Myrinet: A Gigabit-per-Second Local Area Network

The Myrinet local area network employs the same technology used for packet communication and switching within massively parallel processors. In realizing this distributed MPP network, we developed specialized communication channels, cut-through switches, host interfaces, and software. To our knowledge, Myrinet demonstrates the highest performance per unit cost of any current LAN.

Nanette J. Boden

Danny Cohen

Robert E. Felderman

Alan E. Kulawik

Charles L. Seitz

Jakov N. Seizovic

Wen-King Su

Myricom, Inc.

Myrinet is a new type of local area network based on the technology used for packet communication and switching within massively parallel processors. Think of it as an MPP network spanning campus-wide dimensions, rather than as a wide-area telecommunications network operating in close quarters. In creating Myrinet, we devised robust, 25-meter communication channels that provide flow control, packet framing, and error control. We also developed self-initializing, low-latency, cut-through switches. Myrinet's host interfaces map the network, select routes, translate network addresses to routes, and control packet traffic. Its streamlined software allows direct communication between user processes and the network. This work produced a new type of LAN that should significantly benefit distributed computing, image transport, and other communication-intensive applications.

Origins

Myrinet's genealogy explains how it differs from conventional LANs such as Ethernet and FDDI (fiber data distributed interface). Myrinet arose from two US Advanced Research Projects Agency-sponsored research projects. These were the California Institute of Technology Mosaic, an experimental, fine-grain multicomputer,[1] and the University of Southern California Information Sciences Institute (USC/ISI) Atomic LAN,[2,3] which was built using Mosaic components. Members of these two research projects founded Myricom, the startup company that developed Myrinet. (Myrinet, LANai, and MessageWay are trademarks of Myricom, Inc.)

Multicomputer message-passing networks. A multicomputer[4,5] is an MPP architecture consisting of a collection of computing nodes, each with its own memory, connected by a message-passing network. The Caltech Mosaic pushed the envelope of multicomputer design and programming by devising a system with up to tens of thousands of small, single-chip nodes, rather than hundreds of circuit-board-size nodes. The fine-grain multicomputer places more extreme demands on the message-passing network due to the larger number of nodes and a greater interdependence between the computing processes on different nodes. The message-passing network technology developed for Mosaic[6] achieved its goals so well that several other MPP systems adopted it. These included the medium-grain Intel Delta and Paragon multicomputers, the Stanford Dash (Directory Architecture for Shared Memory) multiprocessor, and the MIT Alewife multiprocessor.

In common with LANs, multicomputer message-passing networks send and receive data in packets. Any node may send a packet to any other node. A packet consists of a sequence of bytes starting with a routing header; routing cir-

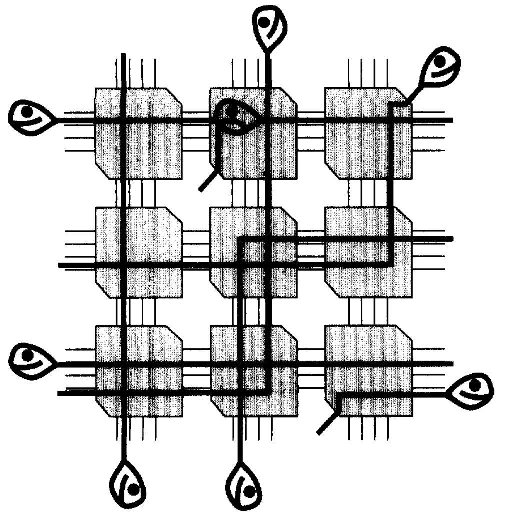


Figure 1. Packets flowing concurrently through part of a 2D-mesh MPP network.

cuits examine the routing header to steer the packet through the network. An arbitrary-length payload that follows the header includes the data delivered to the destination. A trailer or tail terminates the packet, and may include a checksum. Packet-arbitration fairness in routing circuits and buffer-size limits in the sending and receiving nodes may limit the packet's maximum length, which is also called the maximum-transmission unit. If a program sends a message larger than the MTU, the operating system fragments the message into a series of packets that the receiving node reassembles.

In contrast to LANs, the distinctive characteristics of MPP message-passing networks include high data rates, very low error rates, and flow control on every communication link.

High data rates. Channel data rates today range from hundreds to thousands of megabits per second. Message-passing networks commonly organize these individual channels in full-duplex pairs called links. A Myrinet link composed of a full-duplex pair of 640-Mbps channels thus becomes a 1.28-Gbps link, compared to 10-Mbps 10-Base or 100-Mbps 100-Base Ethernet links, which carry packets in only one direction at a time.

Regular topologies and scalability. Elementary routing circuits connected by links in a mathematically regular topology form the network. After the early hypercubes, most multicomputers adopted low-dimension topologies, such as the 2D mesh illustrated in Figure 1. In the figure, the packets injected at the lower left and ejected at the upper right of routing circuits are those to and from the computing nodes. As soon as the cut-through-routing circuit decodes the header, it advances the packet into the required outgoing channel if it is not already in use. Otherwise, the packet is blocked, as in the top-center node, until the outgoing channel becomes available. The *x*-then-*y* dimension-order routing illustrated eliminates cyclic dependencies in the routes, hence avoiding deadlock.

Unlike LANs such as Ethernet and FDDI, in which all packet traffic shares a single physical medium, a 2D-mesh network is scalable. The aggregate capacity grows with the number of nodes because many packets may be in transit concurrently along different paths. The regularity of the network allows simple, algorithmic routing that avoids deadlocks. See Scherson and Youssef[7] for a discussion of these technical issues.

Very low error rate. Inasmuch as a multicomputer message-passing network operates in an intracomputer environment, typically as an active backplane, bit errors or lost packets are extremely rare. Undetected communication errors are rarer still. This high reliability, in contrast with the usual assumption in LANs that communication is unreliable, has many implications. If a multicomputer's message-handling software were to assume unreliable physical communication, it would require more complex communication protocols to make end-to-end communication reliable. These protocols also require additional storage to keep temporary copies of sent packets. At MPP communication rates, executing such protocols would add significant overhead for each packet.

Cut-through routing. In an environment with reliable communication, the routing circuits can employ cut-through routing, an aggressive form of routing. Conventional store-and-forward routing buffers the entire packet and verifies its checksum in each intermediate node before sending the packet on the required outgoing channel. In cut-through routing, the packet advances into the required outgoing channel as soon as the header arrives and is decoded. In either case, the required outgoing channel may already be in use, meaning that the routing circuits must hold or block the packet until the channel becomes available.

Flow control on every communication link. If the required outgoing channel is already in use, a store-and-forward packet must remain queued in a routing circuit or node that presumably has a substantial amount of memory for packet buffering. In cut-through routing, the routing circuit can block a packet with flow control if the required outgoing channel is unavailable. In this way, the cut-through-routing circuit requires no packet buffering, although each link must provide flow control. To provide flow control, an MPP network commonly acknowledges each flow-control unit, which is typically a byte.

USC/ISI Atomic LAN. MPP routing networks and LANs evolved from different requirements, assumptions, and techniques. There is, however, a need for higher speed LANs. In the 10 years since Unix workstations first appeared on the market, processor speeds have advanced by at least a hundredfold; but most workstations still come with the same common-denominator, 10-Mbps Ethernet. New applications, such as the increasing use of distributed computing and the storage and communication of video, place additional demands on network performance, both for bandwidth and latency.

In 1991, a research group at USC/ISI used MPP components to construct a high-speed LAN. Within weeks of receiving several Mosaic host interfaces and a software toolkit, the Atomic (ATM over Mosaic) project team demonstrated a small network that performed standard TCP/IP (transmission-control protocol/Internet protocol) communication at burst rates within the network of 400 Mbps. The Mosaic interfaces, chained in one dimension, could address packets to any other node up or down the chain. The testbed grew to include the equivalent of crossbar switches based on 2D-mesh Mosaic multicomputer arrays. To deal with the irregular topology of chains of varying length connected through switches, Atomic added automatic network mapping and translation from network addresses to routes. Mapping and address-to-route translation were the key insights needed to adapt an MPP network to a LAN environment.

The Atomic testbed also demonstrated an experimental result of particular interest: the transfer of more than 10^{15} bits without a single bit error or dropped packet. Although Atomic was an innovative research project, the Atomic testbed had several practical limitations.

1. The communication links employed asynchronous request/acknowledge signaling, designed for distances up to 1 meter. The links operated correctly over much longer distances, but at progressively slower data rates due to the flight time of the asynchronous signals. In addition, the absence of an acknowledgment on a disconnected channel caused any packet directed to this channel to block, leading eventually to deadlock of the entire network.
2. The network topology—1D chains of interfaces connected by 2D-mesh multicomputers used as switches—was complex for network mapping. It did not permit powering off hosts within chains. No good use surfaced for the substantial computing power and memory hidden in the Atomic switches.
3. The lack of a DMA engine in the host interface limited Atomic's performance. However, the ability of the interface to execute a complex control program was crucial for managing the network interface.
4. The TCP/IP protocol stack in the host operating system, not surprisingly, limited the end-to-end data rates that the network could achieve.

Components and operation

We based Myrinet's design directly on the Atomic experiences, good and bad. There was, however, no constraint for Myrinet to use an existing MPP network. Instead, our group developed communication schemes, routing techniques, custom-VLSI chips, and network-control software specifically for the LAN environment, while still drawing heavily on MPP technology. The specifications governing Myrinet LAN operation are published and open.[8]

As Figure 2 shows, a Myrinet LAN consists of point-to-point, full-duplex links that connect hosts and switches. The multiple-port switches may connect by links to other switches and to the single-port host interfaces in any topology, including those with cycles.

Electrical cable links. The standard Myrinet cable has 18 twisted pairs, nine in each direction, and is driven and sensed in a balanced, differential mode with ~1V signals and ~2V differences. Physically, the cable is ~1 cm in diameter, shielded, flexible, and UL-approved type CL-2. These load-terminated transmission lines have a characteristic impedance of ~105 Ω, and a propagation velocity of ~0.6 *c*.

Transmission is synchronous at the sending end at a rate of 80 million 9-bit characters/s. The 9-bit character may be either an 8-bit data byte or one of five control symbols. The signal encoding is non-return-to-zero (NRZ), in which a signal transition encodes a binary 1, and the absence of a transition encodes a binary 0. The all-zeros character is the Idle control symbol, which is discarded by not being detected.

The receiver circuits are asynchronous; a character may arrive at any time relative to the clocks in the receiving system. Pipeline-synchronizer circuits later synchronize that character

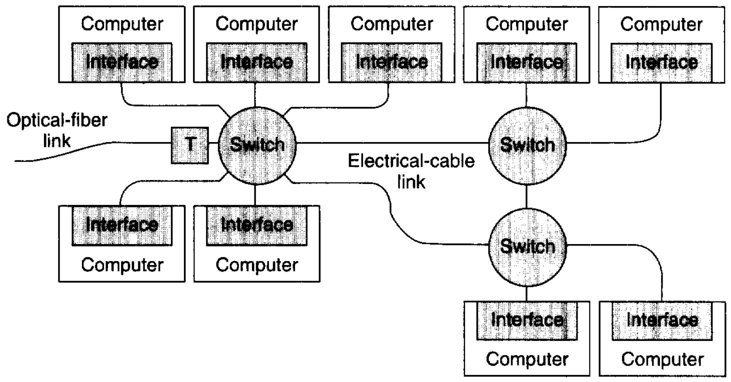


Figure 2. A possible Myrinet LAN configuration.

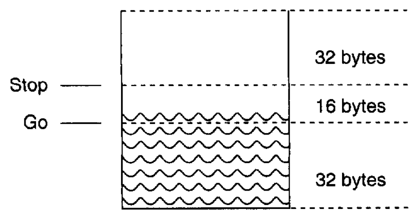


Figure 3. Slack buffer operation.

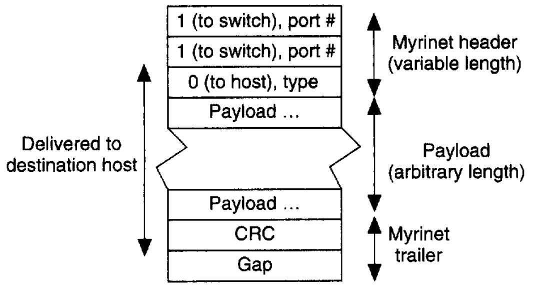


Figure 4. Myrinet packet format.

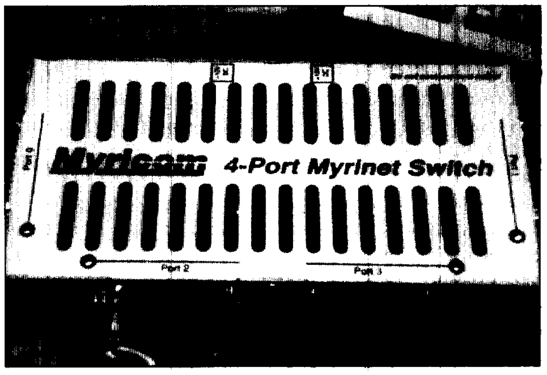


Figure 5. A 4-port Myrinet switch. Note the keyboard for scale. This switch has a bisection bandwidth of 2.56 Gbps, and consumes 15W.

to the clocks.[9] Delay variations and crosstalk between cable pairs and delay variations between line drivers and receivers skew the transitions at the receiving end of a channel. The receiver circuits employ a sampling-window technique that tolerates skew up to half the 12.5-ns character period. These conventional encoding and sensing techniques produce a bit-error rate well below 10^{-15} on cables up to 25-m long. The signal distortions—some of which are quadratic with length—and cumulative skew make low-error-rate operation over longer distances progressively more difficult. Consequently, we chose 25 m as the maximum length for electrical cables. Cable repeaters or the optical-fiber links described later let Myrinet span longer distances.

Myrinet manages flow control by having the receiver inject Stop and Go control symbols into the opposite-going channel of the link. Up to 23 characters can be in transit on a round trip of a 25-m Myrinet cable, and the Stop and Go symbols may require several character periods to generate and decode. The receiver, accordingly, includes a queue-organized slack buffer that operates conceptually as pictured in Figure 3. If the downstream flow is blocked so that the slack buffer fills to the Stop line, the receiver generates a Stop control symbol that stops the flow before the buffer overflows. When the downstream flow resumes, the receiver generates a Go control symbol when the level reaches the Go line. The top part of the buffer prevents overruns; the bottom part prevents data starvation. The buffer positions between Go and Stop provide hysteresis to assure that Stop and Go control symbols will not consume excessive bandwidth on the opposite-going channel.

The Gap control symbol marks the end of a packet. It may, along with Go and Stop, appear redundantly. The sender must emit a non-Idle character periodically as a mechanism analogous to carrier sensing for detecting open links. Also, long-period time-outs detect packets blocked for more than ~50 ms, as may occur if a software error or a bit error in a header causes deadlock. This long-period time-out drops the blocked part of the sender's packet and sends a forward-reset (Fres) control symbol to the receiver.

These Myrinet links completely resolved the first set of practical limitations of the Atomic LAN.

Packet format and routing. Figure 4 illustrates the format of a Myrinet packet. When a packet enters a switch, the leading byte of the header determines the outgoing port before being stripped off the packet. When a packet enters a host interface, the leading byte identifies the type of packet: mapping packet, network-management packet, a packet with an IP packet as its payload, or data carried by a lightweight protocol. The most-significant bit of each header byte distinguishes between to-switch and to-host bytes. If all packets traveled known routes, this bit would be redundant. However, the redundancy in the header encoding simplifies network mapping by allowing switches to drop to-host packets, and interfaces to detect and deal with faults that cause misrouting.

The payload of a Myrinet packet is of arbitrary length; hence, it can carry any other type of packet—such as an IP packet—without an adaptation layer.

The Myrinet-packet trailer carries an 8-bit cyclic-redundancy-check character computed on the entire packet. Because the packet header is modified at each switch, Myrinet recomputes the CRC on each link. If the CRC on a packet is incorrect when it enters a switch, it will be incorrect in the same bit positions when it leaves. Thus, if there is an error on any link on a routing path, the error can be detected at the destination.

Switches. Myricom is currently shipping 4- and 8-port switches, and developing 16- and 32-port switches. These switches employ exactly the same blocking-cut-through (wormhole) routing[6] used in the message-passing networks of such MPP systems as the Intel Paragon and the Cray T3D. The worst-case path-formation latency through an 8-port switch is 550 ns. The core of the switch is a pipelined crossbar that introduces no internal conflicts between packet flows. In addition, recirculating-token arbitration assures fairness (no head-of-line priority problems).

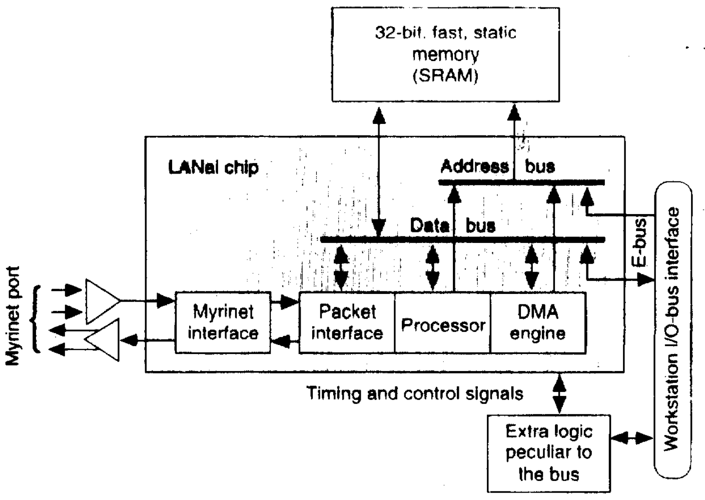

32-bit, fast, static memory (SRAM)
LANai chip
Address bus
Data bus
Myrinet port
Myrinet interface
Packet interface
Processor
DMA engine
E-bus
Workstation I/O-bus interface
Timing and control signals
Extra logic peculiar to the bus

Figure 6. Host interface block diagram.

Two custom-VLSI chips in 0.8-μm CMOS technology are the basis for the switches. The crossbar-switch chip forms the switch itself. The dual-Myrinet-interface chip performs the autonomous parts of the Myrinet-link protocol, including flow control. CMOS technology provides ample headroom for continued evolution of these switches.

Figure 5 shows a 4-port switch. The principal design criteria for these switches were that they be physically small, low-power, self-initializing devices that can reside in wiring closets, over ceilings, or in any other location convenient for cabling the network. This 4-port switch requires ~15W of +12V ±3V power from either or both of two supplies, making battery backup easy to provide. These switches operate correctly at ambient temperatures up to 55°C. Because the switch contains only transient state and no software, no security issues arise concerning which hosts can download programs or routing tables into the switch. The switch simply steers packets.

The Myrinet packet format and switches have entirely resolved the second practical limitation of the Atomic LAN.

So much has been written and argued about MPP-network topologies—hypercubes, meshes, tori, fat trees, and others—that the question of how to classify Myrinet often arises. The Myrinet topology is arbitrary, but is based on high-degree switches. To make a specific performance comparison, let's look at how well Myrinet can simulate or replace a 2D mesh.

A single 32-port-switch chip (~900 pins) could replace the 16 mesh-routing chips (132 pins each) in the 4x4 routing module that serves 16 nodes in the Intel Delta multicomputer. Sixteen switch links would connect to the nodes, and 16 would simulate the edges of the 4x4 mesh. The high-degree crossbar switch would replace 16 chips with one, halve the total pin count, introduce fewer internal conflicts, and reduce the diameter of the network. Indeed, since the external links could be long, we would not even need to connect the 16-node modules in a mesh, but could use a lower diameter network. The precise details of the topology have less influence on network performance than the degree of the switches that form the network.

Host interfaces. Figure 6 shows the organization of a host interface and the internal details of the custom-VLSI LANai chip on which the interface is based. This microarchitecture provides a flexible and high-performance interface between a generic bus called the E-bus and a Myrinet link. Myricom is shipping Myrinet/Sbus interfaces for Sun Sparcstations—a postcard-size circuit board, thanks to the level of integration provided by the LANai chip—and is developing interfaces to several other buses, including PCI.

The static RAM stores the Myrinet Control Program (MCP) and packets. This 32Kx32 memory is accessed twice in each clock period, once by the E-bus and once by the processor or packet interface. Because E-bus accesses are not arbitrated, the LANai as viewed from the E-bus is synchronous memory, and can reside as a bus slave on most 32-bit memory or I/O buses. In addition, when using the DMA engine to provide addresses, the LANai can act as a bus master to transfer data blocks between the E-bus and SRAM. The DMA engine also computes the Internet checksum of the data it transfers.

Myrinet host interfaces retain all of the best characteristics of the Atomic host interfaces. The addition of the DMA engine with Internet-checksum computing resolves the third practical limitation of the Atomic LAN.

Optical-fiber interfaces. Myrinet optical-fiber interfaces connect on one side to an electrical Myrinet port and on the other side to a fiber-pair link. In terms of the ISO Reference Model for computer networks, the optical-fiber link and interfaces can act both as a level-2 bridge and as a level-3 router.

A LANai chip, its associated memory, and a specialized version of the MCP handle the optical-fiber interface's electrical Myrinet port, and provide buffering for the ~1500 bytes per kilometer of slack required to maintain flow control over the optical link. When two optical-fiber interfaces connected by fiber serve as a level-2 bridge, they maintain the same logical characteristics as a Myrinet link.

However, optical-fiber links may connect different Myrinets, for example, in different buildings. Each host in one building may not need to maintain a map of the network in another building, so the optical-fiber interfaces allow the networks to be separate. As a level-3 router that employs a MessageWay protocol, an optical-fiber interface participates in the mapping of its own network, but advertises itself as a route to another Myrinet. A packet routed to an optical-fiber interface may then contain an address in another network, and the translation from this address to a route in the other network completes after the packet gets across the optical-fiber link.

Software

Two categories of software provide access to and control a Myrinet: the MCP executing on the processors in the host interfaces, and the device drivers and operating systems executing in the hosts.

MCP. The device driver initially loads the MCP when the host boots up. The MCP starts executing as soon as the device driver releases a reset signal, and thereafter interacts concurrently with its host and the network.

On the host side, a set of single-producer, single-consumer command and acknowledgment queues control the interface. A typical command from the host is for the MCP to

- perform a gather operation on a set of data blocks at specified host addresses and word counts,
- generate the Internet checksum on the resulting packet and insert it in a specified location in the packet,
- send the resulting packet to the host that has a specified network address, and
- signal the completion of these operations in an acknowledgment queue and, optionally, by producing an interrupt for the host.

Performing such a command involves controlling the DMA engine, a translation from a network address to a route, prepending the route and packet type to the outgoing packet, and controlling the packet interface. In the receiving direction, the MCP checks the validity of the incoming packet, interprets headers, and transfers packet data to specified scatter buffers in the host memory. It then signals the arrival of a packet in an acknowledgment queue and optionally by producing an interrupt for the host.

Interleaved with these interactions with the host, the MCP performs continuous mapping, monitoring (remapping), and route selecting that makes the network self-configuring and self-healing. It may also perform store-and-forward multicasting. One host interface in each Myrinet maps the network by sending mapping packets to other interfaces and to itself. Either the network manager intervenes to manually select the mapper, or Myrinet itself does so automatically. In particular, if the mapping interface's host is turned off, or if faulty links cut the Myrinet into disjoint networks, Myrinet automatically selects one or more new mappers. The mapping interface distributes that map of the network to the rest of the network. Each interface then computes the routes from itself to all other interfaces; these routes are guaranteed to be deadlock-free.

The only difficult part of the mapping algorithm is identifying switches that do not connect to a host. Host interfaces have a network address that identifies them uniquely, but switches do not. If a switch connects even to one host interface, it can be identified easily. If a switch connects only to other switches, however, the mapper must infer the identity of a switch from the routes through it. The details of these operations are beyond the scope of this article. Myricom regularly provides the MCP source code and programming tools to its research customers so that they may tailor interfaces to their own needs. As part of our policy of making Myrinet interfaces open, we plan to eventually publish the MCP source code.

In the interest of network security, the MCP includes no back doors that might allow another network host to substitute a different version of the MCP. Indeed, the segment of interface memory in which the MCP resides is write protected from the MCP itself and can load only from the host.

Host software. The Myrinet host software provides the interface between Unix user processes and the host-interface board. Myricom delivers both a standard Internet Protocol interface, and a streamlined Myrinet application programming interface (API). Figure 7 is a copy diagram that illustrates the steps involved in moving information from a user process to the network. For receiving rather than sending, just reverse the arrows.

In the standard, one-copy TCP/IP interface, Unix copies the specified block from user space to kernel space, computes the checksum either on a separate pass through the block or as part of the copy operation, and gives the host interface a command to send the packet. The MCP copies the data to the LANai memory using the DMA engine, then transfers the packet to the network using the packet interface. The data moves at least three times, but the DMA and packet-interface operations are (nearly) free for the host. Consequently, programmers call this scheme a one-copy implementation. In

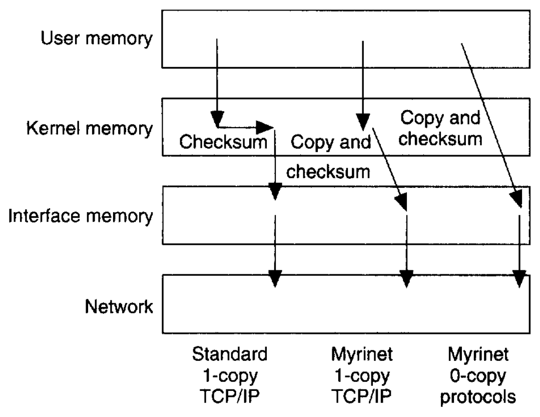


Figure 7. Copy diagram for different software-interface implementations.

operating-system implementations that allow it, the Myrinet host interface can off-load the computation of the Internet checksum from the host. Indeed, we designed the Myrinet host interface to allow a zero-copy mode of operation directly from user space. Workstation manufacturers, however, do not distribute a version of the TCP/IP or UDP/IP protocol stack that can fully exploit the interface capabilities.

Approaching the performance of the I/O bus, let alone of the network, requires avoiding Unix system calls. The Myrinet API sets itself up by system calls that allocate a number of unswappable pages of memory used as a data-exchange area, and thereafter avoids Unix system calls. User processes can manage the command and acknowledgment queues using a library of Myrinet-API functions.

Table 1 tabulates several benchmarks performed on a Myrinet connecting Sparc-2 workstations. The Sparc 2 has a 20-MHz SBus, but supports 16-word burst transfers, so the DMA performance surpasses 25-MHz SBus machines implementing only smaller burst sizes. A raw speed-test program achieves one-way, end-to-end rates on 8-Kbyte data blocks that are nearly as fast as the SBus. Myrinet API functions achieve one-way, end-to-end rates of 250 Mbps on 8-Kbyte blocks, a performance level limited by the number of instructions the interface's processor executes to handle each packet.

The Myrinet-API performance between Unix processes on workstations compares favorably with those between nodes of commercial MPP multicomputers. Standard UDP/IP-interface performance with the hardware handling the Internet-checksum computation is less than one tenth of the channel bandwidth, but is high enough to satisfy the needs of many applications. We achieved the zero-copy TCP/IP result with an experimental protocol stack developed by researchers at Sun Microsystems.

Table 1. Myrinet performance measured using a Sparc-2 system with a 20-MHz Sbus. Measurements were taken on August 4, 1994.

Parameter	Data rate (Mbps)
Channel data rate	640
DMA data rate	444
Raw speed test	380
Myrinet API (8-Kbyte packets)	250
TCP/IP (0-copy, hardware checksum)	70
UDP/IP (1-copy, hardware checksum)	55

The Myrinet software deals with certain aspects of the fourth practical limitation of the Atomic LAN, notably by the MCP handling much of the network control in the interfaces themselves, and by the streamlined access provided by the Myrinet API. To reach the full potential of fast networks, however, workstation and PC manufacturers must provide faster I/O buses, and particularly operating systems that exploit the capabilities of interfaces interacting directly with user processes.

TO OUR KNOWLEDGE, MYRINET demonstrates the highest performance per unit cost of any current LAN. Features such as self-configuration and fault tolerance also make it useful as an I/O fabric, high-reliability MPP network, or platform bus. As might be expected from its MPP genealogy, Myrinet is ideal for cluster-computing applications. With the addition of optical-fiber links, Myrinet also provides high-bandwidth, low-latency, low-error-rate communication of mixed packet types on networks up to kilometers in diameter. Myrinet does not need to scale beyond the cabinet-to-campus range of diameters to be useful. For larger areas, designing gateways between Myrinet and wide area networks is straightforward. For smaller areas, we can continue to use buses.

References

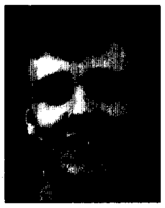
1. C.L. Seitz et al., "The Design of the Caltech Mosaic C Multicomputer," *Proc. Univ. Washington Symp. Integrated Systems,* MIT Press, Cambridge, Mass., 1993, pp. 1-22.
2. R. Felderman et al., "Atomic: A High Speed Local Communication Architecture," *J. High Speed Networks,* Vol. 3, No. 1, 1994, pp. 1-29.
3. G.G. Finn, "An Integration of Network Communication with Workstation Architecture," *Computer Comm. Rev.,* Oct. 1991.
4. W.C. Athas and C.L. Seitz, "Multicomputers: Message-Passing Concurrent Computers," *Computer,* Vol. 21, No. 8, Aug. 1988, pp. 9-24.

5. C.L. Seitz, "Multicomputers," in *Developments in Concurrency and Communication,* Chapter 5, C.A.R. Hoare, ed., Addison-Wesley, Reading, Mass., 1990.
6. C.L. Seitz and W-K. Su, "A Family of Routing and Communication Chips Based on the Mosaic," *Proc. Univ. Washington Symp. Integrated Systems,* MIT Press, 1993, pp. 320-337.
7. *Interconnection Networks for High-Performance Parallel Computers,* I.D. Scherson and A.S. Youssef, eds., IEEE Computer Society Press, Los Alamitos, Calif., 1994.
8. "Myrinet Link and Routing Specification," http://www.myri.com/ myricom/documents.html.
9. J.N. Seizovic, "Pipeline Synchronization," *Proc. Int'l Symp Advanced Research in Asynchronous Circuits and Systems,* CS Press, 1994.

Nanette J. Boden was a post-doctoral research fellow at Caltech before founding Myricom. Her research interests focus on concurrent programming ranging from the applications level to the design and implementation of concurrent, low-level operating systems. Boden earned her BS in applied mathematics from the University of Alabama (Tuscaloosa), and her MS and PhD degrees in computer science from Caltech.

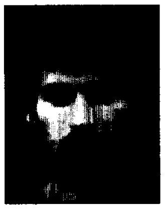
Danny Cohen has served on the computer science faculties of Harvard, Technion, and Caltech, and has done research at USC/ISI during the 19 years prior to founding Myricom. His research interests include real-time graphics, simulations, real-time networking and internetworking, and VLSI. Cohen earned his BSc in mathematics from Technion, and his PhD in computer science from Harvard.

Robert E. Felderman spent two years at USC/ISI working on the Atomic LAN project before founding Myricom. He is an adjunct assistant professor in the Computer Science Department at University of Southern California. His research interests include parallel and distributed systems, high-speed networks, and performance modeling and analysis. Felderman earned his BSE in EECS at Princeton, and his MS and PhD in computer science at the University of California at Los Angeles.

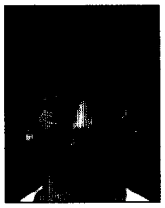


Alan E. Kulawik did research at Caltech on high-performance networks before founding Myricom. His research interests focus on concurrent computers and communication. Kulawik earned his BS in electrical engineering from Caltech.

Charles L. Seitz served on the Caltech computer science faculty during the 16 years prior to founding Myricom. His research interests include computer architecture, VLSI, and self-timed systems. Seitz earned BS, MS, and PhD degrees from MIT. He was elected to the National Academy of Engineering "for pioneering contributions to the design of asynchronous and concurrent computing systems."



Jakov N. Seizovic was a research engineer at Caltech prior to founding Myricom. His research interests include VLSI design and concurrent object-oriented programming systems. Seizovic earned his BS in electrical engineering from the University of Belgrade, Yugoslavia, and his MS and PhD in computer science from Caltech.



Wen-King Su served as a member of the professional staff at Caltech's Computer Science Department until founding Myricom. His research interests include hardware and software designs for concurrent computers and networks. Su earned his BS in computer and electrical engineering from the University of California at Davis, and his MS and PhD in computer science at Caltech.

Send correspondence about this article to Charles L. Seitz, Myricom, Inc., 325 N. Santa Anita Avenue, Arcadia, CA 91006; chuck@myri.com.

Reader Interest Survey

Indicate your interest in this article by circling the appropriate number on the Reader Service Card.

Low 159 Medium 160 High 161