Compute farm software for ATLAS IBL calibration

M Bindi[1], T Flick[2], J Grosse-Knetter[3], T Heim[2], S-C Hsu[5], M Kretz[4], A Kugel[4], M Marx[5], P Morettini[6], K Potamianos[7] and Y Takubo[8]

[1] Dipartimento di Fisica e Astronomia, Università di Bologna, Bologna, Italy
[2] Fachbereich C Physik, Bergische Universität Wuppertal, Wuppertal, Germany
[3] II Physikalisches Institut, Georg-August-Universität, Göttingen, Germany
[4] ZITI Institut für technische Informatik, Ruprecht-Karls-Universität Heidelberg, Mannheim, Germany
[5] Department of Physics, University of Washington, Seattle WA, United States of America
[6] INFN Sezione di Genova, Genova, Italy
[7] Physics Division, Lawrence Berkeley National Laboratory and University of California, Berkeley CA, United States of America
[8] KEK, High Energy Accelerator Research Organization, Tsukuba, Japan

E-mail: `moritz.kretz@cern.ch`

Abstract. In 2014 the Insertable B-Layer (IBL) will extend the existing Pixel Detector of the ATLAS experiment at CERN by over 12 million additional pixels. For calibration and monitoring purposes, occupancy and time-over-threshold data are being histogrammed in the read-out hardware. Further processing of the histograms happens on commodity hardware, which not only requires the fast transfer of histogram data from the read-out hardware to the computing farm via Ethernet, but also the integration of the software and hardware into the already existing data-acquisition and calibration framework (TDAQ and PixelDAQ) of the ATLAS experiment and the current Pixel Detector.

We implement the software running on the compute cluster with an emphasis on modularity, allowing for flexible adjustment of the infrastructure and a good scalability with respect to the number of network interfaces, available CPU cores, and deployed machines. By using a modular design we are able to not only employ CPU-based fitting algorithms, but also have the possibility to take advantage of the performance offered by a GPU-based approach to fitting.

1. Introduction

The Insertable-B-Layer (IBL, [1]) will add an additional layer of over 12 million pixels to the already existing ATLAS Pixel Detector. The front-end chips (FE-I4s, [2]) will be connected to newly designed readout hardware that takes advantage of recent enhancements in hardware components. While the basic architecture with pairs of Back-Of-Crate (BOC, [3]) and Read-Out-Driver (ROD, [4]) cards within a VME crate remains unchanged ([5]), the task of performing computationally demanding fits on calibration histograms, that are gathered in the RODs, has been shifted from ROD DSPs ([6]) to an external compute farm (FitFarm) to allow for greater flexibility.

Gigabit Ethernet interfaces enable the two Slave FPGAs on a ROD to bypass the slow VME-based communication and send out the histogram data via TCP/IP to the FitFarm machines. The FitFarm software reformats the incoming data streams and starts the fitting procedure when the first valid fraction of data has arrived. Once a scan is completed, it publishes the results for further use by the TDAQ infrastructure.

2. Existing DAQ Software

There are two major software frameworks used for the IBL and FitFarm development: TDAQ and IBLDAQ.

The TDAQ framework provides an environment for running distributed software with IPC, an information service (IS) to share information between applications, and a process manager (PMG), that controls and monitors remotely spawned processes. IBLDAQ is a fork of the PixelDAQ framework and the active software development happens here. It contains features specific to the Pixel Detector and IBL, such as a scan and calibration engine. It is foreseen to be merged with PixelDAQ.

3. FitFarm Design Considerations

For software as well as hardware specifications it is important to note the beneficial granularity of the problem set and the hardware interfaces:

3.1. Fitting Problems

Each pixel histogram poses an independent fitting problem and can therefore be processed in parallel independently from other histograms. As histogramming steps will oftentimes only be performed on a fraction of pixels (mask-stepping), the fitting procedures can already start as soon as the first part of valid data, belonging to a subset of all frontend pixels, has been transferred to a FitFarm machine, even before the scan has been completed. This process is illustrated in Figure 1.

3.2. Network Transfer

Each ROD features two slave FPGAs, each with one Gigabit Ethernet interface. Preliminary measurements ([7, p. 41ff]) have shown that the software based IP stack running on the slave FPGAs is able to deliver about 200 MBit/s. This yields a total output rate of 6 GBit/s for all 15 RODs (one ROD belonging to the Diamond Beam Monitor — DBM — telescopes), a rate

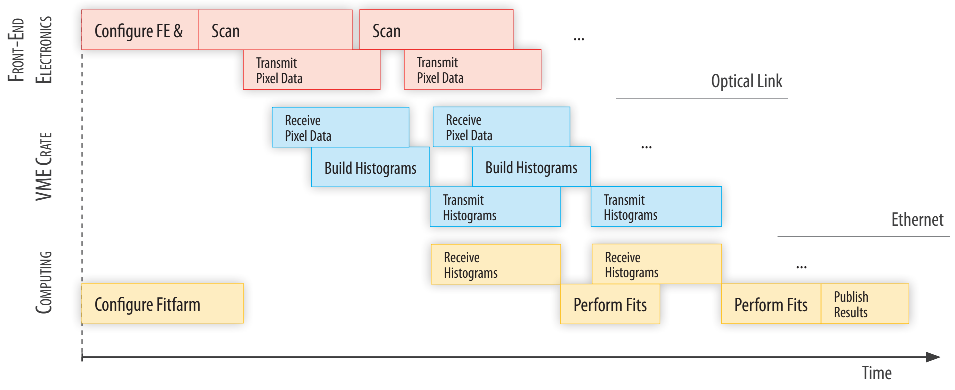

Figure 1. The steps involved in a calibration scan that are being performed on the different parts of the system. The front-end electronics are connected via optical links to the readout hardware that is located in a VME crate. Histogram data is being sent out from the Readout-Drivers (RODs) to the FitFarm via a Gigabit Ethernet connection and the fitting starts as soon as the first valid data is available.

that can easily be handled by current hardware. Different approaches to effectively parallelizing the fitting problems (CPU and GPU-based) have already been investigated ([8], [7, p. 57ff]) and proven fruitful. As there are 60 TCP connections originating from the slaves, the number of FitFarm computing nodes is very flexible and can easily be scaled up. The current design aims for 1-4 nodes in the final setup. Figure 2 depicts the involved hardware components. All of the ROD-BOC pairs for the IBL are located in a single VME crate. Network connectivity between slaves and FitFarm machines will most likely be realized via a separate network segment, allowing for a possible use of jumbo ethernet frames.

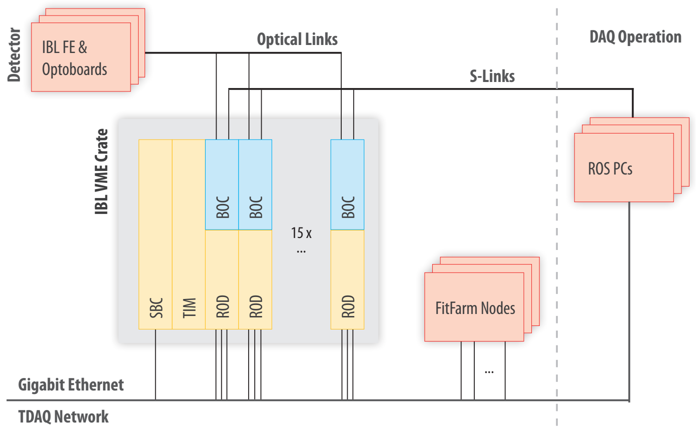

Figure 2. Hardware architecture of the off-detector electronics and computing components. The 15 ROD-BOC pairs will fit in one VME crate. ROD slave FPGAs and FitFarm computing nodes will be connected via Gigabit Ethernet connections.

4. FitFarm Software

The architecture of the FitFarm software reflects two main goals - flexibility and scalability. Figure 3 gives an overview of the planned software structure of a FitFarm node. A FitController component receives information about the scan that is to be performed. Networking threads are being created and configured to receive data from the RODs. In the figure a design for a standalone application, that is controlled by the Process Manager, is shown, but an integration of the components into an already existing application (PixActionsServer) is feasible. The FitServer processes perform the actual fitting routines and report the results. They can easily be substituted to take advantage of different fitting methods (ROOT [9], ported DSP code from the current Pixel Detector RODs, GPU-based approach, etc.). The results are then being published to a database for further analysis or will be used directly for steering more advanced tuning procedures.

5. Development Approach

In order to simplify and speed up the development of the FitFarm software we use a standalone tool that can simulate the network output generated by the slave FPGAs. Setting up a full readout chain would not only require to have the involved hardware (front-end chips, VME crate with ROD and BOC cards, etc.) available, but also a working DAQ software, which is currently under heavy development. The tool generates fake threshold scan histograms that can be used for realistic tests of the FitFarm fitting routines.

As the FitFarm software will run in a distributed environment, we make use of virtual machines that are based on SLC5 and run TDAQ/IBLDAQ. This allows a fast and convenient deployment of additional FitFarm nodes for testing functionality and debugging. For assessing the network as well as the fitting performance we will need to run measurements on dedicated hardware, though.

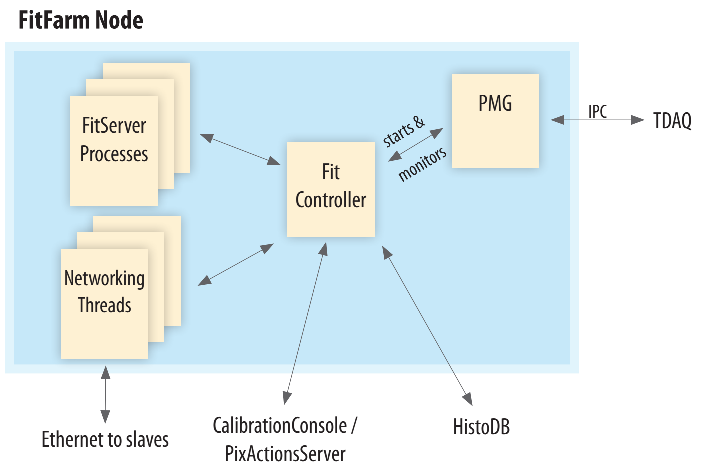

Figure 3. Design of the software running on a FitFarm compute node. A FitController module coordinates the fitting of the incoming histogram data and assigns work packages to FitServer processes.

6. Current Status & Outlook

Currently the FitFarm software supports one out of the four readout modes provided by the histogramming units, that is being used during threshold scans. It publishes fitted threshold scan histograms to the TDAQ online histogramming service (OH). For fitting the histograms ROOT or the lmfit library can be used, resulting in fit times of 15 μs/pixel with lmfit on a machine with dual Intel E5620 CPUs. A modified implementation of the DSP fitting algorithm in CUDA yielded fit times of up to 2.3 μs/pixel on an Nvidia GTX480 GPU.

Future work will focus on a tighter integration of FitFarm into the IBLDAQ framework and the support of all available readout modes. By making use of multiple threads, the FitFarm will be able to serve more than one histogramming unit at a time. For assessing the needed hardware of the final setup, benchmarks of the networking throughput with the emulator software as well as the real slaves have to be undertaken.

References

[1] M. Capeans, G. Darbo, K. Einsweiler, et al. ATLAS Insertable B-Layer Technical Design Report. Technical Report CERN-LHCC-2010-013. ATLAS-TDR-019. CERN, Geneva, September 2010.
[2] M. Barbero et al. The FE-I4 Pixel Readout Chip and the IBL-Module. ATL-UPGRADE-PROC-2012-001. Proceedings of Science. 2012.
[3] J. Dopke, D. Falchieri, T. Flick, et al. The IBL Readout System. ATL-INDET-PROC-2010-031. October 2010.
[4] D. Falchieri, G. Bruni, M. Bruschi, et al. Proposal for a Readout Driver Card for the ATLAS Insertable B-Layer. ATL-INDET-PROC-2010-039. November 2010.
[5] A. Gabrielli, G. Bruni, M. Bruschi, et al. ATLAS IBL: Integration of new HW/SW Readout Features for the Additional Layer of Pixels. September 2010.
[6] J. Biesiada, J. Dopke, H. Gray, et al. The Implementation and Performance of ROD DSP Software in the ATLAS Pixel Detector. ATL-INDET-INT-2009-006. February 2010.
[7] M. Kretz. Studies Concerning the ATLAS IBL Calibration Architecture. CERN-THESIS-2012-67. June 2012.
[8] J. Dopke, D. Falchieri, T. Flick, et al. Study of FPGA and GPU Based Pixel Calibration for ATLAS IBL. May 2010.
[9] R. Brun and F. Rademakers. ROOT - An Object Oriented Data Analysis Framework. In AIHENP96 Workshop, Lausane, volume 389, pp. 8186. 1996.