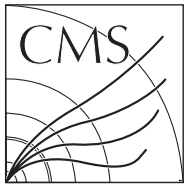
Available on CMS information server

CMS CR -2009/123



The Compact Muon Solenoid Experiment

Conference Report

Mailing address: CMS CERN, CH-1211 GENEVA 23, Switzerland

14 May 2009

The CMS L1 Trigger Emulation Software

V. M. Ghete (on behalf of CMS Collaboration)

Abstract

The CMS L1 Trigger processes the muon and calorimeter detector data using a complex system of custom hardware processors. A bit-level emulation of the trigger data processing has been developed. This is used to validate and monitor the trigger hardware, to simulate the trigger response in monte-carlo data, and for some components, to seed higher-level triggers. The multiple-use cases are managed using a modular design, implemented within the modular CMS offline software framework. The requirements, design and performance of the emulators are described, as well as the iterative process required to bring the emulators and hardware into agreement.

Presented at *CHEP09: International Conference On Computing In High Energy Physics And Nuclear Physics,21-27 Mar 2009,Prague,Czech Republic,15/05/2009*

The CMS L1 Trigger emulation software

V M Ghete
(on behalf of CMS Collaboration)
Institute for High Energy Physics, Austrian Academy of Sciences, Vienna, Austria
E-mail: Vasile.Mihai.Ghete@cern.ch

Abstract. The CMS L1 Trigger processes the muon and calorimeter detector data using a complex system of custom hardware processors. A bit-level emulation of the trigger data processing has been developed. This is used to validate and monitor the trigger hardware, to simulate the trigger response in monte-carlo data, and for some components, to seed higher-level triggers. The multiple-use cases are managed using a modular design, implemented within the modular CMS offline software framework. The requirements, design and performance of the emulators are described, as well as the iterative process required to bring the emulators and hardware into agreement.

1. The CMS Experiment

The Compact Muon Solenoid (CMS) experiment is a general-purpose experiment at the Linear Hadron Collider (LHC) at CERN, Geneva.

It contains a tracker system, composed of a pixel detector and a silicon strip detector, and a calorimeter system, both situated inside a solenoid providing a 3.8 T magnetic field. The superconducting solenoid, together with the Drift Tubes (DT) system and the Resistive Plate Chambers (RPC) in the barrel part, the Cathode Resistive Plate (CSC) chambers and the RPCs in both endcaps forms a high-performance muon spectrometer. The calorimeter system consists of a high resolution lead-tungstate crystal electromagnetic calorimeter (ECAL), covering a region with pseudorapidity $|\eta| < 3$, an endcap preshower and a hadron calorimeter (HCAL) covering a region with pseudorapidity $|\eta| < 5$.

A detailed description of the detector and its performance can be found in Ref. [1].

2. The CMS Trigger System

The main task of the CMS trigger system is to reduce the event rate from 40 MHz to $\mathcal{O}(100)$ Hz, covering the widest possible range of discovery physics. The trigger system is part of the Trigger and Data Acquisition System (TriDAS), shown in Fig. 1. The Data Acquisition system (DAQ) is modular; it can be deployed in up to eight parallel slices, each slice corresponding to 12.5 kHz. Some of the slices can be staged, depending on the financial situation of the experiment and on the needs resulted from the accelerator parameters.

The trigger system is organized in two levels: a custom hardware trigger, the Level-1 Trigger (L1), and a software trigger, the High Level Trigger (HLT).

The L1 trigger reduces the event rate from 40 MHz to 100 kHz, when the TriDAS system is complete, or to 50 kHz, when the DAQ system is 50% staged. Total processing time is of the

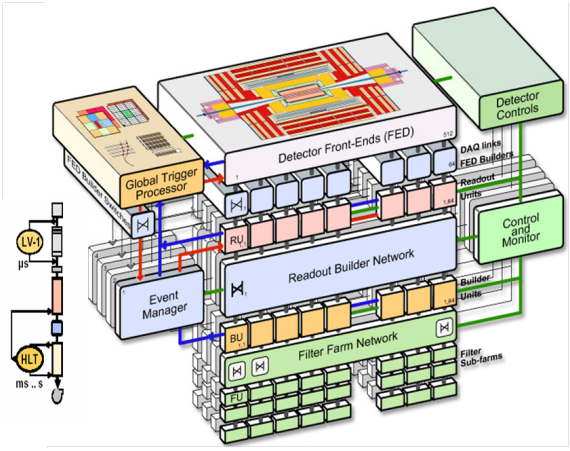


Figure 1. Architecture of CMS Trigger and Data Acquisition System.

order of 3 μs. The selection of the events is based on coarse information from the calorimeter and the muon detector, whose response is fast enough to allow the required latency time.

The High Level Trigger is developed in the CMS offline software framework (CMSSW). The allowed output event rate is $\mathcal{O}(100)$ Hz, within an average processing time of the order of 40 ms per event. The HLT has access to all event data, with full precision and granularity, in contrast to the L1 trigger which access only reduced information. Each HLT path is seeded by a logical combination of L1 algorithms (conditional reconstruction) and the event is reconstructed only around the L1 seed candidates (partial/regional reconstruction).

2.1. The Level-1 Trigger System

The L1 trigger system is shown in Fig. 2. It is a custom-design, largely programmable electronics, based mainly on FPGA technology to allow maximum flexibility, and on ASICs and programmable memory lookup tables. The configuration and operation of the L1 trigger is controlled by a software system, the Trigger Supervisor.

There are three main components of L1 trigger: the muon trigger, the calorimeter trigger and the global trigger. The muon trigger and the calorimeter trigger have each local, regional and global components.

The local components, called Trigger Primitive Generators (TPG) are integrated with the detector readout; they provide trigger tower energy sums in ECAL, central HCAL and forward HCAL (HF) for the calorimeter trigger, and track segments and hit patterns in muon chambers for the muon trigger.

The calorimeter TPG information is sent to the Regional Calorimeter Trigger (RCT), which finds isolated and non-isolated candidate electromagnetic (e-γ) objects, transverse energy sums per calorimeter region and tau-veto bits, and sends them to the Global Calorimeter Trigger (GCT). Central, forward and tau jets (identified using the tau-veto bits) are found in GCT. Jets and e-γ objects are then sorted based on their transverse energies and the highest-rank four objects from each category are sent to the Global Trigger (GT). Transverse energy sums, HF bit counts and energy sums in HF are also computed and sent to GT.

The Local DT and CSC triggers provide local trigger information, which is then assembled by the regional muon triggers DT Track Finder (DTTF) and CSC Track Finder (CSCTF) in muon tracks with physical parameters assigned. The RPC trigger chambers deliver their own tracks, based on regional hit patterns. Each of the three systems sends muon candidates to the Global Muon Trigger(GMT) - four barrel and four forward RPC muon candidates, four DTTF

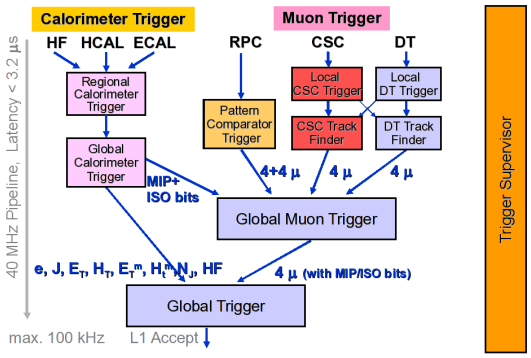


Figure 2. CMS Level-1 Trigger System.

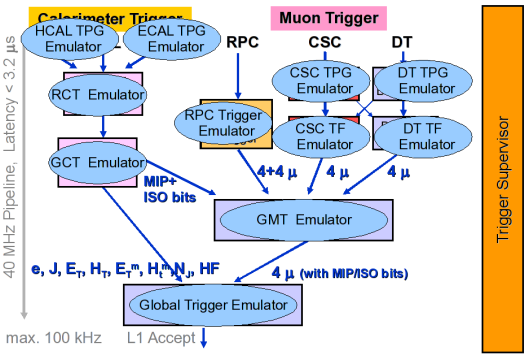


Figure 3. CMS Level-1 Trigger Emulators.

and four CSCTF muon candidates - which combines them to improve the momentum resolution and efficiency. The four highest-rank muon are sent to GT, with rank depending on momentum and muon quality.

Using the objects sent by GMT and GCT, the GT evaluates up to 128 sets of physical requirements ("L1 algorithms") for each LHC bunch crossing. Each algorithm is a logical expression of templated object physics requirements, so-called GT "conditions", with AND, OR and NOT logical operators. Combining the decisions of the L1 algorithms with the decisions of up to 64 technical triggers (signals sent directly to GT), the GT computes a FinalOR signal which, if true, accepts the event and initiates the readout procedure of the event in the whole CMS detector (L1 accept, L1A). Prescale factors can be applied for each algorithm and technical trigger; trigger masks (removing an algorithm or a technical trigger from the computation of FinalOR), and veto trigger masks (FinalOR false if a given bit is true) can also be applied before the FinalOR. Trigger rules, limiting the number of L1A in various ranges of time (number of bunch crossings) are also implemented.

3. The Level-1 Trigger Emulation

The L1 trigger emulation task is to emulate bit-by-bit each L1 trigger subsystem. The following requirements were devised for the development of the L1 trigger emulation:

- It must be integrated in CMS modular offline software framework CMSSW.
- The design of the emulator must be modular, following the modularity of the hardware subsystems; every module must use the same input in the same format as the hardware modules and must produce the same output as the hardware, with identical format. One can in this way run on the data output of the hardware modules or one can run on the output of an emulator module.
- If possible and required, the emulator modules should produce additional information, not computed for various reasons by the hardware, or intermediate information, used in the hardware but not saved in the readout record.
- Stringent timing requirements are imposed for all modules, especially those modules running online, in the Event Filter farm, as part of the data taking or data monitoring for each event.

Following the requirements outlined above, for each L1 trigger hardware subsystem an emulator was developed. The mapping of the emulators on the hardware subsystems and their inter-connections are shown in Fig. 3.

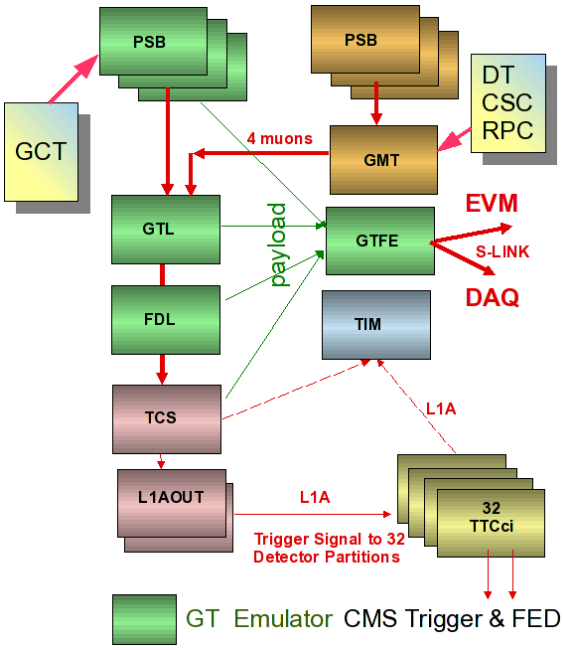


Figure 4. L1 Global Trigger Hardware System.

The configuration of the emulator modules must be consistent with the configuration of the hardware modules at the level of individual parameters.

The L1 hardware modules are configured from the Online Master Data Storage (OMDS) database. By design, OMDS is not accessible from the offline environment, where the L1 trigger emulator is running, therefore a procedure to transfer in real time those parameters for hardware configuration used also in the emulator was devised. The "Online to Offline" (O2O) application runs in the Run Control as last step of the trigger configuration and transfers the parameters from OMDS to ORCON (Offline Reconstruction Online subset database) / ORCOF (Offline Reconstruction Offline subset database). In the same process, the Interval of Validity (IoV) for the transferred records is set such that they are associated with the data via the "Event Setup" mechanism provided by the CMSSW framework. Some more details about configuration implementation are presented in the subsection 3.1.

For some modules, it must be possible to overwrite for emulators some hardware parameters.

3.1. Example: Emulation of Level-1 Global Trigger

A typical example for the emulation of a L1 trigger subsystem is the emulation of the L1 Global Trigger.

Fig. 4 shows the hardware design of the GT. It is based on FPGA technology and consists of several VME boards mounted in a VME9U crate, together with the GMT and the central Trigger Control System (TCS). The GCT objects are received via Pipeline Synchronize Buffer (PSB) cards. The PSB cards transmit the data, after synchronization, to the logic board (Global Trigger Logic board, GTL) via the backplane. The GMT objects are sent directly to GTL via the backplane. The GTL board receives the objects via three programmable receiver chips and send them to two "condition" chips, each evaluating the results of up to 64 algorithms. The results are sent to the Final Decision Board (FDL) where, combining the technical triggers (received directly via a dedicated PSB) with the algorithm results and applying prescale factors and trigger masks, the FinalOR is computed. The Global Trigger Front End (GTFE) board

Figure 5. L1 Global Trigger Emulator - UML-like scheme.

receives the event inputs from PSBs, GMT and FDL and combines them in a readout record sent via a SLink64 to the DAQ system; another record, built from FDL, TCS inputs, information from the LHC Beam Synchronous Timing system is sent to the CMS Event Manager System (EVM).

The UML-like scheme of the GT emulator is shown in Fig. 5. The data formats, common with the formats defined for unpacking the hardware raw data, are defined in the block "DataFormats/GlobalTrigger". The classes for GT configuration are defined in the block "CondFormats/L1TObjects" while the corresponding database records are defined in the block "CondFormats/DataRecord". These records are filled via an O2O application, reading the values of the parameters from OMDS; each record has assigned a "cache identifier" integer, changed whenever the content of a record is changed. The configuration parameters for emulator modules are read from the records via the "event setup" mechanism provided by the CMSSW framework for each event; in order to reduce the time of reading the configuration, the parameters are cached and re-read only if the cache identifier changes. The GCT objects obtained unpacking the raw GCT data are read in the PSB class, while GMT objects, obtained from unpacking GMT raw data are read directly in the GTL class. The GTL run method computes the results of the logical conditions for the corresponding object type (L1GtConditionEvaluation), then evaluates the result of the logical expression for each algorithm (L1GtAlgorithmEvaluation).

The main class L1GlobalTrigger, driving the emulator execution, produces also the GT DAQ record and the GT EVM record, in the same format as the hardware module. In addition to these records, the GT also saves a record containing the objects which actually fired a given "condition" from a given algorithm. The three records are saved in the event as framework EDM products.

4. Use Cases for Level-1 Trigger Emulation

The most important use cases for the L1 trigger emulators are: validation and monitoring of the L1 trigger hardware, production of L1 trigger object maps, and Monte Carlo (MC) simulation of the L1 trigger response.

For the validation of the L1 trigger hardware, one runs the emulator on same input data as hardware and compare the results. In case of disagreements, one looks for errors in the hardware or in the emulator. A precise method for validation which can be done also offline before putting the hardware in production is to run pattern tests in emulator and trigger hardware and to compare the results. For example, one can generate muons, pass them through detector simulation and the trigger modules before the GT, then convert the output to a format appropriate to load in the FPGA chips. The patterns are loaded in the hardware, passed through the hardware and the results recorded. In parallel, the same patterns are run through the GT emulator.The emulator and the hardware results are then compared. The method allows to test thoroughly the hardware by choosing adequate pattern tests, covering all the expected experimental situations.

After the emulators are properly validated against the hardware, one can use them for monitoring of the trigger hardware during data taking. This implies running the emulators during data taking in the Event Filter farm, together with the HLT processes. Due to severe timing constraints of the HLT, not all the emulators can be run for each event; depending on the interest, one can run them selectively, or on a prescaled set of events. The results from the comparison of the emulator and the hardware module are managed by the Data Quality Monitoring (DQM) system in real time, such that the data manager can spot problems in the hardware during data taking and take appropriate measures. The L1 emulator-based DQM can also be run offline on every event from the datasets saved at Tier-0, alongside prompt reconstruction.

Due to the complexity of the treatment and taking into account the computing constraints

on the FPGA chips from the L1 GT trigger, the list of trigger objects which actually fired in a given algorithm could not have been implemented in the hardware. Instead, the L1 GT emulator is run in the EF farm during data taking on each event selected by the L1 trigger hardware and "trigger object maps" are produced. The "trigger object maps" provide the list of objects which fired a "condition" for all the "conditions" in an algorithm, as well as the result of the algorithm before prescaling (not saved by the hardware in the readout record, the hardware saves only the result after prescaling). These trigger objects maps are used in the HLT to start the evaluation of the HLT paths from the L1 trigger objects which fired, avoiding the "volunteers" (trigger objects used in the selection, incompatible between L1 and HLT). This use case requires perfect agreement between the hardware results and emulator results and imposes severe timing constraints on the L1 GT emulator. Using the L1 trigger objects as seeds, one can also reduce the time needed by HLT by performing conditional (reconstruct only when needed) and regional (reconstruct only in the region of the L1 seeds) reconstruction.

CMS has developed a complete MC event sample production, starting from the generation of events (GEN) according to chosen physics models, followed by a precise detector simulation (SIM), trigger simulation and reconstruction. The L1 trigger emulators are run on the digitized quantities (DIGI) from the corresponding detectors and provide the L1 trigger response; in the production chain, L1 trigger is followed by the HLT. One can perform detailed physics studies and detailed trigger studies on the MC samples (trigger efficiency, turn-on curves, etc), optimizing the physics program of the CMS. As part of the physics optimization, extensive studies are done to develop the L1 trigger menus and HLT trigger menus, determining the trigger requirements, the prescale factors and computing the expected trigger rates such that the hardware constraints in timing and allowed rates are also fulfilled. New proposed trigger quantities can also be studied with the L1 trigger emulators, before they are implemented in the L1 trigger hardware.

5. Validation and Performance Improvement of Level-1 Trigger Emulation

As mentioned before, the L1 emulator modules running online have stringent timing requirements, as they are running in the EF farm and the average HLT time per event must be of the order of 40 ms. Extensive effort has been invested to reduce the timing for each L1 trigger emulator. Tools, methodology, and tests for timing reduction were also provided by the CMS offline software framework developers, see Ref. [2].

The most effective method to reduce the timing is to streamline the code, improving the algorithmic organization of the code. Timing profilings of the code were regularly produced for emulator modules, identifying the time contribution for each method.

Timing improvement can also be obtained by using a tailored configuration. There are multiple configurations possible for emulator modules, depending on the environment and the task to be performed. As a rule, for online use, one sets the parameters which produce the minimum required results. For example, in order to seed the HLT with L1 objects, only objects from the bunch crossing corresponding to L1A are needed and the L1 trigger object maps. The L1 GT DAQ and EVM records are not needed for seeding. By default, the hardware is configured to include in the readout records data for three or five bunch crossings (one or two bunch crossings before and after bunch crossing with L1A, respectively). The GT emulator was therefore configured to compute only the results for one bunch crossing and to not produce the unnecessary records, overwriting the corresponding parameters read from the hardware configuration.

General techniques to reduce the timing were also used, among them avoiding string comparisons and string parsing, using constant references to avoid useless copying, etc. The choice of the STL containers was also done carefully considering the effect on timing; careful declaration of container size to avoid resizing was used whenever the size of the containers was possible to be estimated.

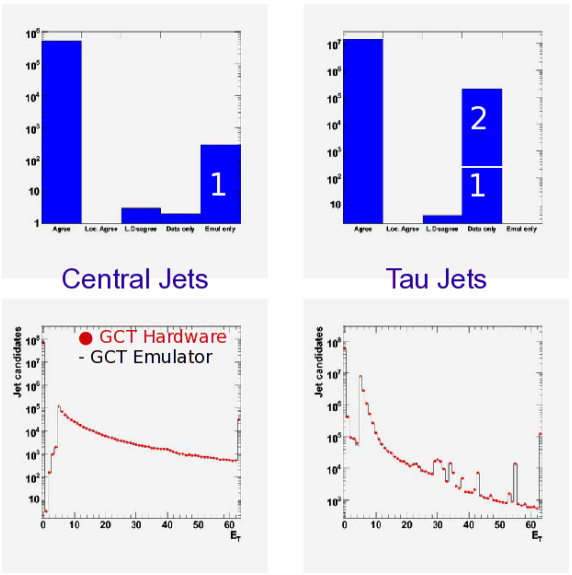


Figure 6. Example of problems seen in GCT system.

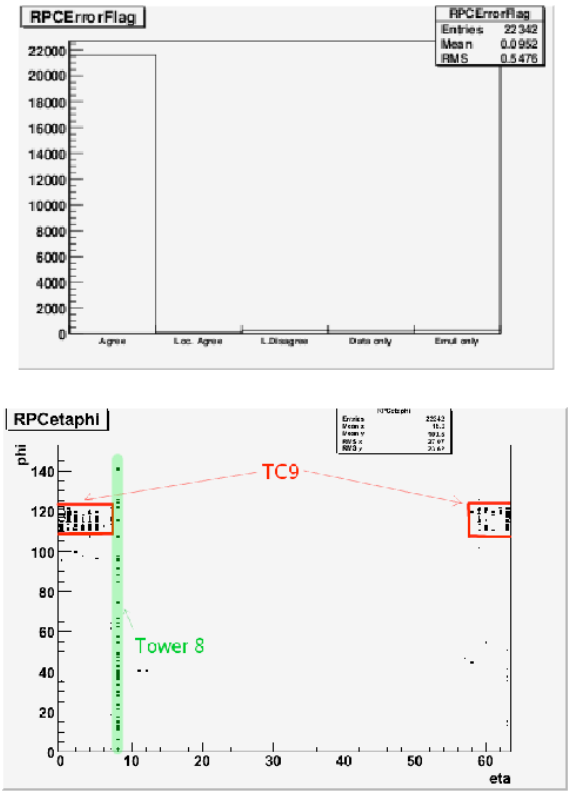


Figure 7. Example of problems seen in RPC system.

Caching of event setup, as described in Section 3.1, was also used extensively.

The tools used extensively for timing and memory optimization were IgProf (Ignominious Profiler) [3] and Valgrind [4].

As an example for timing evolution: the initial implementation of the GT emulator was running on average in ~ 40 ms per event. After streamlining the code and reducing the number of bunch crossings to one, the average timing per event was ~ 14 ms. The next two optimization cycles, using the methods described above, brought the timing first to ~ 7 ms per event and then to $1 - 3$ ms per event, depending on the complexity of the input.

The validation of the emulators and the timing reduction is an iterative process, using data collected in cosmic data taking ("Global Runs") and various MC samples.

In the following, some examples of the usage of L1 trigger emulators are given.

Fig. 6 shows the results of the GCT hardware validation for a run from 2008 CRAFT (Cosmic Run at Four Tesla) data taking for the collections of central jets and tau jets. The emulator and the hardware are in good agreement for the majority of the events, as can be seen from the upper left plot (column "Agree"). However, one observes few hundred jets seen in the central jet collection in the emulator only (denoted by "1"), while for tau jets one sees some tau jets in the data only ("1" in upper right plot). In this case, due to tau-veto bit handling errors in the hardware, some central jets were misidentified as tau jets; after the firmware was updated, the problem disappeared. The discrepancies denoted by "2" in the upper right plot were tracked to a bug in jet finding algorithm, producing extra candidates at $\eta = 0$ boundary, present in both emulator and hardware. The upper right plot shows the comparison after the emulator was patched, but the firmware was not yet patched. After both emulator and firmware were

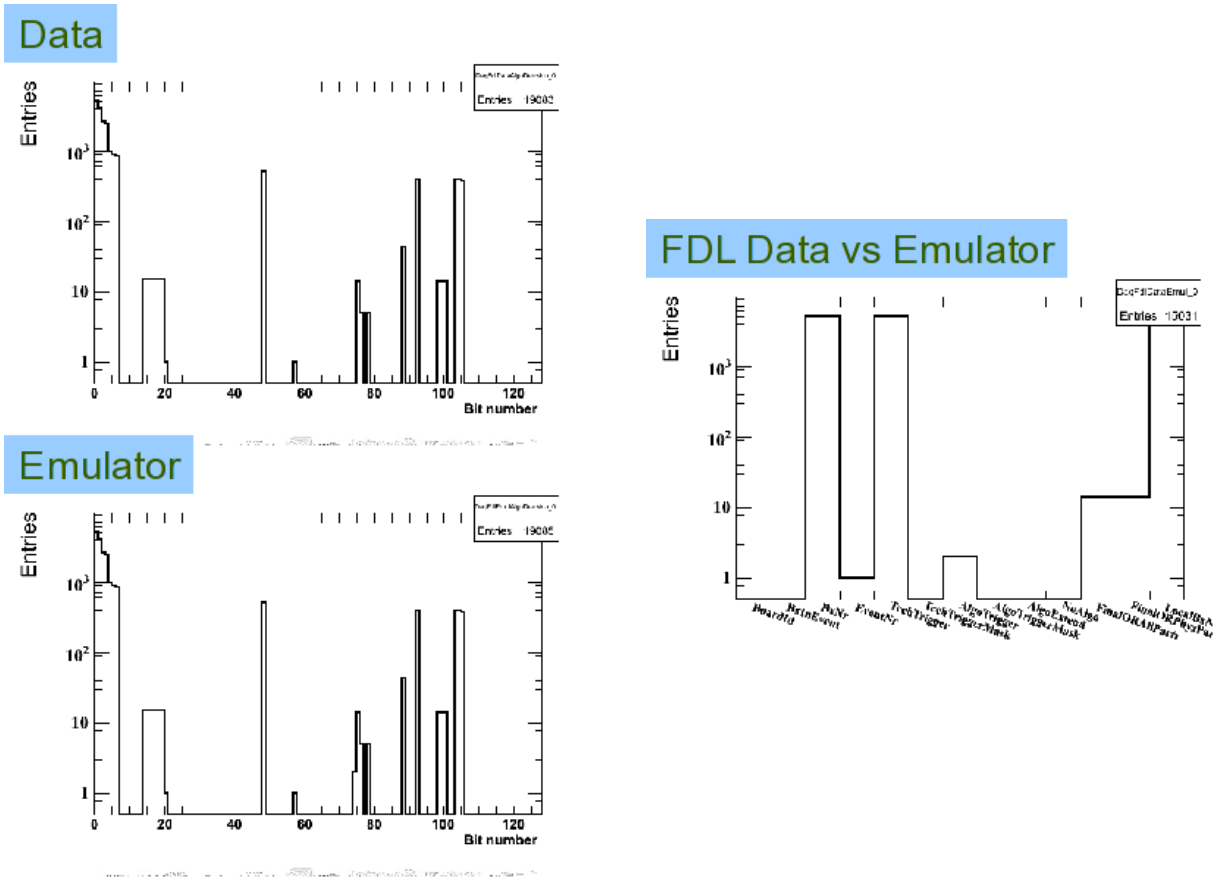


Figure 8. Results of comparison between GT data and emulator.

patched, the emulator and the hardware are in good agreement, as can be seen in the jet E_T distributions from the lower two plots.

Fig. 7 shows the results of the RPC hardware validation for some runs from CRAFT 2008. There is a 98% agreement between hardware and emulator results, as can be seen in the upper plot. By plotting the track $\eta-\phi$ distribution for the events not in agreement, one can see that the errors are localized in the $\eta-\phi$ region corresponding to a particular trigger crate (TC9). Some disagreement is also seen in another region, identified as Tower 8. The disagreements were very likely due to transmission errors. After updating the RPC firmware, with better transmission implemented, no errors were observed in the following runs.

Fig. 8 shows the comparison of the GT results for physics algorithms in a CRUZET (Cosmic Run at Zero Tesla) from early 2008. The left plots show the number of triggered events for each bit number corresponding to the physics algorithms. The right plot shows the number of discrepancies between emulator and hardware for various GT quantities. There were discrepancies in the bunch crossing number (at that time, the number was not yet implemented in the emulator), in the results for technical triggers (there were no technical triggers available in the hardware), as well us 10 events with disagreements in the FinalOR (due to a misconfiguration for the GT database key used for emulator). All discrepancies seen were fixed in the meantime.

6. Summary

The CMS Collaboration has developed an extensive system of L1 trigger emulation, which performs a bit-level emulation of L1 trigger data processing. For each L1 trigger subsystem an

emulator was developed, using the same input and producing an output in the same format as the hardware.

The L1 trigger emulators are actively used for validating and monitoring the L1 trigger hardware, for Monte Carlo simulation of the L1 trigger response, with special emphasis on the development of the L1 trigger menu and for L1 seeding of HLT during data taking.

The results from the cosmic data-taking runs started in the year 2007, with the L1 emulator-based DQM running throughout online and offline, prove the usefulness of L1 trigger emulation for debugging L1 trigger. More problems were spotted and solved in the L1 trigger hardware configuration and in the L1 trigger running. Starting from this year, the L1 emulation is also used for the validation of every event taken by CMS, running in the offline DQM. It is also used in large scale production of MC samples, allowing to define and refine the CMS physics program.

References

[1] The CMS Collaboration, Chatrchyan S *et al* 2008 The CMS experiment at the CERN LHC *JINST* **3** S08004 doi: 10.1088/1748-0221/3/08/S08004

[2] Elmer P *et al*, 2009 *CMS Software Performance Strategies* these proceedings

[3] Eulisse G and Tuura L 2004 *IgProf profiling tool* Proc. CHEP04, Computing in High Energy Physics, Interlaken

[4] N Nethercote and J Seward 2007 *Valgrind: A Framework for Heavyweight Dynamic Binary Instrumentation* Proc. PLDI, San Diego, California, USA, June 2007. See also http://valgrind.org/