

FPGA Development for the LHCb Vertex Locator Upgrade

Nicholas Mead
8064141

School of Physics and Astronomy
University of Manchester

January 4, 2016

Abstract

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur blandit purus ut lacus aliquam, a sodales ante sodales. Etiam a elit nunc. Mauris ipsum tellus, ullamcorper et arcu at, cursus malesuada elit. In tempus pellentesque nisi, vel egestas enim cursus tempus. Sed velit urna, luctus sed efficitur sed, laoreet vitae magna. Mauris elementum dignissim lacus vitae tempus. Curabitur laoreet molestie dictum. Donec sit amet auctor nisl.

Duis pellentesque euismod pellentesque. Praesent volutpat tincidunt eros, at faucibus tellus eleifend a. Quisque molestie sed ante sit amet sodales. Duis sed justo quam. Curabitur tellus felis, laoreet et bibendum a, posuere eget nisi. Donec suscipit lacinia porttitor. Aenean posuere sem nibh, et iaculis nisl faucibus eu. Donec ac posuere sapien. Aenean suscipit, nisi eget porttitor viverra, dui sapien vulputate lectus, ut dapibus purus orci nec arcu. Etiam placerat sapien non massa fringilla, et malesuada nibh hendrerit. Vestibulum et porttitor mi. Aliquam turpis velit, rutrum vitae erat at, scelerisque cursus lacus. Praesent libero urna, sodales efficitur eros id, sodales lacinia sem. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

Contents

24	1 Introduction	1
	1.1 The Standard Model of Particle Physics	1
26	1.2 The LHCb Experiment	1
	1.3 LHCb Upgrade	3
28	1.3.1 VELO Upgrade	3
	1.3.2 The Role of FPGA's in the VELO Upgrade	3
30	1.3.3 Data Flow and Low Level Interface	3
	2 Scrambler	3
32	2.1 Scrambler Options	4
	2.2 Cross Checks	5
34	2.3 Algorithm Analysis	5
	2.3.1 Measurements of the Algorithms	6
36	2.3.2 Statistical Predictions	7
	2.3.3 Results of Analysis	9
38	2.4 Conclusion	12
	References	14

1 Introduction

1.1 The Standard Model of Particle Physics

Central to the modern study of particle physics is the standard model,

$$\begin{aligned}
L_{GWL} = & \sum_f (\bar{\Psi}_f (i\gamma^\mu \partial_\mu - m_f) \Psi_f - e Q_f \bar{\Psi}_f \gamma^\mu \Psi_f A_\mu) + \frac{g}{\sqrt{2}} \sum_i (\bar{a}_L^i \gamma^\mu b_L^i W_\mu^+ + \bar{b}_L^i \gamma^\mu a_L^i W_\mu^-) \\
& + \frac{g}{2x_w} \sum_f \bar{\Psi}_f \gamma^\mu (I_f^3 - 2s_w^2 Q_f - I_6 e_f \gamma_5) \Psi_f Z_\mu - \frac{1}{4} |\partial_\mu A_v - \partial_v A_\mu - ie(W_\mu^- W_v^+ - W_\mu^+ W_v^-)|^2 \\
& - \frac{1}{2} |\partial_\mu W_v^+ - \partial_v W_\mu^+ - ie(W_\mu^+ A_v - W_v^+ A_\mu) + ig' c_w (W_\mu^+ Z_v - W_v^+ Z_\mu)|^2 \\
& - \frac{1}{4} |\partial_\mu Z_v - \partial_v Z_\mu + ig' c_w (W_\mu^- W_v^+ - W_\mu^+ W_v^-)|^2 - \frac{1}{2} M_\eta^2 \eta^2 - \frac{g M_\eta^2}{8 M_W} \eta^3 - \frac{g'^2 M_\eta^2}{32 M_W} \eta^4 \\
& + |M_W W_\mu^+ + \frac{g}{2} \eta W_\mu^+|^2 + \frac{1}{2} |\partial_\mu \eta + i M_Z Z_\mu + \frac{ig}{2c_w} \eta Z_\mu|^2 - \sum_f \frac{gm_f}{2M_W} \bar{\Psi}_f \Psi_f \eta. \quad (1.1)
\end{aligned}$$

The standard model, shown in equation 1.1, is a quantum field theory that describes the fundamental particles and how they interact. While this report does require, or attempt, a detailed understanding the intricate detail of the standard model; the aim of many particle physics experiments is to verify, measure and expand the model. Despite being the current best theory to explain particle interactions, the model is not complete. There are many undescribed phenomena, such as the matter domination in the universe, that require physics beyond the standard model in order to be described. To that end, major international efforts, namely in the form of the Large Hadron Collider, aim to gain further knowledge and understanding of the underlying physics of the universe. [1]

1.2 The LHCb Experiment

One experiment at the Large Hadron Collider is Large Hadron Collider beauty (LHCb). Located at intersection point 8, LHCb is designed to study rare particle physics phenomena, such as lepton flavour violation and CP violation. The decays studied in the LHCb are via exotic hadronic decays of Bottom or Charm quarks that form short lived hadrons. These hadrons, commonly B mesons, travel in the order of mm's in the detector before decaying. As such, B meson decays can be identified by decay products that propagate via a secondary vertex.

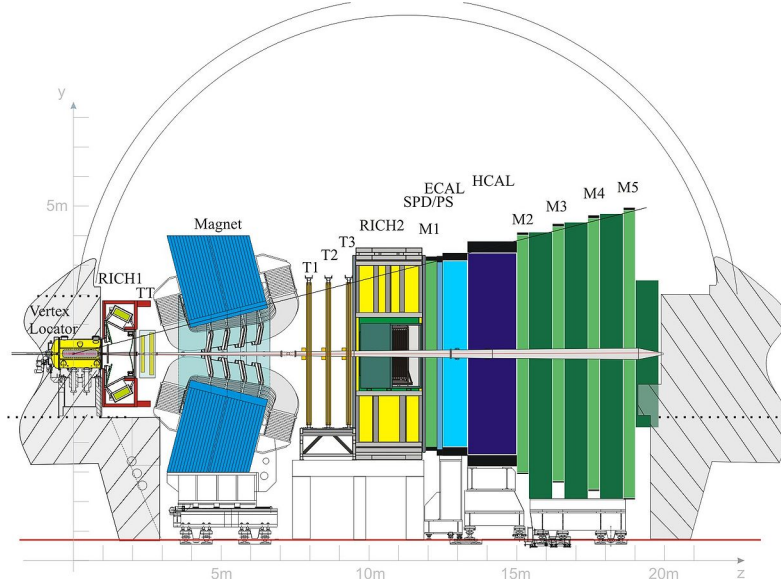


Figure 1.1: The LHCb Detector along the bending plane.

As B mesons are light (in comparison to other particles studied in the LHC), the decay products are produced at a shallow angle relative to the beam pipe; this is the driving factor in the design of the experiment. LHCb is a single arm forward spectrometer. Surrounding the point of collision is the Vertex Locator (VELO), this high precision detector uses silicon strips to detect ionising particles as they propagate from a collision and provides the coordinates of the particle in terms of R^1 and ϕ^2 . By reconstructing the paths of particles back to the intersection point, it can be identified whether or not the particular decay products are a product of the primary vertex³, or a secondary vertex⁴.

The Rich detector, comprised of two subdetectors either side of the magnet, uses Cherenkov radiation to deduce the velocity of the particle. The silicon trackers, labeled TT and T1-3 in Figure 1.1, calculate the angle deflection by the magnet. By combining the velocity and angle of deflection, the mass, momentum and energy of the particles can be deduced from simple relativistic kinematics.

The muon detectors, labeled M1-5 in Figure 1.1, are important to detect muons. This is of particular importance on LHCb as muons can be easily misidentified as charged pions, due to their similar mass.

HCAL and ECAL, shown in Figure 1.1, are hadronic and electromagnetic calorimeters respectively. Both measure the total energy of incoming particles. As the calorimeters are absorbing the particles they detect, any leptonic particle reaching the M2-5 muon detectors can be assumed to be a muon. Electrons and Photons are absorbed by the ECAL and any Tauons would have decayed long before reaching the far muon detectors.

¹Radial distance from the beam pipe.

²Azimuthal angle.

³The position at which the protons collided.

⁴The decay point of a short lived particle. i.e. B Meson.

1.3 LHCb Upgrade

With the advancements in accelerator technology, the detectors must also advance in order to make best use of the accelerators. The LHC is scheduled to increase its luminosity during Long Shutdown 2 (LS2), and as such LHCb will have to cope with this greater luminosity. The front end electronics of LHCb implement a hardware trigger and this is limited to a 1MHz maximum readout speed. Post LS2, LHCb will have to cope with a luminosity of $\mathcal{L} = 2.10^{33} \text{cm}^{-2} \text{s}^{-1}$, this is significantly greater than the current $\mathcal{L} = 4.10^{32} \text{cm}^{-2} \text{s}^{-1}$. A simple luminosity increase will not significantly increase the statistics for some statistical error dominated channels. To achieve this, greater resolution of the VELO and fully software triggers are required. Detailed in the ‘*LHCb VELO Upgrade Technical Design Report*’ [2] the main goals of the 2019 upgrade are as follows:

- Increase the luminosity to $\mathcal{L} = 2.10^{33} \text{cm}^{-2} \text{s}^{-1}$.
- Read data from the detector at the bunch crossing frequency, 40 Mhz.
- Convert to a fully software based trigger.

1.3.1 VELO Upgrade

Common with its predecessor, the upgraded VELO uses thin, retractable modules. The advantage of this approach is that during collisions, the modules can sit closer to the beam line than otherwise possible. The modules retract for the beam fill, avoiding the radiation damage from the wider fill beams. In order to gain greater resolution of secondary vertices, the upgraded VELO will sit at 5.1 mm from the beam at the closest pixel [2]. The current VELO achieves 8 mm [3].

As previously mentioned, the current VELO uses silicon strips to detect particles. The upgraded VELO, however, will use silicon pixels. At

1.3.2 The Role of FPGA’s in the VELO Upgrade

1.3.3 Data Flow and Low Level Interface

2 Scrambler

Due to radiation levels inside the detector chamber, the main data processing takes place in a concrete bunker away from the detector. To facilitate this, 20 optical links (per module) are used to transfer the data from the front end VELO to the Data Acquisition FPGA (DAQ). When communicating data digitally, the transferring module (TX) and the receiving module (RX) must have synchronised clocks. In these cases, the GWT serialiser is

the TX, and the DAQ is the RX. When achieving synchronised clock, there are two main approaches:

- Transmit the TX clock with the data to the RX modual - used in I²C and SPI communication.
- Use bit-changes in the data to continuously synchronise the RX clock.

The former of these options, although widely used in convertional electronics, requires a finely tuned clock accounting for all possible delays. The latter, while negating cons of the former, requires data with a high density of tranitions to reduce the likelihood of a desynchronisation event. Becuase delays in the data are possible, the latter option has been selected.

A it is nessesary to ensure that the data has large density of transitions before being transmitted from the front-end detector to the DAQ modual. However, as the majority of super pixel hitmaps are empty, the data has a bais towards '0's. This reduces the frequency of transitions in the data - increasing the probability of a desynchronisation event. It is therefor nessecary to scramble the data prior to transmtion and descramble the data in the DAQ FPGA.

Scrambling and later descrambling the data is not a trivial exercise. The scrambling (TX) modual and descrambling (RX) modual must use a sycronised 'key', that is used in both the scrambling and descrambling processes. In the FPGA, the 'key' is derived from the previous states of the data. There are two methods when generating this 'key':

Additive The 'key' is generated by evolving the previous 'key' at each itteration of data using the incoming frame.

Multiplicative The 'key' is generated from the previos n frames. (Here n is a variable specific to the algorithm).

2.1 Scrambler Options

Three scrambling algorithms have been concidered:

Additive Scrambler

This scrambler is was origionally impremented and used two sets of two-input XOR logic gates. As the name implies, this scrambler used additive key generation which is dependent all previous input frames since the last reset signal.

Intermediate Scrambler

Created by Karol Hennessy, and deriving its name arbitrarily from the order of consideration, this multiplicative scamber combines the current and previous frames to generate the 'key'. Therefor, in the event of desynchronisation, only two frames are

lost before the ‘key’ is automatically recovered. This feature alone is a significant improvement over the Additive Scrambler.

VeloPix Scrambler

This is the current implemented scramble algorithm in the DAQ and VeloPix code. Like the Intermediate Scrambler, it uses multiplicative ‘key’ generation. However, the VeloPix scrambler is compatible with further constraints enforced by the ASIC, including the number of combinational logic operations. The Intermediate Scrambler was design purely for simulation purposes and as such does not meet these constraints.

2.2 Cross Checks

The main priority when scrambling data, is ensuring that the data is recoverable. For all three scramblers, the algorithm was synthesised in Quartus⁴ and simulated in Modelsim⁵. The aim of synthesising and simulating the scramblers in these programs was to ensure that the design was both physical in term of on-board logic gates, and to check that the scrambled data was recoverable, respectively.

Furthermore, a C++ simulation was created for the three scramblers. This simulation had two main purposes: firstly to cross check the output of the C++ against the Modelsim simulations; secondly to simulate the scrambler over a much larger simple of data as Modelsim simulations are less time effecient. In attition to the cross checks, the C++ code allowed for the injection of a descronisation event, in which the ‘key’ is lost. As expected, the Additive Scarmbler was unable to recover any data post descronisation, however the intermediate and VeloPix scarmblers both recovered the ‘key’ after two frames and contiinoud to recover data.

2.3 Algorithm Analysis

For analytical purposes, it is assumed that fully scrambled data is indistinguishable from randomly generated data. For this reason, the three algorithm are not only tested against eachother and the pre-scrambled QWT data but also randomly generated binary. The randomly generated data was created using the Python ‘random’ library, selecting a ‘0’ or ‘1’ with equal probubility. While the Python ‘random’ library is only sudo-random, on the scale of this example (i.e. >> 100,000 frames), it is by far sufficient.

A more mathematically rigorous approuch, however, is to evaluate the system abstractly in the framework of statistical physics. In this abstraction, the 120 bit frame (with the header and parity removed) is concidered an ensemble; microstates are the particular form of the frames; and macroscopic quantities can be calculated by averaging a large number of frames (i.e. the desync data). For the analysis outlined in section 2.3.1, predictions will be made using these principles and outlined in section 2.3.2.

In the context of the statisical model, it is reasonable to concider the degree of ‘scrambled-

ness' analogous to entropy. This analogy is not dissimilar to the common interpretation
of entropy as a measure of disorder.

$$S \sim \ln(\Omega) \quad (2.1)$$

where Ω is the number of microstates associated with the macrostate, we learn that
this state of maximum entropy is a macrostate with the maximum number of associated
microstates.

The entropic argument of Equation 2.1 is not only mathematical founded. For a scramble
algorithm to hold for all possible data sets, it must also be capable of outputting all
possible permutations. As such, assuming all possible output are equally likely, the count
of each macroscopic output will be proportional to the number of microstates associated.

2.3.1 Measurements of the Algorithms

To compare the efficiency of the three algorithms in section 2.1, the algorithms were
run over the same input data and compared for the following measures:

Number of Transitions Per Frame

This measure counts the total number of bit transitions (i.e. $bit(n) \neq bit(n-1)$)
in a 120 bit frame. The header and parity information was not included as they are
not scrambled. This is an important test as one of the roles of the scrambler is to
maximise the number of transitions.

Common Bit Chain Length

One of the downsides of the 'Number of Transitions Per Frame' analysis is that the
two hypothetical 20 bit frames,

a) 10101010101111111111,

b) 10011001100110011001,

both with 10 transitions, are considered equal. However, (b) is clearly a more
suitable output for data transfer as (a) has a large probability of desynchronisation
due to the long chains of '1's in the right most bits. It is therefore also necessary
to evaluate the length of common bit chains within the scrambled data as shorter
chains are more suitable for data transfer.

Bit Asymmetry

Pre-scramble, the data had a large bias towards '0's due to the majority of the
bitmaps being empty. Scrambled data, via entropic arguments, *should* show zero
bias eitherway. Therefore, by investigating how the number of '1's - '0's evolves
over many frames, any bias in the scrambler can be found.

2.3.2 Statistical Predictions

216 Number of Transitions Per Frame

218 Consider a particle in a symmetric, discrete time-dependent, two state system,

$$p_0(t) = p_1(t) = 0.5 \quad : \quad \forall t \in \mathbb{N}, \quad (2.2)$$

At each time iteration,

$$p_{i \rightarrow j}(t) = 0.5 \quad : \quad i, j = [0 \ 1], \quad \forall t \in \mathbb{N}. \quad (2.3)$$

220 However, assuming zero bias and detailed balance, as $p_{1 \rightarrow 0}(t)$ is equal in both
222 probability and importance to $p_{0 \rightarrow 1}(t)$, the probability of a bit change shall herefore
be referred to as $p_\tau(t)$.

Over a n step process, analogous to a n bit frame, the probability distribution of
224 the number of transitions N_τ is given by Binomial statistics,

$$f(N_\tau) = \frac{n!}{N_\tau!(n - N_\tau)!} p^{N_\tau} (1 - p)^{n - N_\tau} \quad (2.4)$$

Simplified for the special case $p = p_\tau = 0.5$,

$$f_\tau(N_\tau) = \frac{n!}{N_\tau!(n - N_\tau)!} (p_\tau)^n \quad (2.5)$$

226 For $n = 120$, we can calculate,

$$\langle N_\tau \rangle^{Binomial} = \sum_{N_\tau=0}^{n-1} N_\tau f(N_\tau) = n p_\tau = 60 \quad (2.6)$$

$$\sigma_{N_\tau}^{Binomial} = \sqrt{n p_\tau^2} = 5.48 \quad (2.7)$$

228 Furthermore, when considering the entropic argument in section 2.3 equation 2.1,
the number of microstates corresponding to each macrostate N_τ can be related to
equation 2.5,

$$\Omega_\tau \sim \frac{n!}{N_\tau!(n - N_\tau)!} \quad (2.8)$$

$$\langle N_\tau \rangle^{Entropic} = MAX[S_\tau] = MAX[\Omega_\tau] \quad (2.9)$$

230 This can be numerically solved,

$$\langle N_\tau \rangle^{Entropic} = 60 \quad (2.10)$$

232 While the result of equation 2.10 does not contribute anything new, it is important
 as a ‘*sanity check*’. Because the system can be described as in section 2.3, it would
 indicated a problem in the theoretical framework if the result did not match.

234 Common Bit Chain Length

236 The probability of a chain of length n is,

$$p_n = p_1(1 - p_\tau)^{n-1}, \quad : \quad n \in \mathbb{N}, \quad n > 1 \quad (2.11)$$

238 where p_1 is the number of chains of length 1. As $p_1 = N_0(1 - p_\tau)$, where N_0 is the
 total number of chains,

$$\frac{N_n}{N_0} = (1 - p_\tau)^n, \quad : \quad n \in \mathbb{N}, \quad n > 1 \quad (2.12)$$

where N_n is the number of chains of length n . Taking the log of both sides,

$$\begin{aligned} \log\left(\frac{N_n}{N_0}\right) &= n \log(1 - p_\tau), \\ \log(N_n) &= n \log(1 - p_\tau) + \log(N_0). \end{aligned} \quad (2.13)$$

240 Therefor, for a graph of $\log(N_n)$ against n for a large sample of data, the gradient
 would be $\log(1 - p_\tau)$. In this case, as $p_\tau = 0.5$,

$$\log(1 - p_\tau) = -0.30. \quad (2.14)$$

242 Bit Asymetry

244 $A_{1,0}$, the assymetry of ‘1’s and ‘0’s is defined as,

$$A_{1,0} = N_1 - N_0, \quad (2.15)$$

246 where N_1 and N_0 are the number of ‘1’s and ‘0’s respectively. We can consider
 the evolution of $A_{1,0}$ with frame t of size n as a stockastic iterative map with zero
 deterministic growth [6],

$$A_{1,0}(nt + n \Delta t) = A_{1,0}(nt) + \mathcal{N}(nt) \quad (2.16)$$

248 Where \mathcal{N} is an independant random variable picked from a gaussian distribution.
 While $A_{1,0}(t) \in \mathbb{Z}$, in the limit of large nt we can approximate that $A_{1,0}$ is continious.

250 If we consider the moments of $A_{1,0}$,

$$\langle A_{1,0}(nt = M n \Delta t) \rangle = \sum_{m=0}^{M-1} \mathcal{N}(m n \Delta t), \quad (2.17)$$

$$\begin{aligned} \langle A_{1,0}(nt = M n \Delta t)^2 \rangle &= \sum_{m=0}^{M-1} \sum_{m'=0}^{M-1} \mathcal{N}(m n \Delta t) \mathcal{N}(m' n \Delta t) \delta_{mm'} \\ &= \sum_{m=0}^{M-1} \langle \mathcal{N}(m n \Delta t)^2 \rangle. \end{aligned} \quad (2.18)$$

252 Clearly, in Equation 2.17, $\langle A_{1,0} \rangle = 0$. In Equation 2.18, we assume the variance is of form $(n \Delta t)^\alpha$ [6]. Then,

$$\langle A_{1,0}(nt = M n \Delta t)^2 \rangle = M(n \Delta t)^\alpha. \quad (2.19)$$

254 Running the analysis over the frames $t = 0$ to t_f , the number of bits sampled is $M = t_f/n \Delta t$. Substituting this into Equation 2.19,

$$\langle A_{1,0}(nt = M n \Delta t)^2 \rangle = t_f (n \Delta t)^{\alpha-1}. \quad (2.20)$$

Considering the three cases of α in the approximation of continuous $n\Delta t$:

- 256 • $\alpha > 1$: Here $A_{1,0} \rightarrow 0$ as $\Delta t \rightarrow 0$.
- $\alpha < 1$: Here $A_{1,0} \rightarrow \infty$ as $\Delta t \rightarrow 0$.
- 258 • $\alpha = 1$: This is the only sensible choice.

With $\alpha = 1$,

$$\langle A_{1,0}(nt = M n \Delta t)^2 \rangle = M(n \Delta t). \quad (2.21)$$

260 And thus,

$$\sigma_{A_{1,0}} = \sqrt{\langle A_{1,0}^2 \rangle - \langle A_{1,0} \rangle^2} = \sqrt{\langle A_{1,0}^2 \rangle} = \sqrt{n \Delta t}. \quad (2.22)$$

2.3.3 Results of Analysis

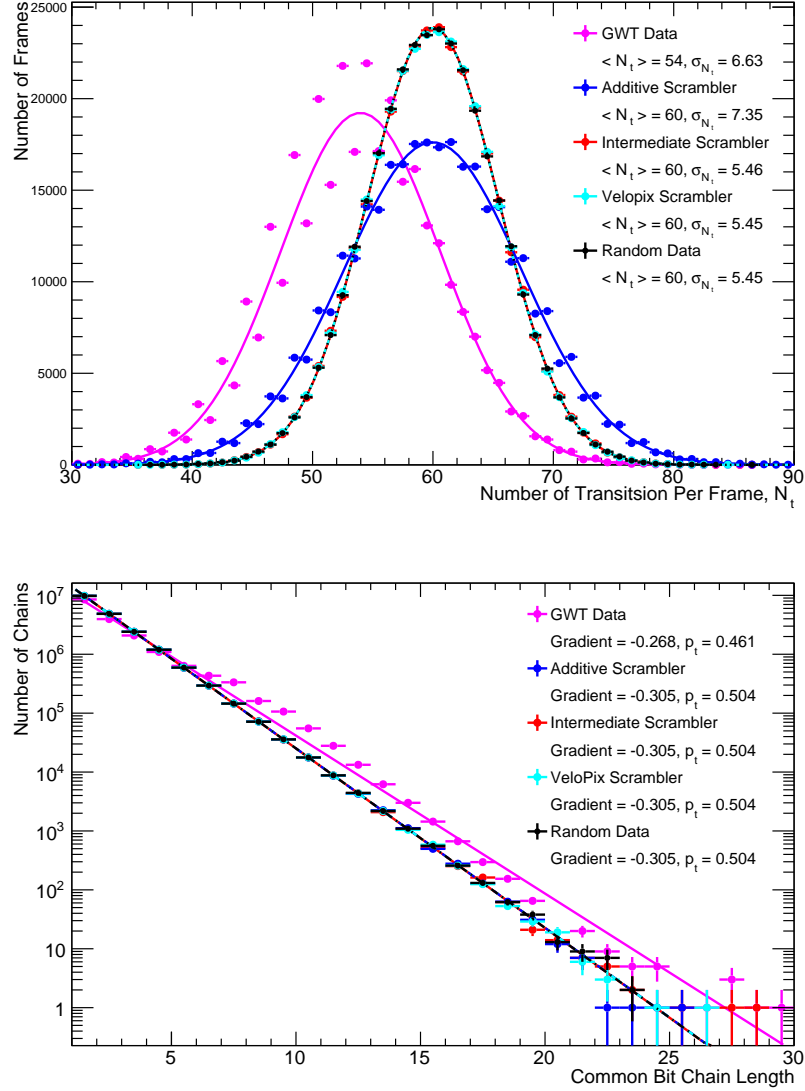


Figure 2.1: Results of the ‘*Number of Transitions Per Frame*’ analysis (Top) and the ‘*Common Bit Chain Length*’ analysis (Bottom). The results for the Random Data, Intermediate Scrambler and VeloPix Scrambler overlap for the ‘*Number of Transitions Per Frame*’ analysis. The results for the Random Data, Additive Scrambler, Intermediate Scrambler and VeloPix Scrambler approximately overlap for the ‘*Common Bit Chain Length*’ analysis.

The results from the ‘*Number of Transitions Per Frame*’ analysis, shown in Figure 2.1, show a strong correlation between the Intermediate and VeloPix Scramblers with the randomly generated data. These results are within 1% agreement with the theoretical predictions for $\langle N_\tau \rangle = 60$ and $\sigma_{N_\tau} = 5.48$, made in Section 2.3.2. The remarkable consistency between the theoretical predictions and the randomly generated data provides confidence in both the theory, and the scrambled nature of the Intermediate and VeloPix scrambler outputs.

All three scramblers, the random data, and the theoretical predictions are all consistent to within 1%. Comparing the two results for the Additive Scrambler, it is shown that while the frequency of longer chains is consistent with random data; but as the variance of transitions is larger than predicted, the long and short trains are more locally clustered.

The ‘*Bit Asymmetry*’ of each scrambler, shown in Figure 2.2, is consistent with the theoretical prediction. The deviation of $A_{1,0}$ for the predicted mean of 0 is fully consistent with stochastic noise. The random data also shows consistency. This gives confidence in the assumptions made in Section 2.3.2.

One notable feature of Figure 2.2 is the steep gradient of the additive scrambler at $t \sim 6 \cdot 10^6$. However, as the data stays within the theoretical limits and the ‘*drop*’ is of approximately $\Delta A_{1,0} \sim 60 \cdot 10^3$ over the range $n \Delta t \sim 1.2 \cdot 10^8$ it would be difficult to construct any argument claiming that this feature is of statistical significance.

(I am tempted to run χ^2 analysis for a fit of $y=0$ to show that the data is consistent with the model, but am not sure this will actually add to the argument?)

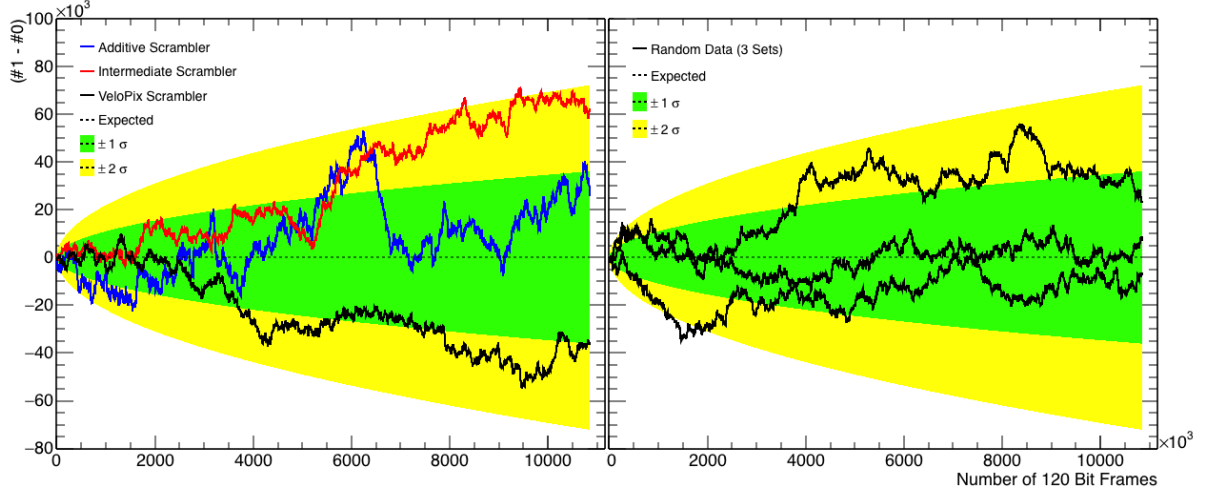


Figure 2.2: The results of the ‘*Bit Asymmetry*’ analysis.

2.4 Conclusion

	$\langle N_\tau \rangle$	σ_{N_τ}	Gradient	p_τ
GQT data	54	6.63	-0.268	0.460
Additive Scrambler	60	7.35	-0.305	0.504
Intermediate Scrambler	60	5.45	-0.305	0.504
VeloPix Scrambler	60	5.46	-0.305	0.504
Random Data	60	5.45	-0.305	0.504
Theoretical Prediction	60	5.48	-0.3	0.5

Table 2.1: The combined results of the algorithm analysis.

The consistency of random data and the theoretical predictions justifies the assumptions and approximations made in Section 2.3 and Section 2.3.2. Furthermore, the conformation of the statistical model allows for accurate comparisons to be made from predicted values and their measured counterparts.

The Additive Scrambler, while consistent with the ‘*Chain Length*’ and ‘*Bit Asymmetry*’ analysis, has a variance in the transition frequency that leads to the conclusion that long and short chains are locally clustered. This is not ideal for data transfer. Many sequential long chains increase the probability of TX-RX clock desynchronisation. Furthermore, the additive scrambler will not recover from this loss of synchronisation, as the ‘*key*’ will never be recovered without a common reset signal.

The Intermediate Scrambler produced an output consistent with random data. This

296 makes the algorithm suitable of data transfer. As already mentioned⁵, however, the
scrambler is designed for computer simulated. As such, it is not suitable for implemen-
298 tation as it does not meet the additions requirments of the ASIC.

The VeloPix Scrambler, like the Intermediate Scrambler, produces a statistically scram-
300 bled output. Furthermore, the algorithm is inline with the additional requirments of the
ASIC. As such, it ideal for implementation, and hense is currently the choice algorithm
302 for use in the 2019 VELO upgrade.

⁵Note to Marco: this is in the scrabler options section

References

- 304 [1] Cern. *The Standard Model*. 2015. URL: <http://home.cern/about/physics/standard-model> (visited on 12/2015).
- 306 [2] LHCb Collaboration. *LHCb VELO Upgrade Technical Design Report*. Tech. rep. CERN-LHCC-2013-021. LHCb-TDR-013. Geneva: CERN, Nov. 2013. URL: <https://cds.cern.ch/record/1624070>.
308
- [3] CERN. *LHCb VELO Project*. 2015. URL: <http://lhcb-vd.web.cern.ch/lhcb-vd/html/project.htm>.
310
- [4] Altera. *Quartus Prime Software*. 2015. URL: <https://www.altera.com/products/design-software/fpga-design/quartus-prime/overview.html> (visited on 12/2015).
312
- 314 [5] Mentor Graphics. *ModelSim - Leading Simulation and Debugging*. 2015. URL: <https://www.mentor.com/products/fpga/model/> (visited on 12/2015).
- 316 [6] Kurt Jacobs. *Stochastic Processes for Physicists - Understanding Noisy Systems*. Cambridge University Press, 2010. ISBN: 9780521765428.