Brief update on projects

Daniel Hugo Cámpora Pérez
dcampora@cern.ch

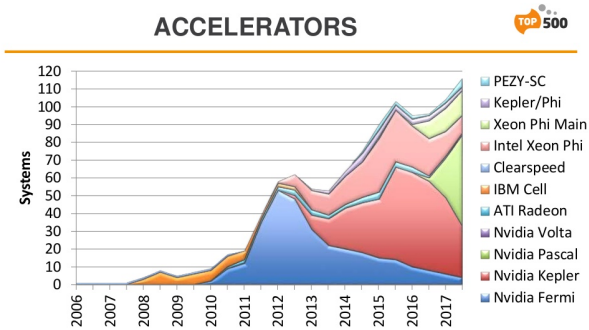
COMCHA, 12th September, 2018

Universidad de Sevilla
CERN

Motivation

GPUs are competitive from a price-performance standpoint

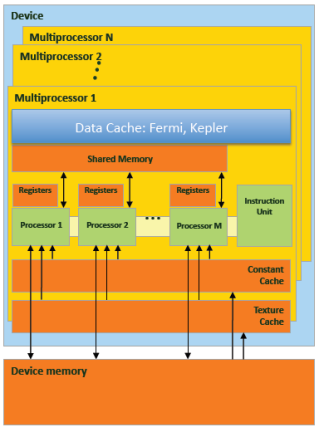
There are many results in literature that indicate a significative gain in price-performance from moving to GPUs.



A different programming paradigm

Even though GPUs are in essence SIMD machines, they are programmed following a slightly different programming paradigm (SIMT). Vectorized code vs. GPU code will look very different to one another.

Memory hierarchy also differs between CPUs and GPUs. L1 memory can be partitioned into shared memory. Constant and texture memories also exist.



Setting the bar high

The purpose of this project is to produce a full HLT1 on GPUs by the end of the year, and evaluate the cost and possible implementations of such project.

- Full HLT1: Data preparation (VELO, UT, SciFi, Muon) $\rightarrow$
  VELO tracking / PatPV3D $\rightarrow$ VeloUT $\rightarrow$ Forward $\rightarrow$ Kalman $\rightarrow$ MuonID

Ongoing developments

- SciFi reconstruction
- Muon reconstruction
- FPGA project (José Mazorra)

Building a small team

- Join the e-group *lhcb-parallelization*
- `https://gitlab.cern.ch/lhcb-parallelization/cuda_hlt`
- `https://twiki.cern.ch/twiki/bin/view/LHCb/GPUStudies`

Status

Velo tracking - Physics and performance

Search by triplet (minbias):

```
TrackChecker output    :     8058/   246296   3.27% (  2.82%) ghosts
Velo                   :   215012/   242881  88.53% ( 88.99%),      2215 (  0.52%) clones, hit eff  97.28% pur  99.02%
Long                   :    65408/    67648  96.69% ( 96.96%),       643 (  0.49%) clones, hit eff  98.13% pur  99.11%
Long, p > 5 GeV        :    41907/    42621  98.32% ( 98.52%),       377 (  0.45%) clones, hit eff  98.40% pur  99.11%
Long strange           :     2860/     3294  86.82% ( 87.68%),        23 (  0.40%) clones, hit eff  98.01% pur  97.67%
Long strange, p > 5 GeV:     1458/     1601  91.07% ( 91.31%),         6 (  0.21%) clones, hit eff  98.63% pur  97.61%
Long from B            :      105/      107  98.13% ( 97.20%),         0 (  0.00%) clones, hit eff  98.54% pur  99.02%
Long from B, p > 5 GeV :       67/       67 100.00% (100.00%),         0 (  0.00%) clones, hit eff  98.39% pur  98.82%
```

Performance:

- GTX 1080 Ti: 84 kHz

Velo UT - Physics and performance

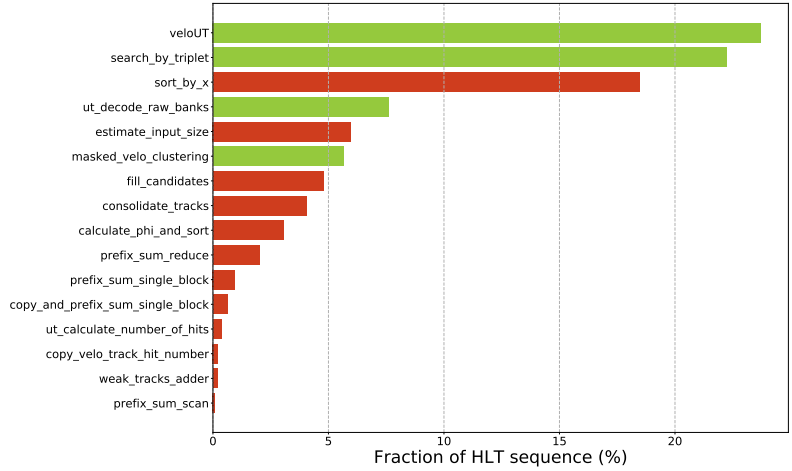
Search by triplet (minbias):

```
TrackChecker output    :      1790/    69548   2.57% (  2.13%) ghosts
Velo                   :     63170/   242881  26.01% ( 25.26%),       512 (  0.41%) clones, hit eff  67.65% pur  99.12%
Velo+UT                :     60752/   102595  59.22% ( 57.65%),       480 (  0.40%) clones, hit eff  66.76% pur  99.16%
Velo+UT, p > 5 GeV     :     40513/    48096  84.23% ( 82.90%),       334 (  0.41%) clones, hit eff  68.22% pur  99.18%
Velo, not long         :     17208/   175233   9.82% (  9.46%),       150 (  0.44%) clones, hit eff  67.47% pur  98.94%
Velo+UT, not long      :     14967/    39167  38.21% ( 37.10%),       118 (  0.39%) clones, hit eff  63.89% pur  99.07%
Velo+UT, not long, p > 5 GeV:      7588/     8743  86.79% ( 86.34%),        60 (  0.40%) clones, hit eff  66.35% pur  99.14%
Long                   :     45962/    67648  67.94% ( 66.20%),       362 (  0.39%) clones, hit eff  67.72% pur  99.19%
Long, p > 5 GeV        :     33089/    42621  77.64% ( 75.94%),       274 (  0.41%) clones, hit eff  68.67% pur  99.19%
Long from B            :        88/      107  82.24% ( 83.06%),         0 (  0.00%) clones, hit eff  66.84% pur  99.12%
Long from B, p > 5 GeV :        62/       67  92.54% ( 89.99%),         0 (  0.00%) clones, hit eff  68.69% pur  98.75%
```

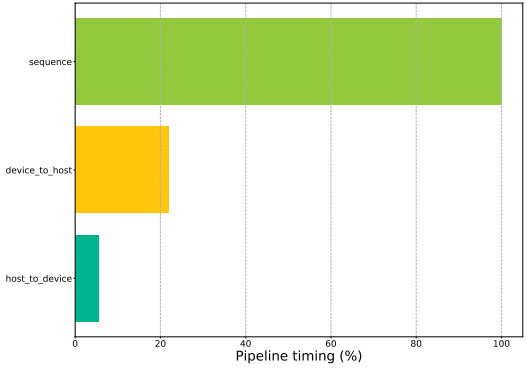
Performance:

- GTX 1080 Ti: 43 kHz

Cuda HLT1 current sequence



Cuda HLT1 pipeline



OpenCL on FPGA

FPGA processing

- First experience during July.
- Access granted to a TechLab machine at CERN.
  - Name: techlab-fpga-altera-01.
  - Board: Nallatech 385A.
  - FPGA: Arria 10.
- Tested out three design examples provided by Intel/Altera.
  `https://www.intel.com/content/www/us/en/programmable/products/design-software/embedded-software-developers/opencl/developer-zone.html`
  - Hello World: create and run basic FPGA kernel.
  - Vector Addition: use simple kernel in host software.
  - Multithread Vector Operation: one kernel per thread.

FPGA processing

- Projects composed of two types of files.
  - OpenCL kernel files (*.cl) coding the FPGA processing.

```
81  __kernel void vector_add(__global const float *x,
82                           __global const float *y,
83                           __global float *restrict z)
84  {
85      int index = get_global_id(0);
86      z[index] = x[index] + y[index];
87  }
```

  - C++ source files (*.cpp) coding the CPU processing.

```
174     scoped_array<cl_kernel> kernel; // num_devices elements
175
176     ...
177
178     const char *kernel_name = "vector_add";
179     kernel[i] = clCreateKernel(program, kernel_name, &status);
180     checkError(status, "Failed to create kernel");
```

Manufacturer SDK defines types and functions for FPGA.

FPGA processing

- Two different tools are used to compile the whole project.
  - OpenCL kernels use the SDK specific command "aoc" producing a bitmap for FPGA configuration (*.aocx).

```
aoc device\vector_add.cl -o bin\vector_add.aocx --board <board>
```

  - Command "aoc" can also be used for management.

```
aoc --list-boards
```

    More details on the Intel FPGA SDK for OpenCL at
    `https://www.intel.com/content/www/us/en/programmable/products/design-software/embedded-software-developers/opencl/support.html`

  - Host application compiled with gcc/g++ 4.4.7 or later (examples provided with corresponding Makefile).

FPGA processing

- During last LHCb Week cross kalman algorithm tested on FPGA.
- OpenCL branch as is requires more RAM than available in FPGA.
- Simplified version (predict state) successfully run on FPGA.
- Future developments:
  - Include resource usage in test output.
  - Study OpenCL coding optimization for FPGA.
  - Code other algorithms with potential improvement in FPGA processing.