer tool. {Lecture}

The Programming Model

- **Window and Streaming Data APIs**

  Describes window and streaming APIs and reviews the various window operations for kernels. Also discusses using overlapping data and various data movement use cases. {Lecture}

- **The Programming Model: Single Kernel**
  Reviews the AI Engine kernel programming flow for programming and building a single kernel. Also illustrates the steps to create, compile, simulate, and debug a single kernel program using the Vitis IDE tool. {Lecture, Lab}
- **The Programming Model: Single Kernel Using Vector Data Types**
  Illustrates Versal AI Engine kernel programming in detail, reviewing the scalar kernel code and comparing with vector kernel code that utilizes intrinsic functions and vector data types. {Lab}
- **The Programming Model: Introduction to the Adaptive Data Flow (ADF) Graph**
  Provides the basics of the data flow graph model and graph input specifications for AI Engine programming. Also reviews graph input specifications, such as the number of platforms and ports. {Lecture}
- **The Programming Model: Multiple Kernels Using Graphs**
  Describes the ADF graph in detail and demonstrates the steps to create a graph and set the runtime ratio and graph control APIs from the main application program. {Lecture, Lab}

Vitis Tool Flow

- **Vitis Model Composer for AI Engine Development**
  (On-Demand – Coming Soon)
  Introduces Vitis Model Composer, a Xilinx toolbox for the MATLAB® and Simulink® software, and how it can help with developing AI Engine kernels and modeling a heterogeneous device. {Lecture}