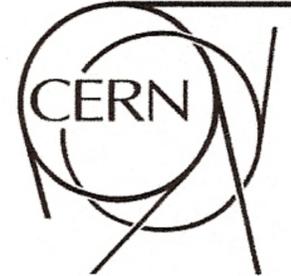


ATLAS NOTE

CERN-PH/2006-000

April 18, 2008



Data Preparation for the ATLAS High-Level Trigger Calorimeter Algorithms

The ATLAS Collaboration

Abstract

various

This note describes the data preparation layer for the ATLAS High Level Trigger Calorimeter Algorithms. Different calorimeter based algorithms (electrons, photons, taus, jets, missing E_T and muons isolation) use this same infrastructure. Fast processing and robustness are fundamental prerequisites for the operation of the trigger reconstruction algorithms.

ET

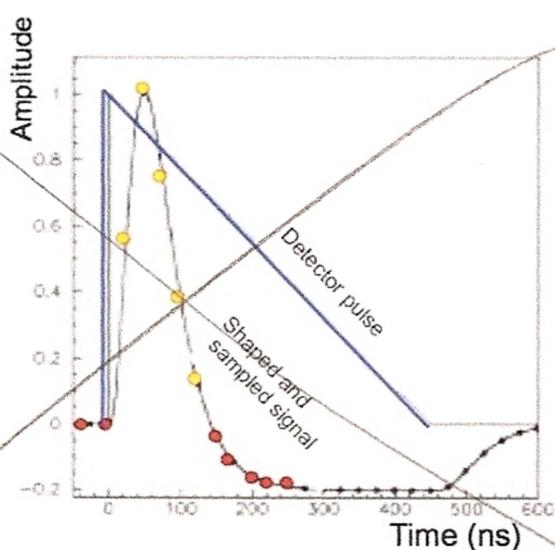


Figure 1: The LAr Calorimeter triangular original detector pulse is shaped and sampled (dots) at the bunch crossing rate (25ns). No more than 5 samples of the signal are necessary to obtain an accurate peak measurement.

1 Introduction

The ATLAS [1,8] detector is one of the two general purpose experiments at the Large Hadron Collider (LHC) at CERN. The bunch crossing nominal rate is 40 MHz between crossings. ^{At} Due to the designed luminosity, an all inclusive event rate of 1 GHz is expected and must be reduced by the trigger system to 200 Hz for recording and offline processing. Some of the ATLAS sub-detectors data acquisition electronics (the calorimeters and part of the muon systems) were designed ~~considering the interaction~~ ^{to have} with the hardware-based part of the trigger (the level 1 - LVL1) [2-5]. All sub-detectors participate ~~on the software levels~~ (level 2 - LVL2 and Event Filter - EF) [6].

^{high Level Software trigger} One important phase for any trigger software algorithm is the data preparation step which provides the conversion of the bytes of data produced by the detector electronics into a form manageable by the trigger algorithms. In the ~~present case~~ ^{as}, the digital information provided by the detector must be converted into calorimeter cells ^{to be} input to the reconstruction algorithms. A good data preparation step will provide the input to the trigger software in an organized manner, so that access to the prepared data is optimized. This note describes such step for the calorimeter trigger software. The same software interface is used for the LVL2 and EF for many different algorithms (electrons, photons, taus, jets and even muon isolation algorithms) [7].

1.1 Calorimeters Readout

The LAr calorimeter readout unit is the calorimeter cell. The cell electrodes receive the current due to the drift electrons in the liquid argon and form a triangular shaped signal (See Figure 1) [2]. The shaping and readout of this signal is performed by the Front-End Electronics. To preserve the dynamic range and the energy resolution, the signal is shaped with 3 possible gains (high, medium and low for small, medium and large signals, respectively). The Front-End Boards (FEBs) save analog samples of the signals coming from the detector at the bunch crossing ~~speed~~ ^{rate accepted} (every 25 ns). Each FEB can process up to 128 LAr calorimeter cells.

Those signals are converted by the FEBs to digital if the event is ~~approved~~ ^{approved} by the LVL1 Trigger. The digital information is sent to the ReadOut-Drivers (RODs). These are Digital Signal Processor (DSP) based machines, fast enough to deal with a number of input channels (2 FEBs feed one ROD DSP). From the pulse shape digitized at the FEB, the energy deposited in any cell can be calculated.

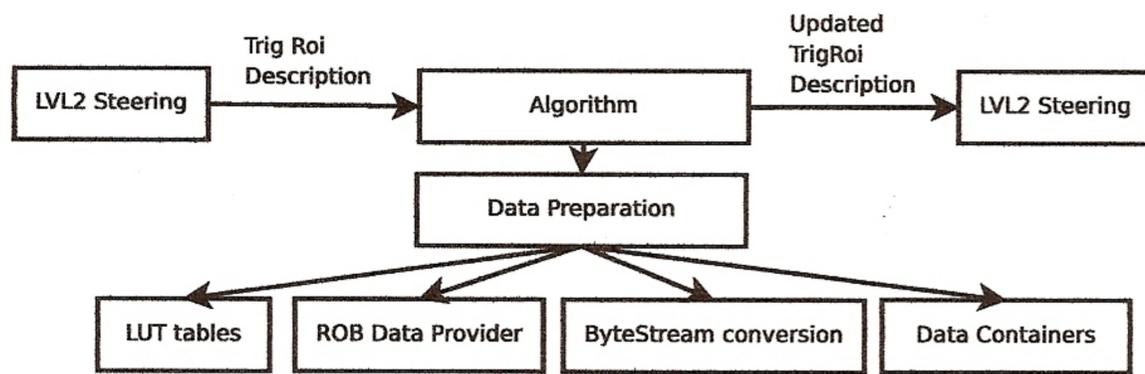


Figure 2: Different parts of the data preparation processing and their relation to the calorimeter algorithm at the LVL2. For details, see text.

Data from one ROD with at most 256 channels are sent to a ReadOut Buffer (ROB). The ROB keeps this data fragment until requested by the LVL2 or by the Event Builder (EB). The EB will request the fragments from the whole detector in case the event is approved by the LVL2 and send it for further processing at the EF farm.

The photons produced in the Tile Calorimeter [3] scintillators are treated by photomultipliers, which produces a negative shaped pulse. The height of the pulse is proportional to the number of photons received and lasts around 175 ns. The electronic signal already digitized is saved into an on-detector memory waiting for the accept signal from the LVL1 trigger. For each Tile Calorimeter module (in a total of 256 modules), there is a drawer with up to 48 photomultipliers and all the readout electronics inserted in the back of the calorimeter structure.

The shaped analog pulses from the calorimeter cells are readout by the front-end electronics for digitalization. They are also summed up by detector regions. A coarse granularity version of the calorimeter output is provided in analog mode to the hardware LVL1 processing. These coarse granularity units are called Trigger Towers (TT). Except for the very forward regions, the TT size is 0.1×0.1 in $\eta \times \phi$. This seed is used to open a region (usually defined in terms of TT coordinates) which is used by the LVL2 algorithms. In these regions called Regions of Interest (RoI) the full detector granularity is used by the reconstruction algorithms.

The LVL1 hardware algorithm uses some minimal TT energy and isolation quantities to define a possible egamma candidate. A pointing to the found candidate $\eta \times \phi$ position is sent as a seed for software trigger processing.

2 Data preparation

From a general point of view the data preparation for the LAr and Tile Calorimeters is similar. Figure 2 depicts the global scope of the Algorithm and Data Preparation for the LVL2 calorimeter algorithm. The extra details of the Event Filter data preparation will be detailed later [11].

The LVL2 steering receives the LVL1 information on the acceptance of an event with the RoI $\eta \times \phi$ coordinates. The algorithm then gathers a list of ROB identifiers which contain data for a given RoI. Each ROB may contain data from Trigger Towers not pertaining to the RoI (ROB data access is not usually defined by the RoI, but rather by the hardware cabling). Hence, an optimal way to map cells to the towers and to the addresses of the ROB must be provided.

These mappings of the ROB and Trigger Towers are part of the geometry description and are also used in an offline context. Detector Description databases are used to translate any physical position into a set of identifiers. These tools are typically very slow, as the description is comprised of a great amount of data. In order to have a faster access, compatible with the LVL2 speed requirements, a look-up table

identifiers are translated

are subsequently requested.

is prepared in the initialization of the algorithm. This table is called the Region Selector.

The ROB addresses are input to the ROB Data Provider Service. This translates the ROB addresses to network addresses of the ROB machines and sequentially requests the data. The algorithm processing is blocked while the network acquires the data. Studies on timing measurements were performed [6, 12] indicating the advantages of a multiprocessing environment, where a process could run while the processor is waiting for incoming network data to another process.

indicated
are

When data is received, pointers to the beginning of the different fragments are made available to the data preparation algorithm. Such pointers are passed to the detector specific code (LAr or Tile ByteStream conversion codes) which interprets the data format coming from the detector RODs and converts it into an easy to use format (calorimeter cells) for the algorithms.

The last step of Data Preparation for the LVL2 trigger pervades all the rest of the algorithms processing. This is the data providing to the algorithm which must optimize the calorimeter cell access.

Awkward sentence!

2.1 Data Processing in the Read-Out Drivers

The LAr DSPs are able to prepare data in different formats, the most important one being the physics mode. In this mode the DSPs process the nominal 5 samples per cell provided by the front-end electronics. These samples are used to compute the energy deposited in the cell by the particles using an optimal filtering (OF) algorithm [9]. This processing is a simple weighted sum of the samples that include the electronics calibration constants and describes the relation between the charge deposited in the cell and the height of the shaped signal. Also, a normalization factor that converts ADC counts to MeV is included. A different set of OF coefficients allows for the determination of the time at which the energy deposit occurred in each cell. This is only used for cells with energy above a programmable threshold. Also, for such cells, the RODs also provide a quality factor that reflects the difference between the actual samples and their predicted values calculated from the previously estimated pulse shape (chi-square of the fit). Finally, for each cell, the electronic gain applied to the analog signal in the FEB is transmitted to the acquisition system.

noise auto-correlation and

the timing of the signal and the quality of pulse shaped compared to the expectation are calculated

TT already defined.

Beyond the cell based data, the DSP can also extract global information at an FEB or Trigger Tower (TT) level, which can be used to improve the trigger LVL2 and EF processing speed. The DSP sums up the Ex, Ey and Ez of each cell using cell position based projection coefficients loaded in the DSP from the online database. These quantities can be used to compute jets (at the LVL2) or missing E_T at the LVL2/EF when unpacking the full set of cells data is too time consuming. Providing Ex, Ey, Ez at the TT level is under evaluation to improve jet (LVL2) and missing E_T (LVL2 and EF) resolutions.

The pulses from the Tile Calorimeter photomultipliers are also sampled and digitized at 40 MHz by 10-bit Analog to Digital Converters (ADCs). During a physics data taking, 7 samples (175ns) of the signal pulses are acquired and transmitted to the RODs. The information received is also processed using DSPs by applying online algorithms such as the Optimal Filtering energy reconstruction [10]. Again different formats are possible, the main one being the online cell energy reconstruction output.

2.2 Region Selector

LAr

ROB identifier?

As mentioned in the previous sections, to optimize the access to the detector description, part of the information is cached into lookup tables. In the Liquid Argon calorimeter case, the information unit to be correlated to the LVL1 position is the Trigger Tower. The $\eta \times \phi$ minimum and maximum of each TT is arranged in a large matrix. Also, the information about the Read-Out Driver address for each TT is included in this table. In the LAr case, table pages corresponding to the different calorimeter layers are available. Also, only for the Liquid Argon case, it is possible that a given TT (in the calorimeter crack region) is served by more than one ROD (one ROD in the Barrel and another in the EMEC). In the Tile Calorimeter case, the geometry information is associated with the calorimeter module identifier

End cap

and, again, the ROB identifiers. In order to fill these tables, detector specific code prepares the table with detector description geometry from conditions databases.

2.3 Data Containers

The data structure of Calorimeter Cell includes a part common to LAr and Tile, and parts specific to LAr and Tile.

~~The basic information unit for the Calorimeter information is the Calorimeter Cell. Some specific implementations for the LAr and Tile Calorimeters are available in the form of LArCells and TileCells.~~ The geometry (cell size and position) information is provided via a Calorimeter Detector Description Element attached to the cells.

In the software these cells are organized in vectors, each one called a collection. The definition of a collection size and the corresponding detector covered area are subdetector dependent. There are LAr and Tile CellCollections. Finally, the Collections are organized in a vector and this vector is called a container.

For the Liquid Argon Calorimeter each ~~LArCellCollection~~ ^{Collection} holds data for a LAr ROD, corresponding to two FEBs or, at most 256 cells. In the case of the Tile cell collection, there are either 23 cells (in the Barrel) or 13 cells (in the Extended Barrel) per Collection. Data for 4 ~~TileCellCollections~~ ^{Collections} are associated to a single ROD. A Tile Calorimeter ROD has data for at most 92 Tile Cells.

The containers for LAr and Tile are stored permanently in memory and the cells and collections are never deleted. This way, we avoid on-the-fly memory allocation, which is a typically slow operation in a computing system. One problem with reusing collections is that the container must keep track of which collections have already been decoded or not in each event. This information is provided by the tools that access the container. Collections already decoded, if needed again, will not be redecoded.

in the same event

2.4 ByteStream Conversion

The ByteStream conversion process begins by providing the ROD fragment (~~via the ROBDataProvider-Svc~~) containing bytes where the energy information per cell is coded to the ByteStream conversion code. Also, based on the ROD fragment identifier, a cell collection pointer is requested to the proper container (LAr or Tile containers). With the ROD fragment pointer and the collection to be filled, the subdetector specific code is used to perform the data unpacking.

2.5 LAr ByteStream Conversion

should not be a separate subsection

The LAr ByteStream conversion code automatically identifies the fragment type using the ROD version marked in the ByteStream itself. Depending on the detected format, the correct internal infrastructure is automatically selected. ~~A general interface of such infrastructure, via a method called getNextEnergy, is used.~~ Different DSP block formats will be processed by different pieces of code under the same interface.

Code

The ~~method getNextEnergy~~ simply unpacks the energy information from the memory block using the format standard as described in 2.1. It then returns the energy of the cell, the channel gain and the time and quality information (if available) for each of the ROD fragment channels. The channel number ~~returned by the method~~ is used as an index to the cell position in the CellCollection, so that each LAr channel is associated to a single ~~LArCell~~ ^{cell} object in the collection. Each ~~LArCell~~ ^{cell} energy can then be set ~~to the decoded energy value.~~ ^{updated with the current values.}

2.6 Data Providing

cells

After the data is prepared, all available ~~LArCells~~ are filled with the energy information to be used in the algorithms. However, a ~~LArCellCollection~~, comprising data from more than 1 FEB, may extend

LAr collection

TT

Therefore

over many Trigger Towers, typically, many more than those composing the RoI. The trigger algorithms, during the shower shape variables calculation, typically require a per layer data access.

All such conditions would produce a code very complex to run, with many branching points to check cell position and layer. ~~However~~, the information required to evaluate these conditions is based on the geometry of the detector and can be prepared before the algorithm runs. So some maps between TT identifiers and groups of cells associated with the given TT are assembled during algorithms initialization. Using the TT identifier list obtained from the Region Selector, a chain of cells for those TTs can be obtained, simplifying the algorithm code.

2.7 Tile Byte Stream Conversion

The Tile Byte Stream conversion also checks the ROD format via a fragment type identifier in the ByteStream. This is used to select which methods have to be used to unpack the data. The selected method will completely decode the bytes and fill the energy values of a pre-allocated raw channel structure. This is again used to avoid online memory allocation. The energy, time and quality are stored together with the ADC identifier for each cell in a Tile Calorimeter drawer. This vector with the raw channel information is passed to a method to copy them into the cells. The mapping of raw channels to cells is the same for every drawer in a given calorimeter face. In order to expedite the processing a mapping is built to the indexes of the cells that correspond to each raw channel.

Each TileCal drawer is unpacked into a TileCellCollection. The collection must be used before accessing the next one (no chaining mechanism). The data providing in this case is much more simple than in the LAr case. ~~Iterators to the begin and end of the collections are provided to the algorithm.~~

Algorithm can iterate through the whole collection.

2.8 Data Preparation in the Event Filter

The Data Preparation tools in the EF make use of the same data unpacking approach that is used by the LVL2. The only particularity is that the calorimeter cells are stored in a ~~CaloCellContainer (a vector of CaloCells)~~. This approach was taken to profit from tools already designed for the offline framework to deal with cells in this format.

There are 4 tools that fill the ~~CaloCellContainer in the TrigCaloRec package~~. They access different detector parts (LAr EM, LAr HEC, LAr FCal and Tile). The reason to split the data unpacking into three different tools is to provide the possibility of skipping, if needed, one of the calorimeter sections.

Once the ~~CaloCellContainer~~ has been filled up with the corresponding calorimeter cells, a set of tools ~~is executed in order~~ to check the container quality, organize it according to the calorimeter section they belong to, and to perform cell based calibrations.

3 Algorithms and Performance

The HLT algorithms are divided in two levels (LVL2 and EF) and each level has two categories : feature extraction (FEX) and hypothesis making (HYPO). The LVL2 egamma reconstruction FEXes, for example, start from the seeded LVL1 RoI and build a cluster object that gathers all the important features for particle identification (e.g. shower shape profile). The HYPO algorithms use those features to reach a decision, normally by performing simple cuts. In the EF case, we will use as an example, the missing E_T algorithm which calculates total vectorial sum of the energy of the calorimeter cells.

Here the feature extraction algorithms are briefly described, as they are presented in more detail together, with the hypothesis algorithms, elsewhere [7].

Whenever possible the LVL2 and EF results are treated separately. The results are based on ByteStream (BS) files prepared with a format similar to the ATLAS raw output data.

data indices

data

data

data

data

NOT clear!

one container for all the cells, designed for the offline reconstruction

are

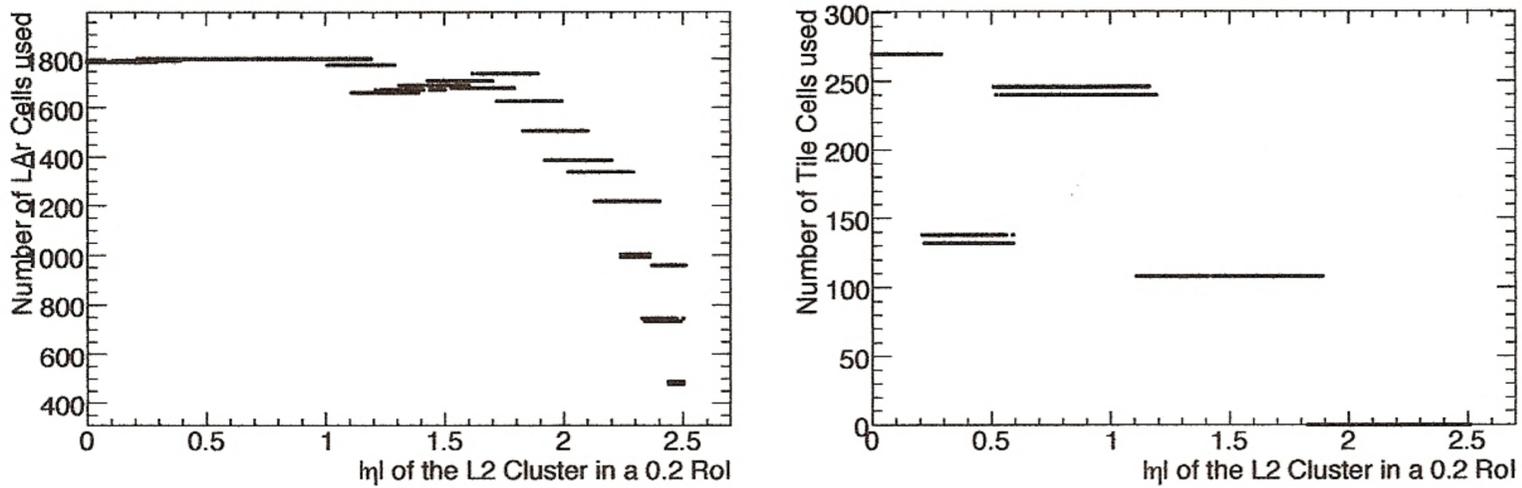


Figure 3: Number of cells used in the T2CaloEgamma Algorithm as a function of η for the LAr (EM, EMEC and HEC) Calorimeters (left) and for the Tile Calorimeter. These distributions were made for the standard LVL2 egamma reconstruction RoI Size (± 0.2).

A detail study of the memory footprint and initialization time was performed. A good fraction of the initialization time is taken by the detector geometry assembling, for example, the filling of the cells coordinates and the Region Selector tables. This accesses databases with detector conditions and possibly files with complementary information.

~~The amount of memory leaked could also be verified for a huge number of LVL2 processed RoIs. For 500K events (about 5-6.7 LHC seconds of LVL2 operation at the LHC), no memory leak was found. We could also measure the memory leak when running LVL2 reconstruction and all the Event Filter algorithms for the Calorimeter. A very small leak of 0.5 bytes per event was found. This is being investigated but it is possibly just caused by some fluctuation on the measure per event (since a different number of cells must be allocated per RoI for the EF).~~

Memory leaks are SW bugs.
I see no point of discussing SW debugging

In order to better understand the processing time performance of the algorithms, it is essential to know how much data is actually necessary. A good way to see that is to check on average how many cells are requested by the algorithms as a function of η . This is shown in Figure 3. We have separated results for LAr (left) and Tile (right) cells. The distribution on the left shows that the barrel number of used from the cells is quite uniform. Since the granularity gets reduced at the EMEC and HEC, less cells have to be unpacked.

in the barrel

The distribution of the number of cells unpacked for the Tile Calorimeter presents a profile determined by the number of drawers to be unpacked. In the central region ($|\eta| < 0.4$), data from $\eta < 0$ ($\eta > 0$) must be accessed to complete the RoI for positive (negative) η . As a consequence, the number of drawers to be unpacked is doubled. A similar effect happens in the region between the TileCal Barrel and Extended Barrel.

For the standard T2CaloEgamma RoI (± 0.2), the algorithm execute time interval for each of its tools could be measured. The results are shown in Figure 4 for the EM tools (left - only LAr access) and for the hadronic tool (Tile and HEC access). More than 15 thousand events were used to make these plots. For the EM part, it can be seen, that even though in the calorimeter crack region (around $\eta = 1.4$), fewer cells are used by the algorithm (see Figure 3), these cells are distributed in more than one ROB (1 ROB from the Barrel and another from the EMEC). This results in a processing time overall increase.

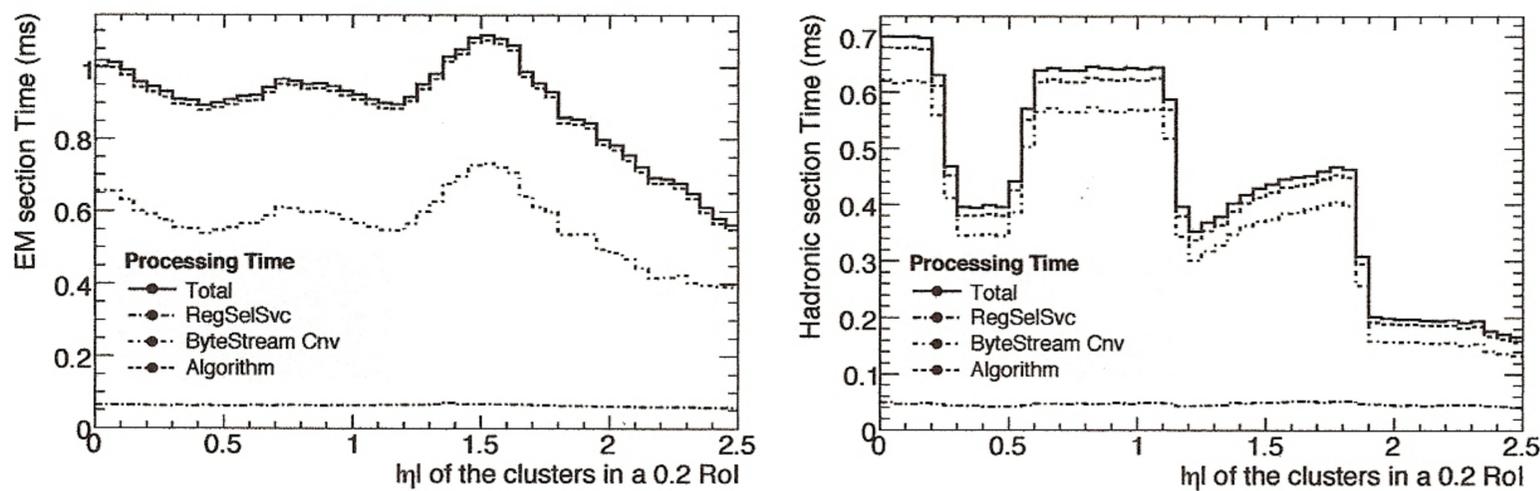
L2 egamma (is this 0.2x0.2 ?? define 0.2 RoI somewhere)

can

1.5

As can be seen in the Figure, for the EM case, the ByteStream conversion makes up a large fraction of the total processing time (about 64%), the rest being used by the algorithm. For the hadronic part, this proportion is much worse (about 90%). The conversion times are especially slow at the Tile Calorimeter regions and in proportion to the number of Tile Calorimeter modules accessed. Timing optimizations are

Why is the Total \neq Sum?



LVL2 egamma

Figure 4: Time spent in the different phases of the ~~T2Calo~~ ^{LVL2 egamma} algorithms as a function of η for the Electromagnetic part (~~EM, EMEC~~) left) of the algorithm and for the Hadronic part (~~HEC, Tile~~) right).

Reco. step	Region Selector	ByteStream Cnv	Algorithm	Total
EgammaSamp2	29 μ s	169 μ s	146 μ s	347 μ s
EgammaSamp1	13 μ s	171 μ s	113 μ s	301 μ s
EgammaEmEn	21 μ s	158 μ s	56 μ s	243 μ s
EgammaHadEn	46 μ s	334 μ s	43 μ s	438 μ s
Total	109 μ s (8.2%)	833 μ s (62.6%)	358 μ s (26.9%)	1.33 ms

Table 1: Processing time for different algorithm steps and for different actions. Improvements for the Tile Cal Data preparation should be envisaged. Time measures ~~not considering~~ ^{Time} ROB data retrieval ~~time~~ ^{ments excludes}.

in progress for this section. The results are also summarized in the Table 1. As in the figure, ROB data fetching times are not included. ROB retrieval times can only be evaluated during ~~the cosmic ray runs~~ ^{real data taking}.

These time measurements confirm the general idea that the TileCal data preparation needs some improvement. Even though, for the barrel region, 6 times fewer cells are accessed from TileCal, the time to run the hadronic tool is comparable (a proportion about 0.9 to 0.4 at $\eta = 0.5$ - choosing a very good case for TileCal) to all EM tools.

Other algorithms like tau or jets need larger RoI. ~~As~~ ^{As} an example, an algorithm using a 1.0×1.0 RoI takes around 10-12 ms.

3.1 Missing E_T in the Event Filter

The Missing E_T EF FEX algorithm, ~~called TrigEFMissingET~~, accesses data from the full acceptance region of the calorimeters and computes the E_T^{miss} with its (E_x, E_y) components as well as the total scalar energy sum. In addition, corrections to the computed energies due to muons can be taken into account by including the results from the EF muon FEX algorithm. (transverse?)

To access the calorimeter data the algorithm uses the same data preparation layer as described to the LVL2 calorimeter algorithms data preparation. Since the ATLAS calorimeters contain about 200 K calorimeter cells, the access to every single cell can become too time consuming at the trigger level. A faster option is to use the E_x , E_y and E_z energy sums at the FEB level, which was introduced in Section 2.1. So far the FEB-wise energy sums have only been implemented in the LAr data unpacking code, where the impact on the data unpacking time should be most significant.

Zero suppression is used for cells below a noisy threshold. In the ~~Trigger~~ ^e EF MissingET ~~framework~~ ^{algorithm}

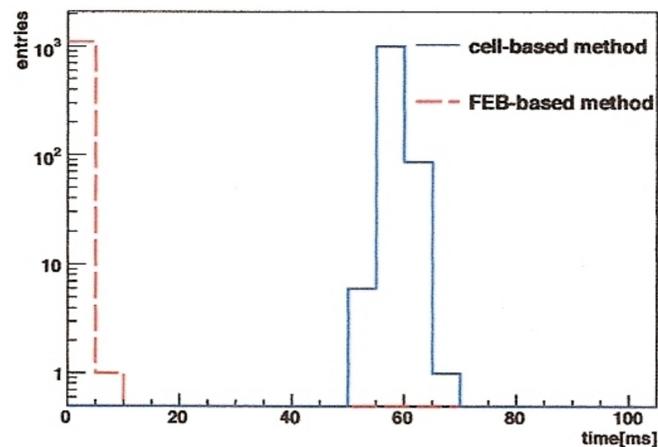


Figure 5: Processing time of the EF MET FEX algorithm. The timing distributions for the cell and FEB method are shown. In average, the time to process the whole calorimeter is 56.6ms for the cell method and 2.36ms to the FEB method (an improvement of 24 times).

two different data ^{event,} unpacking tools can be used. One uses the full cell granularity and unpacks every cells in the ~~ByteStream~~. A faster option uses the FEB information (Energy sums per FEB) from the LAR calorimeter and the cell granularity for the Tile calorimeter. It is important to mention that since this is an EF algorithm, all the event available data is already fetched by the Event Building step, so, no extra network delays are introduced as would be in the LVL2 case. *stick to the same notation*

In Figure 5 the processing time of the EF MET FEX algorithm using cells unpacking is shown. The total processing time peaks at around 57 ms. With the same setup the EF MET processing time was also measured using the FEB information unpacking shown in Figure 5. The distribution of the total processing time in this configuration has a peak at around 2 ms. The processing is almost 24 times faster than that obtained with the cells unpacking.

We also compared the Missing E_T computation and the total scalar sum for the cells unpacking and the FEB unpacking. The missing E_T calculation shows a similar profile whilst the scalar sum suffers from cell noise level definition in the zero suppression part of the algorithm. Due to the advantages in this FEB-quantities based approach, these algorithms are included as an option for the Trigger EF missing E_T algorithms.

4 Summary and Conclusions

from This note described the implementation of the whole data preparation step for the calorimeter trigger ~~since~~ the detector electronics up to the reconstruction algorithms. The High-Level Trigger Calorimeter tools described here have been used in the whole physics studying phase of the ATLAS trigger. An unique interface provides access to detector physics quantities (calorimeter cells) obtained with complex computations from the readout data. Knowledge on the detector details is, of course, fundamental to determine the optimal strategy to be followed in this unpacking procedure. The emphasis of the design approach ~~taken~~ was to satisfy the important LHC processing time allowances. They were plainly satisfied as stated in this work. Even for special algorithms, like the missing E_T which process cells from the whole detector, the data preparation performance is still below the required processing interval restrictions. Specially when optimized solutions, like the usage of Front-End boards summary information is used instead of full detector unpacking. More optimizations are, anyway, still undergoing. *bad worded sentence.*

Recently, tests have been performed to determine the sensitivity of the data preparation layer to failures in reading parts of the detector or corruption caused in the data packages received. Most of the

Whenever FEB summary information can be used, significant improvement can be achieved.⁹

Remove this paragraph.

issues were solved and more testings are now happening.

Also, all the described tools and algorithms have been used during the ATLAS commissioning data taking with cosmic rays. For the moment, this is the only exercise that can emulate the real trigger usage in LHC conditions. Many trigger slices like taus, jets and missing E_T are being successfully explored this way, providing feedback to further improvements of the algorithm developers. This also allow for a detailed study of the data fetching time at the LVL2, which may be a dominant factor for the trigger functioning.

References

- [1] ATLAS Collaboration, Atlas Detector and Physics Performance Technical Design Report, ~~Internal report, CERN, 1999.~~ CERN/LHCC 99-15
- [2] ATLAS Collaboration, Atlas Liquid Argon Calorimeter Technical Design Report, Internal report, CERN, 1999.
- [3] ATLAS Collaboration, Atlas Tile Calorimeter Technical Design Report, Internal report, CERN, 1994.
- [4] ATLAS Collaboration, Muon Spectrometer Technical Design Report, Internal report, CERN, 1997.
- [5] ATLAS Collaboration, Atlas First Level Trigger Technical Design Report, Internal report, CERN, 1998.
- [6] ATLAS Collaboration, Atlas HLT, DAQ and DCS Technical Design Report, Internal report, CERN, 2002.
- [7] ATLAS Collaboration, CERN/LHCC XX (1900).
- [8] ATLAS Collaboration, ~~Atlas Detector Paper, Internal report, CERN, 1900.~~ 'The ATLAS Experiment at the CERN Large Hadron Collider'
- [9] W.E. Cleland, E.G. Stern, Nucl. Instrum. Methods A338 (1994) 467-497. Submitted to JINST
- [10] E. Fullana *et al*, IEEE Trans. Nucl. Sci. 53:4 (2006) 2139-2143.
- [11] D.O. Damazio on behalf of the ATLAS High-Level Egamma Trigger Calorimeter, ICATPP Conference (2007).
- [12] Andre dos Anjos *et al*, IEEE Trans. Nucl. Sci. 51 (2004).
- [13] ATLAS Collaboration, Atlas HLT, DAQ and DCS Technical Design Report, Internal report, CERN, 2002.

get the CERN ID # for the Notes.