Fermi National Accelerator Laboratory

FERMILAB-Conf-91/303

Results from a Data Acquisition System Prototype Project Using a Switch-Based Event Builder

D. Black, M. Bowden*, J. Andresen, E. Barsotti, A. Baumbaugh, A. Booth*, G. Cancelo**, D. Esterline, K. Knickerbocker, R. Kwarciany, G. Moore, J. Patrick, C. Swoboda, K. Treptow, O. Trevizo, J. Urish, R. VanConant and D. Walsh

Fermi National Accelerator Laboratory
P.O. Box 500, Batavia, Illinois 60510

**SSC Laboratory*
2550 Beckleymeade Ave., Dallas, Texas

***Universidad Nacional de La Plata*
Argentina

November 1991

* Presented at the *IEEE Nuclear Science Symposium*, Santa Fe, New Mexico, November 2-8, 1991.

Operated by Universities Research Association Inc. under Contract No. DE-AC02-76CHO3000 with the United States Department of Energy

Disclaimer

RESULTS FROM A DATA ACQUISITION SYSTEM PROTOTYPE PROJECT USING A SWITCH-BASED EVENT BUILDER

D. Black, M. Bowden*, J. Andresen, E. Barsotti, A. Baumbaugh, A. Booth*, G. Cancelo**, D. Esterline, K. Knickerbocker, R. Kwarciany, G. Moore, J. Patrick, C. Swoboda, K. Treptow, O. Trevizo, J. Urish, R. VanConant, D. Walsh

Fermi National Accelerator Laboratory, Batavia, Illinois
*SSC Laboratory, Dallas, Texas
**Universidad Nacional de La Plata, Argentina

ABSTRACT

A prototype of a high bandwidth parallel event builder has been designed and tested. The architecture is based on a simple switching network and is adaptable to a wide variety of data acquisition systems. An eight channel system with a peak throughput of 160 Megabytes per second has been implemented. It is modularly expandable to 64 channels (over one Gigabyte per second). The prototype uses a number of relatively recent commercial technologies, including very high speed fiber-optic data links, high integration crossbar switches and embedded RISC processors. It is based on an open architecture which permits the installation of new technologies with little redesign effort.

INTRODUCTION

The switching network approach to physics event building is based on similar switching networks used in telecommunications systems. The basic component is a synchronous time-division multiplexed switch (TMS). Event data arriving from each independent source is buffered in an accompanying time-slot interchanger (TSI) which assembles and sequences data packets for maximum utilization of switch bandwidth.

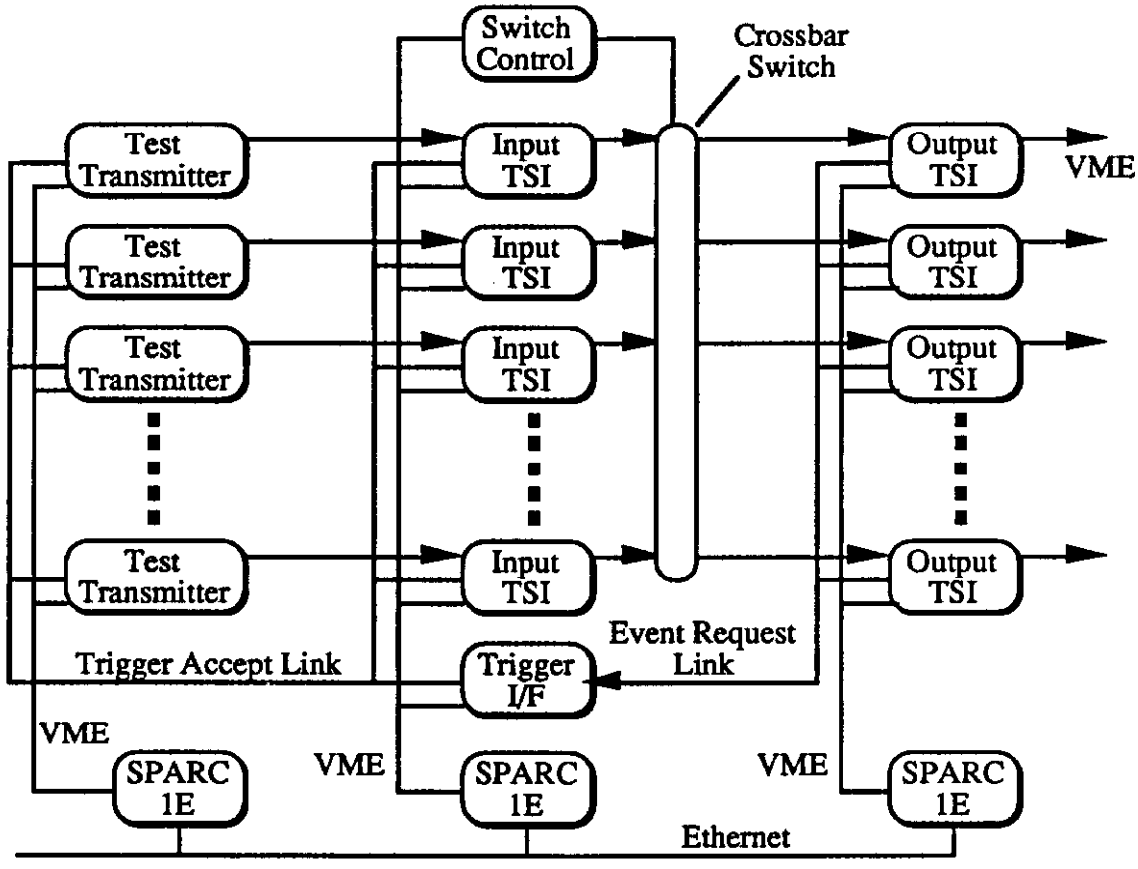

Figure 1 Prototype System

Work performed under auspices of United States Department of Energy

The switch interconnection pattern for event building is completely deterministic, so switch control requirements (in comparison to a telecommunications switch) can be greatly simplified. The switching rotation can be predefined as some subset of a general crossbar (e.g., a barrel shifter) which in turn reduces the number of stages and complexity of the switch. The switch is entirely self-routing. Destinations are assigned to the events by the trigger system and can be added to the event header at the front-end or at the Input TSI.

The goal of future physics event building is to allow the collection of data at high trigger rates with high data throughput. The pipelining and parallelism built into this system allows the data to pass through with minimal hardware or software protocol overhead.

IMPLEMENTATION

A small prototype event builder (Figure 1) using the switching network approach has been implemented. Figure 2 shows simplified block diagrams of each module. Most of the components contain an embedded processor to allow software control of data format and module functionality. These processors are software compatible with the host workstation (SUN SPARC). The "Test Transmitters" contain 64 KW of pattern memory which can be downloaded with simulated event data. For test purposes, a number of different error conditions can be dynamically inserted into the data stream. Each channel drives a 100 meter fiber-optic data link operating at 250 Mbps. The fiber link driver and receiver circuitry is implemented on a small daughtercard which is pin-compatible with a lower-cost twinax ECL version

Event fragments are received by the "Input TSI" modules which store and forward the data to the appropriate "Output TSI". The Input TSI maintains a separate queue for each available switch output channel. A data packet is taken from the appropriate queue depending on the current switch position. The Output TSI uses the same circuitry as the Input TSI but is designed to connect to a standard workstation or multi-processor VME backplane. It contains a 32 bit VME master interface plus data checking logic. TSI buffers are implemented using video DRAM which provides a large, economical dual-port memory for packet mode transfers. Individual packet buffers are managed by the embedded processor using a simple page allocation scheme.

Event flow is controlled by the "Trigger Interface" module which matches event requests from the Output TSI/Processor with triggers from the (simulated) trigger system. The Trigger Interface maintains a priority table of pending event requests and assigns specific events to specific processors based on event type and overall event distribution.

The 64 X 64 channel switch is implemented on a common backplane using eight integrated circuits (only four of these are installed in the prototype system). Each path through the switch is eight bits wide to minimize high-frequency effects. The switch is fully synchronous with a programmable rotation pattern and packet size. The Switch Controller provides a small configuration memory, counter and clocks which define the switch size, rotation pattern and internal packet size.

Two control links regulate data flow in the system. The "Trigger Accept Link" is used to broadcast a destination ID and event type for each event. This message is queued in the Input TSIs and used to route the event fragments as they arrive. The Input TSIs may also distribute events in a fixed order without using the Trigger Accept Link. In the prototype system, the same link is used by the Test Transmitters to select an event in pattern memory. The "Event Request Link" provides a way for individual event processors to control the type of triggers they wish to examine and the rate at which events are supplied. VME connections are strictly for downloading and monitoring of embedded processors. There is no data transferred across the VME bus.

Figure 3 illustrates the major software components. There were four main areas of software development; the user interface (about 20,000 lines), remote procedure calls (18,000 lines), embedded code (4,000 lines) and diagnostics. All software was generated using standard facilities of the SUN host workstation.

The graphical user interface was developed using (VI Corp) DATAVIEWS. This is a commercial product consisting of two parts, DV-DRAW which is used to create the graphical layout of the interface and DV-TOOLS which connects sources of real-time information to the on-screen graphics. Dataviews also provides an interface to the (Neuron Data Corp) Nexpert expert system software for intelligent diagnosis of system errors. The prototype system uses a series of hierarchical displays. The top level allows configuration and initialization of the system. It also selects the operating mode (actual or simulated) and provides an interface to the diagnostic software.

Configuration is accomplished by selecting files which contain the embedded code, data patterns and switch control information. A menu of default files for various switch configurations is provided. Selecting the initialization function invokes a series of remote procedure calls (RPCs) which reset and download each module in the proper sequence. Total initialization time is approximately 10 seconds. Once the system is running, the RPC mechanism is used to retrieve status information at regular intervals. The total event count, instantaneous event rate and peak buffer usage for each channel is displayed using on-screen meters. The same user interface can control a (Cadence Design Systems) Verilog simulation of the data acquisition system. One intended advantage of this approach is that most of the interface software can be developed in parallel with the system hardware. RPCs are executed on the SUN SPARCengine 1E boards which access individual system modules through 16-bit dual-port VME interfaces. SUN rpcgen was used to compile a library of functions for resetting, downloading and monitoring the system.

The embedded code is written in C using the SUN SPARC compiler. Use of SPARC embedded processors allows compilation and debugging of the real-time code without a specialized development environment. Each processor runs a single task, negating the need for a real-time operating system.

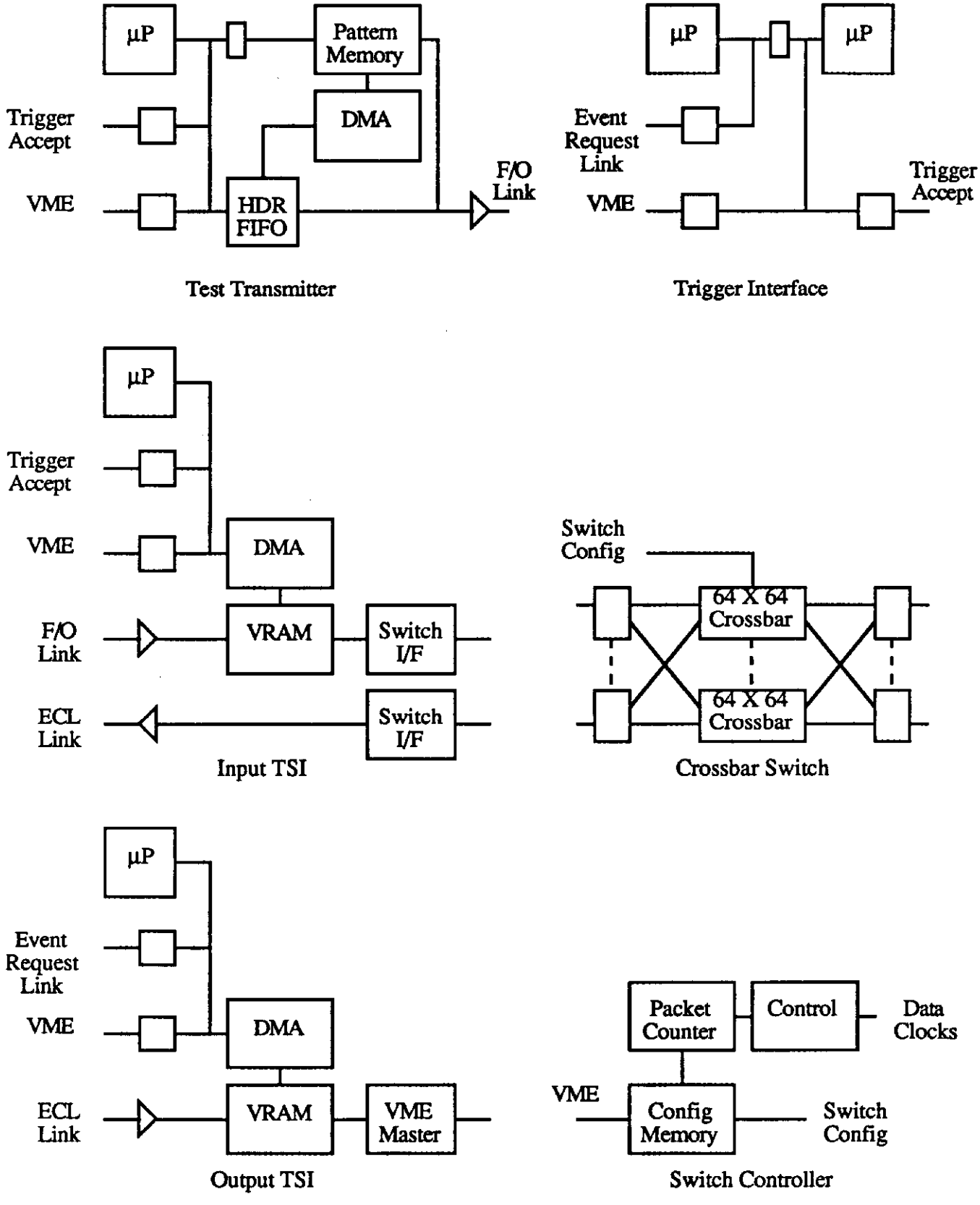

Figure 2 System Components

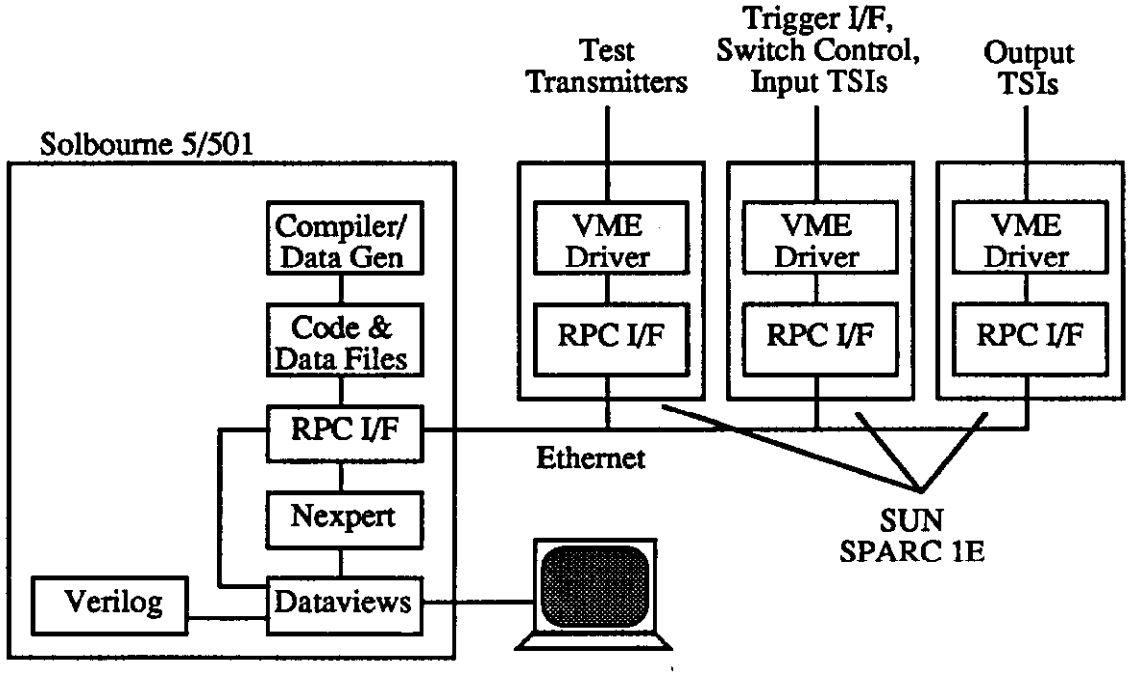

Figure 3 Software Interaction

RESULTS

The prototype switching hardware occupies a single 9U X 400 crate. Up to 64 channels (1 GByte/sec) can be accommodated. By comparison, the previous generation of bus-based event builders handle approximately 20 MBytes/sec in the same physical space. This is a 50X improvement brought about mainly through the use of high-speed (point-to-point) serial communication and parallel data transfer.

The system was operated with 8 inputs and 1, 2, 4 or 8 outputs, demonstrating the expected linear increase in throughput. It can be dynamically reconfigured without interrupting the dataflow. With an event size of 200 KBytes, a rate of 600 Hz (>120 MBytes/sec) in the 8 X 8 mode was easily sustained (figure 4). Average buffer usage in a single TSI is given by the equation $E * (N-1) / N / 2$ where E is the total event size and N is the number of switch channels. This is approximately $E / 2$ for any reasonable switch size. The 200 KByte sample events use an average of only 10% of the available TSI buffer space in the prototype system (figure 5). To handle non-uniform distributions, a buffer size of at least 1.5 times the average event size is recommended. As long as the average event size remains constant over time, the throughput will be very close to the maximum available bandwidth.

Tests were also run at very low rates with software in the Output TSI checking each event for data integrity and routing. Simple error conditions were simulated (by disconnecting an input data link for example) and these errors were detected by both hardware (sync error) and software (missing event fragment) monitors. Using a minimal rulebase, the expert system software is then able to read diagnostic information and localize the errors to a specific switch input or output channel.

The system was operated in two different packet modes. In the first mode, event boundaries and packet boundaries are completely independent. This makes maximum use of data link bandwidth but requires additional processing overhead in the Output TSI to locate event boundaries within packets. In the second mode, events begin on even packet boundaries only. This results in some packets being less than full, but it has the advantage of greatly simplifying the Output TSI logic. For event fragments which are much larger than the packet size, the loss is negligible. Ideal packet size represents a tradeoff; large packets waste bandwidth and small packets require too much processing overhead. A packet size of 1024 Bytes was used for the majority of tests in the prototype system.

Test results in most cases were better than predicted by simulation. The simulations assumed certain overheads for processing packet headers, etc. Many of these overheads were hidden in the actual hardware by pipelining operations and by transferring data at higher internal rates to make up for processing time.

Efficient switch operation is based on two factors. The event data must be evenly distributed among the input channels and the assembled events must be evenly distributed among the output channels. The TSI buffers act to eliminate short-term fluctuations in data rate. However, even when the channels are poorly balanced, the bandwidth of a switch-based event builder will be substantially higher than an equivalent bus-based system.

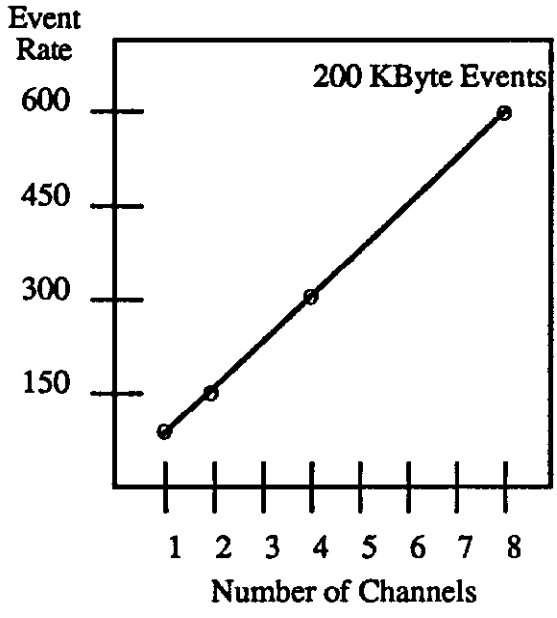

Figure 4 Throughput Scaling

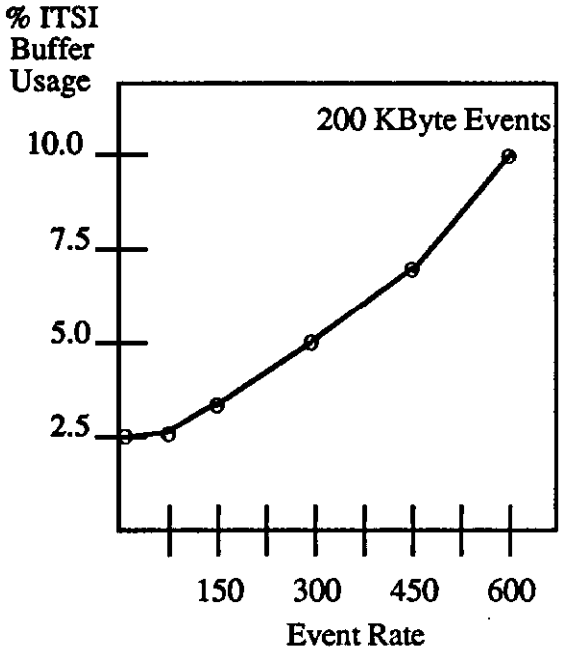

Figure 5 Input TSI Buffer Usage

CONCLUSIONS

The prototype system meets its performance objectives and appears to scale well. Much of the complexity in this particular implementation comes from the large number of programmable features. Using a fixed packet size and header format would allow significantly faster and simpler hardware. The Output TSI is not necessary when using aligned packets. It can be replaced by a serial link to backplane interface.

Any future version would be designed as a modular (8 X 8 or 16 X 16) self-contained switch. This allows almost unlimited expansion. Use of serial switches to reduce the cost has also been considered but tests of receiver resynchronization delays following switching are needed.

Comparisons are sometimes made between the system described here and a similar approach which uses an array of dual-port memories (Figure 6). In fact, these two architectures are basically identical. Both use a crossbar switch and differ only in the placement of buffers. In one case the buffers are distributed throughout the crossbar (embedded queueing), while in the other case they are concentrated at the input (input queueing). We favor the second approach simply because VLSI technology allows implementation of large buffers and integrated crossbar switches at far lower cost than is possible with smaller distributed buffers and discrete interconnections.

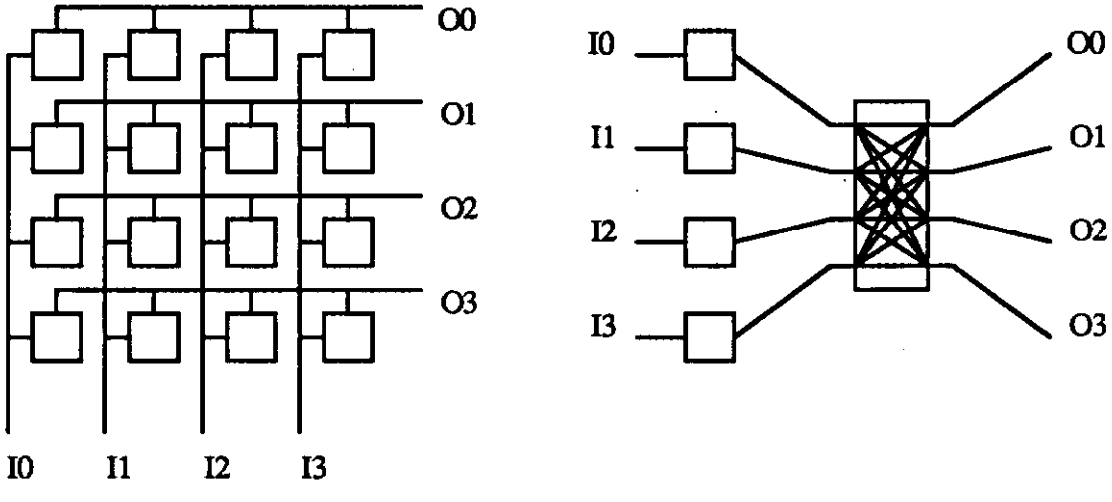

Figure 6 Buffered Crossbars

REFERENCES

1. "A Scalable Parallel Open Architecture Data Acquisition System for Low to High Rate Experiments, Test Beams & All SSC Detetcors", E. J. Barsotti, et al, IEEE Nuclear Science Symposium, San Fransisco, 1989.
2. "A High-Throughput Data Acquisition Architecture Based on Serial Interconnects", M. Bowden, et al, IEEE Transactions on Nuclear Science, Vol. 36, No. 1, February 1989, p760-764.
3. "Verilog-XL",Cadence, Lowell,Massachusetts.
4. "Dataviews", V.I. CORP,Amherst,Massachusetts.
5. "Nexpert",Neuron Data Inc.,Palo Alto, California.