Online Workshop on the new NVIDIA HPC SDK at HPC Thüringen

(28 May 2021, 10:00–13:00)

The new NVIDIA HPC SDK significantly reshapes the diverse GPU metaprogramming approaches previously taken by OpenACC and Thrust/Bolt. With a new GPU-enabled ***std::par*** framework, a developer gets a vendor-neutral access to multicore CPUs and GPUs in a more unified fashion. In this webinar, we explain how to start using *std::par* in existing applications in typical software engineering scenarios. Furthermore, we focus on the new Nsight profiling tools, which are now free of a lot of limitations of the previous generation NVIDIA Visual Profiler. Finally, the webinar covers the new CUDA Fortran features as well as the Python bindings essential for any modern compute application.

The online session will be accompanied by an optional offline practical hands-on, Q&A and submissions review. Attendees will have access to a JupyterLab environment with NVIDIA GPUs for the duration of the sessions. All corresponding presentations and code samples will be available to attendees as a downloadable package.

Session 1 (10:00–11:30): New NVIDIA HPC SDK for More Productive GPU Computing

- New focuses of NVIDIA HPC SDK: performance, portability and productivity
- How new HPC SDK is different from CUDA Toolkit and PGI Compiler Suite
- C++17 std::par constructs as an alternative to OpenACC directives and CUDA Thrust
    - Essential elements: containers, iterators and lamda functions
    - Thurst-like counting iterators
    - Explicit and implicit memory transfers
- NVIDIA HPC SDK in the programming ecosystem
    - Intermixing HPC SDK and CUDA Toolkit
    - CMake support for NVC++ compiler
    - Switching between *std::par* backends, targeting different GPUs with *SyclParallelSTL*

Session 2 (11:40-13:00): Profiling, language and binding features of NVIDIA HPC SDK

- Next generation profiling tools: NSight Systems and NSight Compute
    - Connecting Nsight profilers to remote cloud GPUs
- Fortran 2003/2008 and CUDA Fortran in NVIDIA HPC SDK
- Python bindings for GPU applications
    - Using NVC++ compiler to build Python API around native GPU code
    - Preparing a pybind11 wrapper for C++17 std::par and CUDA Fortran applications