AMD
together we advance_

CUDA to HIP/ROCm Migration

MGRT-HIP (v1.0)

Course Specification

Course Description

This course provides a structured, end-to-end approach for migrating GPU-accelerated applications from NVIDIA CUDA to the AMD ROCm™ platform using the Heterogeneous-computing Interface for Portability (HIP) programming model. Participants gain a comprehensive understanding of the AMD RDNA™ GPU architecture, the ROCm software stack, HIP programming, and the tools required to port, debug, profile, and optimize real-world GPU workloads.

The course emphasizes architectural mapping between CUDA and HIP, practical migration workflows, performance optimization, and correctness validation to ensure a smooth and efficient transition to supported AMD GPU platforms.

The emphasis of this course is on:

- Understanding the AMD RDNA GPU architecture and single instruction, multiple threads (SIMT) execution model
- Exploring the ROCm platform and its open GPU computing ecosystem
- Optimizing memory access, synchronization, and kernel execution on RDNA GPUs
- Applying the HIP programming model for portable GPU development
- Mapping CUDA concepts, APIs, and toolchains to their HIP/ROCm equivalents
- Executing a structured CUDA-to-HIP migration workflow utilizing validation and optimization strategies
- Debugging GPU kernels using ROCgdb at wavefront and lane granularity
- Profiling GPU and system performance using ROCm profiling tools

Level – 1

Course Details

- 2 days ILT or 3 sessions/19

Course Part Number – MGRT-HIP

Who Should Attend?

- CUDA developers migrating applications to AMD GPUs
- GPU performance engineers and HPC application developers optimizing and porting GPU workloads
- AI/ML engineers developing portable GPU workloads
- System architects evaluating AMD GPU platforms

Prerequisites

- Intermediate knowledge of CUDA programming
- Understanding of GPU programming concepts (kernels, threads, memory hierarchy)
- Proficiency in C/C++ development
- Familiarity with Linux® OS-based development environments

Software Tools

- ROCm platform
- HIP runtime and compiler toolchain
- ROCgdb
- ROCProfiler-SDK / rocprofv3
- hipify tools

Hardware

- Strixhalo laptop (or) Korat+ board

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

- Describe AMD RDNA GPU execution concepts and the AMD ROCm software stack
- Develop and compile HIP applications targeting AMD GPUs
- Port CUDA applications to HIP by mapping APIs, execution models, and memory concepts
- Build and validate migrated applications on the ROCm platform
- Optimize and debug HIP kernels using ROCm debugging and profiling tools
- Profile GPU and system performance on the RDNA architecture

Course Outline

Day 1

Foundations and Architecture

- **GPU Fundamentals: Understanding RDNA Architecture and Execution**

  Describes GPU execution using SIMT, wavefronts, and scheduling. Also discusses how the RDNA architecture organizes compute, memory, and execution resources to achieve high throughput. {Lecture}
- **Overview of the ROCm Platform**

  Introduces the ROCm platform as an open GPU computing stack, explains the HIP portability model, and identifies key software components, including the runtime, libraries, and developer tools. {Lecture}
- **RDNA GPU Memory Hierarchy**

  Outlines the RDNA memory hierarchy and its performance implications, reviews optimization techniques using global memory, local data share (LDS), and registers, and covers synchronization using barriers, fences, and atomics. {Lecture}

HIP Programming Fundamentals

- **Introduction to HIP Programming Basics**

  Introduces the HIP programming model, including host-device interaction and memory usage. Also covers core HIP runtime APIs for managing GPU execution. {Lecture, Lab}
- **HIP Compilation and Toolchain**

  Explains the HIP compilation flow, host-device code separation, and key toolchain components. Also compares ahead-of-time, runtime (hipRTC), and multi-architecture compilation strategies. {Lecture}
- **HIP Kernel Programming**

  Describes HIP kernel structures, thread hierarchy, and indexing models, explains kernel launch configuration, synchronization, and memory movement, and reviews common kernel patterns in application workflows. {Lecture, Lab}

Day 2

Migration and Porting

- **CUDA to HIP/ROCm Platform: Concept Mapping and Migration Considerations**

  Compares CUDA and HIP programming models, execution semantics, and toolchains. Also identifies performance tuning, debugging, and validation strategies for successful migration. {Lecture}
- **HIP Porting Overview: Concepts, Workflow, and Best Practices**

Provides a structured workflow for porting CUDA applications to HIP. Also outlines build, validation, and cross-platform portability best practices. {Lecture, Lab}

Debugging, Profiling, and Optimization

- **Debugging RDNA GPU Kernels Using ROCgdb**
  Describes a unified CPU-GPU debugging approach for RDNA architectures. Also demonstrates wavefront- and lane-level debugging techniques, including breakpoints and memory inspection. {Lecture, Lab}
- **ROCm Profiling on the RDNA Architecture**
  Explains GPU profiling methodologies using ROCProfiler and rocprofv3 and differentiates GPU execution profiling from system-level performance analysis. {Lecture, Lab}

.