# A MULTIPROCESSOR ARCHITECTURE FOR HIGH-RATE COMMUNICATION PROCESSING

Eric E. Johnson

New Mexico State University*

ABSTRACT*

High data rate communication processing presents a unique challenge to computer architects, due to the unusually high ratio of I/O to computation required compared to general-purpose computing, and the greater variability and complexity of the processing required per packet compared to high speed packet switching.

The Virtual Port Memory architecture developed at NMSU is a global memory-message passing (GMMP) multiprocessor which is well suited to I/O-intensive real-time processing. We here present a description of the architecture, including its unique memory system structure, and an analysis of its performance for communication processing.

## I. INTRODUCTION

To distinguish communication processing from general-purpose computing on the one hand, and from packet switching on the other, it is useful to consider the ratio of I/O to processing that each requires. A rule of thumb in computer architecture, known as the Amdahl-Case Rule [1], suggests that a general purpose computer system will be balanced when, for each MIPS of computational performance, it has 1 MByte of main memory and supports 1 Mbps of I/O, for an I/O-to-processing ratio of roughly one bit per instruction executed. High-speed packet switching, as discussed in [2], requires only a few operations to set up a Banyan switch for a packet of several thousand bits, resulting in an I/O-to-processing ratio of hundreds of bits per operation. Communication processing, such as protocol processing, packet reassembly, adaptive routing, and so on, has intermediate I/O-to-processing ratios on the order of ten bits per instruction.

These three types of processing may also be contrasted by the regularity and complexity of the tasks involved: general-purpose computing includes the most irregular and complex computational tasks, while packet switching often has a sufficiently small range of required behaviors to be implemented entirely in hardware; communication processing again falls in between.

System architectures for general-purpose computing therefore contain general-purpose processor(s) and I/O subsystems designed to provide, in the case of well-balanced machines, about one Mbps of I/O per MIPS of CPU performance. Packet switches have also been designed using general-purpose processors, but packet switches designed for data rates of Gbps and packet rates of over $10^6$ per second will probably be implemented almost entirely in hardware [2]. The range of tasks performed by a communication processor appears to require general-purpose CPUs, but the I/O load clearly exceeds that for which general-purpose computers are designed. For high data rate communication processing, therefore, it will usually be necessary to enhance the entire I/O subsystem of a high-performance computer to bring its I/O capacity into balance with its processing performance for communication applications.

This paper presents a general-purpose multiprocessor architecture which accommodates an I/O bandwidth of many Gbps through the use of VRAM in the main memory. This bus-based architecture permits incremental adjustments in I/O bandwidth, memory size, and processing power by simply adding or removing I/O controllers, memory modules, and processors. This architecture is described in section II, followed by an analysis of its performance in handling various communication processing tasks, including the 4 x 300 Mbps data stream at the NASA Tracking and Data Relay Satellite System (TDRSS) ground terminal.

## II. THE VIRTUAL PORT MEMORY MULTIPROCESSOR ARCHITECTURE

The architecture described in this paper was originally developed for computer vision applications, which have similar characteristics to communication processing: relatively high I/O rates, with the processing load per bit decreasing at higher levels of abstraction. Because this architecture belongs to a relatively unusual class of MIMD[1] architectures, this section begins with a discussion of these Global Memory-Message Passing (GMMP) architectures.

### *A. GMMP Multiprocessors*

Multiprocessor architectures may be conveniently classified by noting how they address two fundamental questions:

* This work was supported in part by the U.S. Army Research Office.

[1] Multiple Instruction/Multiple Data streams.

1. Do processors have uniform access to the main memory, or is the memory distributed among the processors, producing non-uniform memory access times? The former case describes a global memory (or UMA, for uniform memory access) architecture, while the latter is distributed memory (or NUMA, for non-uniform memory access).

2. Do processes have access to each others' data structures, allowing communication through shared variables, or are process access spaces private, requiring message-based communication and synchronization? The former is termed the shared variable case, while the latter is called message passing.

With two cases for each of two decisions, this scheme results in four classes of multiprocessor architectures:

1. GMSV (global memory-shared variables), typified by bus-based multiprocessors such as the Sequent Symmetry, and by multistage network machines such as the NYU Ultracomputer, both described in [3];

2. DMSV (distributed memory-shared variables), such as the BBN Butterfly [5]; and

3. DMMP (distributed memory-message passing), typified by the many hypercube machines, such as the N-cube [4];

4. GMMP (global memory-message passing), such as the NMSU Virtual Port Memory machine [6].

GMMP architectures attempt to combine the best attributes of the GMSV and the DMMP machines and to avoid their drawbacks, while DMSV architectures may inherit "the worst of both worlds" [7, p.55]. Message passing often involves more software overhead for interprocess communication than does shared variables, and may introduce larger communication delays when large data structures must be sent from one processor to another in the absence of a global memory (the DMMP case); however, errors are often easier to find in message-based parallel programs [7] and the resulting concurrent systems may be more reliable [8]. Furthermore, the message passing model seems intuitively better-suited for expressing communication processing tasks than is the shared variable model.

With a global memory available as a common repository for data, large data structures can be passed by reference, rather than by value; data is then moved between processes via high-bandwidth cache-to-memory paths, rather than over interprocessor communication links. Also, due to the elimination of shared variables, GMMP processors can use very large caches to improve both processor and system performance without a need to keep these caches consistent, while GMSV machines must ensure cache consistency, often at significant costs in hardware or performance.

Analysis of the speedup available from representative machines of each of these four classes of multiprocessors shows that contention for memory, the interconnection network, or both, significantly reduces the speedup available from GMSV, DMMP, and DMSV multiprocessors as a given problem is run on machines of sizes increasing from tens to hundreds of processors (Figure 1). However, with caches unhindered by consistency concerns, and with the high-bandwidth communication resource provided by the global memory, a GMMP machine supports essentially linear speedups for up to at least a few hundred processors. The architectures used for this comparison were as follows:

GMSV: a multistage-switch machine with local memory at the processors for code and private data (to reduce global memory accesses); I/O traffic reaches the global memory via a dedicated network to reduce processor-memory switch congestion.

DMSV: a Butterfly-type machine with shared-access memory and local I/O at each node.

DMMP: a hypercube with worm-hole routers (to improve message-passing performance and reduce contention for the local memory) and local I/O.

GMMP: a Virtual Port Memory machine, as described in the next section.

The same technology was assumed for all four machines: 2.5 MIPS CISC microprocessors (e.g., 16 MHz 68020s), DRAM main memory, the same total inter-processor or processor-memory bandwidths, and so on.

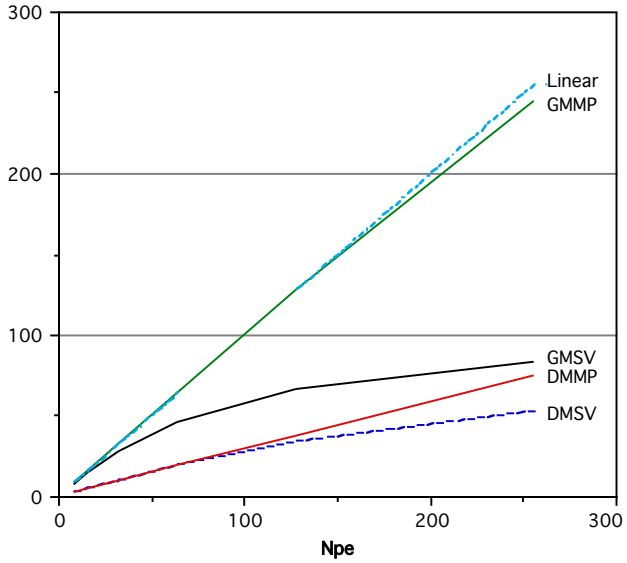


Figure 1: Speedup Bounds of Various Architectures (Constant Problem Size)

*B. The Virtual Port Memory Architecture*

The hardware architecture of Virtual Port Memory machines is designed to efficiently support the

message-passing programming model (processes with isolated address spaces which pass messages for communication and synchronization) while providing uniform high-bandwidth access from the processors to a global memory. A block diagram of a bus-based Virtual Port Memory machine is shown in Figure 2. The interesting architectural features include the following:

a. a split transaction processor-memory channel comprising a transaction request bus (TRB) and a separate data transfer bus (DTB);

b. a dedicated interprocessor message bus (IMB);

c. a centralized address translator (ATran) which performs the final virtual to physical address translation; and

d. "page copy" hardware within the global memory for high-speed intra-memory DMA transfers.

The processors shown here are of three types: processing elements (PEs), which execute application and system processes; user interface processors (UIPs), which interact with users via command interpreters or graphical user interfaces and request the execution of programs on the PEs via messages; and I/O controllers (IOCs), which manage secondary storage and other I/O channels. In order to permit this bus-based architecture to grow to hundreds of processors, the bus system is pipelined: processors send memory transaction requests to the memory controller over the TRB, then release the TRB for other processors' use while waiting for the requested data transfer to occur over the DTB. The TRB need only be wide enough to carry addresses and other transaction data, while the DTB width can be matched to that of a cache line to balance the bandwidths. This structure permits efficient use of the bandwidths of the buses and the memory banks, and naturally accommodates a hierarchy of buses and sub-buses in a large system.

The IMB, along with message passing hardware on each processor (and on the memory controller), provides a broadcast interprocess message channel with high bandwidth (64 MB/s in our prototype machine), whose latency is determined by the speed of the operating system send and receive primitives, not by the hardware.

By centralizing the translation of system virtual addresses to physical addresses in the ATran, the Virtual Port Memory architecture avoids a performance-limiting cache consistency problem common to most other multiprocessors that support virtual memory: when individual processors cache virtual-to-physical address translations in local translation look-aside buffers (TLBs), all cached copies must be found and updated or invalidated whenever a translation is changed [11]. A common approach to this problem, known as "TLB shootdown," involves stalling some or all of the processors in the system while entries are updated, which clearly degrades performance if such changes are frequent. In a Virtual Port Memory machine, however, the ATran is the only virtual-to-physical address translation cache in the system; it is updated to track translation table changes using a single TRB cycle, while the processors continue to execute without incident.

The page copy unit (internal to the global memory in Figure 2) moves blocks of data between memory banks at high speed without consuming bandwidth on the processor-memory channel. Such block copies occur in several circumstances. First, applications may need to replicate data structures. Second, memory banks may be functionally specialized (as discussed below), requiring bank-to-bank transfers of data that were received or generated in one bank, but must be sent to another bank for later processing. Third, the ARGOS operating system [9] written for GMMP architectures makes use of a page copier to accelerate message passing ("by value" semantics) on a global memory machine: large data structures are sent "by reference" from one process to another, but are flagged for copy-on-write protection at the ATran. If one of the processes with access to a copy-on-write-protected data structure changes that data structure, the change must not be visible to the other process (because their access environments must appear to be completely isolated). Therefore, such a change is intercepted before the (shared) memory copy of the data is modified; the affected page is then copied by the page copier and the copy is mapped into the virtual memory of the writing process, which then updates its (new) private copy.

### *C. VRAM Main Memory*

Some years ago, memory designers began

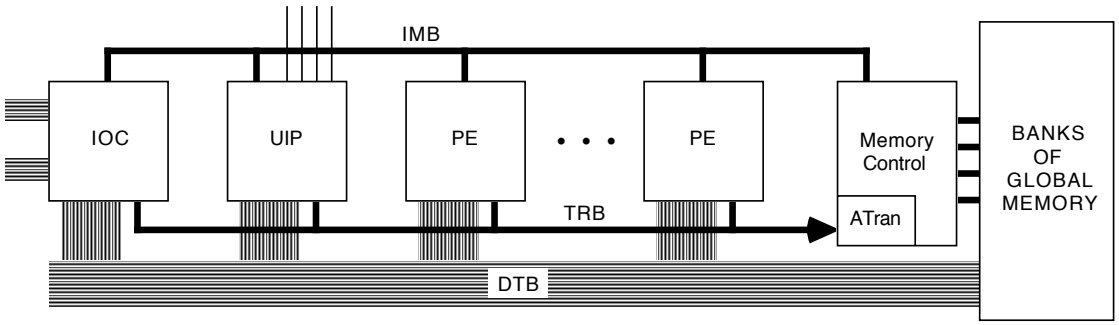


Figure 2: A Bus-Based Virtual Port Memory Machine

implementing a range of enhancements to standard DRAM designs which attempt to harness the greater bandwidth available on an integrated circuit than off-chip for various specialized applications. The so-called "video DRAM" or VRAM shown in Figure 3, adds to the basic DRAM array a (pseudo) shift register which can transfer an entire row to or from the array of DRAM cells in parallel. Due to pin limitations, this shift register only communicates off-chip serially, but it can be clocked at a somewhat higher rate than DRAM accesses can be completed. For example, a 100 ns VRAM (with a 180 ns cycle time) may have an access time from the sequential port of 25 ns, with a cycle time of 30 ns; thus, sequential access (SAM) throughput can be six times greater than random access (RAM) throughput in a pipelined memory system if the restriction to sequential access inherent in the shift register can be accommodated.

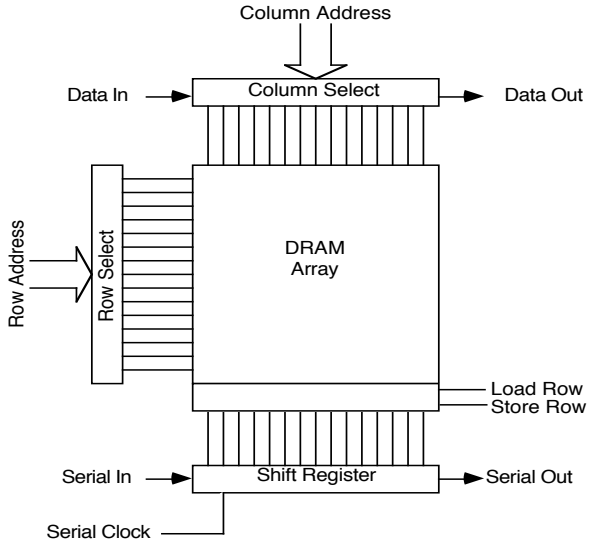


Figure 3: Conceptual Structure of a Video DRAM (VRAM)

In addition to the higher data bandwidth available from the SAM port, it can be used concurrently with the RAM port (except when a the shift register contents must be transferred to or from a row in the DRAM array), resulting in an interesting form of dual-port RAM with one random-access port and one sequential-access port. A recent innovation is the addition of a second SAM port to the structure in Figure 3, resulting in a triple-port DRAM with two high-bandwidth ports [12].

The key to using VRAM to boost multiprocessor performance is the identification of categories of memory access which fit the sequentiality inherent in its serial access mode. One interesting possibility for efficient use of the VRAM SAM port is DMA transfers of contiguous blocks of data to I/O controllers or to other locations in memory.

A Virtual Port Memory machine enhanced with VRAM for high-data-rate communication processing is shown in Figure 4. This machine uses Micron's triple-port DRAM in its main memory as follows: the RAM port is used by the processors for random accesses; one SAM port is used for I/O, while the other SAM port (not shown) is used for inter-bank page-copy traffic. With the 6:1 bandwidth ratio between the RAM port and each SAM port, it is intuitive that this enhanced Virtual Port Memory architecture is well suited for applications in which the rate of the data communication is roughly an order of magnitude greater than the rate at which processors access memory.

The Communication I/O Controllers (Comm IOC) shown in Figure 4 perform serial/parallel conversion and data link layer processing on the data. They communicate with the global memory via SAM ports, which are relatively wide (32 or 128 bit), and which are clocked at 32 MHz in this machine, to give a data rate of 1 - 4 Gbps per bank. One bank of memory is dedicated to each Comm IOC, with inter-bank data transfers taking place over the "page copy" bus. The PEs execute higher-level functions such as protocol conversion and adaptive routing. The disk IOCs manage the flow of data to and from disks for logging, short-term storage, and so on.

## III. PERFORMANCE ANALYSIS

The architecture shown in Figure 4 has been evaluated for several communication processing workloads using queueing network models and customized mean value analysis techniques. The model includes memory and bus contention within the multiprocessor, as well as the performance effects of cache misses. For the analysis reported here, the measure of performance used is the average time to process a packet, including time at the inbound IOC, a PE, and the outbound IOC.

The following three workloads are used: 32 channels of 10 Mbps each ("Ethernet"), 32 x 100 Mbps ("FDDI"), and 4 x 300 Mbps ("NASA"). Packet arrivals and sizes are assumed to be geometrically distributed, with a mean packet size of 4 Kbits. The rate of packet arrivals is varied from 0 to a rate approaching saturation of the communication channels for each workload (i.e., utilization of each channel, U, ranging from 0 to 0.5 or 0.9).

The I/O-to-processing ratio is set to 10 bits of I/O per instruction executed, which represents a significant amount of processing per packet. For each workload, the number of processors is varied, and the fraction "f" of each packet that is read by a processor is set to both 10% and 100%. For the Ethernet workload, relatively low-performance processors (2.5 MIPS) are sufficient to achieve satisfactory processing delays, while the other two workloads require 10 MIPS processors (e.g., a fast 68030) for acceptable performance.

A hypothetical machine using 100 MHz 100 MIPS RISC processors, with 256-bit buses and 50 MHz SAM ports can process 32 channels of 300 Mbps each with only 8 processors.

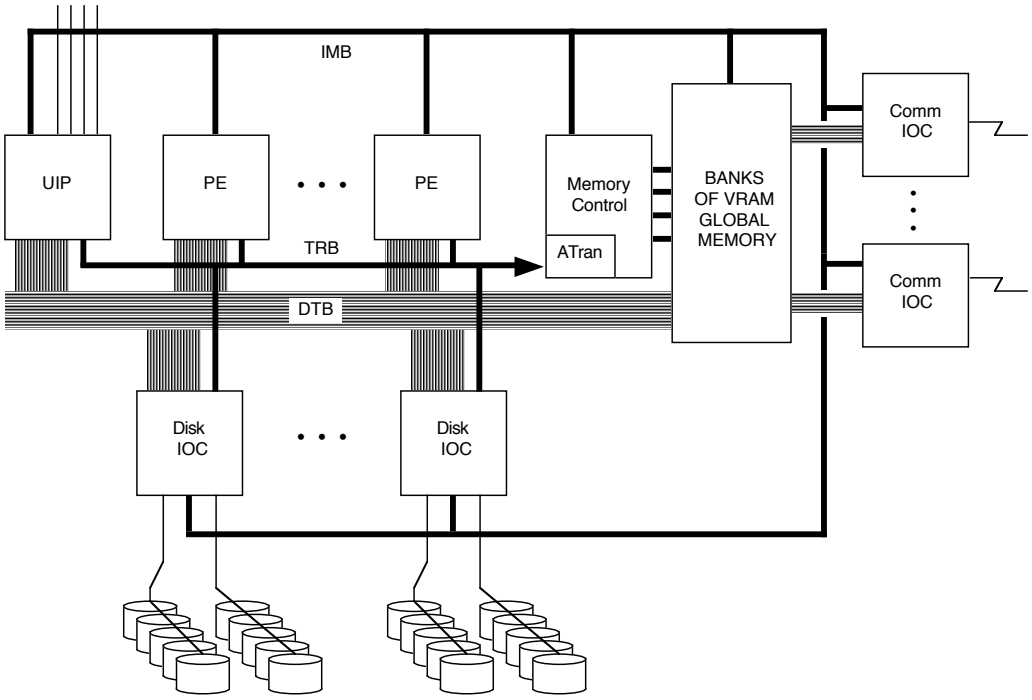


Figure 4: Enhanced Virtual Port Memory Architecture for Communication Processing

Ethernet: 32 channels at 10 Mbps each, with utilization of each channel ranging from 0 to 50%. Either 16 or 32 slow (2.5 MIPS at 16 MHz) PEs were used. The buses in the machine were 32 bits wide. The results are shown in Figure 5.

FDDI: 32 channels at 100 Mbps each, with utilization of each channel ranging from 0 to 90%. Either 32 or 64 fast (10 MIPS at 32 MHz) PEs were used. The buses in the machine were 128 bits wide. The results are shown in Figure 6.

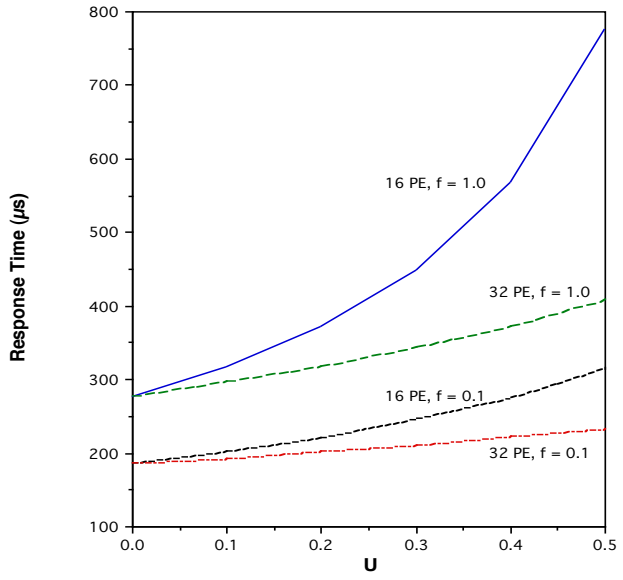


Figure 5: "Ethernet" Application – 32 x 10 Mbps (2.5 MIPS PE)

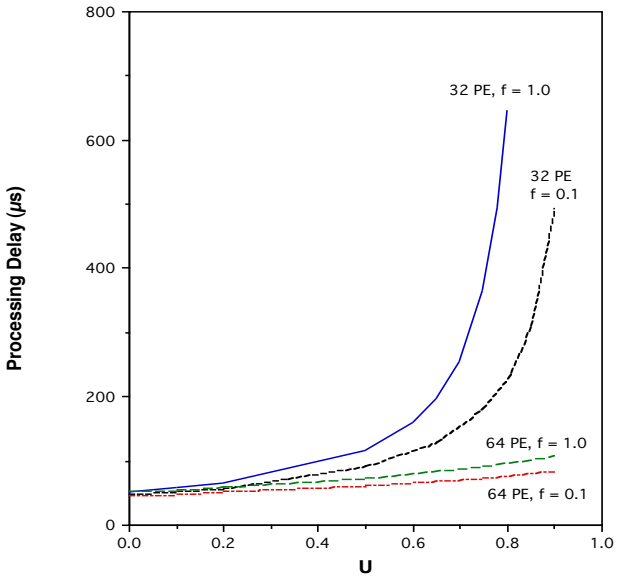


Figure 6: "FDDI" Application – 32 x 100 Mbps (10 MIPS PE)

NASA: 4 channels at 300 Mbps each, with utilization of each channel ranging from 0 to 90%. Either 16 or 32 fast (10 MIPS at 32 MHz) PEs were used. The buses in the machine were 128 bits wide. The results are shown in Figure 7.

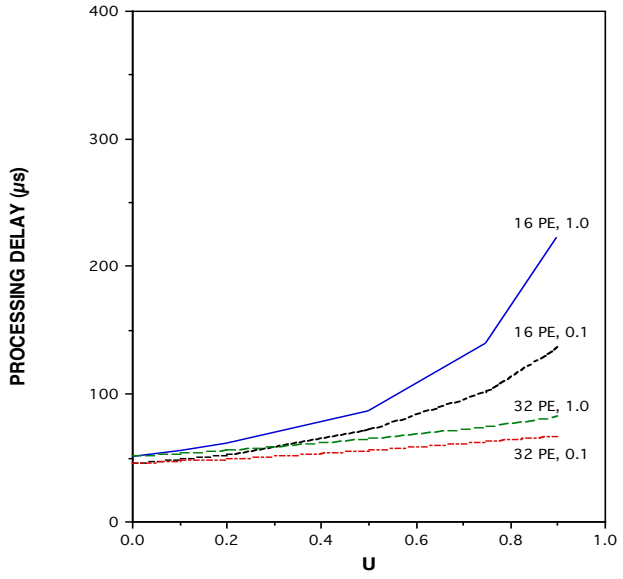


Figure 7: "NASA" Application — 4 x 300 Mbps (10 MIPS PE)

## IV. CONCLUSION

The Virtual Port Memory multiprocessor architecture, enhanced with VRAM global memory, appears to provide the requisite balance of processing and I/O capacity for high data rate communication processing. With one VRAM bank per communication channel, current technology supports data rates of several Gbps per channel.

Analysis of this enhanced architecture for a range of workloads shows that a small number of CISC processors (32 to 64) is sufficient to perform significant processing on packets with an aggregate arrival rate of over 1 Gbps. A machine with eight next-generation RISC processors increases this capacity by an order of magnitude.

## ACKNOWLEDGEMENT

Many people have contributed time and ideas to the development of this architecture. I'd especially like to acknowledge here the contributions of John Polson, a Ph.D. candidate at NMSU, who has helped to investigate communication processing applications on VPM architectures.

## REFERENCES

1. Amdahl, G.M., "Validity of the single processor approach to achieving large scale computing capabilities," *Proceedings, AFIPS 1967 Spring Joint Computer Conference 30* : 483-485, 1967.

2. Tobagi, F.A., "Fast Packet Switch Architectures for Broadband Integrated Services Digital Networks," *Proceedings of the IEEE,* **78**(1): 133-167, 1990.

3. Almasi, G.S. and Gottlieb, A., *Highly Parallel Computing,* Benjamin/Cummings, 1989.

4. *NCUBE Users Manual,* NCUBE Corp., Beaverton, Oregon, 1987.

5. Crowther, W. *et al.*, "Performance Measurements on a 128-Node Butterfly™ Parallel Processor," *Proceedings, 1985 International Conference on Parallel Processing* : 531-540, 1985.

6. Johnson, E.E., "The Virtual Port Memory GMMP Multiprocessor," *Proceedings, International Conference on Computing and Information* : 127-130, 1989.

7. Karp, A.H., "Programming for Parallelism," *IEEE Computer*, May 1987: 43-57.

8. Sager, G.R. and al, e., "The Oryx/Pecos Operating System," *AT&T Technical Journal,* **64**(1): 251-268, 1985.

9. Johnson, E.E., "ARGOS — A Research GMMP Operating System: Overview and Interfaces," Technical Report, NMSU-ECE-89-007A, NMSU, August 1989.

10. Hawkinson, S., "The FPS T Series: A Parallel Vector Supercomputer," Technical Report, Floating Point Systems, Inc., November 13, 1986.

11. Teller, P.J., "Translation-Lookaside Buffer Consistency," *Computer*, June 1990: 26-36.

12. Micron Technology, "Triple Port DRAM," data sheet for MT43C4257/8, October 1990.