Compute Farm Software for ATLAS IBL Calibration

Marcello Bindi[1], Tobias Flick[2], Jörn Grosse-Knetter[3], Timon Heim[2], *Moritz Kretz*[4], Andreas Kugel[4], Marilyn Marx[5], Paolo Morettini[6], Karolos Potamianos[7], Yosuke Takubo[8]

[1]University of Bologna and INFN (IT), [2]Bergische Universität Wuppertal (DE), [3]Georg-August-Universität Göttingen (DE), [4]Ruprecht-Karls-Universität Heidelberg (DE), [5]University of Washington (US), [6]INFN Genova, [7]Lawrence Berkeley National Lab. (US), [8]High Energy Accelerator Research Organization (JP)

Introduction

The Insertable-B-Layer (IBL, [1]) will add an additional layer of 12 million pixels to the already existing ATLAS Pixel Detector. The front-end chips (FE-I4s) 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) and Read-Out-Driver (ROD, [3]) cards within a VME crate remains unchanged, the task of performing computationally demanding fits on calibration histograms, that are gathered in the RODs, has been shifted from ROD DSPs to an external compute farm (FitFarm) to allow for a 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 stream 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.

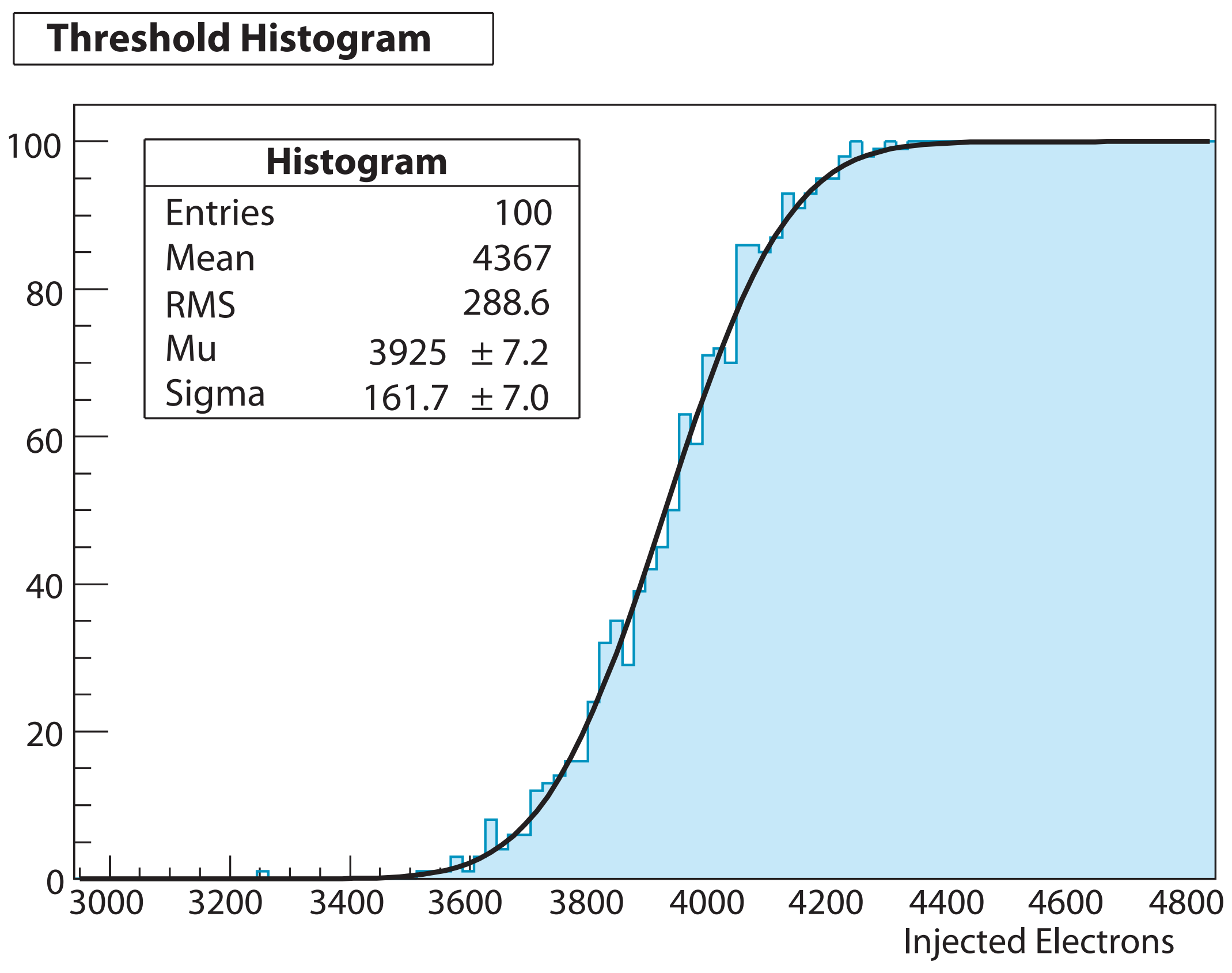


Figure 1: Simulated threshold scan histogram of one pixel and the according fit of an S-curve type function together with the fitting results for threshold and noise parameters Mu and Sigma. Such a fit has to be performed 12 million times for a threshold scan in IBL.

Software Overview

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 again merged with PixelDAQ.

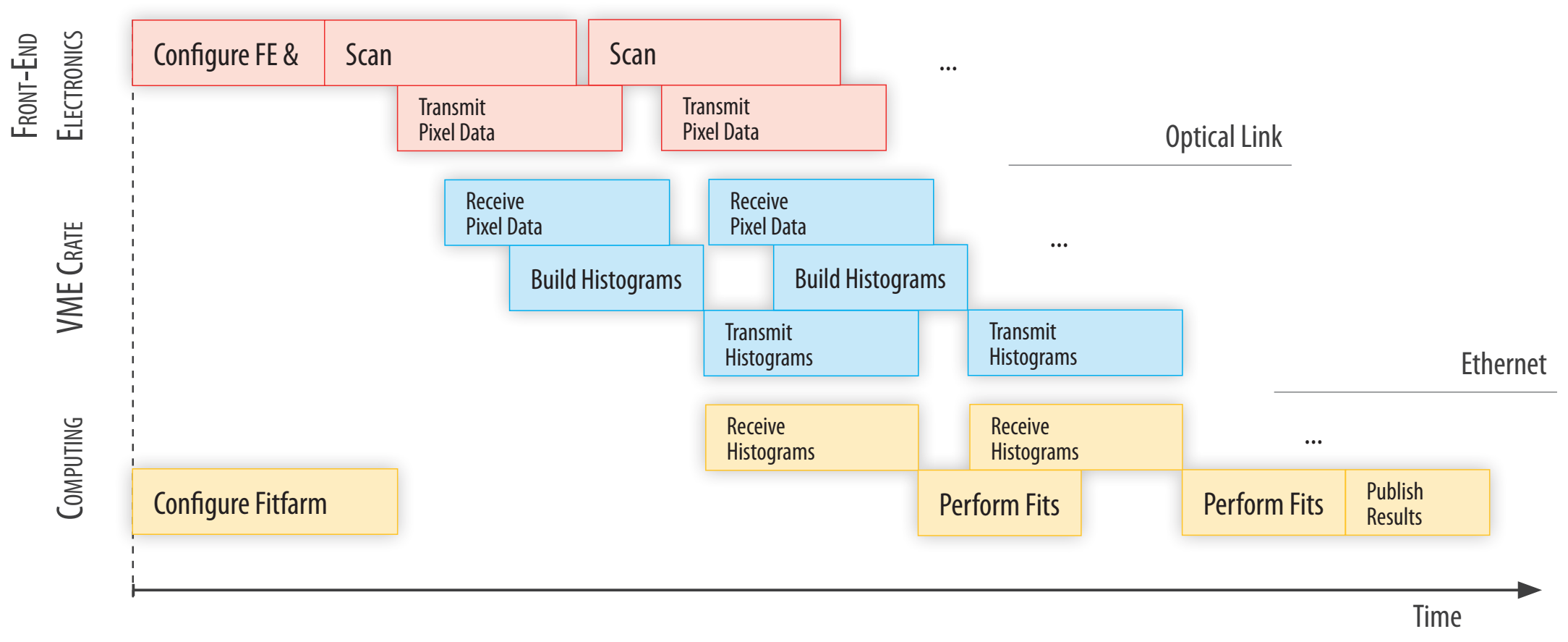


Figure 2: The steps involved in a calibration scan that are being performed on the different parts of the system. The front-end electronics are connected via optical links to the readout hardware that is located in a VME crate. Histogram data is being sent out from the Readout-Drivers (RODs) to the FitFarm via a Gigabit Ethernet connection and the fitting starts as soon as the first valid data is available.

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:

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 has been transferred to a FitFarm machine. This process is illustrated in **Figure 2**.

2. Network Transfer

Each ROD features two slave FPGAs with one Gigabit Ethernet interfaces. As preliminary measurements [6] showed 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, a rate that can easily be handled by current hardware.

ROD Slaves

- 2 Xilinx Spartan-6 devices (XC6SLX150-FGG900) per ROD.
- 2 histogramming units per slave, each serving 8 front-ends.
- 4 different histogramming modes for online and offline operations.
- SSRAM and DDR2-RAM for histogramming and buffering.
- Software based TCP/IP stack (LwIP) running on a soft-core CPU (MicroBlaze).

Different approaches to effectively parallelizing the fitting problems (CPU and GPU based) have already been investigated ([5], [6]) 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 (we aim for 1-4 nodes in the final setup).

Figure 3 depicts the involved hardware components. All of the ROD-BOC pairs for 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.

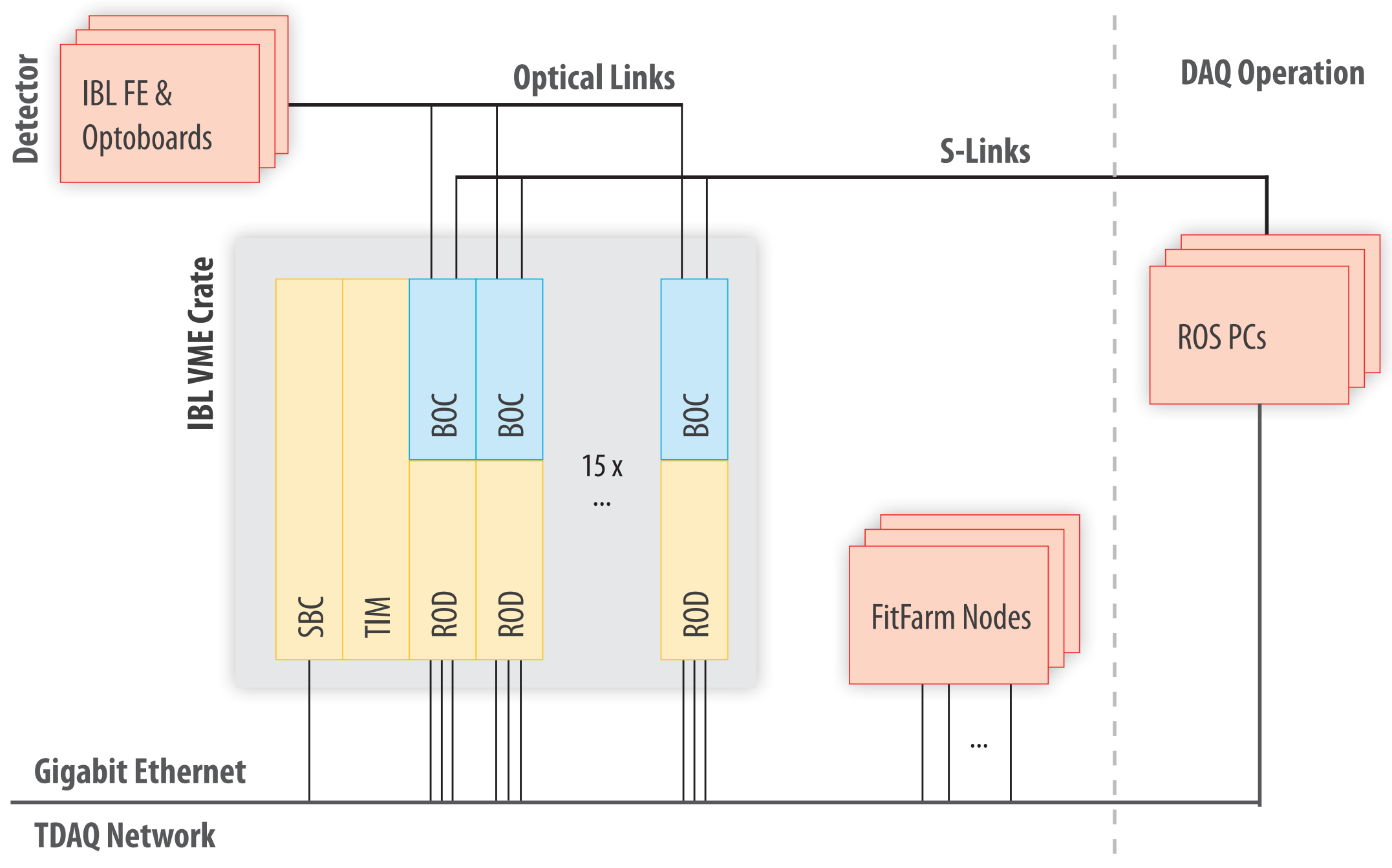


Figure 3: Hardware architecture of the off-detector electronics and computing components. The 15 ROD-BOC pairs will fit in one VME crate. ROD slave FPGAs and FitFarm computing nodes will be connected via Gigabit Ethernet connections.

FitFarm Software

The architecture of the FitFarm software reflects two main goals - flexibility and scalability. **Figure 4** 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, 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.

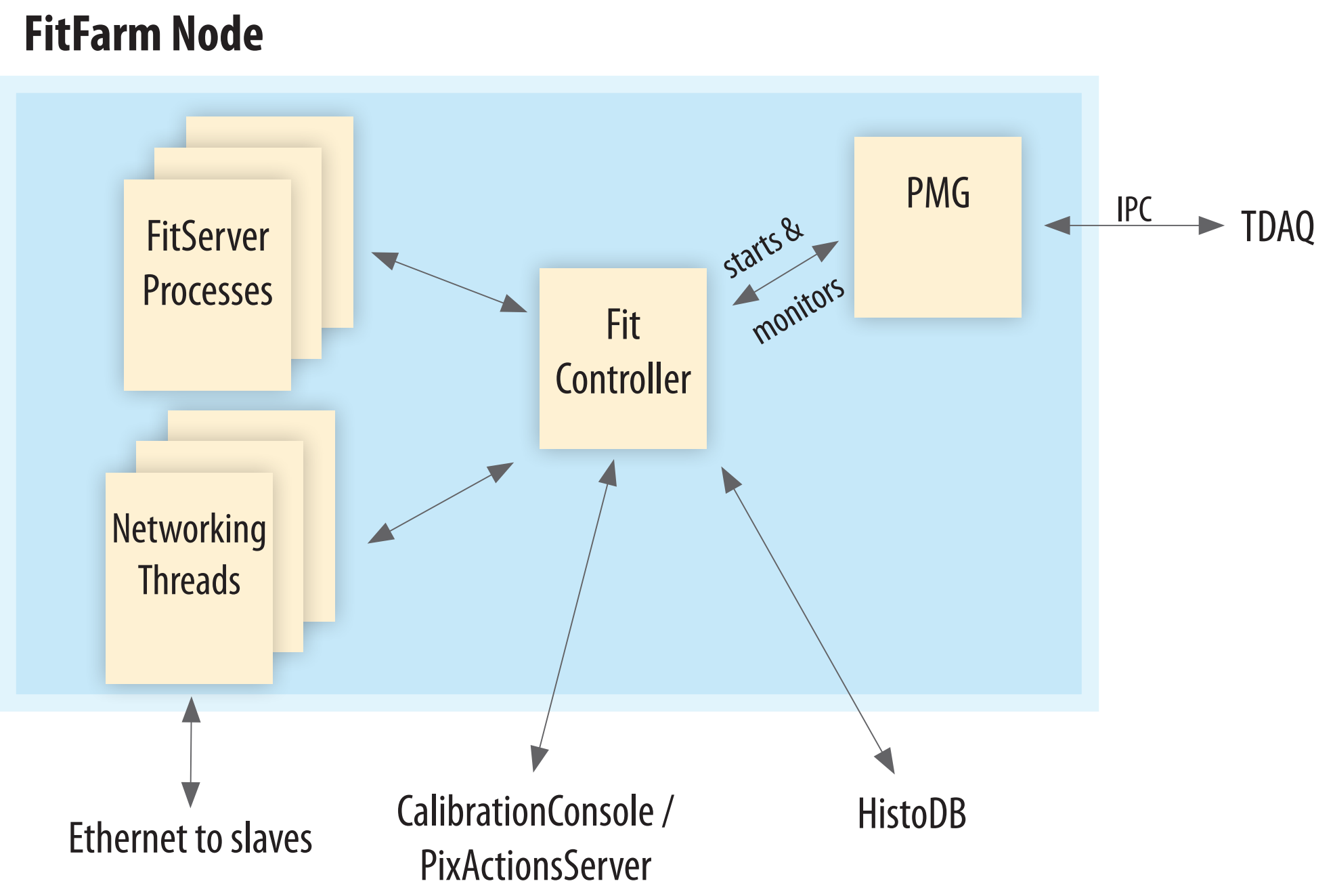
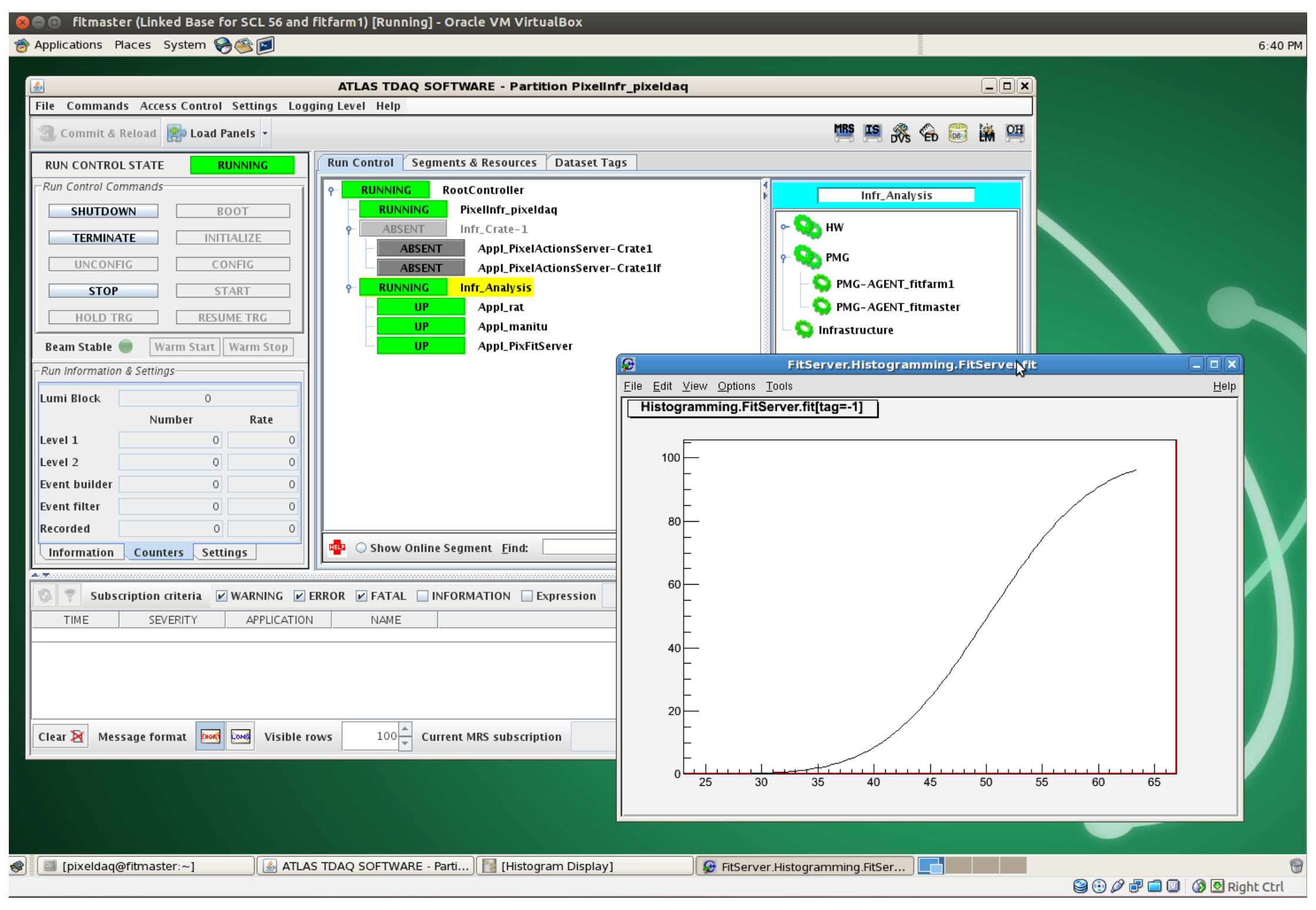


Figure 4: Design of the software running on a FitFarm compute node. A FitController module coordinates the fitting of the incoming histogram data and assigns work packages to FitServer processes.



RUN CONTROL STATE RUNNING

Beam Stable

	Number	Rate
Lumi Block	0	
Level 1	0	0
Level 2	0	0
Event builder	0	0
Event filter	0	0
Recorded	0	0

```
RUNNING  RootController
  RUNNING  PixelInfr_pixeldaq
    ABSENT  Infr_Crate-1
      ABSENT  Appl_PixelActionsServer-Crate1
      ABSENT  Appl_PixelActionsServer-Crate1lf
    RUNNING  Infr_Analysis
      UP  Appl_rat
      UP  Appl_manitu
      UP  Appl_PixFitServer
```

```
Infr_Analysis
  HW
  PMG
    PMG-AGENT_fitfarm1
    PMG-AGENT_fitmaster
  Infrastructure
```

☑ WARNING ☑ ERROR ☑ FATAL □ INFORMATION □ Expression

Visible rows 100

Figure 5: The TDAQ application IGUI shows the state of the active partition and the process manager (PMG) agents running on the two virtual machines "fitmaster" and "fitfarm1". PixFitServer is running on machine "fitfarm1".
In the top window the fit result of a threshold scan histogram that was published to OH is shown.

Slave Simulator Software & Testing Environment

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 for 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.

Current Status of the FitFarm Software

- Supports the most common out of the four readout modes provided by the histogramming units.
- Publishes fitted threshold scan histograms to TDAQ online histogramming service (OH).
- Fitting uses ROOT routines; work on other implementations.

Figure 5 shows a FitServer running within a TDAQ partition and a fitting result that has been published to OH.

Outlook

- Tighter integration of FitFarm into IBLDAQ framework.
- Network throughput measurements with both slave simulator and ROD slaves as sources.
- Extending to multiple network and fitting threads.
- Work towards a complete readout chain.
- More advanced "real-world" system tests.

References

[1] M. Capeans, G. Darbo, K. Einsweiller, et al. **ATLAS Insertable B-Layer Technical Design Report.** Technical Report CERN-LHCC-2010-013. ATLAS-TDR-019, CERN, Geneva, September 2010.

[2] J. Dopke, D. Falchieri, T. Flick, et al. **The IBL Readout System.** ATL-INDET-PROC-2010-031, October 2010.

[3] 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.

[4] 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.

[5] J. Dopke, D. Falchieri, T. Flick, et al. **Study of FPGA and GPU Based Pixel Calibration for ATLAS IBL.** May 2010.

[6] M. Kretz. **Studies Concerning the ATLAS IBL Calibration Architecture.** CERN-THESIS-2012-67, June 2012.

For further information contact
moritz.kretz@ziti.uni-heidelberg.de
or visit
http://www.ziti.uni-heidelberg.de

UNIVERSITÄT HEIDELBERG
ZUKUNFT
SEIT 1386