CSE

CS836GPU ARCHITECTURE AND PROGRAMMING

I

uction - GPUs as Parallel Computers - Architecture of a modern GPU - Why more speed or lism? - Parallel Programming Languages and Models - Overarching Goals – History of GPU ting - Evolution of Graphics Pipelines - GPU Computing

II

el Programming: Goals of Parallel Programming - Problem Decomposition - Algorithm Selection putational Thinking – Introduction to OPENCL: Background – Data Parallelism Model – Device tecture – Kernel Functions - Device Management & Kernel Launch

III

uction to CUDA : Data Parallelism - CUDA Program Structure - A Matrix–Matrix Multiplication ple - Device Memories and Data Transfer - Kernel Functions and Threading - Function ations - Kernel launch - Predefined variables - Runtime API – CUDA Threads : CUDA Thread ization - Using blockIdx and threadIdx - Synchronization and Transparent Scalability - Thread ment - Thread Scheduling and Latency Tolerance – CUDA Memories : Importance of Memory s Efficiency - CUDA Device Memory Types - A Strategy for Reducing Global Memory Traffic - ry as a Limiting Factor to Parallelism

IV

mance considerations: Thread execution – Global memory bandwidth – Dynamic partitioning of sources – Data prefetching - Instruction mix – Thread Granularity- Floating Point considerations: mat – Representable numbers – Special bit patterns and precision – Arithmetic accuracy and ng – Algorithm considerations – Debugging and Profiling: Debugging CUDA programs – ng CUDA programs – CUDA and MPI

V

ch papers from the following journals and conferences from 2013-2015:

ransactions, Elsevier, IEEE/ACM MICRO, High Performance Computer Architecture (HPCA), erformance Computing & Simulation (HPCS);