

E787 Data Acquisition Software Architecture

M. Burke, L. Felawka, R. Poutissou,
TRIUMF, 4004 Wesbrook Mall, Vancouver, BC,
V6T 2A2, Canada

and

S. Adler, J. Haggerty, R. Strzelinski, C. Witzig,
Physics, Brookhaven National Laboratory, Upton, NY,
11973, USA

Abstract

Brookhaven National Laboratory (BNL) Experiment 787's second generation Unix-based data acquisition system is comprised of several independent programs, each of which controls a specific aspect of the experiment. These programs include packages for reading events from the hardware systems, analyzing and reducing the data, distributing the results to various data consumers, and logging the data to tape or disk. Most of these can be run in stand-alone mode, for ease of development and testing. There are also a number of daemon processes for writing special data records to the data streams, and several monitor programs for evaluating and controlling the progress of the whole. Coordination of these processes is achieved through a combination of pipes, signals, shared memory, and FIFOs, overseen by the user through a Motif graphical user interface. The system runs on a Silicon Graphics 4D/320, interfaced to a Fastbus system through the BNL Fastbus/VME interface (BBFC), and runs under Irix and Motif/X-windows.

I. REQUIREMENTS

Brookhaven AGS (Alternating Gradient Synchrotron) Experiment 787's first generation data acquisition system was MicroVax 3200/QBUS based, and relied on a farm of 68020-based ACP nodes for online data processing and compaction. The QBUS bandwidth of 1.5 Mb/s, and the ultimate throughput of 233 kb/s to two tapes were adequate for the first phase of the experiment, ending in 1991. Upgrades starting at that time included a new beam line, offering three times the beam intensity as before, and a detector upgrade with new instrumentation, including 1000 channels of CCD transient digitizers. The event size is expected to triple once the second phase is complete. The old data acquisition system could not handle this throughput, so we undertook to design a new online system in the spring of 1992. The goals were to maximize event throughput within the constraints of the front-end electronics, the periodic beam cycle of the AGS, and a tripled beam intensity.

A. Hardware

We decided on Silicon Graphics' (SGI) Power Series 4D/320 computer to serve as the computing engine of the new data acquisition system. Advantages of this system included multiple, high-performance, upgradeable CPUs, a

versatile and industry standard Unix-based operating system, and a high-speed I/O subsystem including two SCSI buses and a VME bus integrated into its internal architecture. A clear upgrade path exists from this architecture to SGI's new Challenge architecture.

In our 1992 data-taking run, the computer configuration included the Irix 4.0 operating system, two R3000 CPUs, and four Exabyte 8mm tape drives (three model 8200s, and one model 8500).

The front-end electronics were not changed to accommodate the new data acquisition system. In particular, SLAC Scanner Processors (SSPs) [1] continue to be used to accumulate data from the FASTBUS crates during the one second of beam in each 3.2 second spill. A trigger SSP notifies the secondary SSPs when an event of interest has occurred, and signals them to read out data from their respective FASTBUS crates into memory. Between the bursts of beam, a master SSP takes over the process of building events and transferring them over the branch bus to the SGI. A new high-speed fastbus to branch bus interface (BBFC) [2] was designed, coupling the front end electronics to the SGI's VME backplane. Data transfer between FASTBUS and the SGI memory was measured at approximately 17 Mb/s in each direction.

B. Software

Communication with the BBFC was facilitated with a new device driver [3], which, along with the standard high-level IEEE FASTBUS routines, and the BNLSSP routines [4], were ported to the SGI. These packages eased the porting of other code that interacts with hardware on this system.

A philosophical decision was made to develop the online system as a suite of independent processes, rather than as a single, monolithic program. The advantages included greater exploitation of the multi-CPU SGI architecture, and greatly simplified development and testing, since each component is devoted to a single task and can in principle work as a stand-alone program. The main disadvantage, namely the management and synchronization of numerous processes in a multi-tasking environment, is largely handled by the Unix operating system and its built-in facilities for inter-process communication.

Our software wish list included an X11/Motif user interface to the system, in order to enhance ease of operator training, idiot-proofing, and overall trendiness. The *XDe-*

use one of the following buffer strategies: store forward or virtual cut-through, both with or without additional buffer space.

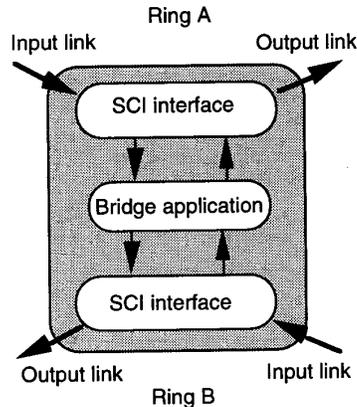


Fig. 5. Model for the ring-to-ring bridge

The basic element in both the node and the bridge is the SCI interface shown in Fig. 6. Each SCI interface has an address decoder, input FIFOs, output FIFOs, bypass FIFO and a mux for the output link as defined in the standard [7]. The address decoder model can recognize an area of addresses (for the bridge) or a single address for the node. The SCI interface model operates at word level. It handles all logical operators for SCI, including busy retries and the "A-B" aging scheme, but the elasticity buffer and clock synchronization as in a real implementation is skipped.

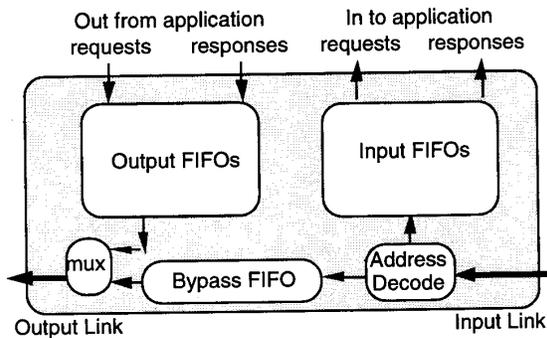


Fig. 6. Model for the SCI interface

The algorithm to route packets through the network uses fixed fields in the header for each dimension in the network. A packet will then go in one dimension until a bridge recognize the field. It will continue through the vertex and in the new dimension until a new bridge recognize the address field and so on until the right vertex and destination node is found.

B. Simulation Parameters

Table 1 gives an overview of the different parameters that were used in the simulations. The table consists of both parameters that were kept constant through all simulations and parameters that were modified, and it shows between which boundaries that the parameters were used. Physical

characteristics of nodes, bridges and wires were kept constant for the different topologies that were simulated. These parameters are based on information from Dolphin SCI Technology and are related to a possible hardware implementation. For simulation of a selected topology a set of 21 simulations was performed with 7 different load situations and with 1, 2 and 4 outstanding requests. Combinations of the parameters k , n and p defined the number of nodes in a topology between 12 and 512 nodes.

For all load situations a uniform distribution of the time from a response to a new request was used within the indicated intervals. Negative exponential and normal distribution were also tried, but there were no significant difference in the results. The load situation when a new request arrives as fast as possible will be equivalent to a worst case startup situation, and the light load situation is close to what one can expect from 50-100 MIPS CPU with caches in a shared memory environment. The load situations used in these simulations can be classified to be heavier than in real DAQ experiments. All requests were assumed to be move64 packets.

For all simulations presented in this paper, we have assumed the bridge be of the type "virtual-cut-through" with double FIFOs for requests and response. This type of a bridge gave the best performance.

IV. RESULTS

In this presentation of the simulations we have focused on results that indicate what kind of performance one can expect from SCI and also what limitations that may appear. We have concentrated on effective system bandwidth and latency, and tried to show how systems scale. We have also compared the simulation results with calculations based on simple theoretical models.

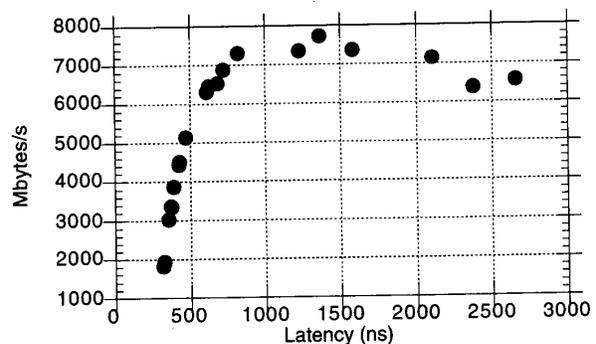


Fig. 7. Effective bandwidth as a function of latency for the topology $6^2 \times 3$ (6.2.3) with 108 nodes. (Longest latency has 4 outstanding requests).

An important system behavior is shown in Fig. 7. which displays the effective bandwidth in the system as function of latency. The figure is for a system with 108 active nodes built from a mesh of 6×6 vertices and 3 nodes in each vertex. The lightest load gives the lowest latency. When the load increases, both latency and bandwidth increase, but the bandwidth reaches a maximum and then slowly decreases if the load is further increased. The reason for this reduced bandwidth is an increasing number of retries, due to heavy load.

Latency is measured from when a packet is placed in the output FIFO and ready to access the network in the sending

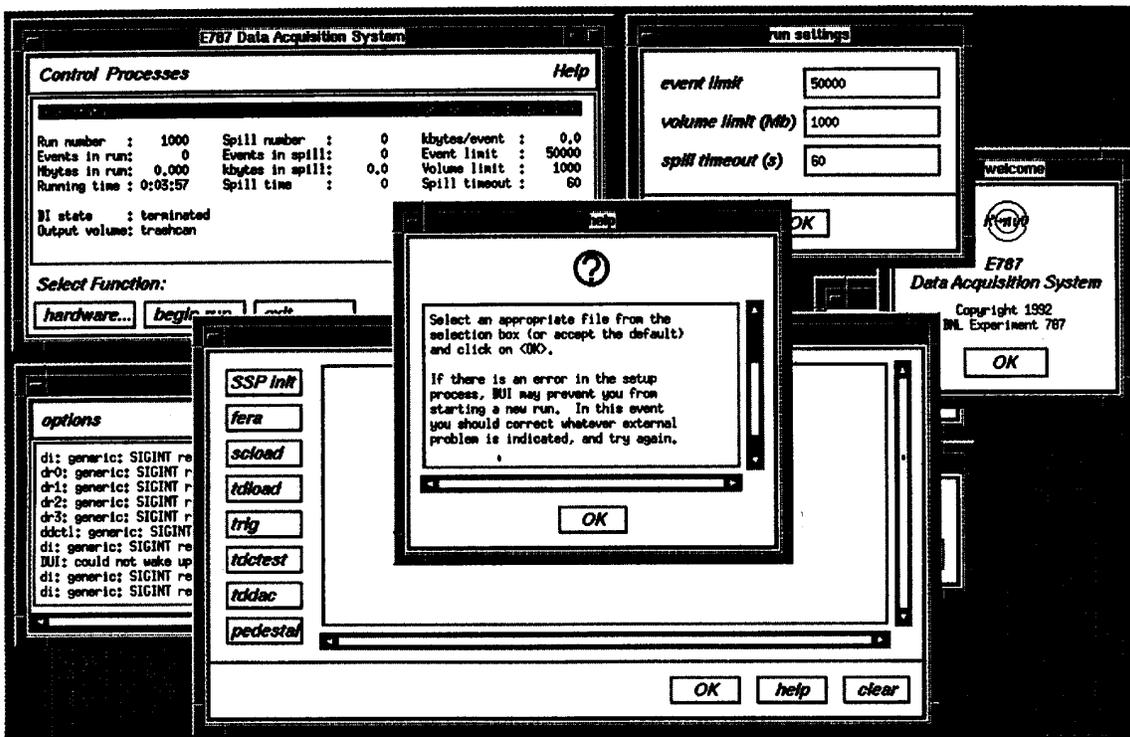


Fig. 2. The DUI (user interface).

B.4. Special Record Daemons

The special record daemons are a special type of data producer that are invoked at certain times to write special records to the data stream. Four types of special records are currently supported: comments, which are written at the beginning of every run and any other time at the user's discretion; scalars, which are written once per spill; begin-runs, which are written at the start of every run; and endruns, which are written at the end of every run. These records are generated and broadcast to all data consumers by daemon-like processes that sleep until signalled by the DUI (see below).

B.5. DUI (user interface)

The user interface (figure 2) is an X11/Motif program that allows the operator to automatically start up and initialize all the various processes of the system, quickly and easily navigate the states of the online system, initialize hardware, and monitor the progress of the runs. It polls the DCOM area to track the states of the various online processes and reports relevant information to the user through a status window. It also monitors pipes that are attached to each of the processes in order to pick up messages and display them for the operator.

At any time the DUI is in one of seven states: initialization, start-up, beginrun, active, paused, endrun, or shutdown (figure 3). The appearance of the user interface changes in each state to show the operator only those options and state transitions that are available at that time.

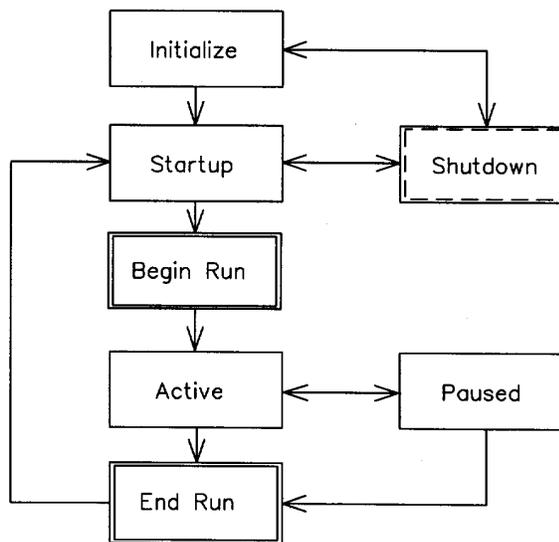


Fig. 3. State diagram for the online system.

Development and testing of the DUI was greatly simplified by the XDesigner user interface builder tool. It proved so effective for generating graphical user interfaces that snappy little GUIs were written for most of our monitor and testing programs. Several, including applications to monitor the DD system status and an interface to the DP0s, have become integral components of the online system.

throughput from the node has reached its limit, while latency increases 6-7 times and the percent of packets ending up getting a busy-retry increase to 80%.

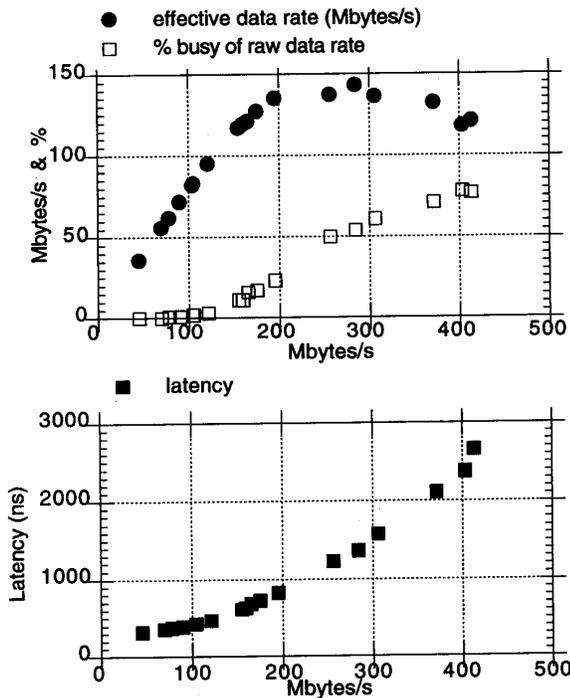


Fig 12. Effective data transfer from a node as a function of raw data rate (included busy-retry). Increase in latency and percentage of busy packets are also shown as function of raw data rate. The topology is a 6-ary 2-cube with 3 nodes in each vertex, i.e. 108 nodes.

V. CONCLUSIONS

The simulations show that complex systems ranging from a few tens and up to about 500 active nodes can use k -ary n -cubes and get a good effective system bandwidth and a relative low latency when the system is not too heavily loaded. The results show that:

- Prediction of latency is good when the systems are not saturated.
- Latency increases and is unpredictable in a saturated system, but no deadlock occurs. Forward progress is guaranteed.
- The ring-to-ring bridge gives good system bandwidth and is efficient for mesh networks (2 dimensions).
- The ring-to-ring bridge is not optimal for larger dimensions, due to low utilization of theoretical available bandwidth.
- Routing algorithms using ring-to-ring bridge is simple and can easily be implemented in hardware.

Theoretical calculations and simulations show close to the same results when the systems are lightly loaded. When the system starts to saturate, the theoretical calculations of latency and bandwidth are much more difficult to do, and simulation will be an easier method to get estimation of system performance.

As a general statement, SCI is a good candidate for DAQ system when the system bandwidth requirements are

in the multi GBytes/s range and when fast access to data in any part of the system is an essential parameter.

VI. ACKNOWLEDGEMENT

This work is supported by Dolphin SCI Technology and Royal Norwegian Council for Scientific and Industrial research (NTNF). We wish to acknowledge the contributions of Andre Bogaerts, Stein Gjessing, David B. Gustavson and Bin Wu.

VII. REFERENCES

- [1] Gustavson, D.B., The Scalable Coherent Interface and Related Standards Projects, *IEEE Micro*, 12(1), February 1992, pp.10-22.
- [2] Aines, K., SCI Node Chip, *Proceedings Open Bus Systems 92*, Zürich, October 1992, pp.49-54.
- [3] Bothner, J. W. and Hulaas, T.I., Various interconnects for SCI-based systems, *Proceedings Open Bus Systems 91*, Paris, November 1991, pp.197-202.
- [4] Bothner, J. W. and Hulaas, T.I., *Topologies for SCI-based systems with up to a few hundred nodes*, Institutt for Informatikk, University of Oslo, February 1993.
- [5] Kristiansen, E.H., Bothner, J. W. and Hulaas, T.I., Behaviour of Scalable Coherent Interface in larger Systems, *Proceedings CAMAC-92*, Warsaw, October 1992.
- [6] Bogaerts, A., Cern, Personal communication and discussions, March 1993.
- [7] IEEE, *SCI - Scalable Coherent Interface, D2.00*, Draft accepted as IEEE std. 1596-1992, March 18, 1992.
- [8] Dally, W.L. and Seitz, C.L., Deadlock-Free Message Routing in Multiprocessor Interconnection Networks, *IEEE Transactions on Computers*, C-36(5), May 1987, pp.547-553.
- [9] Johnson, R.E., *Interconnect Topologies with Point-to-Point Rings*, Computer Sciences Technical Report #1058, University of Wisconsin-Madison, December 1991.
- [10] Johnson, R.E. and Goodman, J.R., Synthesizing General Topologies from Rings, *Proceedings of ICPP*, August 1992.
- [11] Scott, S.L. and Goodman, J.R., *Performance of Pipelined K-ary N-cube Networks*, Technical Report #1010 University of Wisconsin-Madison, February 1991.
- [12] Scott, S.L., Goodman, J.R. and Vernon, M.K., Performance of the SCI Ring, *Proceedings IEEE ISCA 92*, Queensland, May 1992.