Designing a 30 MHz GPU trigger, the LHCb experience

The anatomy of an LHCb event in the upgrade era, and implications for the LHCb trigger

Ref: LHCb-PUB-2014-027

Public Note Issue: 1

Reconstructed yields

Date: May 21, 2014

- b-hadrons
- c-hadrons
- light, long-lived hadrons

Reconstructed yield

0.0317 ± 0.0006

0.118 ± 0.0010

0.406 ± 0.002

Inset (p_T > 2 GeV/c) 85.6 ± 0.6%

51.8 ± 0.5%

2.34 ± 0.08%

Inset (⌧ > 0.2 ps) 88.1 ± 0.6%

63.1 ± 0.5%

99.46 ± 0.03%

Inset (p_T) ⇥ Inset (⌧) 75.9 ± 0.8%

32.6 ± 0.4%

2.30 ± 0.08%

Inset (LHCb) 27.9 ± 0.3%

22.6 ± 0.3%

2.17 ± 0.07%

Output rate 270 kHz

800 kHz

264 kHz

Table 6: Per-event yields determined from 100k of upgrade minimum-bias events after partial offline reconstruction. The first row indicates the number of candidates which had at least two tracks from which a vertex could be produced. The last row shows the output rate of a trigger selecting such events with perfect efficiency, assuming an input rate of 30 MHz from the LHC, as expected during upgrade running. A breakdown of each category is available in Table 14.

Figure 1: HLT partially reconstructed (but fully reconstructible) signal rates as a function of decay time for candidates with p_T > 2 GeV/c (left) and transverse momentum cuts for candidates with ⌧ > 0.2 ps (right). The rate is for two-track combinations that form a vertex only for candidates that can be fully reconstructed offline, i.e.: All additional tracks are also within the LHCb acceptance.
A general-purpose forward spectrometer at the LHC, optimized for heavy-flavour physics

0.5 - 1% momentum resolution

15 + 29/p_{T}[GeV] micron impact parameter resolution

(1+10/\sqrt{E[GeV]})% ECAL resolution

Efficient and high-purity identification of all five stable charged particle types (pion, kaon, proton, electron, muon) over the momentum range 2-100 GeV
The challenge of triggering @ LHCb
The challenge of triggering @ LHCb

Cross-section for processes of interest to LHCb saturates a traditional CALO/Muon trigger.
The LHCb upgrade has to run at $2 \cdot 10^{33}$ cm$^{-2}$s$^{-1}$ — total rethink necessary.
Typical triggers select signal needles in Standard Model haystacks.
LHCb needs to sort and compresses haystacks of needles — Real Time Analysis!
From this follows the LHCb DAQ design for the upgrade.
LHCb upgrade dataflow in more detail

LHC BUNCH CROSSING (40 MHz)

5 TB/s
30 MHz non-empty pp

FULL DETECTOR READOUT
5 TB/s

PARTIAL DETECTOR RECONSTRUCTION & SELECTIONS (GPU HLT1)

0.5-1.5 MHz
70-200 GB/s

BUFFER

REAL-TIME ALIGNMENT & CALIBRATION

FULL DETECTOR RECONSTRUCTION & SELECTIONS (CPU HLT2)

6% CALIB EVENTS
1.6 GB/s

26% FULL EVENTS
5.9 GB/s

68% TURBO EVENTS
2.5 GB/s

OFFLINE PROCESSING

ANALYSIS PRODUCTIONS & USER ANALYSIS

All numbers related to the dataflow are taken from the LHCb Upgrade Trigger and Online TDR
Upgrade Computing Model TDR

LHCb—FIGURE—2020—016
Need to preselect events before running full reconstruction, but based on which criteria?
“Traditional” inclusive selections but based on both transverse momentum and displacement. This of course requires charged particles, so we require 30 MHz tracking at $2 \times 10^{33}$. 

Physics content of HLT1 which runs @ 30 MHz
The LHCb upgrade HLT must handle the same real-time data volume as ATLAS/CMS HL-LHC HLTs.

Pause and compare this to ATLAS/CMS HL-LHC processing.

<table>
<thead>
<tr>
<th>CMS detector</th>
<th>LHC Run-2</th>
<th>HL-LHC Phase-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peak ⟨PU⟩</td>
<td>60</td>
<td>140</td>
</tr>
<tr>
<td>L1 accept rate (maximum)</td>
<td>100 kHz</td>
<td>500 kHz</td>
</tr>
<tr>
<td>Event Size</td>
<td>2.0 MB</td>
<td>5.7 MB</td>
</tr>
<tr>
<td>Event Network throughput</td>
<td>1.6 Tb/s</td>
<td><strong>23 Tb/s</strong></td>
</tr>
<tr>
<td>Event Network buffer (60 seconds)</td>
<td>12 TB</td>
<td>171 TB</td>
</tr>
<tr>
<td>HLT accept rate</td>
<td>1 kHz</td>
<td>5 kHz</td>
</tr>
<tr>
<td>HLT computing power</td>
<td>0.5 MHS06</td>
<td>4.5 MHS06</td>
</tr>
<tr>
<td>Storage throughput</td>
<td>2.5 GB/s</td>
<td>31 GB/s</td>
</tr>
<tr>
<td>Storage capacity needed (1 day)</td>
<td>0.2 PB</td>
<td>2.7 PB</td>
</tr>
</tbody>
</table>
Exploiting the full detector readout

32 Tb/s

1 Tb/s

1 Tb/s

173 Event Builder servers
Three TELL40 readout boards per EB server

200G IB
100GbE
10GbE

16 storage servers

40 HLT2 servers 40 HLT2 servers 40 HLT2 servers 40 HLT2 servers

Up to 100 HLT2 sub-farms (4000 servers)
Exploiting the full detector readout

Data from detectors received by O(500) FPGA readout boards and built into events by a farm of 173 servers.

Leaves three PCIe slots free per server.

Can now choose: send the full 32 Tb/s to a CPU server farm for processing (requires two extra network cards per server) or fill these slots with co-processors (e.g. GPUs) and reduce the data rate locally to 1 Tb/s.

Both options developed and viable — we finally chose the GPU option and I’ll talk about it in more detail now.
Architecture of one Event builder node

GPU-equipped event builder PC, with traffic of all three readout cards.

If we can put the whole trigger software on the GPU, then a GPU accelerator is “transparent” to the Event Builder — like sending data to a network card. Architecture should “automagically” work.
So the architecture should work, will the software?

Essential ingredients for HLT1:

1. Find tracks in the vertex detector
2. Find pp collision points and measure track displacement to them
3. Extrapolate tracks to the UT, and then to the SciFi trackers
4. Perform muon identification and fake track rejection

LHCb events are relatively small, O(100 kB), so we must parallelize both across events and inside each event.

Can we achieve enough parallelism to do the job?
The full sequence contains dozens of components and can accommodate up to $O(100)$ selections. Most algorithms written from scratch for GPU with a logic optimized for SIMT — not ported Cross-architecture by construction, compiles for CPU to enable “for free” simulation.
Interlude on managing software: this began as pure R&D

LHCb had been pursuing individual GPU reconstruction algorithms since 2014, with the most promising work done on the vertex detector reconstruction algorithm and associated infrastructure (see biblio at bottom).

Bibliography: Badalov et al, Badalov et al
By 2017 we had largely concluded this would never work. However porting single algorithms to GPUs was not going to work, mainly because no single algorithm took a large enough piece of the reconstruction sequence to make this cost-effective.

**Not a bene:** compares GPU to a single CPU core!

### Table 1. Comparison of a CPU and a GPU VELO Pixel tracking algorithm.

<table>
<thead>
<tr>
<th></th>
<th>PrPixel</th>
<th>Track forwarding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time per event (ms)</td>
<td>3.6</td>
<td>26.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>batch of 1</td>
</tr>
<tr>
<td>Ghost rate</td>
<td>1.7%</td>
<td>0.8%</td>
</tr>
<tr>
<td>Efficiency for long tracks</td>
<td>98.3%</td>
<td>98.0%</td>
</tr>
<tr>
<td>Efficiency for long tracks over 5 GeV</td>
<td>98.8%</td>
<td>98.4%</td>
</tr>
<tr>
<td>A</td>
<td>0.80</td>
<td>batch of 300</td>
</tr>
</tbody>
</table>

**Bibliography:** Badalov et al, Badalov et al, Badalov et al
Then we decided to give the architecture a fair chance…

At the start of 2018 we decided to try to put the entire HLT1 on GPUs, despite only having a functioning vertex detector reconstruction and two years to get the job done. We hedged our bets, which seemed expensive from the point of view of developer time but in fact made optimal use of people’s diverse skills.

Bibliography: Badalov et al, Badalov et al, Badalov et al, Campora et al
And learned that it can be easier to achieve the harder goal.

Putting “everything” on the GPU unlocked the power of the architecture and made it cost-effective. Classic accumulation of knowledge on a plateau followed by a phase transition as it came together. Similarly, the vectorization of our CPU reconstruction also came together in parallel to meet the required performance.

Bibliography: Badalov et al, Badalov et al, Badalov et al, Campora et al, GPU HLT TDR
O(200) GPUs required to reach 30 MHz so there is plenty of spare capacity!
HLT1 throughput performance

Excellent throughput scaling with theoretical TFLOPS of GPU card

References: LHCb-FIGURE-2020-014
HLT1 throughput performance

The throughput as a function of the occupancy in the SciFi detector is depicted in Fig. 12. The slower throughput decrease in the high occupancy region gives confidence that Allen can be adapted to real data taking conditions, where the detector occupancy might be higher than in simulation (as observed consistently during Runs 1 and 2). If the GEC removing the 10% busiest events is deactivated and all events are processed, the Allen throughput drops by about 20%.

Conclusions

We present Allen, an implementation of the first trigger stage of LHCb for Run 3 entirely on GPUs. This is the first complete high-throughput GPU trigger proposed for a HEP experiment. Allen covers the majority of the LHCb physics programme, using an analogous reconstruction and selection sequence as in Run 2. The demonstrated event throughput shows that the full HLT1 sequence can run on about 500 of either one of the RTX 2080 Ti, V100 or Quadro RTX 6000 Nvidia GPU cards. Consequently, the GPUs can be hosted by the event building servers, significantly reducing the net work cost associated with sending HLT1 output to the EFF.

We show that the performance in terms of track and vertex reconstruction efficiency, muon identification and momentum resolution are sufficient for efficient trigger selections for analyses representative of the LHCb physics programme.

Acknowledgements

We would like to thank N. Neufeld and T. Colombo for many fruitful discussions. We also thank the LHCb RTA team for supporting this publication and reviewing this work. We thank the technical and administrative staff at the LHCb institutes. We acknowledge support from CERN and from the national agencies: CAPES, CNPq, FAPERJ and FINEP (Brazil); MOST and NSFC (China); CNRS/IN2P3 (France); BMBF, DFG and MPG (Germany); INFN (Italy); NWO (The Netherlands); MNiSW and NCN (Poland); MEN/IFA (Romania); MSHE (Russia); MinECo (Spain); SNSF and SER (Switzerland); NASU (Ukraine); STFC (UK); DOE NP and NSF (USA). We acknowledge the computing resources that are provided by CERN, IN2P3 (France), KIT and DESY (Germany), INFN (Italy), SURF (The Netherlands), PIC (Spain), GridPP (UK), RRCKI and YanDEX LLC (Russia), CSCS (Switzerland), IFIN-HH (Romania), CBPF (Brazil), PL-GRID (Poland) and OSC (USA). We are indebted to the communities behind the multiple open-source software packages on which we depend. Individual groups or members have received support.
HLT1 reconstruction performance

References: LHCb-FIGURE-2020-014
Selections nowhere near tuned — of course can only happen once we’ve commissioned the all-new detector hardware

On MC keep > 50% of all reconstructible key B decays with some reasonable parent/child transverse momentum. More than good enough for now!
Integration test — can this be used for stable datataking?

Emulate network traffic & memory pressure by getting FPGA boards to generate data.
Integration test — can this be used for stable datataking? Yes!

(a) Allen throughput  
(b) Memory I/O
Conclusion and outlook

LHCb is ready to tackle the Run 3 signal deluge with a first-level trigger fully implemented on GPUs.

Allen framework is not specific to LHCb, aim to decouple framework and LHCb-specific code in future.

Work ongoing to implement further algorithms on GPU and extend the reach of the HLT for neutral and long-lived particles.

Looking forward to commissioning!
Backup
Comparison of track states executing Allen on GPU and CPU — for vast majority of tracks agreement is at permille level or better. Same is true for most other quantities and we explicitly test for this.

References: LHCb Upgrade GPU High Level Trigger TDR