Monitoring the CMS strip tracker readout system

S Mersi[1], R Bainbridge[2], G Baulieu[3], S Bel[4], J Cole[5], N Cripps[2], C Delaere[4], F Drouhin[4,6], J Fulcher[2], A Giassi[7], L Gross[8], K Hahn[9], L Mirabito[4], M Nikolic[10], S Tkaczyk[11] and M Wingham[2]

[1] INFN and Università di Firenze, Italy
[2] Blackett Laboratory, Imperial College London, UK
[3] Institut de Physique Nucleaire de Lyon, France
[4] CERN, Geneva, Switzerland
[5] Rutherford Appleton Laboratory, UK
[6] Université de Haute Alsace, France
[7] INFN, Sezione di Pisa, Italy
[8] Institut Pluridisciplinaire Hubert Curien, CNRS, France
[9] Massachusetts Institute of Technology, USA
[10] University of California, Santa Barbara, USA
[11] Fermi National Accelerator Laboratory, USA

E-mail: `Robert.Bainbridge@cern.ch, Stefano.Mersi@cern.ch`

Abstract. The CMS Silicon Strip Tracker at the LHC comprises a sensitive area of approximately 200 m^2 and 10 million readout channels. Its data acquisition system is based around a custom analogue front-end chip. Both the control and the readout of the front-end electronics is performed by off-detector VME boards in the counting room, which digitises the raw event data and perform zero-suppression and formatting. The data acquisition system uses the CMS online software framework to configure, control and monitor the hardware components and steer the data acquisition. The first data analysis is performed online within the official CMS reconstruction framework, which provides many services, such as distributed analysis, access to geometry and conditions data, and a Data Quality Monitoring tool based on the online physics reconstruction.

The data acquisition monitoring of the Strip Tracker uses both the data acquisition and the reconstruction software frameworks in order to provide real-time feedback to shifters on the operational state of the detector, archiving for later analysis and possibly trigger automatic recovery actions in case of errors. Here we review the proposed architecture of the monitoring system and we describe its software components, which are already in place, the various monitoring streams available, and our experiences of operating and monitoring a large-scale system.

1. Introduction

The CMS Silicon Strip Tracker (SST) [1, 2] sub-detector extends in the region of radius $r < 120$ cm and $|z| < 270$ cm around the pixel vertex detector and it is completely based on silicon micro-strip detectors, covering a surface of about 200 m^2, the largest micro-strip tracking detector ever designed. Its aim is to reconstruct the tracks and vertexes of charged particles in the high multiplicity environment of the LHC. The key aspects to solving this pattern recognition problem are a low cell occupancy and a large hit redundancy. The low occupancy is obtained by using high granularity sensors ($80 - 200\,\mu$m strip pitch) and fast front-end sampling time

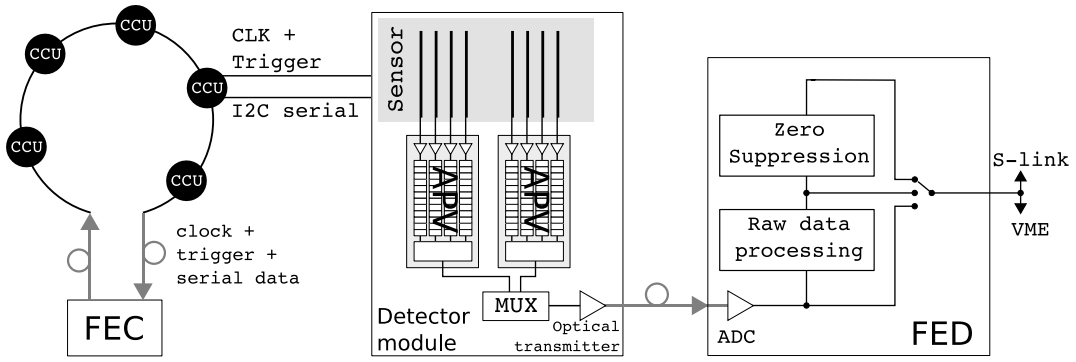

Figure 1. Scheme of the Data Acquisition System.

(about 25 ns, the LHC collision period). The redundancy is guaranteed by the overall design of the tracker (10 cylindrical layers and 12 end-cap layers per side, part of which instrumented with double-sided detectors) which allows many measured points per track (about 12 for 90° tracks [3]) whilst keeping an acceptable material budget, so as to minimise the negative effect on the outer electromagnetic calorimeter performance. We will briefly describe the readout and control components and the data acquisition architecture of the SST.

1.1. Tracker Readout System

The basic element of this device is the detector module (shown in Figure 1): it is equipped with a preamplifier chip (APV [4]) which samples the micro-strips at the LHC frequency of 40 MHz and buffers the data samples into an analogue pipeline for about 4 μs. On receipt of a trigger, the front-end modules multiplex 256 strips at LHC frequency, perform electrical electrical-to-optical conversion of the analogue signals stored by the APVs and propagate the signals to off-detector electronics via optical fibres. The serialised analogue data are marked by a digital header containing a few meta-data, comprising an encoded event time-stamp (the pipeline address) and an error bit. The digitisation is performed far from the detector in the service cavern by the Front End Driver boards (FEDs). Analogue transmission requires that a gain calibration of the optical link is performed during the commissioning and the data taking.

The FED [5] is a VME board implementing the logic needed to read the external clock and trigger and to communicate over the VME and S-link buses. It comprises 8 Front-End Units, which independently process the signal from a ribbon of 12 fibres (256 strips). The VME link can be used both for configuring the FED and to read processed data during the commissioning phase, whilst the S-link is a fast unidirectional dedicated bus used to connect the FEDs to the central data acquisition. The FEDs convert the input optical signal to an electrical signal and perform digitisation on every input line with 10-bits ADCs.

The stream of data produced by the ADCs can be processed in different ways, according to the chosen FED operation mode. Each Front-End Unit writes its processed data to an output FIFO, from which the main FED board reads and serialises the data to one of the output links. The signal processing chain is performed in stages ad the mode selection determines the path of the data through the different processing stages.

Scope mode: This mode is primarily used for debugging: it simply performs data capture for a configurable number of consecutive bunch crossings following the receipt of a trigger. This mode performs no data processing and simply provides the raw APV data stream.

Virgin Raw Data mode: A real-time logic takes care of sensing the arrival of an APV data frame header and triggers the acquisition of the full data packet. The digital header is stored

into a register whose consistency across the 12 channels is checked by the Front-End Unit logic, while the following analogue signal, corresponding to 256 sensor strips, is digitised and placed in the output FIFO to be transmitted to one of the readout links. The number of detected frames and the number of received trigger are recorded. The consistency of the APV pipeline addresses is also checked.

Processed Raw Data mode: In Processed Raw Data mode the pedestal value specific to each strip is subtracted from the Virgin Raw Data. The pedestal and noise values are measured for each strip during a dedicated calibration run and are stored in a 10 bit registers, used both for Process Raw and Zero suppress mode runs.

Zero Suppression mode: After the pedestal subtraction the data are *zero suppressed*: only the strips with a high signal are forwarded to the data analysis, the threshold being usually set proportional to the strip noise.

1.2. Tracker Control System

The tracker has to operate reliably for a long period of time ($\sim$ 10 years) in a harsh radiation environment. With increasing radiation exposure, the response of the silicon sensors, as well as that of the readout electronics, will change. The high radiation flux will also produce Single Event Upset phenomena in the front-end electronics affecting both the logic and pipeline memories of the front-end ASICs at an estimated rate of 100 per hour [6]. Furthermore, intrinsic variations in the fabrication process also necessitates device registers that allow to tune various aspects of the electronics performance. These registers must be tuned individually and periodically through dedicated *commissioning procedures*.

The Tracker Control System [7] provides a two-way communication channel with the front-end detector modules, which is used to adjust the front-end parameters and monitor environmental variables (such as temperature, current, voltages) via front-end ASICs and hardwired probes.

The architecture of the Control System comprises multiple levels (see Figure 1): a branch of detector modules shares the same node, known as Communication and Control Unit (or CCU), and communicate with it through the serial I^2C protocol [8] (100 kHz to 1 MHz clock), while groups of several CCUs are connected between them in a daisy chain through a token ring protocol, forming a control ring. With this design the whole tracker can be accessed through 356 control rings. A custom VME board, the Front End Controller (FEC), acts as a token ring master, the slaves being the CCUs. The FECs are housed in the CMS service cavern, while the CCUs are deployed within the tracker. Like for the analog readout, the transmission of the control signals to and from the CMS detector is performed over optical fibres with a length of the order of 100 m.

The silicon strip tracker has 15 148 front-end modules, thus the potential for problems within the system (both in hardware and software) is largely due to its complexity and scale.

1.3. Tracker Trigger System

As the Control System provides the communication path to the front-end modules, this system is also the natural candidate to dispatch the LHC clock. This is implemented on a communication layer which is independent of any other transaction of information between the Control System and the modules: the LHC clock is used as clock for the CCU token ring and it has been designed to use a robust, completely hardware-implemented protocol.

The LHC clock line also dispatches signals whose latency is critical, such as trigger and re-sync, coded as missing clock ticks. This is know as the Timing, Trigger and Control system (TTC [9]) and is achieved through specific cards (TTCci, LTC) which encode the global fast signals into the LHC clock and dispatch it to the Control and Readout systems.

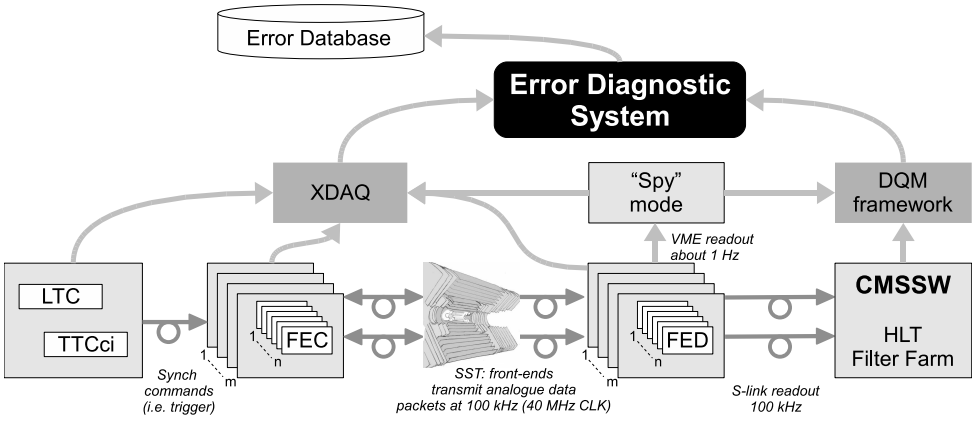

Figure 2. Monitoring architecture schematic.

2. The monitoring system

The hardware components of the strip tracker must be configured, calibrated and synchronised prior to data taking. This is done through several *commissioning procedures* which will be heavily used during the commissioning phase of the experiment, but will also be used to tune the detector periodically. The end result of these procedures should be an optimally configured and calibrated system.

The purpose of the various applications described within this article is to monitor how the operational state and performance of the detector and its readout systems change during data taking with respect to its initial optimal state, as determined by the commissioning procedures.

The CMS online software is split into the Data Acquisition framework (XDAQ [10]) and the event reconstruction framework (CMSSW [11]). XDAQ is used by the applications configuring the hardware and steering the data taking, while CMSSW provides the environment used to implement the software-based High Level Trigger.

The architecture of the monitoring system is shown schematically in Figure 2. Information on the status of the system can be obtained through different data streams from various XDAQ applications and from the Data Quality Monitoring (DQM) tools within CMSSW. The Error Diagnostic System is designed to merge and handle monitoring data from sources implemented within XDAQ and DQM. Details of the monitoring streams are given in Section 3.

The online *configuration database* for the strip tracker sub-detector stores all parameters that are downloaded to hardware in order to configure the detector prior to a physics run.

A CMS *errors database* has been designed in order to keep track of all errors, hardware failures and unusual detector states that could occur during physics runs or configuration runs. Browsing this DB during the data analysis and reconstruction stage will allow to identify data taking periods for which (part of) the detector was delivering data while being in an inappropriate state.

2.1. The Error Diagnostic System

The need for having an Error Diagnostic System was realised soon in the development cycle of the software: The fine granularity of the detector, both in term of modules and of control or readout links, makes problem analysis difficult without the help of an automated failure analysis tool. Giving as quickly as possible a human-readable answer to failures is the main reason for having implemented a complete online error diagnostic system.

The current design of the Error Diagnostic System structure is described below and shown schematically in Figure 3. The whole system has been split into logical and/or physical sub-

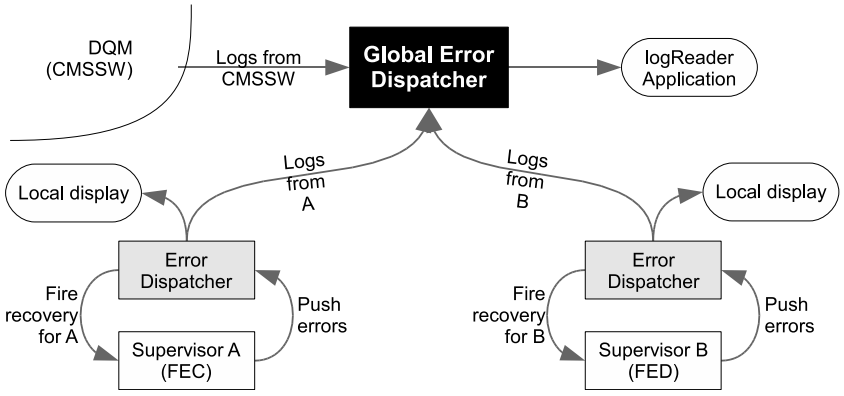

Figure 3. A simple Error Diagnostic System structural scheme.

systems (FEC, FED, Trigger, ...) which are configured through dedicated supervisors. One diagnostic C++ object is embedded into each supervisor providing a common interface between the XDAQ applications steering the data acquisition and the error management. This object is able to route the triggered errors to several outputs (file, database, etc...), to qualify them (as warning, critical, ...) and push them to the proper Level 1 Error Dispatcher.

Error Dispatchers are error analysers and routers. A Level 1 Error Dispatcher should manage only one kind of sub-system. Therefore, in a logical cluster of sub-systems of the same kind, the supervisors will all propagate their error messages to one single dedicated Level 1 Error Dispatcher.

If an error message code received by a Level 1 Error Dispatcher matches one of the known errors, then the requested reconfiguration action (if any) is fired on the appropriate sub-system. If the incoming error message incoming to a Level 1 Error Dispatcher can not be analysed, the error message and/or the status is sent to a Level 2 Error Dispatcher. This second level of diagnostic is intended to process errors which requires to act on at least two different kind of sub-systems.

According to the complexity and the peculiarities of the system, it is possible to layer as many Error Dispatcher levels as requested. Furthermore, it is possible to send specific error messages to a top level Error Dispatcher when an error has been detected via the monitoring analysis. In this way, it should be possible to recover the system through optimised procedures without having to stop the data taking and reinitialise the whole the system.

2.2. DQM-based monitoring of event data

The Physics and Data Quality Monitoring (DQM) is designed to serve as the primary monitoring environment for the CMS data taking. To this end a generic structure was implemented in the form of distributed source-collector-client software architecture illustrated below. The model is a classic server-client chain in which individual source nodes are connected to single or multiple clients by a collector node.

The DQM sources produce all monitoring information and send data in the form of Monitor Elements histograms (aka MEs) to the "collector" (aka server) node. The destination path and port address of the collector node is specified and MEs are tagged for distribution, by the source. MEs are "published" and clients "subscribe" to monitoring data of interest through the collector via TCP/IP sockets.

The DQM collector is standardised and requires no customisation on the part of the user. Once the collector application is initiated, it simply accepts the published MEs from the source and sends information about available content to each connected client node. Updates from the

source nodes are relayed to all connected clients and the user is allowed to add or remove sources or clients from the collector at run time.

DQM clients subscribe to data sent by one or more collectors and perform operations on the MEs. Analysis tools are deployed on the monitoring information on the client side. Histograms can be compared to "reference histograms" in order to set data quality flags. Cuts can be made on quantities of interest in the MEs that raise flags. Those flags are propagated to other applications in the Error Diagnostic System.

3. Overview of the monitoring streams

The data taking process and operational state of the data acquisition systems can be monitored through different streams of data, which are presented here. The monitoring system is designed to be able to merge the information coming from different streams and make them available for diagnostic checks.

3.1. Monitoring the trigger, control and readout systems

The most basic monitoring relies on the monitoring of parameters from each of the applications controlling the hardware of the control, trigger and readout systems, which can be used to identify hardware problems. The actual values to be checked can be easily changed and will ultimately be defined when experience with the full system will be acquired. So far a few basic checks were defined.

The FEC controllers should report that all the control rings are closed, and no I^2C error condition from the CCUs during the configuration.

There are a number of sensors in the FED which read temperatures of its components and voltage levels. These values, along with a number of variables describing the operational state of the FED, such as buffer levels and status registers of the FED FPGAs are checked periodically by the FED supervisors and will be fed to the Error Diagnostic System. An interface is provided to display the current state and a history is stored to produce trend plots over the run. Monitoring information can also be saved to file for subsequent analysis.

3.2. Monitoring on the global online computing farm

Another set of checks can be made on the actual data stream received from the FEDs, such as the meta-data from the Front End electronics. The first place where this information is available for the complete event is from a dedicated event stream provided by the Filter Farm (High Level Trigger). These data are analysed within the CMSSW framework by software processes running on dedicated monitoring nodes in the global online computing farm. These processes provide histograms which are published to the Data Quality Monitoring system.

Some of the measured parameters are: front-end buffer occupancies, possible front-end logic errors and lack of synchronisation between front-end chips. These data can be aggregated at the level of the readout board (FED), the controlling board (FEC) or according to the physical position of front-end chips (APV) inside the detector. Not only will this help detecting data acquisition errors, but also it will aid understanding the cause of the problem. Some of the monitorable elements related to the physics reconstruction can also be used as a feedback to the Data Acquisition System: for example the occupancy of hits not associated to any track will increase in case of a problem with the zero suppression algorithm.

3.3. Monitoring using the FED spy channel

During normal running the FEDs operate in zero suppressed mode meaning that the raw signal level on each strip is not available. The FED spy channel allows raw data to be read out, before any processing has been performed by the FED. This raw data can be used both to diagnose problems with a FED and to monitor the state of the system during the run.

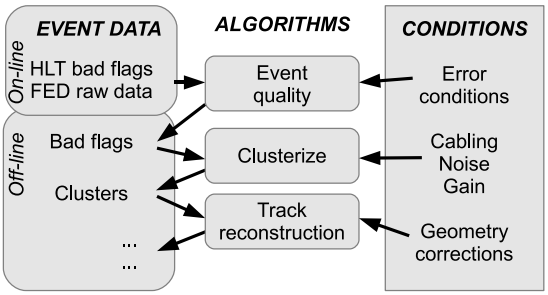

Figure 4. Event data building scheme.

To start capture of data the spy channel is armed by sending the appropriate serial command to the FED. Once the arm command is received, raw data from each fibre is stored in a memory buffer, and is continuously overwritten until the frame finding logic signals that a frame has been found. Then a constant number of samples are stored so that a complete frame, including header. The data is readout over VME from one board at a time; the VME limited bandwidth is not an issue as this readout will be performed at a frequency less than 1 Hz.

Pedestals and noise can be calculated from the spy channel data in the same way as from Virgin Raw data during commissioning even when FEDs operate the zero suppression, so that shifts in pedestals or noise can be detected during the data taking. The presence of the frame header and samples of the baseline level outside the frame can be used to monitor the optical link gain and to verify that the frame finding threshold is still appropriate.

The readout and display of the spy channel can be performed at several levels: access to the channels in each FED supervisor application, access to a module view (2 or 3 channels) from a client application, and finally via a complete event builder, collecting, merging and analysing data from all FEDs (440) during data taking.

4. Using error conditions data during event reconstruction

All non-event data required by event reconstruction are stored in the CMS offline *conditions database*, which is accessible within the CMSSW framework via the *event setup* interface [12]. A large sub-set of configuration and calibration data stored in the configuration database are also considered to be conditions data used by reconstruction. Examples include the calibration constants for various detector and hardware elements, such as gain and noise. This necessitates an Online-to-Offline (O2O) transfer tool that copies data from the configuration database to the conditions database. The conditions database is fully integrated with the CMSSW software framework, which guarantees synchronicity between the conditions data and the event data using a mechanism that defines an *Interval Of Validity* (IOV) range for all conditions data with respect to the event data.

A limited number of checks can be performed in real-time on the High Level Trigger. The software modules that unpack the FED raw data and generate hit information have access to the raw FED payload and the meta-data contained therein. This meta-data will be used to identify problematic detectors and filter these detectors from the online reconstruction performed by the High Level Trigger. One example is the filtering of detectors desynchronised with the rest of the strip tracker sub-detector.

In the CMS architecture after events are first filtered by the High Level Trigger (Filter Farm) they are stored and processed by a worldwide-distributed computing facility organised in levels (Tiers). During the first-pass offline reconstruction at Tier-0, the event data are processed in order to perform event reconstruction. Non-event data available from the conditions database via the event setup interface are used during this reconstruction. Special software modules will

be used to analyse all useful conditions data and condense them into a compacted form for use during reconstruction, as illustrated in Figure 4.

In particular, configuration and calibrations data generated during by commissioning procedures will be merged with error conditions generated by the various monitoring streams to provide information on the operational state and performance of the strip tracker readout system. This information will be used to flag individual detectors and/or complete events as being good or bad. These flags are then used by the reconstruction software to filter out data from affected detectors.

These flags can also be used to generate *event quality* and *run summary* flags in order to provide some condensed, high-level information on the status of the detector for later use.

5. Experiences during detector commissioning at the Tracker Integration Facility

The final integration of the SST sub-detectors was completed recently at CERN in the dedicated Tracker Integration Facility (TIF). This was a clean room equipped with all the infrastructures needed to operate and power up to 1.6 million readout channels. With these devices it was possible to perform for the first time a complete test of a large scale section of the final tracker (nearly 16% of the total readout channels), addressing all the issues related to data handling, system monitoring, data quality monitoring, etc. Also the environmental control could be tested in a realistic scenario, as a cooling system was set in place which was capable to bring the detector's temperature down to $-15°$C.

The SST was operated for several months between March and July 2007. During this period large samples of cosmic ray data were acquired, stored and analysed, so that also the tracking and alignment algorithms could be tested with real physics data.

During all these operations the monitoring system described in this article was setup and developed, even if not fully implemented. The operations at the TIF stimulated the development of new features and also offered a benchmark test.

5.1. Using the Error Diagnostic System at the TIF

The Error Diagnostic System was used *in situ* for the first time during the first large-scale test of the SST (Spring-Summer 2006 – about 1% of the full-scale system) and underwent significant developed since then.

To test the robustness of the implemented solutions we have put in place at the TIF only one GlobalErrorDispatcher module collecting all the logs issued from the whole online software.

The system implemented comprised the main supervisor applications (FEC, FED, the TrackerSupervisor which coordinates the data acquisition, ...), but it was not possible to connect the DQM (and the CMSSW framework) to the Error Diagnostic System because the corresponding development versions were mutually incompatible at the time. Tests are still to be made *in situ* with these two monitoring streams feeding the centralised Error Diagnostic System.

The Error Diagnostic System framework is now stable and well integrated into the online software framework. Next short term efforts will concern the integration of DQM framework and CMSSW framework into this diagnostic system. A mid-term effort will concern the integration into the diagnostic system framework of all the monitoring warnings coming from the power supply and control system along with its independent environmental probes. Once done, having a centralised errors managing process will allow to perform sharp analysis of incidents.

5.2. DQM-based hardware monitoring at the TIF

DQM-based hardware monitoring is presently being implemented and a functional prototype is already in place. This was tested with the data collected at the TIF, even if not run online nor connected to the Event Diagnostic System.

This software monitors the status of the readout by reading the event data, which incorporate the APV meta-data from the frame header as well as status flags produced by the FEDs. For example, available histograms indicate the agreement of APV addressing with the "golden address" provided by a special APV emulator in the service cavern [13] and with the majority APV address of the event (calculated as a median APV address) to ensure the total synchronisation of the readout electronics.

In the eventuality of disagreement of the address checks, finer grain date is available down to the APV level on exactly what the problem is. The errors mode can be lack of synchronicity of APV signals, incorrect APV headers, or error bits sent by the APV itself indicating errors in the signal buffering. In addition, information about the FE FPGA memory status and BE FPGA status is also made available.

A simulation of the entire Tracker Header is also available for development purposes and is implemented by using the Mersenne Twister algorithm [14] to generate bit values and completely specify the header fused on to (zero content) events generated by CMSSW. The user is then free to run a time varying simulation of the header and test and develop the monitoring software without the inherent computing overhead of real events.

The status words and possible error data are decoded from the integer values sent by the registers and histograms are generated for analysis. The DQM source application in this case is a CMSSW data analyser which publishes the available monitoring information to the client which in turn performs quick analysis and displays the resulting histograms for the end user.

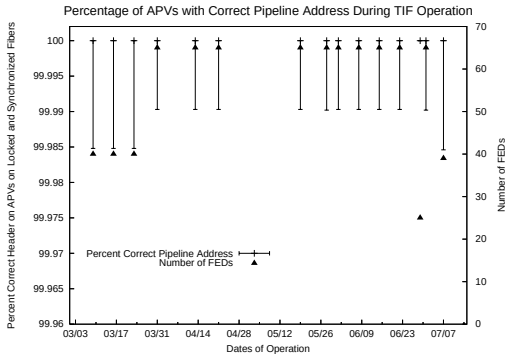

Figure 5. Fraction of front-end chips with a synchronised output during TIF operations.

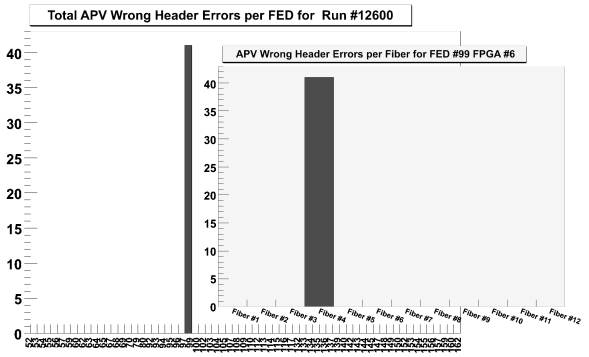

Figure 6. Here a problem that was localised to a single optical fibre.

Monitoring plots produced with TIF data are shown here. In Figure 5 the quality of the data collected in different runs is compared using the percentage in agreement with the majority APV address. This plot shows that the fraction of reliable front-ends was high and stable during the TIF operations. Such a monitoring, when run online, will provide a clear signal in the event of deteriorating performance.

Figure 6 shows the FEDs that have non zero address errors (bigger plot): one can then simply look at the synchronisation state and refer to the FED histogram to see which FED is problematic, and then search the directory structure of the DQM monitoring data file and see exactly which components are failing and the rate of failure (inset). Appropriate actions can then be taken, either in the form of a reset of the APVs or a check of the optical fibres supplying the data.

6. Conclusions

A monitoring system was designed for the data acquisition of the CMS Silicon Strip Tracker. Most of its components are already available and a prototype system was successfully tested during data acquisition at the Tracker Integration Facility, where about 16% of the Tracker was operated.

The basic element of this system is an Error Diagnostic System, which collects data from the applications controlling the hardware and from the online event reconstruction. A "spy mode" readout is also in place and will be able to provide detailed on-line monitoring of the front-end raw data stream.

This system will also be able to automatically trigger the proper error recovery procedures depending on the detected problem. Error conditions are logged to a dedicated database which will be used along with the condition database by the offline event reconstruction, where quality meta-data will be embedded within the event data that are archived to disk.

References

[1] 1998 CERN-LHCC-98-06

[2] 2000 CERN-LHCC-2000-016

[3] 2006 *CMS Physics TDR: Volume I (PTDR1), Detector Performace and Software* (CERN) CERN-LHCC-2006-001

[4] French M J *et al.* 2001 *Nucl. Instr. and Meth.* A **466** 359–365

[5] Coughlan J A *et al. Ninth Workshop on Electronics for LHC experiments* CERN-2003-006

[6] Noah E, Baue T, Bisello D, Facci F, Friedl M, Fulcher J R, Hall G, Huhtinen M, Kaminsky A, Pernicka M, Raymond M and Wyss J 2002 *Nucl. Instr. Meth.* A **492** 434–450

[7] Drouhin F *et al.* 2002 *IEEE Trans. Nucl. Sci.* **49** 846–850

[8] Philips Semiconductors 2000 *I^2C Bus Specification, version 2.1*

[9] Troska J, Corrin E, Kojevnikov Y, Rohlev T and Varela J 2006 *IEEE Trans. Nucl. Sci.* **53**(3) 834–837

[10] Brigljevic V *et al.* 2003 (*Preprint* `hep-ex/0305076`)

[11] CMS Software page `http://cmsdoc.cern.ch/cms/cpt/Software/html/General/`

[12] Jones C D *et al.* 2006 *Proc. for Computing in High Energy and Nuclear Physics, Mumbai, India*

[13] Iles G, Cameron W, Foudas C, Hall G and Marinelli N *Proc. of the Eighth Workshop on Electronics for LHC Experiments* pp 396–369 CERN-LHCC-2002-034

[14] Matsumoto M and Nishimura T 1998 *ACM Trans. Model. Comput. Simul.* **8** 3–30 ISSN 1049-3301