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 when designing 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

What's New for 2024.1

- New module on AI Engine system partitioning
- All labs have been updated to the latest software versions

Level – Embedded Software 4

Course Details

- 2 days live instructor led training (online or in person)
- 16 lectures
- 7 labs

Price – $1,600 or 16 AMD Training Credits

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

Related

- Xilinx Rapid Development Embedded Design
- AMD Vitis tool flow

Software Tools

- AMD Vitis Unified IDE 2024.1
- AMD Vivado™ Design Suite 2024.1

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 AMD 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 compute resources 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}
- **Introduction to the AMD Versal AI Engine Architecture**
  Describes the Versal adaptive SoC at a high level and introduces the architecture of the AI Engine as well as its memory modules and interfaces. {Lecture}
- **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}
- **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 AMD Vitis High-Level Synthesis (HLS) Components**
  Provides an overview of high-level synthesis, 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}
- **AI Engine System Partitioning Methodology**
  Describes the AI Engine system partitioning and planning methodology and mapping system requirements. {Lecture}
- **Analyzing AI Engine Designs Using the Vitis Analyzer**
  Covers the different reports generated by the Vitis Unified IDE and how to use these reports to optimize and debug AI Engine kernels. {Lecture}
- **Versal AI Engine Application Debug and Event Trace**

Shows how to debug the AI Engine design in an application and how to debug via hardware emulation. {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. Packaging 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}

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.

Morgan Advanced Programmable Systems

- 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.