Upgrade trigger: Biannual performance update

Public Note

Issue: 1
Revision: 0

Reference: LHCb-PUB-2017-005
Created: 25 July 2016
Last modified: February 23, 2017

Prepared by: R. Aaij[a], J. Albrecht[b], B. Couturier[a], S. Esen[c], M. De Cian[c], J. De Vries[d], A. Dziurda[a], C. Fitzpatrick[e], M. Fontana[f], L. Grillo[g], C. Hasse[a], C. Jones[h], R. Le Gac[i], R. Matev[a], N. Neufeld[a], T. Nikodem[c], F. Polci[j], L. Del Buono[j], R. Quagliani[k], R. Schwemmer[a], P. Seyfert[g], S. Stahl[a], T. Szumlak[l], M. Vesterinen[c], J. Wanczyk[a], M. Williams [m], H. Yin[n], E. Zacharjasz[l]

[a]European Organization for Nuclear Research (CERN), Geneva, Switzerland
[b]Fakultät Physik, Technische Universität Dortmund, Dortmund, Germany
[c]Physikalisches Institut, Ruprecht-Karls-Universität Heidelberg, Heidelberg, Germany
[d]Nikhef National Institute for Subatomic Physics, Amsterdam, The Netherlands
[e]Insitute of Physics, Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland
[f]Sezione INFN di Cagliari, Cagliari, Italy
[g]Sezione INFN di Milano Bicocca, Milano, Italy
[h]Cavendish Laboratory, University of Cambridge, Cambridge, United Kingdom
[i]CPPM, Aix-Marseille Université, CNRS/IN2P3, Marseille, France
[j]LPNHE, Université Pierre et Marie Curie, Université Paris Diderot, CNRS/IN2P3, Paris, France
[k]LAL, Université Paris-Sud, CNRS/IN2P3, Orsay, France
[l]AGH - University of Science and Technology, Faculty of Physics and Applied Computer Science, Kraków, Poland
[m]School of Physics and Astronomy, University of Manchester, Manchester, United Kingdom
[n]Institute of Particle Physics, Central China Normal University, Wuhan, Hubei, China

Contents

1 Introduction **2**

2 Reconstruction Sequence **3**
2.1 Track types 3
2.2 Changes in the detector description 3
2.2.1 VELO 3
2.2.2 SciFi 4
2.3 Simulated datasets 4
2.4 Changes to the reconstruction sequence 4
2.5 Changes to individual algorithms since the trigger TDR 5
2.5.1 VELO 5
2.5.2 PV reconstruction 6
2.5.3 VELO-UT 6
2.5.4 Forward 6
2.5.5 Seeding 7
2.5.6 Matching 7
2.5.7 Downstream 7
2.5.8 Kalman Fit 7
2.5.9 Ghost Probability Tool 8
2.5.10 Particle ID 8
2.6 Performance Update 9
2.6.1 PV resolution 9
2.6.2 Ghost probability performance 9
2.6.3 Track reconstruction efficiencies 9

3 Timing & throughput measurements **11**
3.1 Global Event Cut 11
3.2 Single process timing comparison 11
3.3 Throughput 12

4 Event Filter Farm Performance Extrapolation **13**
4.1 Performance extrapolation of the trigger TDR 13
4.2 Revised growth factor 13
4.3 Revised performance extrapolation 13

5 Conclusions **16**

6 References **17**

Abstract

This document presents the performance of the LHCb Upgrade trigger reconstruction sequence, incorporating changes to the underlying reconstruction algorithms and detector description since the Trigger and Online Upgrade TDR. An updated extrapolation is presented using the most recent example of an Event Filter Farm node.

1 Introduction

The LHCb trigger during the 2010-2018 running period consists of a hardware, level-0 trigger that reduces the accepted rate of bunch crossings from $30\mathrm{MHz}$ to $1\mathrm{MHz}$ at which point the detector can be read out, followed by a software stage in which a combination of inclusive and exclusive selections are made that further reduce the output made available offline for analysis [1,2].

After the phase 1 upgrade, LHCb will take data at an instantaneous luminosity of $2 \times 10^{33}\mathrm{cm}^{-2}\ \mathrm{s}^{-1}$ from 2021 onwards. The upgraded trigger consists of two paradigms, the *triggerless readout* and *full software trigger*: The $1\mathrm{MHz}$ readout limitation will be removed, allowing for the full event rate to be processed in software on the commodity event filter farm (EFF) without the need for a hardware trigger. In 2014 the LHCb Trigger and Online Upgrade TDR [3] (referred to as the trigger TDR henceforth) determined the resources and development strategy necessary to implement these paradigms, using as input the studies performed as part of the Tracker Upgrade TDR [4]. At the time an extrapolation was used to determine the effective computing power available in the EFF given the specified budget of 2.8MCHF if purchased shortly before data taking. This extrapolation was to be revisited at regular intervals in order to ensure more precise estimates of the available resources and overall performance of the software were taken into account.

Performance studies of this kind are an important benchmark for the LHCb computing TDR in preparation based on the computing TDR roadmap document [5]. The new LHCb software framework and computing model aims to exploit recent trends in computing hardware.

In this document we update the timing studies presented in the trigger TDR with a more realistic simulation and improvements to the reconstruction algorithms. We additionally report the events processed per second by the extrapolated farm (throughput) and its evolution on several EFF benchmark configurations purchased between 2010 and 2016. Using the present software framework we present an updated extrapolation to the computing resources expected to be available at purchase in 2020. The results serve as input to the computing TDR and as a baseline for comparison as we move to the new software framework.

2 Reconstruction Sequence

Since the trigger TDR a number of changes have been implemented in the design of certain subdetectors. Simultaneously there has been an ongoing effort to improve the performance of many of the reconstruction algorithms. In this section we describe the changes to the detector description and the reconstruction sequence.

2.1 Track types

Before discussing changes to the individual subdetectors, the types of LHCb tracks produced by each detector and in combination are summarised here: The most valuable tracks for physics analysis are *long* tracks which are reconstructed in the VELO and the T stations of the Sci-Fi subdetector. They have excellent spatial resolution close to the primary interaction and precise momentum information due to the combined information of the track slope before and after the magnet. Tracks consisting of measurements in the T stations alone are known as *T* tracks. They are not used in physics analyses, but are used as inputs to reconstruct *downstream* tracks and serve as input the matching algorithm. These are tracks which have measurements in the UT and the T stations. Tracks consisting of measurements in the VELO and in the UT are called *upstream* tracks. Finally, *VELO* tracks are composed of VELO hits only.

2.2 Changes in the detector description

There have been several changes in the detector description as the Upgrade subdetectors have developed. The largest changes are to those of the Sci-Fi and VELO which are summarised here. For the remaining subdetectors not described in this section there have been no additional changes or those changes have had a negligible effect on performance. The changes to the VELO and Sci-Fi subdetectors are summarised briefly here:

2.2.1 VELO

The design of the upgraded VELO detector has been updated[a] to include a 45° rotation about the z-axis of the active components on each module (i.e. the sensor-ASIC ladders). To accommodate this rotation, the shapes of the microchannel substrate, hybrid boards, module supports, and RF foil have undergone adjustments. This has only a very small effect on the material budget, with the average radiation length traversed by particles within $2 < \eta < 5$ changing by $\mathcal{O}(10^{-3})X_0$, compared to the overall upgraded VELO thickness of around 20% X_0.

In the near future several more significant changes to the detector description are planned to best reflect the current design expectations. These include:

- a significant increase in the copper ($30\mu\text{m} \to 50\mu\text{m}$) and kapton ($150\mu\text{m} \to 750\mu\text{m}$) layer thicknesses in the hybrid,
- an increase in the silicon microchannel substrate thickness ($400\mu\text{m} \to 500\mu\text{m}$),
- an updated RF foil design including local thinning (uniform $250\mu\text{m} \to 500\mu\text{m}$ thinned to $250\mu\text{m}$ in the beam region),
- small updates to the materials and shapes of cooling connectors, supports, and the addition of a glue layer between the ASICs and substrate.

Overall these changes lead to a significant increase on the amount of material traversed by particles in the VELO region, with the average thickness in $2 < \eta < 5$ increasing by around 5% X_0. Dedicated performance studies indicate, however, that the final effect of this material on the IP resolution (≈3% degradation) and tracking efficiency (≈0.1% absolute reduction for VELO tracks) are not too severe. For the purpose of this document we neglect these changes as they are not yet part of an official release.

[a] As of detector description database version dddb-20161117

2.2.2 SciFi

Since the trigger TDR, the geometry and digitization of the SciFi detector have been modified to both improve the realism of the simulation and to take into account recent changes in the detector construction.

The fiber active material z depth has increased from $1.2\mathrm{mm}$ to $1.3\mathrm{mm}$. This corresponds to the simulation of fiber-mats built with 6 stacked fibers layers rather than 5 in the last trigger TDR. This increases the photon yield. In the past, the fiber modules in the stereo layers were simulated by performing a global 5 degree rotation around the detector z axis, in such a way that the boundary between the upper and lower layers was rectilinear. As this was mechanically difficult and would result in degraded performance, a local rotation of the modules themselves is introduced. This however causes a saw tooth shape around $y = 0$ and the pattern recognition algorithms have to take into account this additional complexity by looking simultaneously in the upper and lower modules when searching for $u - v$ hits at small $y = 0$, which slightly increases their execution time. The gaps between SiPM edges have been reduced from $0.625\mathrm{mm}$ to $0.225\mathrm{mm}$ which decreases the detector dead areas and thus increases the hit efficiency. The shape of the beam pipe hole is changed making it marginally larger than that used in the trigger TDR. This leads to minor changes in the central occupancy and acceptance of the SciFi.

In addition to the mechanical changes to the SciFi detector description, there have been several additions to the digitisation since the simulated samples used in the trigger TDR: Thermal noise in the SiPMs after full irradiation, pixel cross-talk and afterpulses are now included. Additionally, spillover contributions from previous and next bunch crossings are added. The channel charge ADC has been reduced from six bits to two bits, reducing the precision of the weighted (sub-channel) cluster position.

2.3 Simulated datasets

The performance and timing studies in the following sections are performed using fully simulated Upgrade datasets that take into account the modifications to the detector descriptions indicated in the previous section[b]. These datasets consist of three million minimum bias events for timing and throughput measurements, and a sample of 100,000 events containing $\mathrm{B}_\mathrm{s}^0 \to \phi\phi$ signal for tracking efficiency and ghost rate determinations. For all samples an equal quantity of magnet up and magnet down polarity events are generated.

2.4 Changes to the reconstruction sequence

The LHCb trigger for Run II is conceptually different from that of Run I, with real-time detector alignment and calibration, allowing for analysis directly on the trigger output without the need for reprocessing [6]. The Upgrade tracking sequence has been redesigned to take advantage of this concept by splitting the reconstruction sequence into two stages: The first *fast* stage provides the necessary input to the run-by-run calibration and alignment, while the second *best* stage performs the remaining reconstruction. The schematic view of both stages are shown in Fig. 1 and Fig. 2 for the fast and best stages, respectively.

The 'fast' stage is similar to the sequence described in the trigger TDR and serves as the baseline for comparison in subsequent sections of this document. It begins with the full VELO tracking and the simplified Kalman Filter, creating VELO tracks. These VELO tracks act as input to the Primary Vertex (PV) reconstruction. The next step extends the VELO tracks into the UT detector creating upstream tracks. Only tracks with a transverse momentum larger than 300 MeV/c are kept, and for them a charge and momentum are estimated. Finally, the upstream tracks are extended further by looking for hits in the Sci-Fi sub-detector where search windows are determined based on the output of the VELO-UT algorithm. The resulting long tracks are made with a transverse momentum threshold above 400 MeV/c. These serve as the necessary input to the calibration and alignment algorithms.

The second, 'best' stage is presently designed with a focus on performance rather than speed. It is presented for the first time in this document, with performance and timing numbers with respect to

[b]The sample configuration is: Sim09a-Upgrade with spillover, 7 TeV beam energy, $25\mathrm{ns}$ bunch spacing using Pythia8.

minimum bias events as no selection is presently applied between it and the fast stage. Future iterations will be optimised subject to available CPU, disk buffer space, and physics performance requirements which will determine the optimal balance between retention after the fast stage.

In the current best stage, VELO tracks from the Fast sequence are used to find additional long tracks with a decreased transverse momentum threshold of 50 MeV/c. Additional T tracks are found using a seeding algorithm which looks for track seeds inside Sci-Fi detector, serving as input to the downstream tracking and track matching. All tracks are then combined into the 'best' track container with the addition of a clone-killing algorithm and an additional Kalman Filter. These serve as input to the remaining algorithms and subsequent trigger selections.

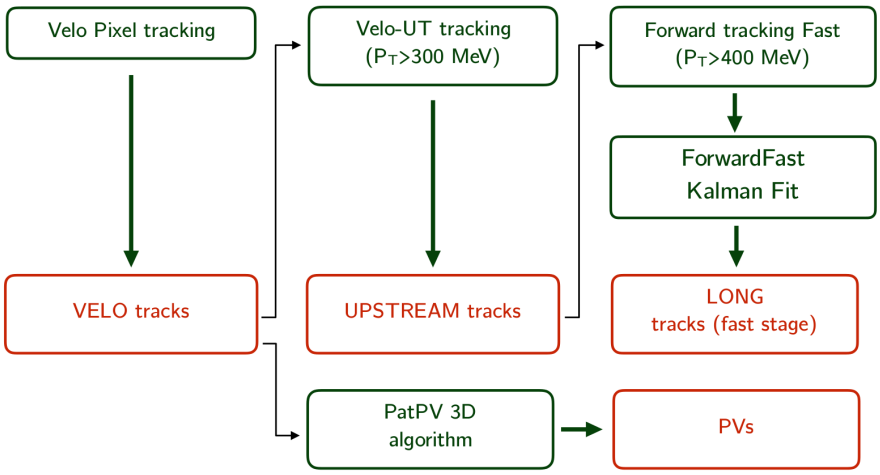

Figure 1: A schematic view of the fast tracking stage.

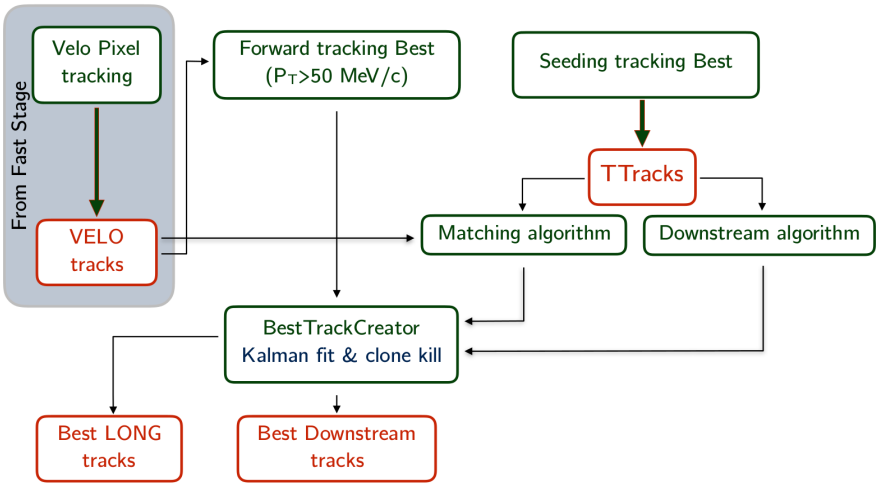

Figure 2: A schematic view of the best tracking stage.

2.5 Changes to individual algorithms since the trigger TDR

The reconstruction algorithms that form the fast and best sequences serve the same purpose as they did in the Run 1 and Run 2 trigger. As such, many of those described in the trigger TDR are under ongoing development to serve both the Run 2 trigger, and in preparation for the upgrade. Changes to these algorithms since the trigger TDR are described here:

2.5.1 VELO

VELO tracks are reconstructed using a simple track following algorithm, using seed tracklets formed from pairs of clusters on adjacent modules within a certain range of slopes. At the end of the track

finding stage, the accepted track candidates are fitted using a Kalman filter with a fixed amount of scattering. A detailed description of the algorithm can be found in Ref. [7]. The present implementation combines high reconstruction efficiency ($> 99\%$ for long tracks) and low ($\sim 2\%$) ghost and clone rates. While the VELO tracking algorithm is unchanged since the trigger TDR, the PV reconstruction has been adapted to make use of the same Kalman fitter as used here.

2.5.2 PV reconstruction

The Primary Vertex (PV) reconstruction is sensitive to the LHC running conditions, in particular to the multiplicity of the particles produced in the pp collisions. The average number of PVs in the single event increases by a factor of five between Run II and the Upgrade, which makes the reconstruction even more challenging.

The primary vertex algorithm begins with a set of input tracks. In Run II the PV reconstruction is based on VELO Kalman Filtered tracks. This strategy which is new with respect to the trigger TDR, has been incorporated into the Upgrade reconstruction [8]. Two Kalman Filters have been investigated: One using the full GEANT simulation detector description, and a much faster simplified Kalman Filter (a part of the VELO algorithm described in Sec. 2.5.1). Since no impact to the PV performance is found, the simplified version is used.

The PV reconstruction has been reoptimized to have the highest PV reconstruction efficiency as well as to minimize vertex misidentification. In principle, the new optimal working point has been found for the PV radial distance introduced in the reconstruction for Run II. This additional cut removes about 70% of misidentified PVs while maintaining 99% efficiency for true PVs. The new optimal working point has been found for a radial distance of r = 0.3 [mm], which is looser than the previous one r = 0.2 [mm].

The performance of the optimized reconstruction is described in Sec. 2.6.1.

2.5.3 VELO-UT

The VELO-UT algorithm extends VELO tracks into the UT. Due to a significant magnetic field between VELO and UT an estimate of the charge and the momentum of a particle can be provided. The charge and momentum estimate is used in further algorithms, as Forward tracking, to reduce search windows and thus combinatorics in pattern recognition. This improves the timing of Forward tracking and also reduces the number of fake tracks (ghosts).

The core of the algorithm is taken from the current VELO-TT code, which is adapted to the modified geometry of the UT. Several updates have been implemented since the description in Ref. [9]. The size of search windows related to the slope of tracks and tolerances of hits is tuned to have a higher reconstruction efficiency, especially for low momentum particles. The code has been optimized for speed by reducing the construction of temporary objects. The memory access has been reduced and the HitManager has been modified to allow a faster processing of hits.

The VELO-UT algorithm is only run in the Fast stage as input for Forward tracking. As a consequence, it has been tuned to reconstruct particles with a transverse momentum above 500 MeV/c. This also improves also the timing of the Forward tracking due to the reduction in input tracks.

2.5.4 Forward

The Forward tracking is one of two algorithms which produces long tracks in the best stage and the only algorithm which produces long tracks in the fast stage. Two configurations are implemented: In the best stage the algorithm is tuned to reconstruct all tracks with the highest possible quality. In the Fast stage the algorithm is sped up by about a factor of ten by focussing on reconstructing tracks with a transverse momentum above 500 MeV/c.

Since the trigger TDR a detailed timing study has been performed in which hotspots were identified and removed where possible. The data flow has been improved to reduce memory access. Using auto-vectorization several resource-intensive portions of the code have been sped up. Additionally a

significant improvement is observed from splitting the reconstruction process into two stages, as is done in the current Forward Tracking for Run II data taking: In a first stage tracks with at least five x-hits are searched for. In a second stage tracks with only four x-hits are reconstructed. Overall a timing improvement of about a factor of two is achieved.

A Neural Net classifier has been implemented in order to identify ghost tracks. Using this ghost track identification in the pattern recognition allows selection cuts to be relaxed. The number of ghost tracks is reduced by 44% while the efficiency is enhanced by 1.7%.

2.5.5 Seeding

The seeding algorithm performs a stand-alone track reconstruction using only hits in the SciFi, placed downstream the LHCb dipole magnet. The output of the seeding algorithm is T tracks, which are then used to feed the downstream and the matching algorithms.

The seeding algorithm used for the trigger TDR, called *PrSeedingXLayers*, has been replaced by a new algorithm *PrHybridSeeding* that makes several significant improvements in both speed and performance [10]. In particular a new implementation of the T-track search algorithm has been introduced which is two orders of magnitude faster and reduces the ghost rate by a factor of three with respect to the previous algorithm.

The tracking efficiencies for long tracks with $p > 5$ GeV/c is improved by around 5 $\%$, while for $p< 5$ GeV/c the gain is around 15 $\%$. Furthermore, the efficiencies for T tracks produced by daughters of long lived particles are significantly enhanced: for those having $p > 5$ GeV/c, the gain is around 7%, while it rises up to $15 - 20\%$ including also lower momentum ones. The *PrHybridSeeding* algorithm combined with the matching and the VELO tracking performs slightly better than the forward in the best tracking stage, having almost the same speed, the same efficiency for higher momentum tracks, higher efficiency for lower momentum tracks (3%) and a lower ghost rate (2-3 %). Globally, the *PrHybridSeeding* is about 3 times faster than the *PrSeedingXLayers*.

2.5.6 Matching

The track Matching algorithm is an alternative approach to the Forward pattern recognition. It takes T and VELO tracks as input and extrapolates them to the focal plane of the magnet and very close to the second T station.

The algorithm is improved with respect to the trigger TDR with an optimization of the track fit parameters and addition of a Neural Net for final track selection after a loose χ^2 requirement. The algorithm and its current performance is documented in Ref. [11].

2.5.7 Downstream

The algorithm that reconstructs downstream tracks, i.e. tracks from daughters of long-living particles like K_s and Λ, is called *PrLongLivedTracking* and is improved with respect to the one used for the Tracker Upgrade TDR. It was originally developed for Run II and improved the efficiency and ghost rate of the former algorithm (*PatDownstream*), used in Run I, while also being about 50% faster. The code has been ported for the upgrade, with minor tunings. The Run II algorithm is documented in Ref. [12].

2.5.8 Kalman Fit

While not benchmarked directly in the trigger TDR, a Kalman fit is required in order to produce the final fitted tracks and their associated covariance matrices. In order to correctly account for energy loss and multiple scattering, the Kalman fit relies on knowledge of the detector geometry. In Run 1 the trigger used a simplified geometry online in which the real detector volumes were replaced by effective volumes with effective materials designed to reproduce on average the performance of the full detector for a much lower timing. Offline, a full detector geometry was used as part of the reconstruction. In

Run 2 the simplified geometry was used in both the online and offline contexts. The Run I and Run II Kalman Fit framework is general enough such that the upgrade tracking detectors can be implemented, which until recently has used the full detector description in the trigger reconstruction sequence. To achieve a reasonable timing performance however, a "simplified geometry" for the upgrade is needed. The finalization of this upgrade simplified geometry is currently still ongoing: The number of effective volumes is unlikely to change, but their effective material descriptions will be optimised according to Ref [13]. The physics performance is likely to improve as a result, but the current timing and throughput performance is unlikely to change.

2.5.9 Ghost Probability Tool

The ghost probability is a multivariate classifier used to separate mis-reconstructed tracks (ghost track) from good tracks, using information from different stages of the track reconstruction and tracking sub-detectors.

For the upgrade, the tracking algorithms have changed as described earlier. Therefore, many of input variables used in the Run-I and Run-II ghost probability training have been replaced or modified, requiring a new classifier training. The optimization of ghost probability is still ongoing, with the present performance given in Sec. 2.6.2.

2.5.10 Particle ID

While not profiled explicitly in the trigger TDR, particle ID is a necessary component of the trigger in order to distinguish between different final states. The particle ID relies on reconstruction from three subdetectors, resulting in three particle ID algorithms: Muon, Calorimeter and RICH. The Calorimeter remains unchanged for the upgrade, and so the same underlying algorithms as used in Run II are used by default. These may in future be reopitimised to improve speed but are likely to be of similar performance.

The muon reconstruction algorithm has been revised during Run 2 in order to remove possible inconsistencies with the offline reconstruction and to optimise for CPU time consumption. The logic of the muon reconstruction is an algorithm (*isMuon*) that gives a boolean response. Based on an extrapolation of a long track into the muon stations it makes a statement about whether a track is consistent with a muon hypothesis. The expected hit positions in muon stations M1-M5 are evaluated from the extrapolation and only those within the acceptance are considered. Among the hits in the entire muon system, a search for hits within field of interest (FoI) around the extrapolated hit is performed for the M2-M5 stations.

The same algorithm is now available to the best reconstruction sequence for the Upgrade with minor modifications, i.e. the M1 station has been removed from the decoding of the hits. The timing performance as measured during Run 2 is expected to be approximately the same for the upgrade, as on one hand the number of decoded hits decreases by 50% due to M1 removal, while the number of hits in the other stations increases due to the increased luminosity.

The RICH reconstruction sequence has been rewritten to take advantage of the new software framework and modern programming paradigms.

This rewrite was completed at the end of 2016 where a first version of a new RICH reconstruction sequence, deploying the same basic processing steps but under the new framework, was made available. This new sequence gives statistically the same performance as the original Run I sequence but, due to various improvements the new framework allows, is almost a factor of two faster on Run II data. Future work on the algorithms, to improve the data structures, should lead to more speed improvements. The RICH reconstruction sequence is the first to have made the move to the new framework.

The rapid and smooth development of the new sequence means work is ongoing to deploy the new sequence already, for the 2017 data taking period.

2.6 Performance Update

In light of the changes described in the previous section to both the simulation and the reconstruction algorithms the overall performance of the reconstruction must be quantified. We describe here the studies to-date:

2.6.1 PV resolution

The PV resolution is strongly correlated to the number of tracks associated to the vertex, N, and can be described using the function:

$$\sigma(N) = \frac{A}{N^B} + C \tag{1}$$

where A, B, C denote free parameters. Figure 3 shows the PV resolution comparison between Run II and Upgrade simulation samples. Overall, a better resolution is found for the Upgrade sample.

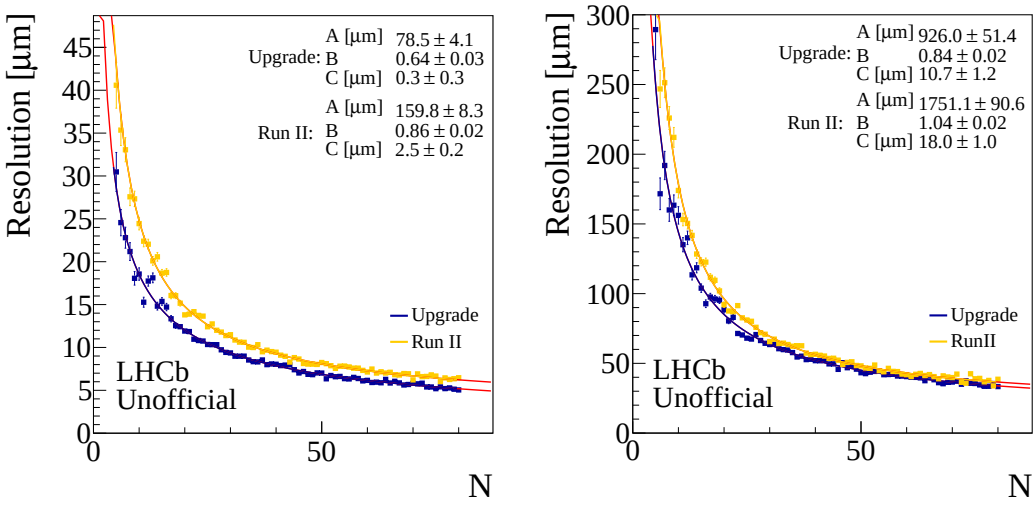

Figure 3: Comparison between the PV resolution in x, left, and z, right, measured in Run II and Upgrade Monte Carlo samples. The solid orange (blue) line corresponds to Run II (Upgrade). The resolution is parameterized by Eq. 1, and the result of the fit is indicated in the plot.

2.6.2 Ghost probability performance

The performance of trained ghost probability is studied by calculating the signal efficiency and ghost track rejection efficiency. The signal efficiency and ghost rejection efficiency for each track type are shown in Fig 4. For long tracks, with 70% ghost rejection, the signal efficiency is around 95%, which is degraded with respect to the signal efficiency at the same ghost rejection in Run-2, where the efficiency is 99%. This is expected to improve with further study.

2.6.3 Track reconstruction efficiencies

Table 1: Track reconstruction efficiencies and ghost rates of the fast and best stages as compared to the trigger TDR. The best stage efficiencies and ghost rates are shown for several values of the ghost probability requirement.

	Trigger TDR	Fast stage	Best Stage			
Ghost probability			< 0.9	< 0.75	< 0.3	< 0.1
Ghost rate	10.9%	5.6%	18.8%	15.2%	7.8%	4.2%
long	42.7%	42.9%	91.1%	90.8%	88.2%	84.3%
long, from B	72.5%	72.7%	94.8%	94.6%	93.1%	90.6%
long, from B, $p_T > 0.5GeV$	92.3%	92.5%	96.5%	96.4%	95.4%	93.6%

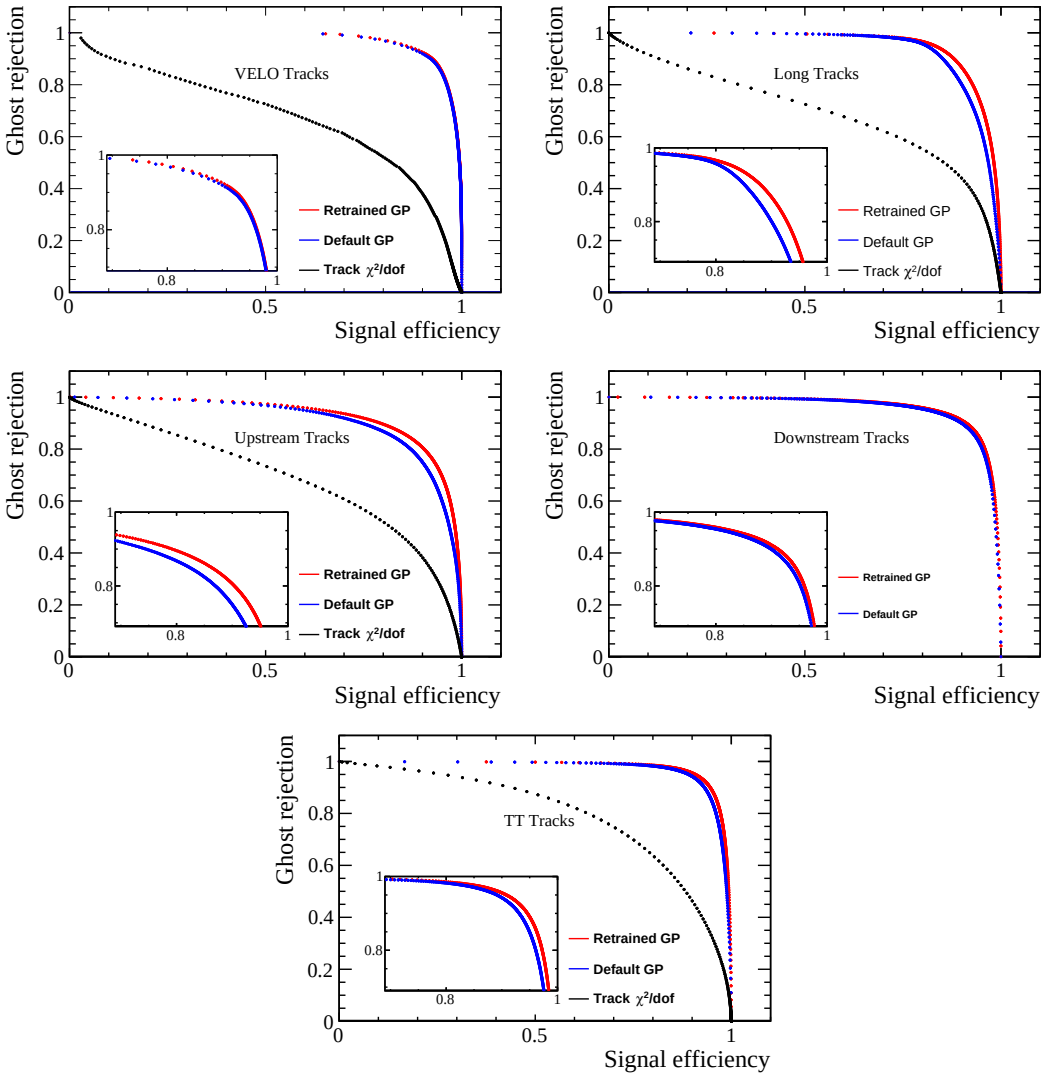

Figure 4: Comparisons of signal efficiency and ghost track rejection efficiency from the most recent ghost probability training compared to the previous version. The track χ^2/ndof distribution is overlaid, and an insets are included showing the region around 90%.

The tracking efficiency is determined on the $B_s^0 \to \phi\phi$ signal sample. The efficiency is determined as the number of reconstructed candidates with respect to MC particles that have left sufficient clusters in the subdetectors to be considered reconstructible. The efficiency includes the ghost probability requirement. In table 1 it can be seen that the efficiency for all long tracks is considerably higher in the best stage due to the looser momentum requirement. However, high momentum particles from beauty hadrons are reconstructed in the fast stage with an efficiency very similar to what is possible with significantly more computing time in the best stage. In all cases the performance is equal to or better than that of the trigger TDR with the fast stage having about half the ghost rate: Improvements in the algorithms counteract the more realistic detector description including spillover, larger gaps and modified acceptance in the Sci-Fi. For the best stage several efficiencies are provided as a function of the ghost probability requirement. This will be tuned to meet physics performance and throughput requirements.

3 Timing & throughput measurements

In the trigger TDR the timing was determined for the reconstruction sequence that would be needed prior to Kalman fit and selections. The timing measurement is repeated here using the updated detector description, digitisation and algorithm improvements. We present a direct comparison between the fast sequence and the sequence used in the trigger TDR, and include for the first time the best sequence timing to server as a baseline.

The extrapolations presented in the trigger TDR used the per-event timing of a single process to determine the expected farm capacity in 2021. This relies on three multiplicative factors: The scaling from a single process on a single core to multiple processes on the present number of logical cores, the extrapolated growth rate at equal cost for a node purchased in 2020, and the number of farm nodes expected to be purchased with the available budget. A more useful extrapolation is to measure the *throughput* on a fully loaded EFF candidate node, which then removes the need to measure the first multiplicative factor, only needing the growth rate at equal cost. We present here both the timing and throughput, using the throughput as input to the updated extrapolation.

In the trigger TDR the candidate EFF node consisted of a dual-socket Intel X5650 'Westmere' server node in which each processor has 6 physical cores with clock frequencies of 2.67GHz and two virtual processing units, sharing 24GB of RAM. The X5650 nodes were purchased in 2010, at a unit cost of 2.5 kCHF. At the time of purchase this was the most cost-efficient solution. In this note we compare to a more recent EFF candidate node purchased in 2016 consisting of a dual socket Intel 'E5-2630 v4' server node in which each processor has 10 physical cores with clock frequencies of 2.20GHz and two virtual processing units, sharing 128GB of RAM. In 2016 these nodes had a unit cost of 3 kCHF.

3.1 Global Event Cut

In the trigger TDR a Global Event Cut (GEC) was used based on our experience of using a similar cut in Run I and Run II. This GEC reduces the average event multiplicity which reduces the average time needed to process an event, with the added advantage that cleaner events have a higher tracking efficiency. We apply the same GEC here as was used in the trigger TDR for consistency, consisting of a cut on the sum of clusters in the ECAL and HCAL. The value can be adjusted to meet timing requirements.

3.2 Single process timing comparison

The timing of the reconstruction sequence is performed in a manner identical to that of the trigger TDR: A candidate EFF node on which no other processes are running is used, and the timing is determined by taking an average over 20 runs using 20000 minimum bias events in each run. The average time per event for the sequence is compared to the trigger TDR in table. 2 using the X5650 trigger TDR benchmark node.

Table 2: Timing of the fast stage compared to the one of the trigger TDR with GEC cuts applied. Both timing tests are performed using an X5650 EFF node described in the text.

Timing [ms]	Trigger TDR	Fast stage
VELO tracking	2.0	2.0
VELO-UT tracking	1.3	0.5
Forward tracking	1.9	2.3
PV finding	0.4	1.1
total	5.6	6.0

The VELO-UT tracking has increased in speed considerably with no loss in performance as described in the previous section. The new PV finding algorithm is slower, however, it is yet to be fully optimized: We expect this to improve. In spite of the additional complexity of the digitisation, the overall timing is similar to that measured in the trigger TDR, while still showing the potential to improve.

3.3 Throughput

The timing test gives us confidence that the overall software performance estimated in the trigger TDR still holds. However, in the EFF multiple instances of the sequence are executed in parallel in Run 2, with a considerable effort in optimising the number of concurrent processes. For the Upgrade the new software framework will go further, leveraging a fully multiprocessor paradigm. While the timing is useful to determine the areas in which further optimisation may be needed, the overall performance is best determined by throughput, as measured in Events/s on a fully-loaded EFF node.

The throughput tests are performed using compressed datasets. In the raw measurements there is an overhead due to uncompressing the event and loading the objects into memory. When running during data taking, this overhead is not present as uncompressed events are streamed to the farm nodes. The measured overhead, which accounts for about 10 ms/Event, is subtracted from the raw throughput numbers. To do this the throughput is measured in the absence of the reconstruction sequence. The reconstruction sequence throughput is then calculated as:

$$\text{Throughput} = \left[\frac{1}{\text{Throughput}_{\text{total}}} - \frac{1}{\text{Throughput}_{\text{no reco}}}\right]^{-1} \tag{2}$$

For both the X5650 and the E5-2630 v4 candidate nodes, the number of processes that maximises the throughput has been studied. In both cases the optimal number of processes is slightly higher than the number of virtual processors: 26 in the case of the X5650 and 44 in the case of the E5-2630 v4. The scaling from a single process to the maximum throughput is given in table 3 for the X5650 candidate node. The fast and best sequences are shown, however no selection is applied between the fast and best sequence, which will not be the case in future. The best sequence performance is shown only to study its scaling from single- to multi- process operation. It can be seen that the fast stage throughput increases by a factor of 15 when moving from a single process to the maximum. Including the Kalman fit with full detector geometry reduces this scaling by a large amount due to the additional memory access load the detector geometry adds to the system. The Kalman fit using simplified geometry is also shown, and highlights this effect: The reduced memory footprint of the simplified geometry increases the relative performance from $8\times$ to $14\times$. The best stage has a negligible effect on the scaling, indicating that it is not memory bound in the present configuration.

Table 3: Throughput on the x5650 candidate EFF node for different sequences, the full sequence, only the fast stage and the fast stage without Kalman fit. The Kalman fit is shown for both the full and simplified detector geometries.

Throughput [Events/s]	Single Core	Complete node	Ratio
Fast stage	133	1991	15.0
Fast stage + Kalman Fit	8.5	73.1	8.6
Fast stage + Kalman Fit (simplified)	59	823	13.9
Fast+Kalman+Best	0.9	7.3	8.1
Fast+Kalman+Best (simplified)	4.6	72.8	15.8

For the purpose of updating the extrapolation in a later section, the same throughput test is applied to the more recent E5-2630 v4 node and a comparison between this and the X5650 are shown in table 4. The E5-2630 v4 is found to be on average a factor of 1.7 faster than the X5650 candidate node when both are measured at their maximum throughput for the fast reconstruction sequence. The relative gain in performance is larger for sequences that are more memory intensive, with the full fast + best sequence including the complete detector description in the Kalman fit exhibiting a speedup of 2.1 between the X5650 and E5-2630 v4 candidate nodes.

Table 4: Throughput comparison between the X5650 and E5-2630 v4 EFF candidate nodes when fully loaded.

Throughput [Events/s]	X5650	E5-2630 v4	Ratio
Fast stage	1991	3376	1.7
Fast stage + Kalman Fit	73.1	134	1.8
Fast stage + Kalman Fit (simplified)	823	1611	2.0
Fast+Kalman+Best	7.3	15.9	2.2
Fast+Kalman+Best (simplified)	72.8	150	2.1

4 Event Filter Farm Performance Extrapolation

In this section we first summarise the original performance extrapolation for the trigger TDR, and produce a revised estimate based on the throughput measurements made in the previous section. We use the fast sequence for the extrapolation as this was the same sequence used in the trigger TDR, and the best sequence is not yet representative of the sequence we expect to run in 2021.

4.1 Performance extrapolation of the trigger TDR

The trigger TDR performance extrapolation determined the time per event for a single process to be $t = 13$ ms total. The extrapolation used was:

$$t = \frac{g^y \times n \times N}{R} \tag{3}$$

Where g is the performance growth per year at equal cost, $y = 9$ is the number of years before purchase, $n \times N$ is the total number of processes per node times the number of nodes and R is the bunch crossing rate of $30\mathrm{MHz}$. For the extrapolation n was assumed to be equal to the number of virtual cores, 24 in the case of the X5650 candidate node, and $N = 1000$ was the number of X5650 nodes purchasable with the 2.8MCHF budget.

The performance growth per year at equal cost was estimated in the trigger TDR to be $g = 1.365$. This figure was based on experience with procurement for the LHCb event-filter farm between 2007 and 2010. This number was higher than the number used in many extrapolations by the CERN IT department, which at the time was 1.25, motivated by the fact that our farm-nodes used a significantly cheaper configuration than the ones generally bought by CERN IT. Moreover Moore's law, in the form of doubling the number of transistors on equal surface every 18 months, suggested an even higher growth rate of almost 1.6.

4.2 Revised growth factor

The revised growth factor is determined using the throughput ratio of the fast sequence between the 2010 X5650 and 2016 E5-2630 v4 EFF candidate nodes shown in table 4, where the performance growth at equal cost is measured to be 1.7 over six years, corresponding to a growth per year at equal cost of $g = 1.1$. This is considerably lower than the trigger TDR estimate and lower even than the CERN IT estimate of 1.25. The X5650 nodes purchased in 2010, were excellent value for money: The X5650 CPU was only 30% slower than the fastest then-available CPU while costing 40% less.

In recent tenders, at equal cost per server, LHCb has been quoted lower-end CPUs. In 2016 the E5-2630 v4 nodes purchased are less than half as fast than the best CPU available and over 80% cheaper [14]. This means that, while machines are available that could be considered to fulfill the extrapolation of the trigger TDR, their prices are so high, that they are not economically feasible for our particular workload any more. This trend is also seen in the regular procurements of the CERN IT department. Figure 5 shows this trend: The HEPSPEC06 benchmark as applied to recent CERN IT procurements shows a continuing price decrease for compute power, for our type of application. The decrease is not constant however. There are several points at which the rate at which compute becomes cheaper changes. One major change took unfortunately place right after the reference node year that was used for the trigger TDR.

This means that the X5650 candidate node served to overestimate the growth rate at equal cost between 2007 and 2010. while for the extrapolation between 2010 and 2016 it may serve to underestimate. For the updated extrapolation we use the measured value of $g = 1.1$.

4.3 Revised performance extrapolation

The revised extrapolation is based on the measured throughput of the fast sequence on the 2016 E5-2630 v4 candidate node.

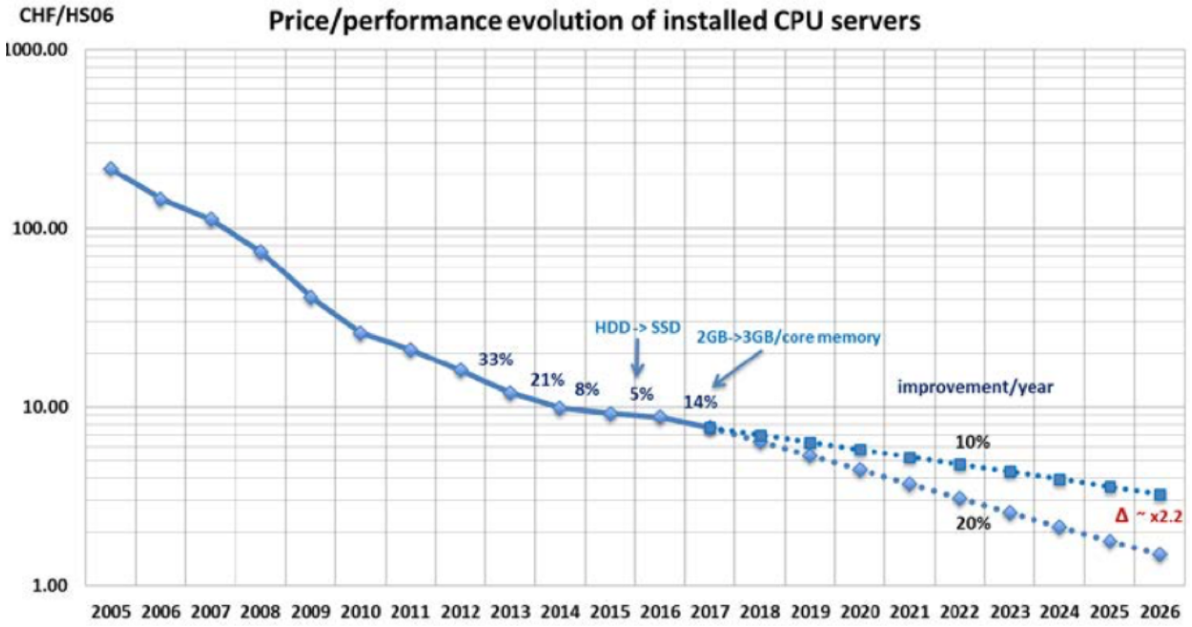

Figure 5: The evolution of the cost of one HEPSPEC06 unit in CHF as a function of the year of procurement in purchasing procedures conducted by the CERN IT department. Two extrapolations are shown assuming a 10% and 20% growth rate. Source: Bernd Panzer CERN/IT

The throughput based extrapolation is defined as:

$$T = T_{\text{meas}} \times g^{y} \times N \tag{4}$$

Where $y = 4$, $g = 1.1$ as determined previously, $N = 1000$ and $T_{\text{meas}} = 3376\text{Hz}$ as measured in Sec. 3.3. The extrapolated fast sequence throughput is then 4.9 MHz. This corresponds to a factor of six discrepancy between the extrapolated throughput and the throughput necessary to run the fast sequence at the full event rate of 30 MHz.

A major contributor to the discrepancy is the change in processor design in recent years. The present LHCb software framework was written at a time when CPU growth came almost exclusively through higher CPU frequencies [15] [16]. Today's growth comes via increasing core counts and vectorized processing units [17]. Figure 6 illustrates the consequences of this trend. It shows the progression of Run 2 trigger decision throughput and CPU floating point operations (FLOPS) as a function of time, measured on candidate EFF nodes.

At around 2010 a transition occurred between increasing CPU frequencies and increasing core counts[c]. It was also the start of a shift to increasing FLOPS via SIMD instructions rather than scalar processing on a large scale [18]. This is the major contributor to the gap that is visible in Figure 6

Another non-negligible contribution comes via the gap between memory performance and CPU power [19]. Since CPUs and the attached memory no longer scale in frequency the performance of both components has become decoupled. While CPU performance went up at the usual rate, memory bandwidth and latency have started to lag behind. With the high core counts of modern CPUs, contention of the memory system causes significant penalties to program performance unless optimised. This effect is illustrated in Figure 7. Even though cache sizes have increased, last level cache hit rate has decreased with every successive generation of CPU for the present LHCb software and has reached a point where almost half of all clock cycles are lost due to waiting for memory access. Most of this effect is currently hidden by hyper-threading, which partially covers these stalls by scheduling a different process. We have however observed diminishing returns on the effectiveness of hyper-threading on the last two generations of CPUs.

[c]It should be pointed out that pure scaling through frequency already ended several years earlier. What we mean here is that versions of this CPU generation had a fixed number of cores and the performance within that generation varied via clock frequency rather than core count

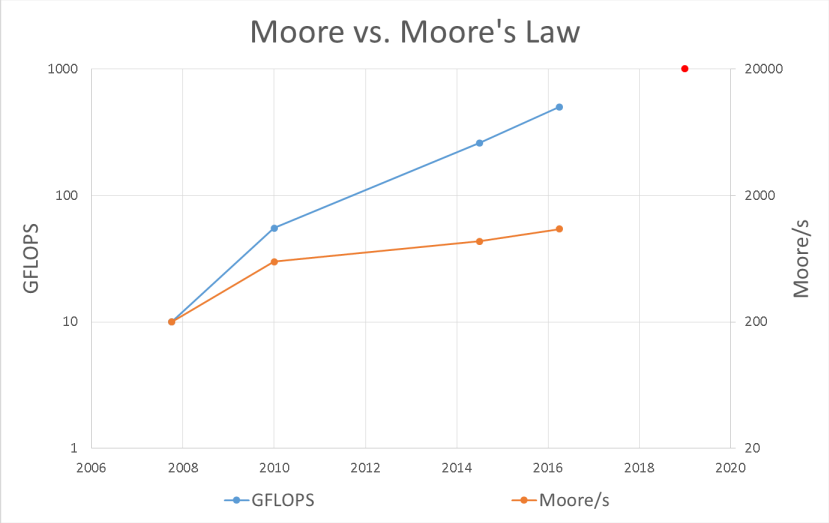

Figure 6: Comparison of CPU performance in Floating Point Operations per Seconds (FLOPS) and trigger decisions per second (Moore/s) for the 2014 version of the HLT.

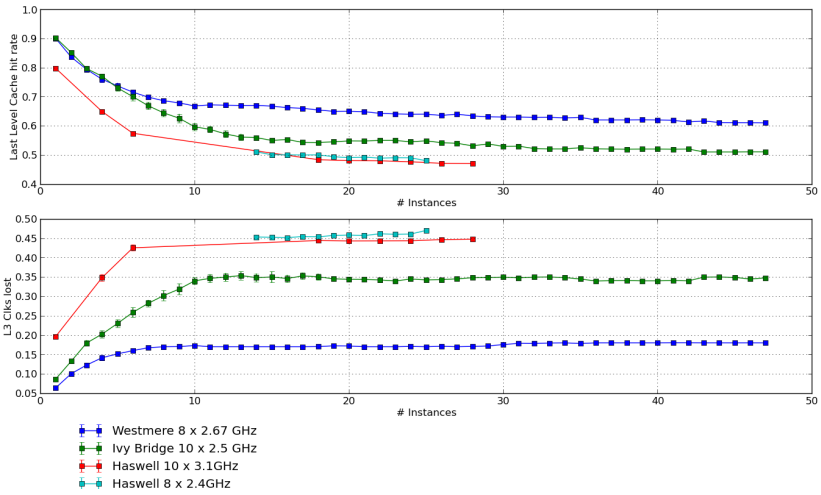

Figure 7: Cache hit rate and CPU stalls due to cache misses vs. number of running HLT process instances on different Intel micro architectures.

These factors are known to LHCb, hence the drive to modernise the software framework as indicated in ref [5]. There is clear potential for improving our gains from modern CPUs by better adopting our software to the new hardware situation. The RICH reconstruction sequence is the first to have moved to the new framework, and shows a marked improvement in performance, indicating that further gains are not unreasonable at the algorithm level prior to making full use of vectorisation.

5 Conclusions

In this document we have presented an updated extrapolation with respect to the trigger TDR. We find, using a more realistic detector description and improved reconstruction algorithms, that the overall performance is unchanged. However, we also find that with the present LHCb software as used in Run 1 and Run 2, the growth rate at equal cost has decreased significantly.

The present shortfall for the fast reconstruction sequence is found to be a factor of six with respect to the 30MHz throughput required. This is attributed to two factors: An overestimate in the original extrapolation and a fundamental change in the way in which modern CPUs are designed, which the present software framework was not designed to capitalise upon. The reconstruction sequence and its throughput described in this document is intended to serve as a baseline for the new software framework. Which has already shown significant performance increases in a preliminary version of the RICH reconstruction algorithm.

Significant effort will be expended in the coming years to port the current reconstruction sequences to the new software framework. The updated throughput of these sequences will dictate the optimal output rate of events selected at the fast stage in combination with an extrapolation of the available disk buffer. These studies will be presented in a future performance document.

6 References

[1] R. Aaij *et al.*, *The LHCb Trigger and its Performance in 2011*, JINST **8** (2013) P04022, arXiv:1211.3055.

[2] J. Albrecht, V. V. Gligorov, G. Raven, and S. Tolk, *Performance of the LHCb High Level Trigger in 2012*, arXiv:1310.8544.

[3] LHCb Collaboration, R. Aaij *et al.*, *LHCb Trigger and Online Upgrade Technical Design Report*, Tech. Rep. CERN-LHCC-2014-016. LHCB-TDR-016, May, 2014.

[4] LHCb Collaboration, R. Aaij *et al.*, *LHCb Tracker Upgrade Technical Design Report*, Tech. Rep. CERN-LHCC-2014-001. LHCB-TDR-015, Feb, 2014.

[5] C. Bozzi *et al.*, *Upgrade Software and Computing TDR: a roadmap*, Tech. Rep. LHCb-INT-2016-016. CERN-LHCb-INT-2016-016, CERN, Geneva, Mar, 2016.

[6] R. Aaij *et al.*, *Tesla : an application for real-time data analysis in High Energy Physics*, Comput. Phys. Commun. **208** (2016) 35, arXiv:1604.05596.

[7] T. Bird *et al.*, *VP Simulation and Track Reconstruction*, Tech. Rep. LHCb-PUB-2013-018. CERN-LHCb-PUB-2013-018, CERN, Geneva, Oct, 2013.

[8] A. Dziurda and J. Wanczyk, *Primary Vertex Reconstruction for Upgrade at LHCb*, Tech. Rep. LHCb-PUB-2017-002. CERN-LHCb-PUB-2017-002, CERN, Geneva, Jan, 2017.

[9] E. E. Bowen, B. Storaci, and M. Tresch, *VeloTT tracking for LHCb Run II*, Tech. Rep. LHCb-PUB-2015-024. CERN-LHCb-PUB-2015-024. LHCb-INT-2014-040, CERN, Geneva, Apr, 2016.

[10] R. Quagliani, Y. Amhis, F. Polci, and P. Billoir, *Description of the hybrid seeding for the SciFi*, Tech. Rep. LHCb-INT-2015-025. CERN-LHCb-INT-2015-025, CERN, Geneva, Jun, 2015.

[11] S. Esen and M. De Cian, *A Track Matching Algorithm for the LHCb upgrade*, Tech. Rep. LHCb-PUB-2016-027.

[12] A. Davis, M. De Cian, A. M. Dendek, and T. Szumlak, *PatLongLivedTracking: A tracking algorithm for the reconstruction of the daughters of long-lived particles in LHCb*, Tech. Rep. LHCb-PUB-2017-001. CERN-LHCb-PUB-2017-001, CERN, Geneva, Jan, 2017.

[13] B. Couturier, C. Hasse, T. Szumlak, and E. Zacharjasz, *Simplified Detector Description for LHCb Upgrade*, Tech. Rep. LHCb-PUB-2017-003. CERN-LHCb-PUB-2017-003, CERN, Geneva, Mar, 2017.

[14] Microway, *Detailed Specifications of the Intel Xeon E5-2600v4 Broadwell-EP Processors*, , Online; accessed 25-Jan-2017; https://www.microway.com/knowledge-center-articles/detailed-specifications-of-the-intel-xeon-e5-2600v4-broadwell-ep-processors/.

[15] G. Barrand *et al.*, *GAUDI - A software architecture and framework for building HEP data processing applications*, Comput. Phys. Commun. **140** (2001) 45.

[16] I. Corporation, *Intel ARK - Product Overview - Westmere family of processors*, , Online; accessed 25-Jan-2017; https://ark.intel.com/products/codename/38530/Broadwell#@Server.

[17] I. Corporation, *Intel ARK - Product Overview - Broadwell family of processors*, , Online; accessed 25-Jan-2017; https://ark.intel.com/products/codename/38530/Broadwell#@Server.

[18] P. E. McKenney, *Is Parallel Programming Hard, And, If So, What can You Do About It?*, 2-Jan-2017. Online; https://www.kernel.org/pub/linux/kernel/people/paulmck/perfbook/perfbook.2017.01.02a.pdf.

[19] M. V. Wilkes, *The memory gap and the future of high performance memories*, ACM SIGARCH Computer Architecture News **29 Issue 1** (2001) 2.