



Development of a Data Acquisition System for the BABAR CP Violation Experiment

R. Claus¹, P. Grosso², R.T. Hamilton³, M.E. Huffer¹, C.P. O'Grady¹, J.J. Russell¹, I. Scott⁴

¹Stanford Linear Accelerator Center, Stanford, CA 94309, USA

²INFN Torino, Italy

³The University of Iowa, Department of Physics and Astronomy, Iowa City, IA 52242, USA

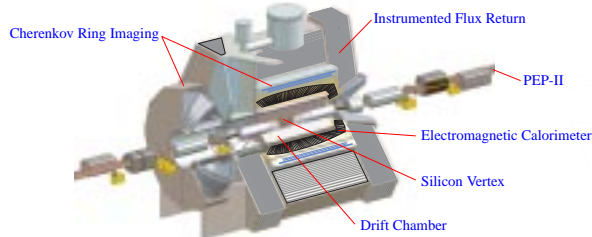
⁴Royal Holloway & Bedford New College, University of London, Egham, Surrey, TW20 0EX, UK



Abstract

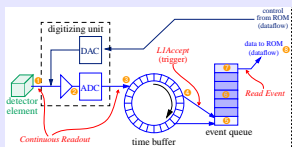
Experiences developing a data acquisition system for the BABAR CP violation experiment located at the Stanford Linear Accelerator Center are presented. The BABAR detector consists of multiple independent subdetectors joined with a data acquisition system consisting of a large number of embedded PowerPC single board computers residing in VME crates. The data acquisition software is layered on the VxWorks real time operating system. It is partitionable to allow subsystems (as well as test stands) to operate independently. Data is assimilated into events through a combination of shared memory and a high performance network. This system presents data to a UNIX farm via a high speed non-blocking ethernet switch at a rate of 2 kHz.

Topics such as bootstrapping and loading 200 processors, NFS file access for these processors and software development and deployment are discussed.

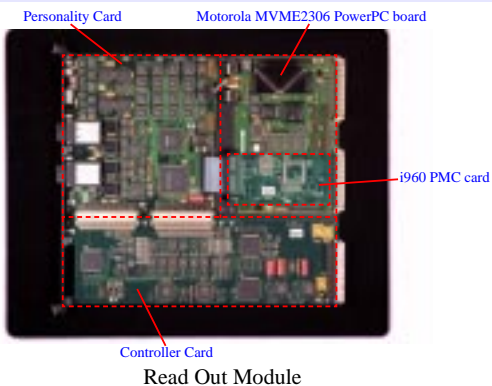


The BABAR detector on the PEP-II beamline

Data from the 5 subdetectors is digitized on the detector in so called Front End Elements (FEEs).



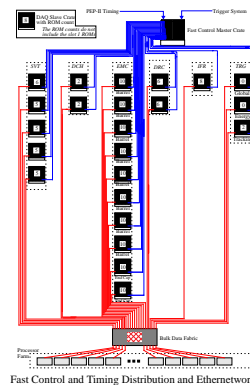
FEEs are connected to Read Out Modules (ROMs) with optical fibers, one for control and one for data. A ROM can handle up to 32 FEEs.



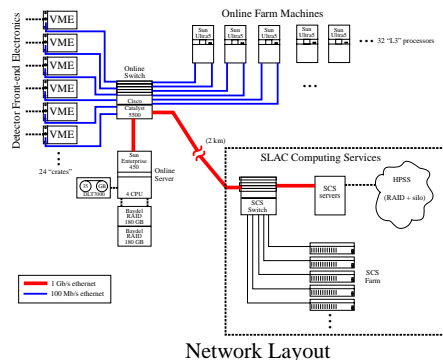
Read Out Module

ROMs are 9U VME cards composed of four independent circuit boards. There are two varieties of Personality Cards to interface to fibers: triggered (TPC) and untriggered (UPC). The Controller Card provides access to the custom P3 backplane, carrying control and timing signals. The processor card is a commercial Single Board Computer: Motorola MVME2306 with 300 MHz 604r PowerPC, 32 MB DRAM, 1+4 MB flash memory, DEC 21140 ethernet interface, Tundra Universe II PCI/VME bridge, serial port, 2 PMC expansion slots. The serial port and network access are brought out through the P2 connector over a custom flex-circuit attached to back side of SBC. VxWorks RTOS is booted from flash. PMC daughter card contains i960 to transport data from PC to SBC. There are some 170 ROMs in 24 VME crates. ROMs are used for feature extraction and partial event building.

The Fast Control and Timing System distributes messages such as the L1Accept trigger and PEP-II derived timing signals and collects ROM-generated inhibit signals (*FULL*). Messages are accompanied by a 56 bit, beam-crossing derived timestamp. The FCTS was designed to allow *partitioning* of the system. The hardware trigger system generates L1Accept from deposited energy and tracks plus geometrical constraints. The L1Accept timestamp is used to tag event *segments* in all ROMs. L1Accept rate is 2 KHz. This reduces the 2.35 GB/sec from the detector to 65 MB/sec. The Data Acquisition system is governed by the Finite State Machine model. State transition requests are passed around using the FCTS messaging scheme.



The network consists of 100 Base Tx, full duplex ethernet and Cisco Systems Catalyst 5500 switches. The TCP/IP protocol stack is used for the network transport. UDP is used to transfer event data from ROMs to the Level 3 trigger processor farm. Packetization, packet acknowledgement and flow control were implemented to make the transport reliable and allow events of almost arbitrary size.



Event segments from data-taking ROMs are built into event *fragments* in VME slot 1 ROMs according to their timestamps. Event fragments are dispersed to a Level 3 trigger processor by indexing into an available farm node table using the timestamp information. This ensures that all fragment contributions of an event go to the same L3 node. L3 nodes are Sun machines running the Solaris operating system. The physics rates resulting from the L3 trigger is 100 Hz. The data rate going to persistent store is 3.2 MB/sec. Buffering around each stage of event data processing smooths out variations in data rate. When a stage's buffering is exhausted, the next stage upstream is notified. This process continues (called back pressure) until FEEs can no longer deposit data into ROMs. At that point the *FULL* signal is asserted to disable L1 triggers.

Lessons learned and Suggestions

- To allow concurrent independent development of subsections of the system, partitioning was implemented
- RTOS kernel was put in flash and network parameters were tuned to achieve reliable booting in fastest possible times.
- To ensure that a multiprocessor system is coherent, third party loading to *broadcast load* kernel, code and constants should be used.
- Connections should be brought into the backs of modules, not via the front panel, to reduce connector failures and lower the probability of incorrect wiring.
- Version management is used (e.g., cvs) to keep track of changes to code and data files.
- Loadable objects are (or will be) tagged with a version "number" so that it can be determined what has been loaded after the fact.
- Strived for scalability so that errors in system parameter guesses can be compensated for.
- Used object oriented methodology and C++ for all BABAR software. Not all compilers support all features of the language. This caused unavailable features in one compiler to be avoided for use with other compilers.
- Some C++ features caused more difficulties than they solved, e.g. templates and inlined functions. Dependency trees became unwieldy.
- Included header files cause compilation time to increase. Programmers are often not aware of how deeply nested header files are. The effect on compilation time causes some programmers to replicate from header files instead of including them.
- RISC programming techniques are different from CISC techniques. Avoid memory references. Make the best use of cache by accurate timing measurements. Watch for prefetch and write posting side effects. Make sure that memory cacheing attributes are properly set.
- Looking at the compiler's assembler output teaches one how to trick the compiler into generating the best code.
- Shareable libraries and run-time linking allow one to avoid interdependencies between packages. Clients use a common copy rather than linking to private copies. Much easier for package owners to maintain.
- For packages that must communicate together, dynamic version management can help. Add a version number to data communicated between packages so that clients can recognize whether they're incompatible with the server's output format.

Go to Rowan Hamilton's talk
[The BABAR Data Acquisition System](#)
 for more information