

MIT Open Access Articles

Performance of the CMS Event Builder

The MIT Faculty has made this article openly available. **Please share** how this access benefits you. Your story matters.

Citation: Andre, J-M et al. "Performance of the CMS Event Builder." Journal of Physics: Conference Series 898 (2017): 032020 © IOP Publishing

As Published: <http://dx.doi.org/10.1088/1742-6596/898/3/032020>

Publisher: IOP Publishing

Persistent URL: <https://hdl.handle.net/1721.1/121947>

Version: Final published version: final published article, as it appeared in a journal, conference proceedings, or other formally published context

Terms of use: Creative Commons Attribution 3.0 unported license



PAPER • OPEN ACCESS

Performance of the CMS Event Builder

To cite this article: J-M Andre *et al* 2017 *J. Phys.: Conf. Ser.* **898** 032020

View the [article online](#) for updates and enhancements.

Related content

- [Upgrade of the CMS Event Builder](#)
G Bauer, U Behrens, M Bowen et al.
- [The CMS event builder and storage system](#)
G Bauer, B Beccati, U Behrens et al.
- [The NOvA DAQ Monitor System](#)
Michael Baird, Deepika Grover, Susan Kasahara et al.



IOP | ebooks™

Bringing you innovative digital publishing with leading voices to create your essential collection of books in STEM research.

Start exploring the collection - download the first chapter of every title for free.

Performance of the CMS Event Builder

J-M Andre⁵, U Behrens¹, J Branson⁴, P Brummer², O Chaze²,
S Cittolin⁴, C Contescu⁵, B G Craigs², G-L Darlea⁶, C Deldicque²,
Z Demiragli⁶, M Dobson², N Doualot⁵, S Erhan³, J F Fulcher²,
D Gigi², M Gładki², F Glege², G Gomez-Ceballos⁶, J Hegeman²,
A Holzner⁴, M Janulis^{2a}, R Jimenez-Estupiñán², L Masetti²,
F Meijers², E Meschi², R K Mommsen⁵, S Morovic², V O'Dell⁵,
L Orsini², C Paus⁶, P Petrova², M Pieri⁴, A Racz², T Reis²,
H Sakulin², C Schwick², D Simelevicius^{2a} and P Zejdl^{5b}

¹ DESY, Hamburg, Germany

² CERN, Geneva, Switzerland

³ University of California, Los Angeles, California, USA

⁴ University of California, San Diego, California, USA

⁵ FNAL, Chicago, Illinois, USA

⁶ Massachusetts Institute of Technology, Cambridge, Massachusetts, USA

^a also at Vilnius University, Vilnius, Lithuania

^b also at CERN, Geneva, Switzerland

E-mail: remigius.mommsen@cern.ch

Abstract. The data acquisition system (DAQ) of the CMS experiment at the CERN Large Hadron Collider assembles events at a rate of 100 kHz, transporting event data at an aggregate throughput of $\mathcal{O}(100\text{ GB/s})$ to the high-level trigger farm. The DAQ architecture is based on state-of-the-art network technologies for the event building. For the data concentration, 10/40 Gbit/s Ethernet technologies are used together with a reduced TCP/IP protocol implemented in FPGA for a reliable transport between custom electronics and commercial computing hardware. A 56 Gbit/s Infiniband FDR Clos network has been chosen for the event builder. This paper presents the implementation and performance of the event-building system.

1. Introduction

The Compact Muon Solenoid (CMS) experiment at CERN is one of the two general purpose experiments located at the LHC. CMS is designed to study both proton-proton and heavy ion collisions at the TeV scale [1]. The detector comprises about 55 million readout channels. The online event-selection is performed using two trigger levels: a hardware-based first-level trigger accepting up to 100 kHz of events and a software-based high-level trigger (HLT) selecting $\mathcal{O}(1\%)$ of these events.

The CMS data-acquisition system from run 1 (DAQ 1) [2] was upgraded during the first long-shutdown of the LHC (2013/14). The main motivation for the upgrade is the aging of the existing hardware (both PCs and network equipment are more than 5 years old), and the need to accommodate sub-detectors with upgraded off-detector electronics that exceed the original specification of the DAQ system. The new DAQ architecture (DAQ 2) [3, 4, 5] uses state-of-the-art network technologies for the event building.



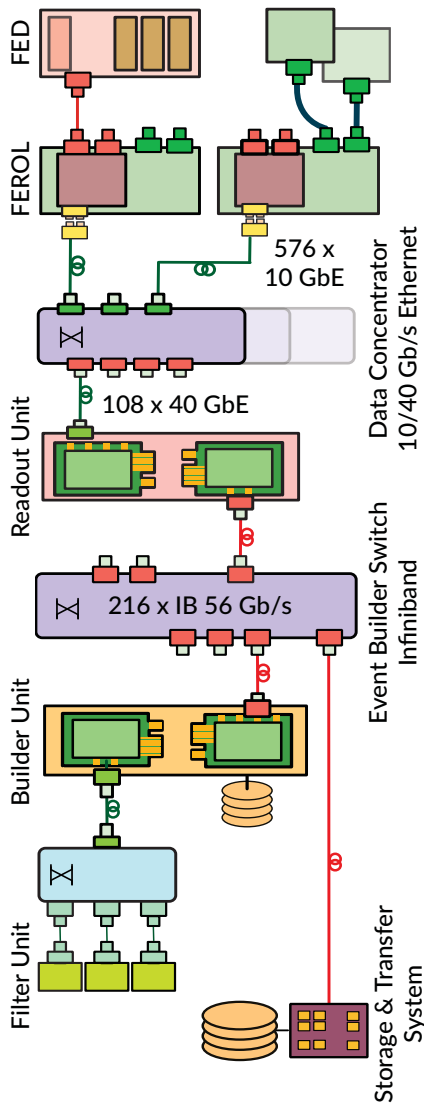


Figure 1: Schematic of the DAQ2 system (see text for explanation.)

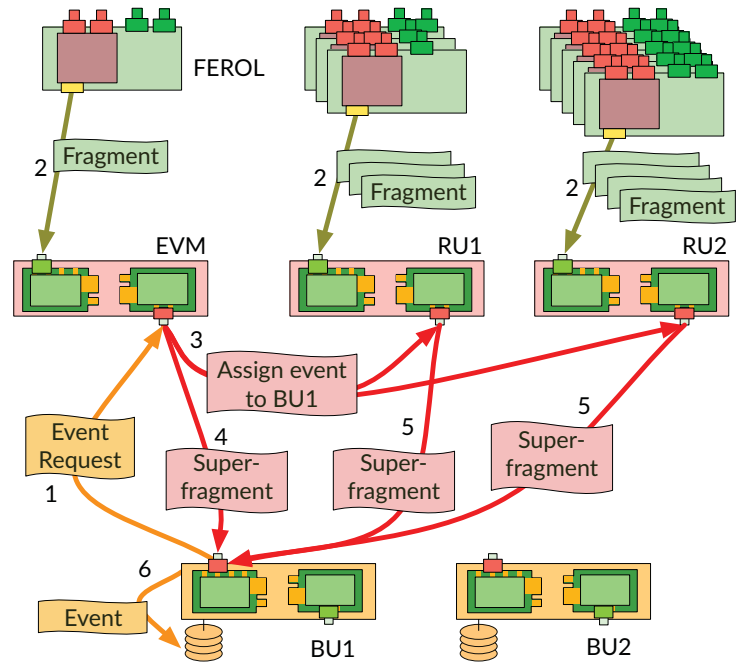


Figure 2: Simplified event-building protocol for run 2. A number of Front-End Readout Optical Link (FEROL) boards is assigned to a given readout unit (RU). The event manager (EVM) gets only the trigger-control data from one FEROL. A number of builder units (BUs) are responsible for the event building. See text for a detailed explanation.

2. The CMS Event Builder

Figure 1 shows a schematic of the DAQ 2 system. The DAQ collects data from about 740 custom detector Front-End Drivers (FEDs). The FEDs deliver for each trigger an event fragment of 0.1-8 kB, depending on the subsystem and the instantaneous luminosity, yielding an event size of up to 2 MB. The event building is done in two steps: First, the data concentrator aggregates data from 1-18 FEDs into a super-fragment, and second, the event builder assembles the super-fragments into complete events.

FEDs are connected over point-to-point links to custom front-end readout optical link (FEROL) boards [6]. One or two legacy FEDs are connected via copper cables (200/400 MB/s), while the new μ TCA-based FEDs uses 10 Gbit/s optical fibers. In both cases a custom protocol is used. The FEROL translates this protocol into an in-FPGA, one-directional 10 Gbit/s TCP/IP connection [7] which is routed to commercial network equipment. The FEROL sends the data from each FED as a TCP/IP stream to a pre-defined readout unit (RU) computer. A series of

switches is used to concentrate the data from 1-18 streams into 40 GbE and to transport the data to the surface. The RU splits the streams into event fragments, checks the consistency and assembles the data belonging to the same event into a super-fragment. The super-fragment is sent over the event-builder switch to the builder unit (BU) machines. The event-builder switch uses Infiniband [8] FDR at 56 Gbit/s. It employs a Clos-network structure with 12 leaf and 6 spine switches, providing 216 external ports and 12 Tbit/s bi-sectional bandwidth. The BU assembles the super-fragments into complete events and makes them available to the file-base filter farm [9].

3. Event-Building Protocol and Implementation

Figure 2 shows a simplified schematic of the event-building protocol. (1) Any builder unit (BU) which has resources to build events sends a request for a predefined number of events to the event manager (EVM), which orchestrates the whole event builder. (2) The EVM receives asynchronously the event fragment from the FEROL connected to the trigger-control readout. (3) Once the EVM received enough event fragments from the trigger control to satisfy the number of requested events, or if a timeout is reached, the EVM sends a message to all readout units (RUs) participating in the event building. This message contains the level-1 trigger numbers of the events together with the identifier of the BU which requested the events and the identifiers of all RUs which contribute super-fragments to the event. In the current running scheme, all RUs provide a super-fragment for all events. (4) At the same time, the EVM packs the event fragments corresponding to the requested events into an Intelligent Input/Output (I2O)[10] message and sends it to the BU. (5) When the RU receives the event assignment from the EVM, it combines all FEROL fragments into a super-fragment. Each RU knows which FEROLs participate in the readout and waits until it has received all fragments. The super-fragments for all events assigned to the given BU are packed into one or several I2O messages with a maximum size of 131,072 Bytes. They are then sent to the BU. (6) Once the BU has received the super-fragments from all RUs, it builds the event. Any super-fragment message contains the identifiers of all RUs which participate in the event building. Therefore, the BU knows on an event-by-event basis if it got the super-fragments from all RUs. The BU checks the consistency of the event by verifying the correct structure of the FED data, the trigger number embedded in each FED fragment, and the CRC of the FED payload. Finally, the event is written to a file residing on the RAM disk.

If the EVM or any of the RUs fail, the event building stalls. In this case, the run needs to be stopped, and if the machine cannot be recovered, a new DAQ configuration has to be made using a spare machine. The latter operation needs manual intervention and takes a few minutes. However, no downtime was created due to these single points of failure during run 2. BUs can leave or join an ongoing run at any time.

The event-building applications are built upon the XDAQ framework [11]. XDAQ is a middleware that eases the development of distributed data acquisition systems. The framework has been developed at CERN and builds upon industrial standards, open protocols and libraries. It provides services for data transport, configuration, monitoring, and error reporting. In particular, it also provides the zero-copy architecture for Infiniband [12].

The event-building system has been optimized in order to exploit the full capability of modern computing and network hardware. The applications use multiple threads to assemble and handle events in parallel. In addition, the number of memory copies is minimized and the rate of control messages needs to be kept at bay. To achieve this, several logically independent messages for the same destination are grouped into a single message frame before it is handed to the transport layer. In order to cope with the non-uniform memory architecture (NUMA), each thread and memory structure is bound to specific CPU core. Moreover, the interrupts from the network interface cards are restricted to certain cores that are not used for any data handling. Finally,

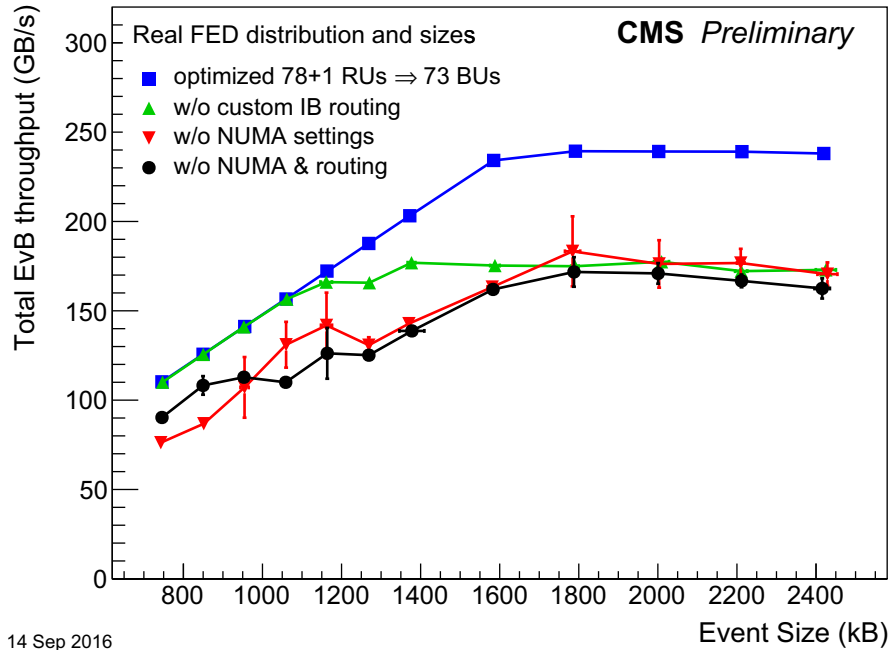


Figure 3: Illustration of performance optimizations using a production-like event-building setup. The blue curve shows the throughput of the fully optimized system as function of the event size. The green curve uses the standard Infiniband routing instead of the custom scheme taking into account the event-building traffic pattern. The red curve depicts the performance when not binding each thread and memory structure to specific CPU cores. Note that the system is not only performing worse for smaller events, but is also much less stable as indicated by the error bars. The black curve shows the results if both optimizations are switched off.

we worked out a custom routing scheme for the Infiniband to account for the uni-directional event-building traffic. Figure 3 shows the impact of these settings on the performance when emulating a production-like event-building setup (see section 4.4.)

4. Measurements

The performance of the new event-building system is evaluated on a small-scale test bed and on the full-scale production system. The hardware on both systems is identical. The readout unit (RU) is a Dell PowerEdge R620 machine with dual 8 core Xeon CPU (E5-2670 0) running at 2.6 GHz and with 32 GB of memory. The builder unit (BU) is a Dell PowerEdge R720 machine with the same CPU type. The BU uses an asymmetric memory configuration with 32 GB attached to CPU 0 and 256 GB attached to CPU 1. An amount of 220 GB on CPU 1 is used for the RAM disk. The network interface cards for 40 GbE and 56 Gbit/s Infiniband are Mellanox Technologies MT27500 Family [ConnectX-3]. We use Mellanox SX1024 & SX1036 for 10 and 40 GbE switches and Mellanox SX6036 for the Infiniband Clos network.

The measurements use the production XDAQ software and settings. They are carried out with stand-alone scripts to setup the system, issue run-control commands, and perform measurements. Each FEROL generates dummy event fragments. Fragments are pushed whenever the downstream system is able to digest them, i.e. the sources are not synchronized by a trigger signal. The system is stopped and restarted for each fragment size, and is then kept running for 60 seconds before the first measurement is taken. Each measurement is the average over one second. Each data point is averaged over 20 measurements taken 10 seconds apart.

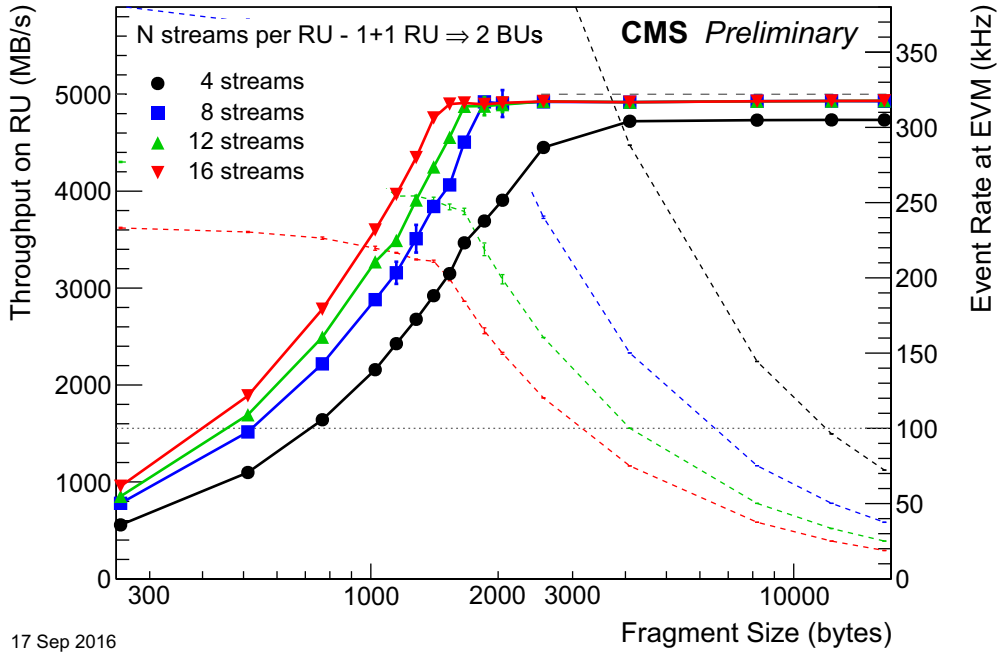


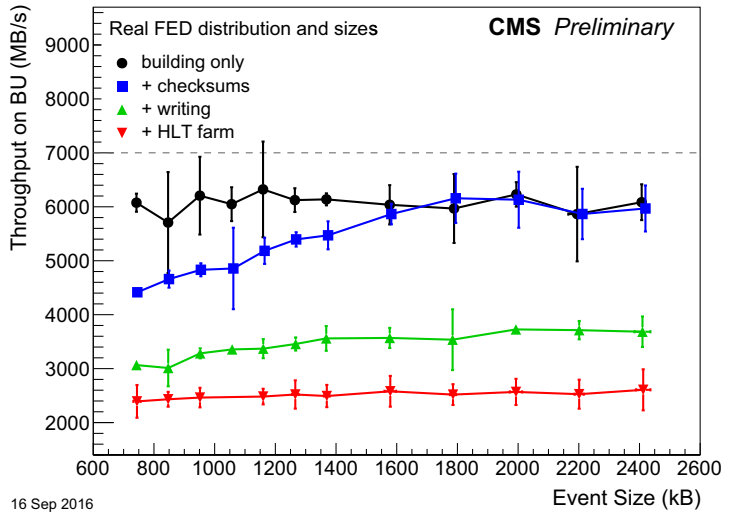
Figure 4: Performance measurement of the data concentrator: the thick lines show the throughput on the RU as function of the fragment size for different numbers of FEROLs connected to the RU. For fragment sizes of $\gtrsim 1.5$ kB, the 40 GbE line speed is reached (horizontal dashed line.) The dashed lines show the equivalent event rate in kHz (right axis.) The point where they go below the requirement of 100 kHz (dotted line) indicates the maximum fragment size which can be handled by the given setup.

4.1. Data Concentrator

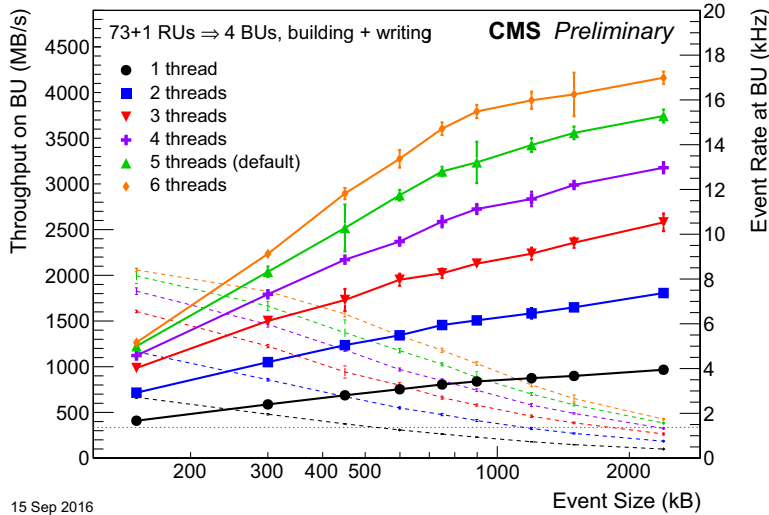
To measure the performance of the data concentrator, we use one EVM and one RU sending data to two BUs. Between 4 and 16 FEROLs are assigned to the RU. The EVM gets a fixed-size fragment of 1 kB. The BUs only assemble events, but do not write them to files. Figure 4 shows the throughput on the RU as function of fragment size. The throughput increases with the number of TCP streams as more threads can work independently. In most cases it reaches the 40 GbE line speed. The 4-streams case is limited by the message rate of the individual TCP streams.

4.2. Performance of the Builder Unit (BU)

Figure 5 shows the throughput on the BU as function of the event size. In figure 5a, 78 readout units are sending super-fragments with sizes similar to the one found in proton-physics running, while in figure 5b all super-fragments originating from 73 readout units have the same size. This explains the small difference between the green curves in the figure which corresponds otherwise to the same settings. Figure 5a shows the throughput as function of the increasingly demanding tasks: building only (black) is mostly unaffected from the event size, while calculating the checksums (blue) is more efficient for larger fragments. In both cases, the throughput reaches nearly the 56 Gbit/s Infiniband line speed (horizontal dashed line.) Writing the data to the RAM disk (green) is limited to ~ 3 GB/s, and running an emulated HLT processing (red) reduces the throughput to ~ 2.5 GB/s. This limit is given by the throughput with which the HLT process can access the data from the RAM disk over NFS [9]. Figure 5b depicts the dependency on the number of threads used for building and writing events. A near linear scaling is achieved up to



(a) Performance of increasingly demanding event-building tasks



(b) Dependency on the number of threads used for building and writing events

Figure 5: Throughput of the builder unit (BU) as function of the event size. Note that figure (b) spans a much larger range of event sizes. It also shows the event rate as dashed lines (right scale.) Each BU has to handle ~ 1.8 kHz of events.

5 threads. The required event rate of ~ 1.8 kHz is met for all event sizes when using at least 4 threads.

4.3. Scaling behaviour

The scaling behavior of the system is evaluated by changing the system size from 4 RU sending data to 4 BU to 101 RUs sending data to 73 BUs. Each RU has 8 FEROLs attached, each sending a same-sized fragment (see figure 6.) The 4×4 system is capable of running close to the line speed of the 40 GbE data-concentrator network. Larger square systems (22×22 , 48×48 , and 73×73) are limited to $\sim 60\%$ of the 56 Gbit/s line-speed of the IB network. Adding more builder units (sinks) improves the total throughput, while adding 30 more readout units (sources) does

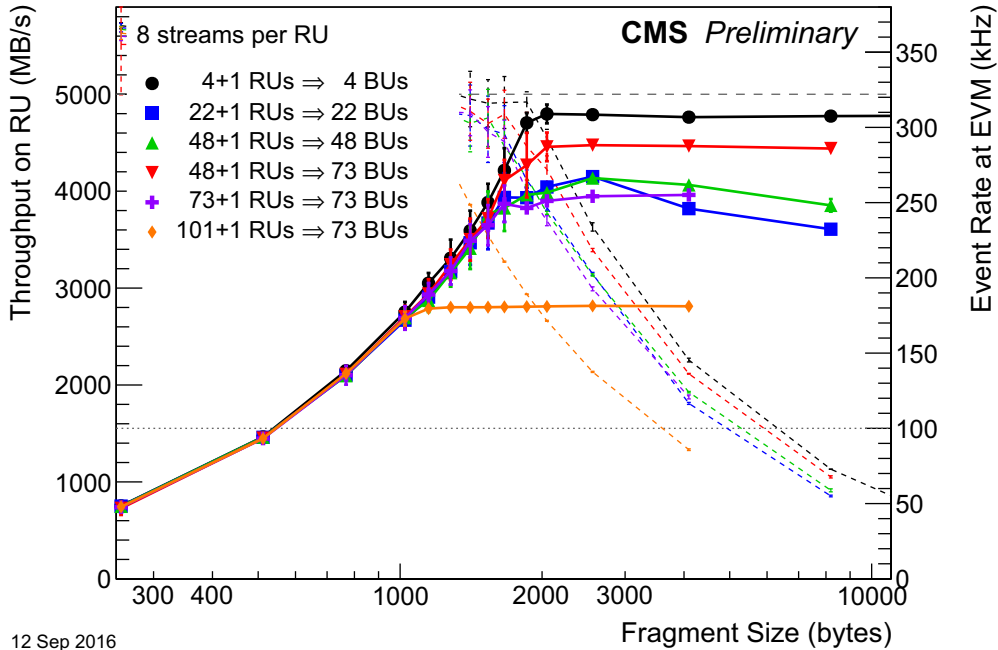


Figure 6: The scaling behavior of the system: the thick lines show the average throughput on the RUs as function of the fragment size for different sizes of the system. The dashed lines show the equivalent event rate (right axis). The requirement of 100 kHz is shown as dotted line.

not change the utilization of the links to the BUs. We believe that the limitation to the BUs comes from head-of-line blocking [13] due to the non-uniform event-builder traffic. In all cases the requirement of 100 kHz is met for fragment sizes of $\lesssim 4$ kB.

4.4. Emulating the production system

In the real system, the distribution of super-fragment sizes is far from uniform because the data-concentration is done by sub-detector partitions. In addition, different detectors and regions show a different dependency of the fragment size on pileup. In order to emulate the behavior of the event builder as realistically as possible, we measured the fragment size as function of pileup for each FEROL stream during physics-data taking. In addition, we measured the relative root-mean-square (RMS) of each stream. These parameters are then used to emulate the fragment sizes for each FEROL stream for different pileup scenarios. Figure 7 shows the total event-builder throughput as function of the event size for 2 different ways to concentrate the same number of FEROL streams. The smaller setting using 63 readout units (RUs) corresponds to the production setting in 2016. The total throughput is limited by a few RUs which handle ~ 4.2 GB/s. Spreading the FEROLs belonging to the highest throughput RUs over 15 additional RUs increases the total throughput to ~ 240 GB/s, which corresponds to an event size of ~ 2.4 MB at 100 kHz trigger rate.

5. Summary

CMS has built and commissioned a completely new DAQ system for LHC run 2. The new DAQ 2 can handle larger events originating from legacy and new detector front-end electronics. It uses state-of-the-art network technology, which allows to shrink the size of the system by an order of magnitude compared to the previous DAQ. However, the use of high-end hardware requires

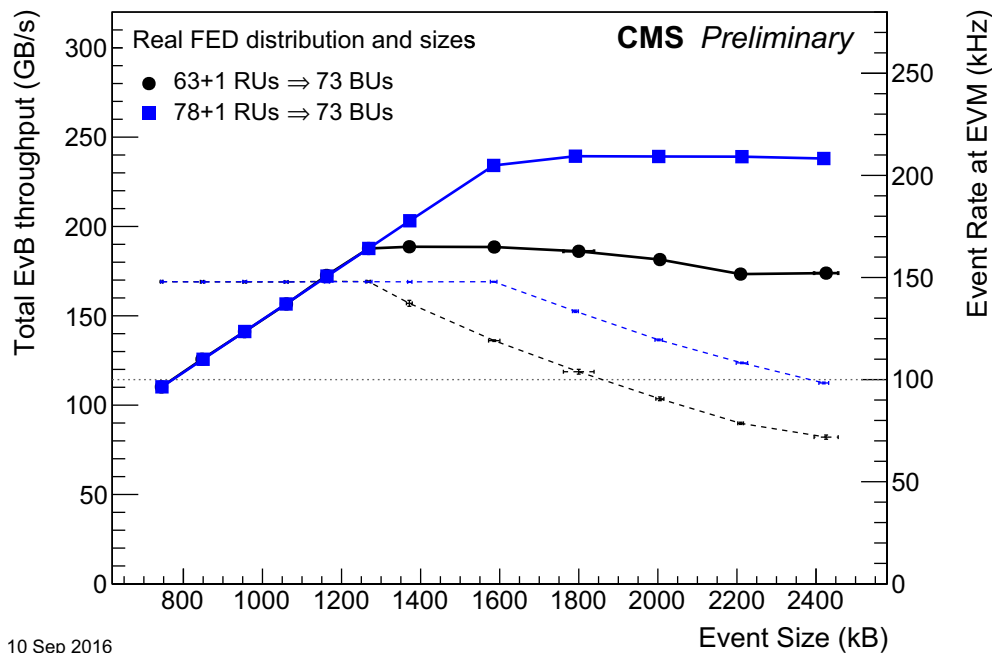


Figure 7: Total throughput of the event builder using FED fragments emulating the distributions as measured in physics-data taking. Two different configurations to concentrate the same number of FEROL streams are shown. The setting with 63 readout units (black) corresponds to the production setting in 2016. Adding 16 more readout units (blue) eliminates the worst bottle necks.

a more efficient event-builder protocol and new software to exploit the hardware capabilities. In addition, a lot of fine tuning is needed to achieve the best performance, which exceeds the requirements of building 2 MB events at 100 kHz trigger rate. The limiting factor is attributed to head-of-line blocking on the Infiniband network due to the non-uniform event-building traffic going to the builder units. Therefore, additional builder units could be installed whenever an increased throughput would become necessary.

Acknowledgments

This work was supported in part by the DOE and NSF (USA).

References

- [1] CMS Collaboration 2008 *JINST* **3** S08004
- [2] Mommsen R K *et al.* 2011 *J. Phys. Conf. Ser.* **331** 022021
- [3] Petrucci A *et al.* 2012 *J. Phys. Conf. Ser.* **396** 012039
- [4] Holzner A *et al.* 2014 *J. Phys. Conf. Ser.* **513** 012014
- [5] Sakulin H *et al.* 2015 *IEEE Trans. Nucl. Sci.* **62** 1099–1103 [7097437(2014)]
- [6] Zejdl P *et al.* 2013 *JINST* **8** C12039
- [7] Zejdl P *et al.* 2014 *J. Phys. Conf. Ser.* **513** 012042
- [8] InfiniBand Trade Association, see <http://www.infinibandta.com>
- [9] Meschi E *et al.* 2015 *J. Phys. Conf. Ser.* **664** 082033
- [10] I2O Special Interest Group, 1999 Intelligent I/O (I2O) Architecture Specification v2.0
- [11] Gutleber J *et al.* 2010 *J. Phys. Conf. Ser.* **219** 022011
- [12] Petrucci A *et al.* 2013 *IEEE Trans. Nucl. Sci.* **60** 4595–4602
- [13] Karol M, Hluchy M and Morgan S 1987 *IEEE Transactions on Communications* **35** 1347–1356 ISSN 0090-6778