Teaching Guidelines for
Advanced Programming in HPC
ACC-HPC June 2025

Duration: classroom hours 36 + 36 lab hours

Objective: To introduce the student to Accelerator Programming OpenACC, OpenCL/SYCL ,CUDA Programming and Tools.

Prerequisites: Knowledge of programming in any language like C, C++, basic Mathematics, statistics.

Evaluation method: CCEE exam– 40% weightage

Lab exam – 40% weightage

Internal exam – 20% weightage

List of Books / Other training material

Reference Book: 1. OpenACC for Programmers: Concepts and Strategies by Sunita Chandrasekaran,

Guido Juckeland

2. Programming in Parallel with CUDA: A Practical Guide by Richard Ansorge

3. Hands-On Gpu Programming with Python and Cuda: Explore high-performance parallel computing with CUDA by Dr Brian Tuomanen Dr. Brian Tuomanen (Author) Python Power: The Comprehensive Guide

4. Learn CUDA Programming: A beginner's guide to GPU programming and parallel computing with CUDA 10.x and C/C++ by Jaegeun Han, Bharatkumar Sharma

Tools for High Performance Computing: Proceedings of the 2nd International Workshop on Parallel Tools for High Performance Computing, July 2008, HLRS, Stuttgart by Rainer Keller, Valentin Himmler, Bettina Krammer , Alexander Schulz

Note: Each session mentioned is for theory and of 2 hours duration.

Lab assignments are indicatives, faculty need to assign more assignments for better practice.

Introduction to Accelerator Programming

Session 1:

- Introduction GPU & GPGPU

Session 2:

- Performance of GPU over CPU

Session 3:

- GPGPU hardware architecture (NVIDIA – AMD)
- Application of Accelerator Programming

Lab Assignment

OpenACC

Session 4:

- Introduction to OpenACC
- Parallelize & Optimize Loops

Lab Assignment

Session 5

- Directive & Clauses
- GPUs and Multi-Core CPUs

Lab Assignment

OpenCL/SYCL

Session 6,7,8 & 9

- Introduction to OpenCL/SYCL, Host Program & Device Program, Single-source programming for both host and device code, Queues to manage execution and synchronization, Buffers and accessors for data management between the host and device, Kernel programming, Parallelism via constructs, Platform portability across CPUs, GPUs, FPGAs, and other accelerators.

  Lab: Case-Studies (Algorithms and Parallelization Approaches): Matrix – Multiplication

CUDA

Session 10

- NVIDIA GPGPU Architecture

Lab Assignment

Session 11 & 12

- GPGPU computing with CUDA

Lab Assignment

Session 13 & 14

- Thread & Memory Hierarchy
- CUDA constructs

Lab Assignment

Tools

Session 15:

- Profiling: GNU (gprof), Intel VTune
- Debugger: GDB (GNU)

Lab Assignment

Session 16:

- Performance measurement and optimization
- Algorithm, memory & network, parallel Processing, compiler flags, libraries (BLAS, LAPACK, MKL)

Lab Assignment

Session 17:

- Benchmarking
- Introduction to micro-architecture analysis tools: LIKWID

Lab Assignment

Session 18:

- Intel Advisor
- Demo of the sample code by using the above tools

Lab Assignment