Accelerating CLUE with oneAPI SYCL

Juan José Olivera Loyola

Supervised by Felice Pantaleo and Wahid Redjeb

Additional Contributors: Aurora Perego, Andre Bocci, Luca Ferragina, Nikolaos Andriotis, Tony Di Pilato

Project Github: Parsifal-2045/heterogeneous-clue (github.com)
Context - Raw Data Processing

1. Raw data has to be reconstructed by algorithms to obtain physics objects

2. Performance demand will increase substantially at HL-LHC
   - 7x higher event rate
   - 4x higher pile-up — The Patatrack Project
**Context - CLUE Algorithm**

**CLUE:** Highly parallelizable clustering algorithm mainly used in HGCal at CMS, CERN to recognize different particles based on the energy deposits in nearby sensors by a particle’s energy shower.

- Implemented with C++/CUDA to run on CPU’s and Nvidia’s GPU’s.
Problem Definition

Previous State: CLUE C++ CPU and CUDA

How to run in?

Coding multiple implementations is:
- Prone to errors
- Hard to understand code
- Hard to update
- A lot of expertise required (eg. SDK’s)

Objectives:
- Run on different hardware accelerators and architectures
- Avoid code duplication
- Be performant
- Be correct across hardware architectures
- Support of harnessing multiple accelerator devices concurrently
Solution

One code to run them all
with a performance portability framework (OpenCL, SYCL, Alpaka, etc)
# Solution - Tech Stack

<table>
<thead>
<tr>
<th>CLUE SYCL</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYCL</td>
</tr>
<tr>
<td>Device Streaming Framework</td>
</tr>
<tr>
<td>Intel oneAPI - DPC++ Compiler</td>
</tr>
<tr>
<td>Nvidia GPU's</td>
</tr>
</tbody>
</table>

Also done:
- Correctness Validators
- Benchmarking Scripts
- SYCL Porting and Compiling Guides

Advantages:
- Long-term support
- Support for Intel CPU's, GPU's, FPGA's
- Tools like VTune and Intel GDB
Task Details

- Replaced CUDA functions with equivalent SYCL
- Used SYCL features to match host-device synchronization model
- Built Intel’s DPC++ open source compiler to support Nvidia GPU’s backend
- Correctness validation was done by automatically comparing new results with a correct reference for many inputs
Results - SYCL CLUE RUNS EVERYWHERE!

```
[jolivera@patatrack01 heterogeneous-clue]$ ./sycl --maxEvents 1
Found 9 SYCL devices:
- Intel(R) FPGA Emulation Device
- Intel(R) Core(TM) i9-9900K CPU @ 3.60GHz
- Intel(R) Core(TM) i9-9900K CPU @ 3.60GHz
- Intel(R) UHD Graphics 630 [0x3e98]
- Intel(R) Core(TM) i9-9900K CPU @ 3.60GHz
- Intel(R) FPGA Emulation Device
- Intel(R) UHD Graphics 630 [0x3e98]
- Tesla K40c
- SYCL host device
```

Running CLUE algorithm with the following parameters:

Listed devices by SYCL CLUE application are Intel GPU's, CPU's, FPGA's emulators, Nvidia GPU's in patatrack01 VM
Results - SYCL CLUE IS PERFORMANT!

Latency of Processing 1 Event

CPU's and GPU's

Only GPU's

Log total execution time [ns]

Number of points per layer (\times 10^3)

CUDA
CPU
SYCL CUDA
SYCL CPU
SYCL INTEL GPU

CUDA
SYCL CUDA
SYCL INTEL GPU

Number of points per layer (\times 10^3)
Results - **SYCL CLUE CAN BE DEVICE STREAMED!**

Throughput increased by number of streams

![Graph showing throughput vs. number of points per layer for different stream counts.](image-url)
Conclusions

- Highly parallelizable clustering application (CLUE), developed with C++/SYCL capable of running on different hardware architectures, that supports job streaming to multiple devices.

- SYCL Knowledgebase of porting patterns, common pitfalls and tools setup

- Work along an amazing people
References

- SYCL Specification Revision 5: SYCL™ 2020 Specification (revision 5) (khronos.org)
- Intel LLVM Getting Started: llvm/GetStartedGuide.md at sycl · intel/llvm (github.com)
- Patatrack Wiki: https://patatrack.web.cern.ch/patatrack/wiki/

Git Repositories
- Original CLUE: https://gitlab.cern.ch/kalos/clue
- SYCL CLUE: https://github.com/Parsifal-2045/heterogeneous-clue
QUESTIONS?
jjoulk@gmail.com
Github: JJOL (github.com)
LinkedIn: jjolivera