AMD together we advance

Morgan Advanced Programmable Systems

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

AIE-ARCH (v1.0)

Course Specification

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.

The emphasis of this course is on:

- Describing the AI Engine (AIE) architecture
- Illustrating the Versal AI Engine tool flow
- Designing with single AI Engine kernels and analyzing the performance of scalar and vectorwized kernels using the Vitis™ unified software platform
- Designing with multiple AI Engine kernels using data flow graphs with the Vitis Unified 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
- Describing the AIE-ML architecture
- Illustrating the programming model for the AIE-ML

What's New for 2024.2

- Versal AI Engine Memory and Data Movement module: Added information on array interface topologies and bandwidth performance
- Versal AI Engine Tool Flow module: Added information on the Advanced Flow now available in the AMD Vivado™ Design Suite
- Analyzing AI Engine Design Reports module: Added information on estimating throughput and latency using the Analysis View of the AMD Vitis™ Unified IDE
- All the labs have been updated to the latest software versions

Level – ACAP 2

Course Details

- 2 days live instructor led training (online or in person)
- 15 lectures
- 6 labs

Price – $1,600 or 16 AMD Training Credits

Course Part Number – AIE-ARCH

Who Should Attend? – 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

Software Tools

- Vitis Unified IDE 2024.2

Hardware

- Architecture: Versal adaptive SoCs

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

Course Outline

Day 1

Versal Adaptive SoC Architecture

- **Overview of the AMD 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. {Lecture}

Versal AI Engine (AIE) Architecture

- **Versal AI Engine Architecture**

  Introduces the architecture of the AI Engine and its components. {Lecture}
- **AI Engine Interfaces**

  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}

Design Analysis

- **Versal Adaptive SoC: Application Mapping and Partitioning**

  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. {Lecture}

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. {Lecture, Labs}

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**

  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. {Lecture}
- **IO Buffers and Streaming Data APIs**

  Discusses the input/output buffers and streaming data APIs and reviews the various buffer operations for kernels. {Lecture}

Design Analysis

- **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}

The Programming Model

- **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. {Lecture, Lab}

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. {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}

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. {Lecture}
- **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. {Lecture, Lab}

Register Today

Morgan Advanced Programmable Systems, Inc. (Morgan A.P.S.) delivers public and private courses in locations throughout the central US region; including Iowa, Illinois, Kansas, Minnesota, Missouri, Nebraska, North Dakota, South Dakota, and Wisconsin.

Visit **morgan-aps.com/training**, for full course schedule and training information.

- You must have your tuition payment information available when you enroll. We accept credit cards (Visa, MasterCard, or American Express) as well as purchase orders and Xilinx training credits.

Student Cancellation Policy

- Student cancellations received more than 7 days before the first day of class are entitled to a 100% refund. Refunds will be processed within 14 days.
- Student cancellations received less than 7 days before the first day of class are entitled to a 100% credit toward a future class.
- Student cancellations must be sent here.

Morgan A.P.S. Course Cancellation Policy

- We regret from time-to-time classes will need to be rescheduled or cancelled.
- In the event of cancellation, live on-line training may be offered as a substitute.
- Morgan A.P.S. may cancel a class up to 7 days before the scheduled start date of the class; all students will be entitled to a 100% refund.
- Under no circumstances is Morgan A.P.S. responsible or liable for travel, lodging or other incidental costs. Please be aware of this cancellation policy when making your arrangements.
- For additional information or to schedule a private class contact us here.

Online or in person training with real hardware

- Morgan Advanced Programmable Systems, Inc. has set up a training VPN where engineer participants can take classes online using the same computers and devCards used during in-person training.
- Even better, and upon request, you can use these computers after hours on training days to experiment with labs. This is not possible for in-person training.
- Additionally, just like in-person training, the laptops and devCards, tools, OS, and licensing are set up in advance.
- In some ways, live online-training is better than in-person…for example, you can grant the instructor permission to look at your Vivado, PetaLinux terminal, or Vitis for extended periods of time if your lab is not going exactly has planned to a missed step.
- This is often more comfortable than two engineers crowding around a laptop screen.
- Taking remote training also allows you to learn some tips and tricks for working remote. Whether your devCard is in the lab down the hall, or across the world via VPN, you can control your Xilinx based device quickly and efficiently.