Detector Simulation in SWIFT-HEP

Ben Morgan

WARWICK
THE UNIVERSITY OF WARWICK

SWIFT-HEP Work Package 3 on Simulation

- **3.1: Build on ExCALIBUR-HEP (0.4FTE, 15 months) R&D on algorithms for Electromagnetic Physics/Calorimetry on NVidia GPUs**
  - *Build, Profile, Validate Geant4+GPU(AdePT/Celeritas) application to demonstrate hybrid CPU+GPU workflow (**0.5FTE, months 5-24, 25-36**)*
  - *Continue and strengthen collaborations with Geant4 R&D, CERN-SFT (**AdePT** Project) and US ECP (**Celeritas** Project)*
- **3.2: Contribute to development of example application using Geant4+Opticks for hybrid CPU+GPU optical photon simulation**
  - *In collaboration with FNAL and Geant4 R&D*
  - *Build UK links with, and expertise in, Opticks software*
  - ***See proceeding presentations for details!***

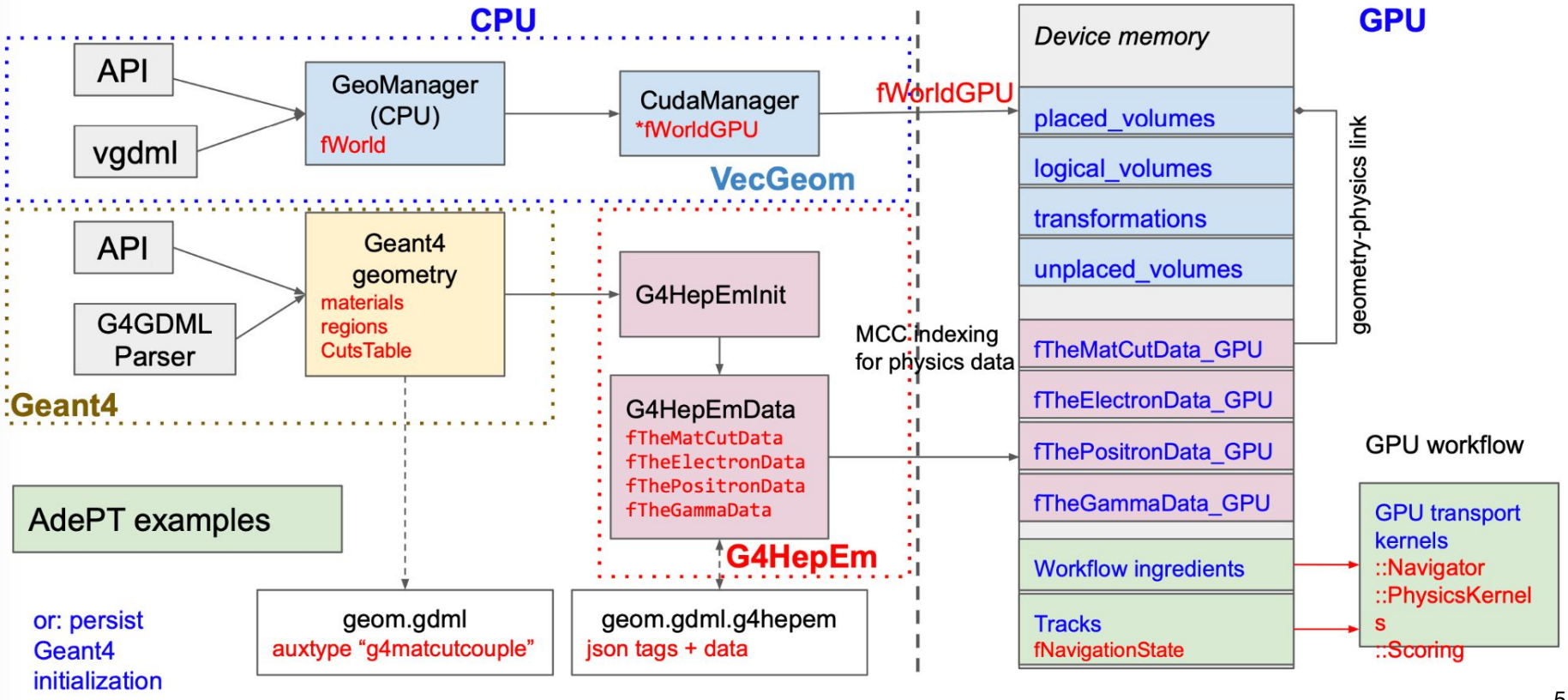
Simulation R&D @ November 2021

- ExCALIBUR-HEP through SWIFT-HEP came at a good time, with R&D projects of similar aim starting up - enabled links and collaborations to be formed:
- **AdePT Project (CERN-SFT)**
  - *https://github.com/apt-sim*
- **Celeritas Project (ECP: ORNL, FNAL, Argonne, LBL)**
  - *https://github.com/celeritas-project*
- Regular working and strategy meetings between projects, plus wider engagement through Geant4 and HSF workshops/meetings
- ***Can only give brief overview of work, follow links in the slides for full details!***

AdePT Objectives

- **See [Andrei Gheata's Geant4 Workshop presentation] for a full overview**
- Demonstrate realistic EM ($e^-/e^+/\gamma$) shower simulation workflow on GPU
  - *GPU "plug in" for Geant4 EM transport for specific regions (e.g. EM calo)*
  - *Possible standalone use for specific applications*
- Implement technical solutions for core simulation components on GPU
  - *Geometry (**VecGeom**), EM Physics (**G4HepEm**), Field Propagation*
  - *Data structures and workflow for track-level parallelism on GPU, CPU+GPU*
- Evolve a **demonstrator** through series of examples integrating components
  - *First complete simulation running, including all EM interactions, tracking in non-constant magnetic field (optionally), sensitive detector functionality*
  - ***Q1 2022**: HSF meeting to plan next steps, seek convergence of efforts on common project*

The AdePT "cookbook"



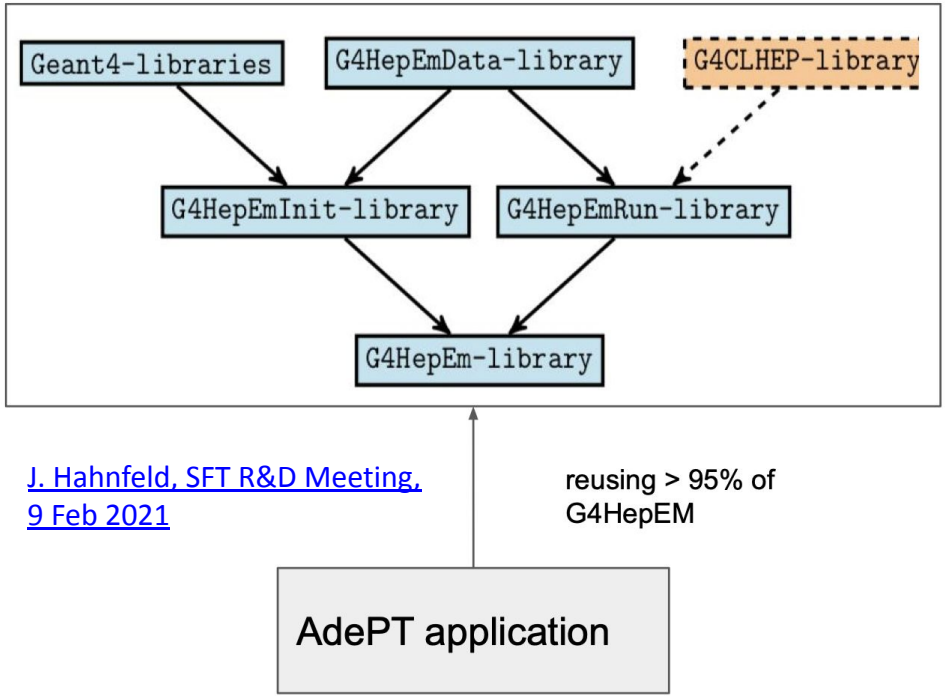
AdePT Ingredient: VecGeom

- Using VecGeom as GPU-aware library describing the detector geometry
  - *Same CPU and GPU algorithms compiled separately for host and device*
  - *Geometry data reconstructed on GPU based on host transient data*
  - *Navigation layer customized for GPU use*
- Improving gradually GPU support (see A. Gheata, Geant4 Workshop 2021)
  - *Developed custom optimised navigation state, single-precision support*
  - *Moving from a simple looper to an optimized BVH navigator*
  - *Adopting modern CMake GPU support*
- **Use in AdePT and Celeritas identified key areas to improve for optimizing performance and portability on GPUs**
  - *See baseline performance later*

AdePT Ingredient: G4HepEM

- Geant4 EM Physics R&D project on specialized EM processes for e+/e-/g only, targeting HEP detector applications
- Data structures and interfaces designed with CPU/GPU in mind
  - *Same code/data on host/device, aiding comparison/validation*
- **Integration in AdePT now complete**

M. Novák, Geant4 Workshop, Sep 2021

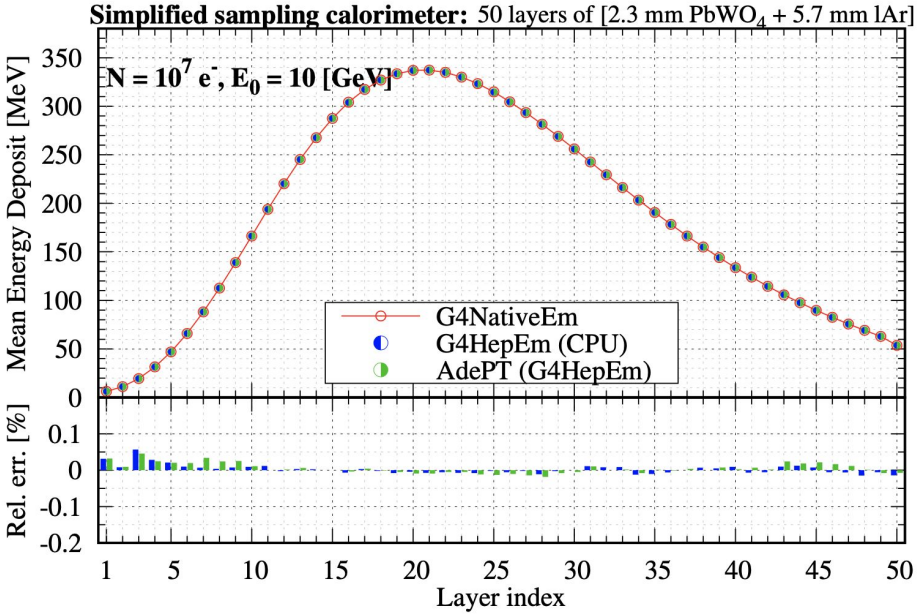


J. Hahnfeld, SFT R&D Meeting, 9 Feb 2021

reusing > 95% of G4HepEM

Adept: G4HepEM Physics Validation

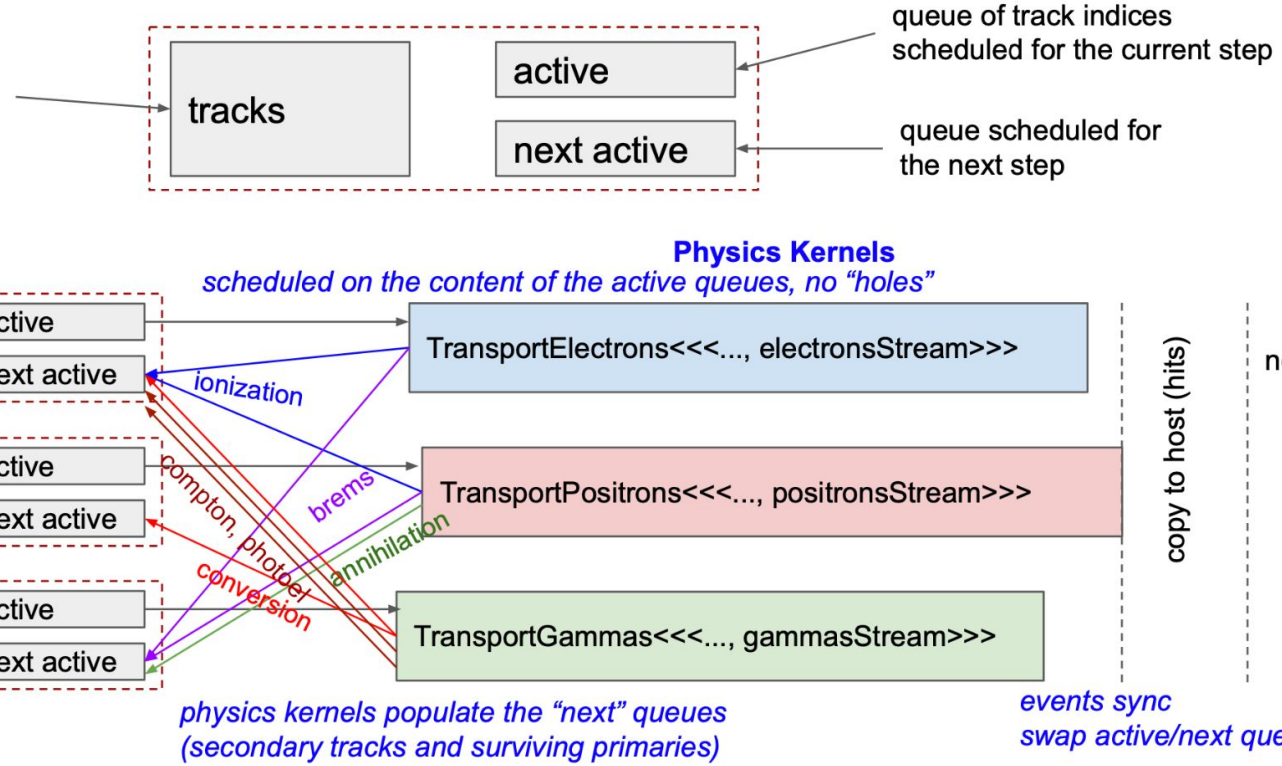
- TestEm3: 50 layer Pb/lAr sampling calorimeter
  - *B-field: 0T or 1T(*) constant*
- Comparison between
  - *Geant4 (CPU)*
  - *Geant4 w/G4HepEm (CPU)*
  - *AdePT+G4HepEM (GPU)*
- (*) couple parts-per-thousand discrepancy present in layers with highest energy deposition, number of particles
- Validation also in progress using CMS geometry, currently validated number of secondaries



M.Novak, Geant4 Workshop, Sep 2021
J. Hanhfeld, M. Novák, SFT R&D Meeting, 23 March 2021

AdePT Recipe: stepping loop workflow

- contiguous
- next slot atomic
- no slot reuse
- different for e+/e-/gamma



AdePT: Baseline performance

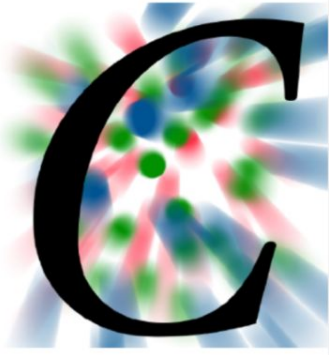
- TestEm3: 50 layer Pb/lAr sampling calorimeter
  - *Zero B-field, 10000 10GeV e-*
  - *Particles processed on GPU in batches of 26*
- AMD Ryzen 9 3900 (12C/24T), GeForce RTX2070 Super (J. Hanhfeld, SFT R&D Meeting, 20 April 2021)
  - *AdePT roughly same runtime as Geant4 with 24 Threads*
  - ***Caveat: AdePT/G4HepEM did not include "safety" geometric cut (MSC)***
  - ***Discussions with Celeritas on common benchmark problems, metrics, software/hardware setups***
- **VecGeom is significant limiter (similar observations in Celeritas)**
  - *Low device occupancy, as low as 10% in extreme cases*
  - *Large thread divergence, serialized calls to solids*
  - *Virtual function calls prevent compiler optimizations*
  - *Design, primarily virtual functions, not optimal for portability to, e.g. SYCL*

AdePT Recipes: Geant4 Integration

- Use case: standard Geant4 simulation of detector, use AdePT as “fast simulation” offload for computationally expensive region(s), e.g. EM calorimeter
  - *Initial design/work underway (W. Pokorski, Geant4 Workshop 2021)*
  - *One important difference to “normal” fast sim is need to batch particles*
  - *Starting discussion with Celeritas on potential for shared design and implementation.*
- Related work in Geant4/G4HepEM to implement “per-particle” tracking/stepping (M. Novak, J. Hahnfeld, Geant4 Workshop 2021)
  - *Enables use of “faster” physics code (e.g. G4HepEM) on CPU, but designed with asynchronicity/batching/offload to GPU in mind.*
  - *Initial interface in Geant4 11.0 (December release)*

Celeritas overview

- **GPU-targeted** re-implementation of a subset of Geant4 physics leveraging both HEP physics community and HPC/GPU particle transport domain knowledge
  - Data layout and memory access patterns are **critical** to GPU performance
  - Program flow requires structural reorganization
- Short-term application: offloading EM tracks from Geant4 to GPU (*Acceleritas* bridge library)
- Long-term application: direct access library for higher performance integration into LHC analysis workflows



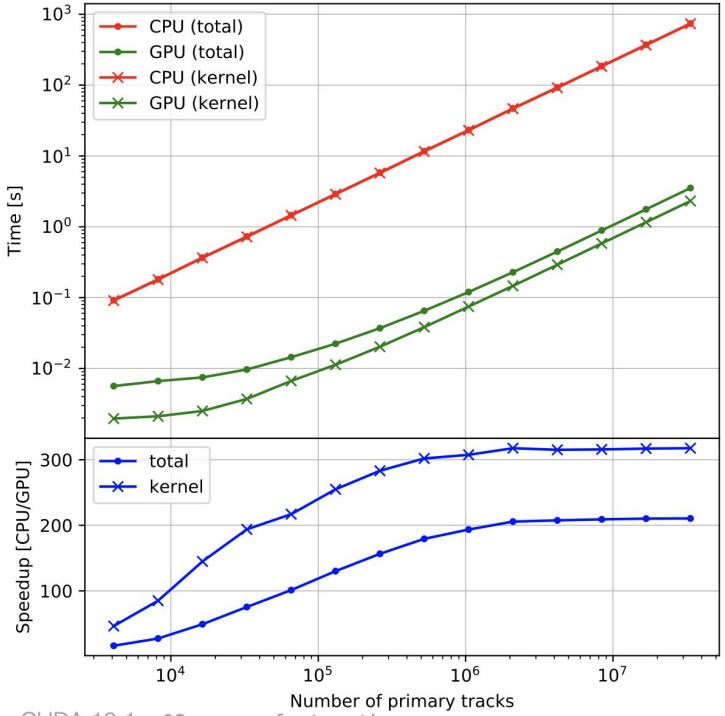
For a full overview, see Seth Johnson's HSF and vCHEP talks

Celeritas status

- Fully implemented on device and host:
  - Standard EM physics models for e, γ except MSC
    (bremsstrahlung, ionization, photoelectric with atomic relaxation, …)
  - Transport loop components
    (process/model selection, secondary production, energy loss fluctuations, …)
  - Geometry navigation using VecGeom
  - Uniform and nonuniform magnetic field
- Current efforts (through end of calendar year 2022):
  - Testing and benchmarking integrated transport loop
  - Physics validation

Celeritas: Physics Verification/Performance

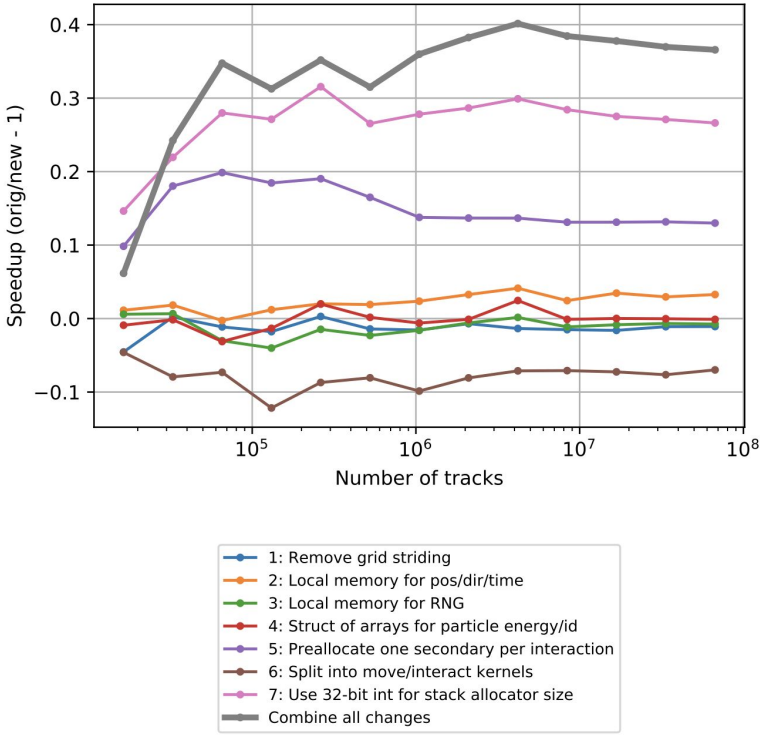
- Simplest physical simulation
  - *Photons only, Klein-Nishina scattering*
  - *Electrons created, but immediately killed*
  - *Single infinite volume of Aluminium*
  - *100MeV monodirectional beam*
- Primary kernel is an entire step
  - *Sequential on CPU, parallel on GPU*
- 1 core 2.3GHz CascadeLake vs 1x Tesla V100
- 200x overall speedup on GPU over CPU, but it is the **simplest** possible case



CUDA 10.1: `-O3 --use_fast_math`
GCC 8.3: `-O3 -march=skylake-avx512 -mtune=skylake-avx512`

Celeritas: Factors influencing GPU Performance

- Exploration of influence of various coding choices on performance
  - *Enabled by flexible design*
- Many “non-traditional” bottlenecks identified in simulation workflows
  - *Atomics, Irregular memory access*
- Demonstrates necessity of designing data/algorithms for “best use” of hardware
- See [Seth Johnson’s vCHEP presentation] for full results



Celeritas: Portability with Alpaka

- ExCALIBUR-HEP: Mark Hodgkinson
  - *Use Alpaka to implement memory management, kernels in Celeritas and Klein-Nishina mini-app*
- Compare three setups
  - *1: Celeritas native*
  - *2: Celeritas + Alpaka memory*
  - *3: Celeritas +Alpaka memory, kernel*
- **Work continuing to understand differences in mem/kernel speeds**
- **Starting investigations with NVIDIA HPC SDK**

```
__global__ void initialize_kn(ParamPointers const   params,
                              StatePointers const   states,
                              InitialPointers const init)
{
    // Grid-stride loop, see
    for (int tid = blockIdx.x * blockDim.x + threadIdx.x;
         tid < static_cast<int>(states.size());
         tid += blockDim.x * gridDim.x)
    {
        ParticleTrackView particle(
            params.particle, states.particle, ThreadId(tid));
        particle = init.particle;

        // Particles begin alive and in the +z direction
        states.direction[tid] = {0, 0, 1};
        states.position[tid]  = {0, 0, 0};
        states.time[tid]      = 0;
        states.alive[tid]     = true;
    }
}
```

```
using namespace alpaka;

//Define shortcuts for some alpaka items we will use
using Dim = dim::DimInt<1>;
using Idx = uint32_t;
//Define the alpaka accelerator to be Nvidia GPU
using Acc = acc::AccGpuCudaRt<Dim,Idx>;

struct initialize_alpaka{
  template <typename Acc>
  ALPAKA_FN_ACC void operator()(Acc const &acc,ParamPointers const params,StatePointers const states,InitialPointers const init) con
    for(int tid = idx::getIdx<Grid, Threads>(acc)[0];tid < static_cast<int>(states.size());tid += blockDim.x * gridDim.x){
      ParticleTrackView particle(params.particle, states.particle, ThreadId(tid));
      particle = init.particle;

      // Particles begin alive and in the +z direction
      states.direction[tid] = {0, 0, 1};
      states.position[tid]  = {0, 0, 0};
      states.time[tid]      = 0;
      states.alive[tid]     = true;
    }

  }
};
```

Setup	Call	Time
1	iterate_kn	3.79 ± 0.04 ms
1	initialize_kn	4.54 ± 0.19 μs
1	cudaMalloc	277.41 ± 6.51 s
2	iterate_kn	77.80 ± 4.41 ms
2	initialize_kn	47.46 ± 0.29 μs
2	cudaMalloc	761.13 ± 302.91 μs
3	iterate_alpaka	92.35 ± 5.91 ms
3	initialize_alpaka	261.98 ± 2.02 μs
3	cudaMalloc	805.38 ± 348.72 μs

In Summary

- Detector simulation is a key area for performance improvement in HEP, but detailed work is needed to realize this on new architectures
- ExCALIBUR-HEP began link up with and contributed to other global projects in this area
- Both AdePT and Celeritas have made major progress over the last year, and SWIFT-HEP is building on contributions to these efforts