Testing: Strategies When Learning Programming Models and Using High-Performance Libraries
(the slides are available under “Presentation Materials” in the above URL)
Date: March 18, 2020
Presented by: Balint Joo (Jefferson Lab)

Q. Any good test programs that work directly on the GPU/Accelerator? Can SYCL take care of that? In other words, put your tests in the DPC++ code?

A.The speaker states that he was not able to run assertions directly on the device: instead put data in a buffer (SYCL) or some other form of memory which can later be visible from the host (e.g. UVM for CUDA, UVMSpace for Kokkos on CUDA) and then access from the host. However, the speaker has only used GoogleTest. Other Testing frameworks may work better. Checking things on the device does have some caveats, such as that any data/methods used in an assertion also compile and run on the accelerator. Speaker has not done any searching in the space of testing frameworks in order to answer this in more detail.

Q. In my experience, ALCF admin can install anything you need. When you said that AURORA does not have CUDA, why can't the admin install CUDA?

A. CUDA is specific to NVidia, and will not support Intel GPUs.

Q. Are you happy with Kokkos for performance portability?

A. On the whole Speaker has tested Kokkos in a QCD Mini App, and ran it on SIMD-Vector like multicore machines (Intel Knight’s Landing, Intel SkyLake) and typically there he had to take care of the SIMD-ization by hand (he wrote his own SIMD type). This seems less of an issue with Kokkos, than with compilers SIMD-izing Complex arithmetic. Modulo tweaks like that Speaker has found Kokkos to be good for the kind of performance portability he needed in his mini-app. Caveats to this are that some of the Kokkos Back-Ends (e.g. SYCL and AMD HIP) are recent and in development, and results there are preliminary. So the Speaker cannot yet comment authoritatively on how performant things will be on those back ends until they mature.

Q. Can you compare Kokkos / SYCL performance?

A. This can only be done in certain cases at the moment, and as always there are caveats. For example, Speaker has published tests on V100 GPUs, which used the POCL OpenCL driver and a particular revision of the Codeplay ComputeCPP SYCL compiler. This setup had comparable performance with Kokkos using the CUDA back-end and with the hand written QUDA library. Recently Speaker has tried the Codeplay variant of the Intel public LLVM DPC++/SYCL compiler which is based on CUDA directly rather than going through OpenCL.

This gave preliminary performances that were lower than Kokkos with the CUDA back end, but this is still being worked on and improvements both to the code and the compiler may reduce this discrepancy.

C. The last 2 questions have more detailed relevant information in the paper presented at SC-19 in the P3HPC workshop. The talk slides are available here (from the P3HPC workshop site) https://drive.google.com/file/d/1rBIzzdGWvVHrQKTwA44o8OLhOSA4nW2P/view