191125 - CCE PPS kickoff meeting

Attendance: Paolo, Meifeng, Torre, Matti, Mike, OLI, Charles, Mark, Tim, Salman

Contact for dune code: Kirby will make sure to have a contact within the WireCell team to help provide the software for testing

Logistics

- Samantha will setup indico for us
- We will setup a Box for documents
- Email lists will also be setup

Paolo: extracts from reviewers comments (the only mildly critical comment)

- *They certainly understand the algorithms and computations that need to be performed, but I am hoping they have access to other staff with deep experience in how to create portable parallelization strategies. They have certainly identified all of the leading programming environments (Kokkos, RAJA, etc.) and have a good plan to evaluate them, but additional experience would be helpful. Hopefully, they can acquire the needed expertise through some means.*

Constraints for the first phase of the project (Salman,Paolo)

- DOE has said that they fund the project, but not at what level
- Currently during CR, funding will not arrive before January (funding is not tied to the CR, but paperwork takes time)
- At the moment, we can do some scoping exercises
- Sort out, what people are available, what kind of expertise they bring to the table

Use cases

- Wire cell: available in container
    - New code for DUNE
    - Self-contained, C++17
    - Follows data flow programming model
    - Not multi-threaded in production, has TBB implementation
    - Main computational kernel: 2D deconvolution of images
    - Done performance analysis on
    - PoC CUDA implementation for signal processing part exists
    - GPU code available

- Patatrack (CMS): container, contains all of CMSSW, 40 GB
    - C++, multi-threaded, C++14 (NVCC) and C++17
    - CUDA directly with multiple CUDA streams
    - 40 kernels executing the seeding, Includes also vertexing
    - Transfer RAW data into the GPU, only transfer tracks and vertices back to the host
        - Input 250 kB, output is 8 MB, rest stays in the GPU, temporary data is 100 MB/event
    - Cannot fill a V100 with one event, using multiple events in parallel using CUDA streams
    - Partly same code executed on CPU and GPU
    - TBB used to schedule for multi-threading
    - Long term plan: use patatrack only on GPUs or both on CPU and GPU
        - Yes, both
- ACTS (ATLAS): self-contained, got it working with OpenMP, installation available
    - ROOT, python, eigen, boost
    - C++11
    - Not uses multi-threaded, used OpenMP (which scaled very nicely) in hackathon (PoC)
        - Doing parallelization within events
        - PoC concept implementation for GPU
    - Data sizes:
    - Paolo was also looking into using python and numbs to do tracking, also using GPUs
    - Will build a container

Portability solutions

- Missing: HIP (AMD programming model, more C than C++, close companion to CUDA, limited hardware compatibility)
- CUDA is not on the list, because it is not universal
- OpenMP
- Long term goal, everything we propose to use would be part of the C++ standard
- OneAPI
    - Changed the bottom layer of OneAPI
    - Used to be OpenCL, now it is called Layer0
    - Much harder to have cross platform support
    - Intel will help or themselves write driver backends for other hardware, time will tell
    - Time will tell if SYCL will develop into a cross platform standard
- Next 18 month, major changes are expected, hopefully converging
    - SYCL, CUDA
- Can get support from Kokkos, Raja developers
- Alpaka

    - Nobody in U.S. is working on it
    - Accepted for early access for Frontier
    - Former Alpaka developer is not at LBNL
    - Main goal is to have Alpaka running well on Frontier
    - Maybe this is going the route of HIP?

Paolo

- Future is uncertain, not important which tool to use first, only that it is supported well
- Important is the impact on the existing code, how much effort it will take to convert
- Simple application might not be enough to explore ➜ focus on the “amount of work” metrics
- Performance is 2nd order thing

Charles

- mini apps: there is gap between them and what the frameworks need to provide. Need to enumerate framework requirements.
    - Concurrent kernels?
    - Chain kernels?
    - Single source accelerator / CPU?
    - Virtual functions?

Matti

- Christian from Kokkos team gave numbers
- Developer could work on 10 lines of code, 20k lines of code per year
- 10% of an application would need to be rewritten
- Matti’s estimate for HEP code is substantial higher
- 600k lines of code ➜ 2-3 person years to port

Paolo

- have one example of a low level library
    - SYCL, CUDA (both
- One example of a higher level library
    - Kokkos

Monday’s at 11 AM

- Next week
    - Decide on high level library
    - Provide instructions and pointers for experiment code for people to look at