COURSE TABLE OF CONTENTS

Designing with Versal AI Engine: Architecture and Design Flow - 1

OEM: AMD • Code: AIE-ARCH

COURSE DESCRIPTION

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

AUDIENCE PROFILE

- Software and hardware developers, system architects, and anyone who needs to accelerate their software applications using our devices

PREREQUISITES

- Comfort with the C/C++ programming language
- Software development flow
- Vitis application acceleration development flow
- Vitis Unified IDE 2026.1
- Architecture: Versal adaptive SoCs

COURSE OBJECTIVES

By the end of this course, participants will be able to:

- Describe the AMD Versal adaptive SoC architecture and the various compute domains at a high level and the motivation behind the AI Engine
- Describe the architecture of the AI Engine
- Enumerate the various interfaces available with the AI Engine
- Describe the data movement and memory access structure for the AI Engine
- Enumerate the toolchain for Versal AI Engine programming and describe the full application acceleration flow with the AMD Vitis Unified IDE
- Explain what AI Engine APIs are
- Program a single AI Engine kernel using scalar and vector data types using the Vitis tool
- Program multiple AI Engine kernels using adaptive data flow (ADF) graphs
- Analyze the performance and implementation reports of an AI Engine design using the Vitis Unified IDE
- Describe the architecture of the AIE-ML
- Describe the programming model for the AIE-ML

- Describe the architecture of AIE-MLv2 with the Versal AI Edge Series Gen 2 devices

COURSE OUTLINE

Module 1: Day 1

AMD Versal Adaptive SoC Architecture

- Overview of the Versal Adaptive SoC Architecture: Provides an overview of the Versal architecture at a high level and describes the various compute domains in the Versal device, such as the processor system, programmable logic, and AI Engines. Also describes how the AI Engine in the Versal device meets many dynamic market needs.

Versal AI Engine (AIE) Architecture

- Versal AI Engine Architecture: Introduces the architecture of the AI Engine and its components.
- AI Engine Interfaces: Describes the AI Engine interfaces that are available, including the memory, lock, core debug, cascaded stream, and AXI-Stream interfaces.
- 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.

Design Analysis

- Versal Adaptive SoC: Application Partitioning and Mapping: Covers the system design methodology and describes how different models of computation (sequential, concurrent, and functional) can be mapped to the Versal adaptive SoC. Also describes what application partitioning is and how an application can be accelerated by using the various compute domains in the Versal device.
- 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. Also covers estimating performance using the Analysis view in the Vitis Unified IDE.

Vitis Tool Flow

- Versal AI Engine Tool Flow: Reviews the Vitis tool flow for the AI Engine and demonstrates the full application acceleration flow for the Vitis platform.

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.
- AI Engine APIs: Describes what AI Engine APIs are, the three types of vector manipulation operations using AI Engine APIs (load and store, element conversion, and lane insertion/extraction), multiplication functions, and application-specific functions.
- Buffers and Streaming Data APIs: Discusses the input/output buffers and streaming data APIs and reviews the various buffer operations for kernels.
- 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 Unified IDE.

Module 2: Day 2

The Programming Model

- 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 AI Engine APIs and vector data types.
- 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.
- 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.

Versal AI Engine (AIE-ML) Architecture

- Introduction to the AIE-ML Architecture: Introduces the architecture of the AIE-ML at different levels, such as the array, tiles, and components.
- AIE-ML Memory Tiles and Programming: Describes the memory tiles in the AIE-ML architecture and illustrates AI Engine-ML programming using both shared buffers and external buffers with the AIE-ML.
- Introduction to the AIE-MLv2 Architecture: Introduces the architecture of the AIE-ML v2 for the Versal AI Edge Series Gen 2 devices at different levels, such as the array, tiles, and components.