The Bordeaux Data Acquisition System

D. Dumora & D.A. Smith
Centre d'Etudes Nucleaires de Bordeaux-Gradignan

28th November 2005

1 Overview

At the GANIL, GSI, and CERN testbeams the CENBG group acquired data using a VME crate controlled by a Motorola MVME 172 (a 68040 processor). The operating system is Lynx OS, version 5, a posix-based real-time unix [1]. Trigger interuppts are sent to the controller via an Industry Pack daughter card installed on the MVME 172 (an ACTIS CIO-32). CsI photodiode signals were digitized using CAEN V785 32-channel peak-sensing ADC's. We also recorded rates using CAEN scalers & discriminators.

The acquisition code consists of different asynchronous processes: a main program spawns an interuppt handling and readout routine, another routine to read the scalers periodically, and a routine that buffers the acquired events, before broadcasting them as TCP/IP packets.

Another program (the Master) running on a linux PC starts and stops the VME programs, issues commands, and receives the data packets, all via TCP/IP. Online monitoring is performed on the linux PC, using a ROOT-based system.

Our architecture is very modular and offers great flexibility. A large number of widely varying DAQ's can be orchestrated from the linux PC (CELESTE was an example, see below), provided they accept commands and issue data via our simple TCP/IP protocol. Our VME acquisition is not particularly fast – 128 kbytes/second to read an event is typical – but it is probably fast enough for what we want to do at CERN

2 Details

2.1 Hardware & Software

The Bordeaux Acquisition system is based on two very modular codes. The first one, called GlastView, running on a Linux PC, is a ROOT based C++ code dedicated to data display and remote control of the acquisition system. GlastView can be used to control the acquisition system by TCP/IP and display sampled data online or, in offline mode, to display recorded data. The second code called GlastAcq, running on a LynxOS MVME-172 VME controller, is the acquisition code. It can be run in remote mode from GlastView or in a standalone console mode (a VT100 console is connected to the MVME-172). In the second case, the acquisition is done in "blind mode" as no display of recorded data is available.

The network communication between the two systems is reduced to the Start and Stop Run command, configuration and RunId packet at the beginning of a run. During the run, GlastView sends a TCP request to ask for data to display and GlastAcq only answers the request by sending a set of data. The data request rate is managed by GlastView, when a data set is displayed then Glastview asks for another one.

2.2 Trigger & signals

The trigger and data readout is managed by coupling a Dual Timer used in "latch mode" and the Industry Pack CIO32 I/O register as shown in the figure below.

The physical trigger initiates the trigger signal in the Dual Timer module, this signal is used to generate gates for the ADCs, increments scalers and sets a level to the CIO32 "a0" port what starts the readout of all the defined VME Modules. When the readout is complete a level is set by the MVME172 to the "c0" port what resets the Dual Timer and allows a new trigger.

Another input of the CIO32 can be used the manage disk writing, during the CERN 2003 beam test the "a4" port of the CIO32 was connected to the beam signal. The status of this port is monitored to start disk writing during the interspill or to take pedestal data during the interspill.

[1] Now called Lynux, see http://www.lynuxworks.com

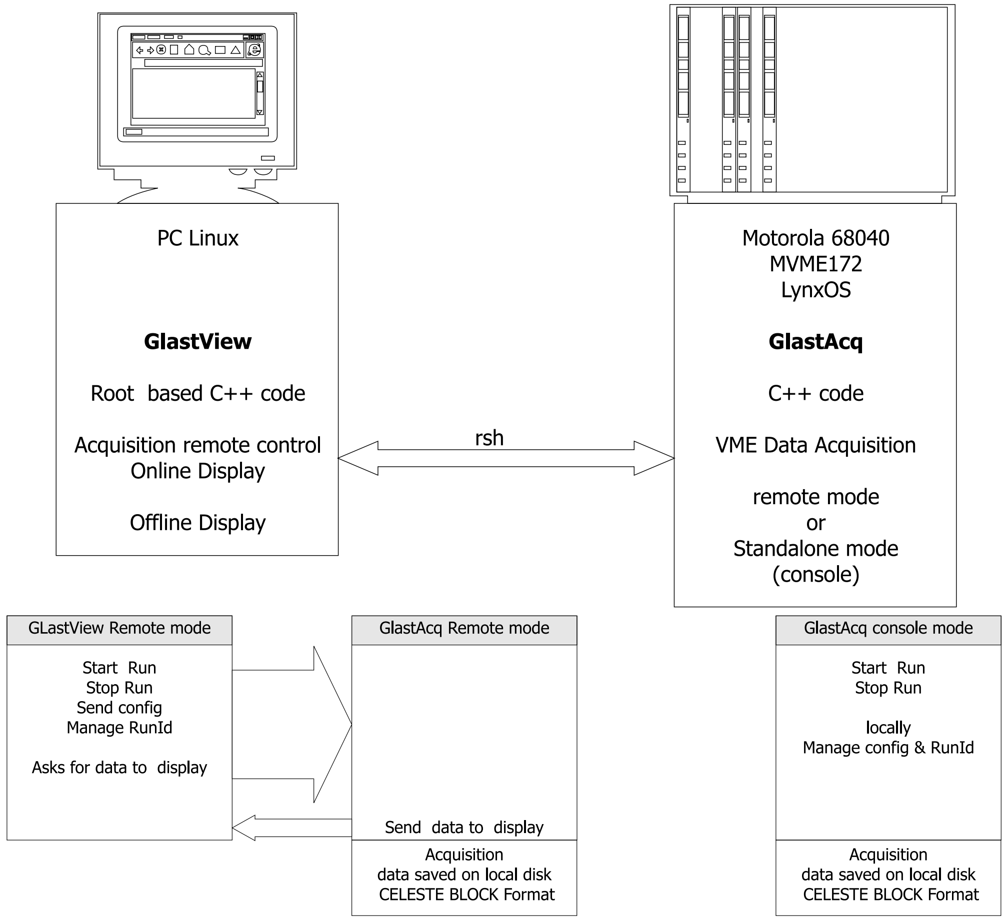

Figure 1: Bordeaux Acquisition system

2.3 Data saving and Format

The data format used for the GANIL,CERN and GSI 2003 is a CELESTE-like Block Format, data are saved to a binary file, each type of data (run header, run configuration, scalers data, ADCs data, ...) is recorded in a specific block. Each type of block has a similar structure, the first eight bytes are used for the Block header, the following bytes are for the data.

Whatever the run type (remote or standalone), data are recorded by the MVME172 on a local scsi-2 disk. In remote mode, only data samples are sent from the Lynx to the Linux PC. During the GSI-2003 beam test, an offline event builder code was used to merge data recorded on different files by the three acquisition systems (EM, FRS, MiniCal).

2.4 GlastAcq on LynxOSX

GlastAcq is a multithread code allowing data acquisition, memory buffer management and TCP/IP communication with GlastView at the same time.

2.4.1 Data reading & VME Module

Each VME Module is described in a class, managing module memory access, configuration and test, and data reading. Data read by each module are saved into the memory buffer managed by the TGlastAcqBuffer class. General VME access is described in the TGlastAcqModule parent class. This modular code structure is very useful for quick development. Adding a known type module (TV 785 for example) is simply done by adding two lines in the configuration file (no need to recompile the acquisition code). Adding a new type of module needs minor developments, the main issue is to code a very short module class describing specific data access, inherited from the general TGLastAcqModule class.

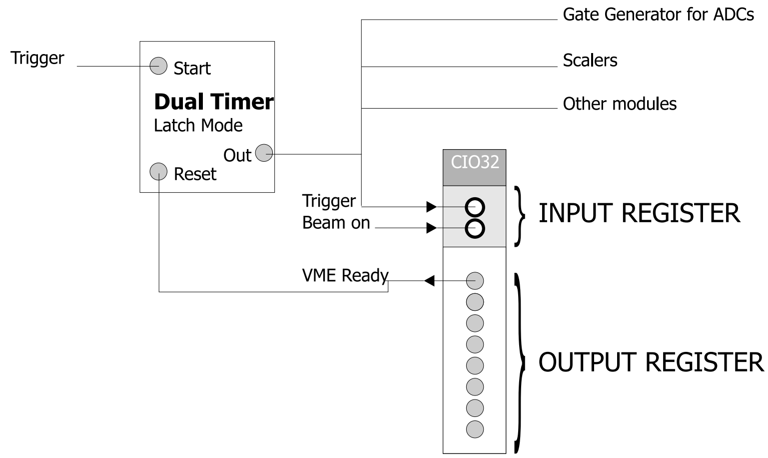

Figure 2: CIO32 as a trigger and disk writing manager

2.4.2 Data buffering & spill management

In order to reduce dead time due to disk access, data are buffered in memory. The buffer management is done in a special class which can control several (two may be enough) memory buffers in FIFO mode. Two threads also manage the communication with GlastView (data sample sending to display) and the recording of full memory buffers. The recording on disk is done when the memory buffer is full or during the beam inter-spill (CIO32 driven).

2.5 Performance

	read out speed
CERN 2003 (7*32 Adc Channels)	Hundreds of Hz
GSI 2003 (4*32 Adc Channels)	better than $\frac{1}{700}\mu sec$

3 History

This VME DAQ was originally developed for the CAT Cherenkov imager by Hugo Delchini at the University of Paris (Jussieu), around 1994. He made the choice of Lynx and a Motorola 68000 processor with the CIO32, and wrote the interuppt and acquisition routines. A group at the Ecole Polytechnique wrote the Master Controller using LabView, at the time on an HP WorkStation. In addition to the VME crate (through which CAT's 3 FastBus crates of ADC's were read out, and through which the phototube high voltages were driven), the Master also controlled two PC's, one for CCD cameras tracking fiducial stars for the telescope pointing, and one for the telescope tracking.

CELESTE adapted CAT's data acquisition and expanded it considerably: we had 4 VME crates, 2 separate high-voltage supplies, and PC's under Windows for PMT current readout, telescope tracking, and weather monitoring. The LabView program on the HP WorkStation was expanded to control all of these. In addition, a linux PC sampled event packets on the fly for online monitoring. It's on this basis that we claim that our system is flexible.

To prepare for the GLAST EM heavy ion beam measurements at GSI, our group built a CsI mini-CAL. We then adapted the CELESTE VME+Lynx system to the mini-CAL needs, and used it at GANIL, again at CERN, and finally at GSI. At GSI the LAT-EM data acquisition and the GSI FRS data acquisition

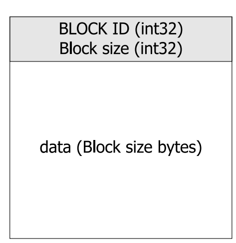

Figure 3: Binary Block Format

were run in parallel. Pains were taken, successfully, to synchronize the two acquisitions. (Using a NIM crate, we imposed a hardware trigger veto after each trigger, for the duration of the longest possible EM readout deadtime, see the note by B. Lott.)

By the time we reached GSI, we had re-written the Lynx VME code in C++ (it had been in c until that time). The re-write included the structure and class definitions described above that have greatly enhanced the modularity and flexibility of the system. Similarly, the LabView code was replaced by GlastView in C++.

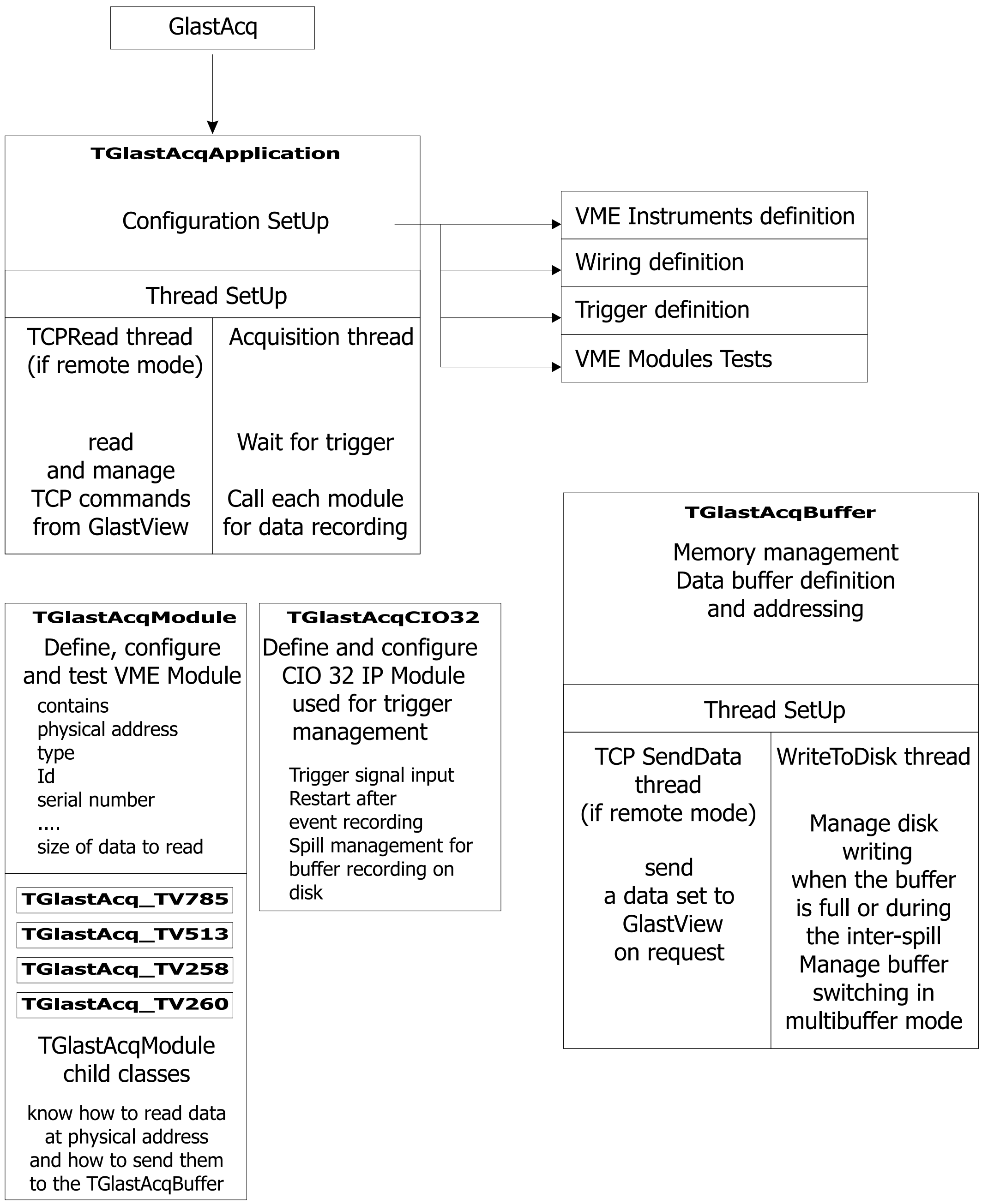

Figure 4: GlastAcq Classes