Applied Parallel Computing LLC
http://parallel-computing.pro
+41 78 925 90 90
contact@parallel-computing.pro

4-day Course on GPU Computing at Continental Corporation

(14–17 January 2019, Regensburg)

Exploiting the potential of GPU computing is inevitable for any modern HPC application. As two leaders in the application of compute & deep learning technologies, NVIDIA and AMD set up the quality standard for massively parallel hardware, development tools and GPU-accelerated libraries. This course provides essential practical experience for scientists to develop, debug and optimize fast and efficient research codes with CUDA and OpenCL frameworks.

All corresponding presentations and code samples will be available to attendees in printed handouts. The hands-on sessions are conducted on remote Jupyter-based platform equipped with NVIDIA Tegra TX2, NVIDIA Tesla V100 and AMD FirePro w9100 GPUs. Attendees may choose to connect the hands-on platform either from their personal laptops, or from a computer classroom provided by customer.

Applied Parallel Computing LLC is an authorized consultancy partner of NVIDIA and AMD delivering GPU training courses in Europe since 2009.

Day 1: Introduction to CUDA and GPU/CUDA libraries

Morning (09:00-12:30)

09:00-11:00: Introduction to CUDA

- Introduction to GPU computing
- CUDA principles and CUDA implementation for C++
- Analogies between MPI+OpenMP and CUDA programming models
- The first CUDA program explained
- CUDA compute grid, examples
- Realistic CUDA application example (wave propagation code)
- Understanding GPU compute capabilities, *deviceQuery*
- Basic optimization techniques
- Overview of CUDA applications development using Visual Studio 2017 and Eclipse IDE

11:15-13:00: Hands-on session

- **Hands-on:** Write & deploy a simple CUDA program
- **Hands-on:** More control on CUDA compute grid

13:00-14:00: Lunch

Afternoon (14:00-18:00)

14:00-15:30: Hands-on session

- **Hands-on:** Write & deploy bilinear image interpolation in CUDA

15:30-16:30: GPU-enabled libraries

- Thrust – the C++ library of GPU-enabled parallel algorithms
- CUBLAS, MAGMA, CUBLAS-XT, CUSPARSE, CUFFT and CURAND
- CUSP and AmgX – Krylov and multigrid solvers

16:45-18:00: Hands-on session

- **Hands-on:** solving Poisson equation with CUFFT

Day 2: GPU memory hierarchy, advanced CUDA, optimization & profiling

09:00-10:30: GPU memory hierarchy

- GPU memory types
- Shared memory
- GPU caches hierarchy and mode switches
- Unified virtual address space (UVA)
- Streams and asynchronous data transfers
- Caveats of memory allocation
- Implications of integrated GPU sharing host memory

10:45-13:00: Hands-on session

- **Hands-on:** Understanding *bandwidthTest* results on NVIDIA Tegra TX2
- **Hands-on:** “fill-in” exercise on reduction with and without shared memory
- **Hands-on:** “fill-in” exercise on CUDA streams

13:00-14:00: Lunch

Afternoon (14:00-18:00)

14:00-15:30: Advanced CUDA

- CUDA 9 cooperative groups
- Warp-synchronous programming in CUDA 9
- Dynamic parallelism
- Warp shuffle instruction. Optimizing reduction with shuffles.
- CUDA C++ compiler pipeline, PTX assembler, SASS, NVVM backend
- Understanding “-Xptxas -v” reports

15:30-16:30: GPU code optimization

- An overview of Pascal/Parker, Volta/Xavier and Turing GPU architectures
- GPU optimizations: compute grid, coalescing, divergence, unrolling, vectorization, maxrregcount, aligning, floating-point constants

- Overview of *NVIDIA Visual Profiler*
- Overview of *nvprof* (command line profiler)
- Common practices of identifying performance hazards in GPU application using NVIDIA Visual Profiler

16:45-18:00: Hands-on session

- **Hands-on:** profile and optimize the bilinear interpolation kernel

Day 3: Introduction to OpenCL and GPU/OpenCL libraries

Morning (10:00-13:30)

09:00-10:30: Introduction to OpenCL and GPU architecture

- Basic OpenCL principles: compiler, host-device interop, kernel program
- Vector addition program step by step
- How OpenCL program maps onto GPU architecture, NDRange and wavefront
- Key OpenCL memory types (*CL_MEM_**)
- SYCL (OpenCL for C++)
- Terminology of CUDA and OpenCL
- NVIDIA and AMD GPUs: architectural differences
- Execution workflow: offline compilation in CUDA, runtime compilation in CUDA and in OpenCL
- CUDA and OpenCL intercomparison, by example of bilinear interpolation kernel
- *libgpurt* runtime library for unified CUDA and OpenCL support

10:45-13:00: Hands-on session

- Getting started with AMD ROCm runtime
- **Hands-on:** “fill-in” exercise on writing simple OpenCL program (*sine_calc*)
- Integrating OpenCL sources and SPIR-V into application binary
- Embedded OpenCL profile, by example of ARM-based RK3399 SoC

13:00-14:00: Lunch

Afternoon (14:00-18:00)

14:00-15:30: OpenACC[1]

- Advantages of OpenACC, programming model, execution model, memory model
- Main directives: *parallel*, *kernels*, *loop*
- Organizing data persistence regions
- Understanding compiler output, forcing data independence
- Profiling GPU kernels in OpenACC application
- Comparison of OpenACC and OpenMP 4

[1]Conducted with PGI OpenACC compiler, Radeon backend.

- AMD GPUs support in PGI OpenACC 16.10
- Dumping LLVM IR from PGI OpenACC and custom compiler toolchain construction

15:30-16:30: OpenCL architecture and memory hierarchy

- Comparison of different algorithms on CPU and GPU, theoretical and practical limitations
- AMD Firepro w9100 architechture. Resources and capabilities. GCN compute unit, GCN chip
- Hardware memory hierarchy. OpenCL memory mapping to hardware

16:45-18:00: Hands-on session

- **Hands-on:** “fill-in” exercise on implementing wave propagation stencil in OpenACC (*wave13pt*)
- **Hands-on:** “fill-in” exercise on implementing dense matrix-matrix multiply (GEMM): naive and tiled in local memory

Day 4: OpenCL libraries, advanced OpenCL, optimization & profiling. MultiGPU

09:00-10:30: GPU-enabled libraries

- *Bolt* (STL for GPU)
- *clMath* (BLAS and FFT), *clMagma* (LAPACK)
- *Eigen* - GPU-enabled eigenvalues library

10:45-13:00: Hands-on session

- **Hands-on:** “fill-in” exercise on producing image histogram with Bolt (*bolt_histogram*)
- **Hands-on:** “fill-in” exercise on key-value sorting with Bolt (*sort_by_key*)

13:00-14:00: Lunch

Afternoon (14:00-18:00)

14:00-15:30: Debugging and profiling

- Principles and terminology
- OpenCL flags for host and device code debug information availablility
- Configuring *codeXL* IDE for debugging
- Breakpoints, function calls history, statistics
- Profiling: performance counters, power profiling
- Static kernel analysis
- Analysing wavefronts layout and GPU resources consumption
- Live demonstration of *codeXL* debugging

15:30-16:30: Optimization

- Optimal global and local worksizes
- Compiler flags for fast math. Double vs float.

- *CL*_-extensions, *CL*_-types
- Atomics performance
- Pipes – FIFO data structures to send data between kernels; usage, limitations
- Live step-by-step optimization of hexapod application

16:45-17:45: MultiGPU

- OpenCL asynchronous execution, command queues.
- Basic principles of CPU multithreading and multiprocessing, by example of OpenMP and MPI.
- Demo: using multiple GPUs in genetic algorithm

17:45-18:00: Final Q & A