For: finn
Printed on: Tue, Aug 23, 1994 15:27:00

Document: Netstation-CCR-OCT-91
Last saved on: Fri, Dec 11, 1992 10:29:59

An Integration of Network Communication with Workstation Architecture

Gregory G. Finn
USC/Information Sciences Institute

Published October 1991
ACM Computer Communication Review

An Integration of Network Communication with Workstation Architecture

Gregory G. Finn [1]
USC/Information Sciences Institute

Abstract

A workstation may be thought of as a group of cooperatively connected subsystems. Point–to–point channels may be used to create a small–scale Gigabit LAN to which these subsystems are attached as nodes. The architectural focus of such a workstation shifts towards its internal LAN. An attractive attribute of this LAN is that its aggregate capacity scales linearly with the number of nodes attached to it.

If the link–layer of the internal LAN is made equivalent to the link–layer of the external LAN, interior nodes become directly accessible externally. Except for latency the distinction between whether a node is inside a workstation versus outside it need not be significant. This property is particularly attractive for distributed communication–intensive applications.

1. Some Architectural Limitations to Communication Performance

Communications resources are centralized in traditional workstation architecture. Typically there exists one network interface, one protocol processor and a system–wide bus to distribute data. This creates bottlenecks for communications–intensive applications. Although these architectural bottlenecks are not a severe problem when LAN bandwidth is 10 Mbits/sec, they become severe as bandwidth rises to the Gbit/sec range.

1.a. *Protocol Processing*

Packets entering a typical workstation pass through a single network interface. They are then read by the central processor that copies them, executes the appropriate protocols and eventually delivers them. This centralization of hardware and software is a hindrance to performance. There is little reason for network video data on its way to a display to pass through a processor that does not need to see it, nor should external traffic intended for processors within a multiprocessor require that one processor act as a filter for all.

For a one Gbit/sec link and an average packet length of 3000 bytes, it is estimated that 60–100 MIPs is the minimum processing capacity required [1].[2] However, LAN bandwidth can advance well beyond the one Gbit/sec range. An additional problem is provision of sufficient internal bandwidth between network interface, memory and processor. A packet is stored into memory and then its data is examined at least once by a destination process. Many operating systems also copy a packet at least once, either from device to kernel

---

This research was sponsored by the Defense Advanced Research Projects Agency under Contract No. DABT63–91–C–0001. Views and conclusions contained in this report are the author's and should not be interpreted as representing the official opinion or policies, either expressed or implied, of DARPA, the U.S. Government, or any person or agency connected with them.

[1] University of Southern California/Information Sciences Institute
4676 Admiralty Way
Marina Del Rey, CA 90292 (Internet address: Finn@venera.isi.edu)

memory or from kernel to user memory. For the single–processor architecture it is unrealistic to expect to sustain a one Gbit/sec network load without three Gbits/sec of available internal bandwidth.

One approach already taken toward these problems is to design specialized pipelines or accelerator boards to process network traffic [2][3][4]. Others point out that rigid layer separation between protocol software and application remains to inhibit performance [5]. It is also true that centralized protocol processing does not adapt well to multiprocessor systems. If network processing were located at each network traffic consuming or producing device, distributed throughout a workstation's architecture, the need for extreme processor speed and memory bandwidth could be greatly lessened.

1.b. *Data Transfer*

A bus is the mechanism commonly used within workstations to distribute data between peripherals, memory, and processors. Buses feature a broadcast data channel that supports one–at–a–time access. At any one time there may be at most one transmitter. The bandwidth of a bus does not uniformly scale. If demand becomes too high, one must modify the architecture to either use another variant of the bus or use another bus entirely.[3]

Distributed capacitances inherent in tapped transmission lines and propagation delay are physical limitations that set hard upper limits on bus performance. Increasing the number of data lines in a bus increases its bandwidth but increases the cost of an interface. Shortening a bus or decreasing the number of taps also allows increased bandwidth but decreases the number of devices that can attach to it.

Point–to–point technology connects adjacent senders and receivers by dedicated channels. This reduces the number of taps on a channel to the minimum of two with a corresponding increase in performance over a bus that uses similar circuit technology. Multiple transfer operations can occur simultaneously on separate channels, so that aggregate bandwidth scales linearly with the number of channels that comprise the network.

Point–to–point technology has been used to build LANs, but providing transfer performance equivalent to a bus was not previously a realistic goal [4][6]. MOSAIC is a point–to–point technology that supports Gbit/sec channels, while the Scalable Coherent Interface, (SCI), supports eight Gbits/sec channels [7][8]. Both support the automatic routing and flow–control of packets in hardware. From a functional perspective it is now reasonable for a point–to–point internal LAN to substitute for a system–wide bus for the purpose of distributing data and control messages between processors, peripherals, and external networks. It is not reasonable to substitute this LAN for the processor–memory transfer functions of a bus.

2. Netstation Architecture

A Netstation is a set of network nodes that function cooperatively as a workstation. A node consists of three parts: LAN Interface, Protocol Processor with memory and a device–specific part. Each node presents to the network an interface to some network–mapped virtual device. Nodes communicate with one another by exchanging messages over their internal or external LAN. A Netstation may be characterized as a heterogeneous message–passing multicomputer.

The Netstation Architecture shares some characteristics of the Nectar effort, but it differs markedly in its scale of application [4]. In a Netstation the LAN is an integral part of the internal architecture of the workstation. The network interface is small, typically a single chip, is capable of protocol processing and must support a transfer rate roughly equivalent to the memory access rate of its attached node.

[2] The MIPs figures offered for super–scalar microprocessors should be approached with caution. Rarely are parallel floating–point or multiply instructions used in protocol processing.

[3] Buses such as the VME and Futurebus support several width variants. This is not scaling in the sense meant here. To increase bandwidth by using a wider variant requires that attached boards match the wider variant.

Each node resides on a LAN and can potentially be accessed by any other node on that LAN and its resources shared. The set of nodes that comprise a Netstation need not be static. If latency between certain nodes must be kept very low for a particular application, those nodes must be physically close to one another, but that is not necessary in general.

2.a. *Internal LAN Characteristics*

The characteristics that make a technology appropriate for use in an internal LAN are similar to those that make it good to use in message–passing multicomputers [7][9]. The LAN must exhibit low latency and high reliability, and it must have better scaling characteristics than a bus. Aggregate LAN bandwidth should scale linearly with the number of attached nodes. Interfaces to it must be small and inexpensive since a Netstation could have several.

The overhead of message routing among nodes must be very small, demanding little if any node resources unless that node is either the source or destination for a message. Nodes should be able to communicate with one another over dedicated channels if needed. This implies support for a flexible topology.

The indefinite number of nodes that can be associated with a Netstation suggests the use of globally unique node addresses. However, global address management and routing can then become problems. Physical addresses can be avoided if a destination is 'addressed' only by its position with respect to the source. Source routing avoids the need for globally unique addresses and has also been identified as a way to obtain a substantial performance gain [10].

A variable packet size is also desirable. The unit of communication between nodes should be appropriate for the application. For example, a unit for communication between a camera and a display might be data for a frame line. Unnaturally sized units increase overhead. Fragmentation and reassembly are computationally expensive.

2.b. *Internal and External LAN Equivalence*

Translation between the internal and external LAN occurs if the two have different link–layer packet formats. The translation function is normally performed by a gateway. This both incurs a latency penalty and recentralizes a great portion of network processing. To prevent this, it is sufficient that both internal and external LANs be link–layer equivalent. It is this equivalence that provides some unique and attractive properties to this architecture.

3. MOSAIC Used as a Point–to–Point LAN Technology

MOSAIC technology was developed by Dr. Charles Seitz's group at Caltech to allow construction of supercomputers that are adapted for massively parallel fine–grain computation. While SCI technology is not oriented toward LAN design, MOSAIC can be used to create LANs that support very large aggregate bandwidth.

3.a. *MOSAIC Nodes*

A MOSAIC–C node is a single chip, containing a 14 MIPs microprocessor, 64K bytes of RAM and 2K bytes of ROM [11]. See Figure 1. It communicates over eight external channels, four in the X–direction and four in the Y–direction. All eight channels may be active simultaneously. Software can be remotely loaded via incoming channels. Channels are nominally rated at 0.64 Gbit/sec, although prototypes operate at 0.88 Gbit/sec. MOSAIC technology can scale upwards, allowing channel speeds well in excess of one Gbit/sec.

Unless a MOSAIC–C node is the source or destination for a message, messages pass through its Asynchronous Router logic on their way to other nodes without interrupting the processor. When a node is either source or destination, packet data is transferred by a DMA controller in the Packet Interface. The

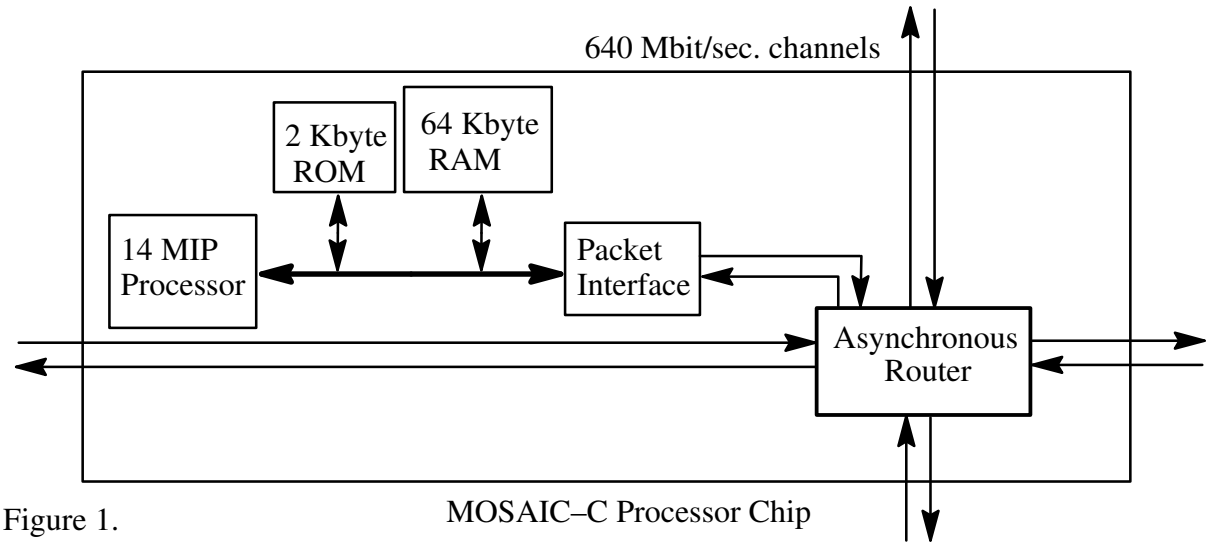

Figure 1.

Asynchronous Router and Packet Interface logic occupies a small fraction of a MOSAIC chip's area. One could include that as part of complex application specific circuits, such as a DSP chip.

A MOSAIC chip adapted to peripheral interfacing also exists. It is distinguished by having an external memory bus and support for fewer channels. This MOSAIC Interface chip uses a 128K external dual–ported memory for program, data and message storage. These two types of components are necessary and sufficient for a Netstation Architecture.

Each MOSAIC chip has a processor and associated memory. This processing capability can be utilized to filter messages, execute protocols, and arrange that data be delivered in the form expected by an application or virtual device specification. A supervisor can control, reload or augment MOSAIC nodes dynamically by sending them messages.

MOSAIC components interface directly to their communications medium. No additional circuitry is required to interconnect two nodes as long as wire lengths between them are kept to within 50cm.[4] It is this characteristic that makes Netstation Architecture practical.

To transmit MOSAIC channel data over distances typical of LANs, it is possible to use bit–serial fiber–optic cables. The design of such a cable is not a trivial task. The MOSAIC channel control signals plus data signals require a data transfer rate approaching 2 Gbits/sec. Currently, this requires that the serial/parallel and parallel/serial conversion take place in non–CMOS technology. The design of an inexpensive laser transmitter that can operate at multi–gigabit rates, air–cooled in an office environment is also a challenge. Work is proceeding on both these issues.

3.b. *Comparison to a Bus*

Figure 2 depicts several messages sent simultaneously between nodes that are interconnected by MOSAIC channels. Channel segments may be used simultaneously. A message sent from node 1 to node 7 interferes neither with messages sent from node 4 to node 6 nor from node 5 to node 2. Neither does the message sent from node 8 to node 9 provide interference. Although the chain of MOSAIC channels 1–6 is topologically similar to a bus with six taps, its bandwidth scaling properties make it much more attractive.

3.c. *Some MOSAIC Properties*

The Asynchronous Router in MOSAIC supports a variable length message format and implements a form of source routing called *cut–through*. The routing is two–dimensional, position–relative and

[4] For comparison, an interface to the 32–bit variant Futurebus+ backplane may require ten chips. Source: National Semiconductor, National Anthem vol. 23, March/April 1991, pp. 3.

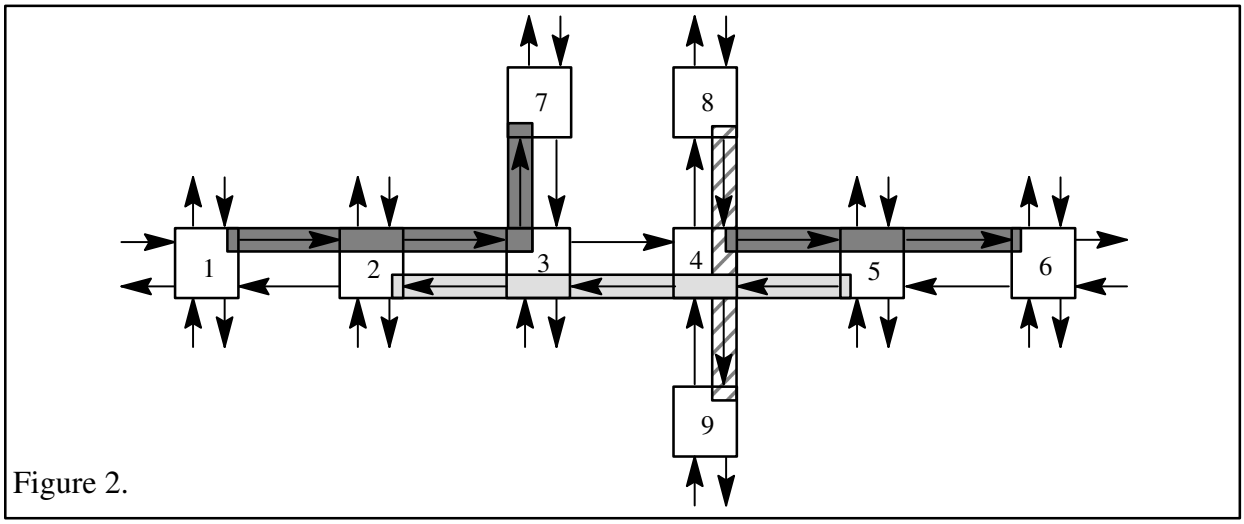

Figure 2.

deadlock–free [12][13][14]. The path a message takes is determined by a two–byte prefix, each specifying a hop count and a direction.

Hop counts are decremented as they pass through each router. A message may be forwarded +/–127 hops in the X direction followed by +/–127 hops in the Y direction. Messages are forwarded in the X direction completely before being forwarded in the Y direction. An interface receives a MOSAIC message when the Y hop–count has reached zero. By implication, MOSAIC messages traverse a restricted two–dimensional topology that allows a single X–to–Y transition.

A path is opened between source and destination routers by advancing a message header hop–by–hop along its path, allocating channels as it progresses. All the routers along that path ship the message toward its destination as a cooperative pipeline. At any node, the head of a message may be blocked because the outgoing channel needed is already used by another message. The arbiter in that node will allow the head of a blocked message to advance when the outgoing channel becomes free. Arbitration operates on a first–come first–served basis.

The Asynchronous Router uses the REQ and ACK signals to implement hop–by–hop, byte–by–byte flow control on each channel. If the head of a message is blocked due to congestion, data flow ceases. This freezes the pipeline until the packet head is again allowed to proceed. Once the head reaches the destination node, all necessary channel resources are allocated and data moves through the pipeline until the message has been received.

Byte propagation time from one MOSAIC router to another is typically 12.5ns, while a routing decision consumes approximately 25ns. A TAIL signal is used to indicate the end of a message. As the tail of a message passes through a node, the channels that were allocated to it are freed. Unused channels are easily terminated.

4. Example: A Multimedia Netstation

A block diagram for a multimedia workstation is shown in Figure 3. Arrows depict major logical data flows that may exist between subsystems. The shaded region depicts flows that could result from conferencing or other network–oriented applications. In traditional workstations those flows must pass through the network interface via the supervisory CPU, where protocol processing occurs. In an ideal architecture, they would pass into and out of the workstation without burdening the supervisory CPU or uninterested subsystems.

Figure 4 depicts a similarly configured Netstation. Each major subsystem uses one MOSAIC Interface chip to attach to the internal LAN. Each node can directly exchange messages with any other. The connection topology in this example is tree–like, but rich topologies are allowed.

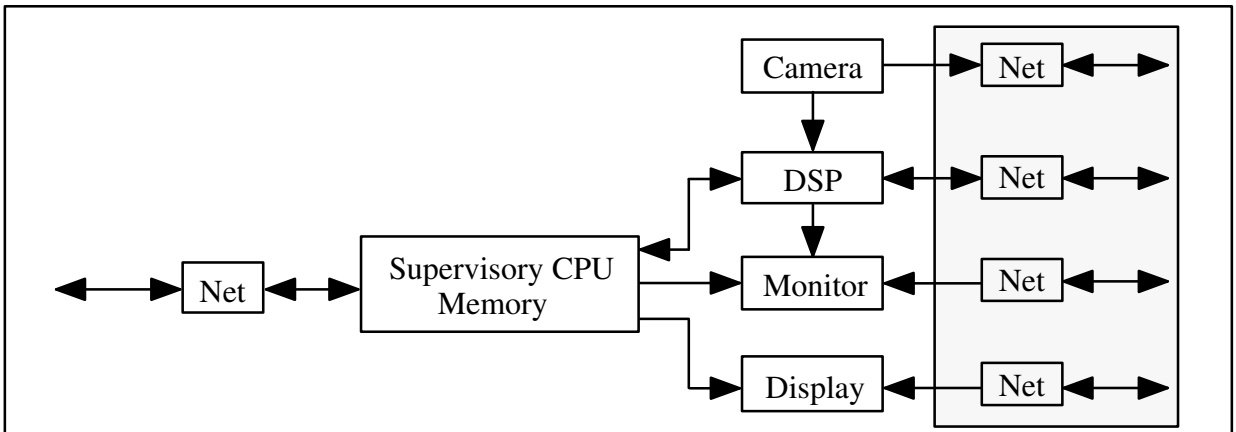

Figure 3. Multimedia Workstation Data Flows

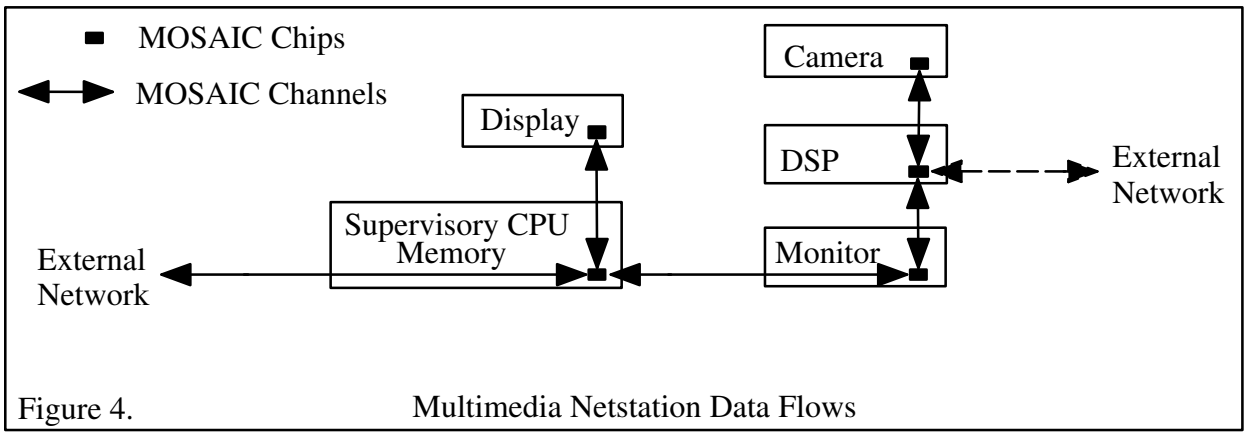

Figure 4. Multimedia Netstation Data Flows

Each node contains memory for data and program storage. The MOSAIC Interface chip accesses that and is dynamically loadable. Each node performs protocol processing tasks specific to the function of its attached device. Traffic between the camera, DSP/Graphics–Engine and monitor nodes interferes neither with the supervisory CPU accessing its memory nor with its traffic to the display node or the external network.

Internal and external LAN equivalence allows external traffic to utilize MOSAIC routing hardware to reach any internal node. Traffic may also leave internal nodes and directly reach the external network. This eliminates both single processor and single interface bottlenecks. No single processor is burdened with the routing, copying and protocol processing tasks of traffic for which it is logically neither source nor destination. That ability, coupled with the ability of nodes to perform protocol processing, closely approaches the ideal discussed above. Multiple external LAN connections are also possible.

As an example of how a display node might be efficiently implemented for a workstation that is capable of conferencing, the MOSAIC Interface shared memory may be block mapped into the frame buffer. Assume that typical traffic for this device specifies {window#, frame–line#} in its packet headers. The MOSAIC processor may interrupt on header arrival, momentarily block further reception, determine where a frame line is to be stored in the frame–buffer, provide that information to its Packet Interface logic DMA registers, and resume reception.

5. ATOMIC as an External LAN

ATOMIC is a prototype LAN that uses MOSAIC technology to provide multi–Gbit/sec service between workstations. ATOMIC routes are composed from one or more MOSAIC source routes. ATOMIC

provides the Y–to–X path transitions not allowed by MOSAIC and therefore allows a much freer topology [15].

5.a. *ATOMIC Network Layer*

The data portion of an ATOMIC packet may be prefixed with a number of source route (SR) directives. When a MOSAIC chip receives a packet, an SR directive tells it to retransmit the packet. In that case, the SR directive is stripped and the packet is retransmitted with the next MOSAIC source route. When the packet reaches its destination node, a leading CODE directive indicates that the packet data is to be sent to a higher layer protocol.

5.b. *Interconnect*

An ATOMIC LAN contains at least one cluster of MOSAIC–C processing elements functioning as a crossbar, as in Figure 5. A cluster is normally organized as a two–dimensional mesh. A special host, called an Address Consultant (AC), assigns paths between Netstations across the cluster. There are many possible paths between any two MOSAIC–C chips at the edge of a cluster. The path from one Netstation *a* to another Netstation *b* will not in general be along the same series of nodes as the path from *b* to *a*. Nor does the *a* to *b* path obtained from the AC at one time need be the same one obtained the next time. This flexibility allows load balancing across a cluster.

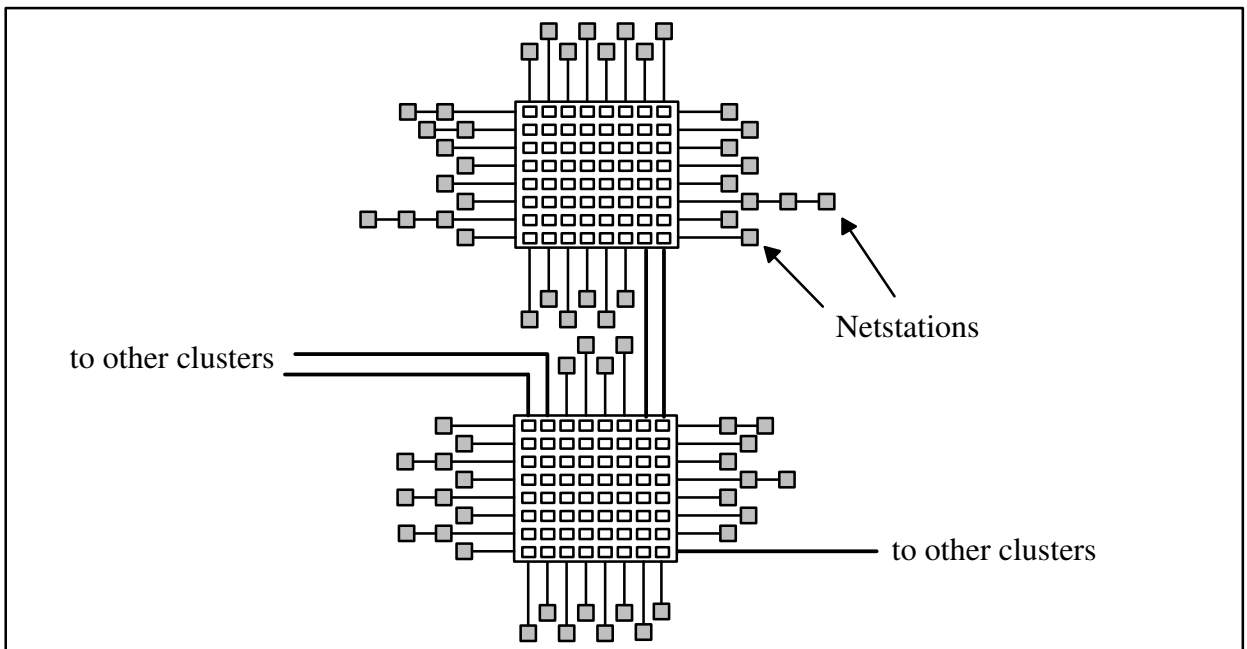

Figure 5. Netstation LAN Topology

Netstations attach to the edge of an ATOMIC cluster via specialized cables. There are 32 attachment ports on an 8x8 mesh board. Each port is the external termination of a separate full–duplex MOSAIC channel. A Netstation may be attached to more than one port on a cluster. Channels can also be used to communicate with other clusters, providing extensibility. As flow between clusters increases, the number of ports used for inter–cluster communication can rise.

Unless blocked by congestion, the maximum time required to set–up a MOSAIC route across an 8x8 cluster is 400 ns. Assuming random source–destination addresses, it has been shown that throughput across a mesh achieves a stable level at nearly 50% of its theoretical maximum capacity as determined by the mesh network bisection [9]. Furthermore, as congestion increases so does latency, but the congestion catastrophe associated with store–and–forward queue overflow is prevented by the flow control provided by the

Asynchronous Routers. An 8x8 ATOMIC cluster should achieve an interconnect bandwidth of up to 10 Gbits/sec under normal operating conditions.

6. Some Implications of Netstation Architecture

Each node of a Netstation is separately LAN addressable. In our examples, MOSAIC chips implement the physical medium dependent layer of the LAN and its link layer, but a node need not implement a complete protocol stack unless required by its function. A node that does not require reliable data delivery need not implement a connection layer. The fact that a node is potentially visible to the external network raises new networking issues, among them the issue of security.

6.a. *Node Visibility and Routing*

A Netstation has potentially many sub–addresses, one for each of its nodes. It is unclear whether at the network level a Netstation should have one address or several. This becomes even less clear when considering multicomputers that are dynamically shared piecemeal.

At the internetwork level, the details of intra–Netstation routing should be hidden. However, to gain the maximum performance advantage, it must be possible for streams to reach individual interior nodes of a Netstation directly via internal LAN routing. Similarly, it must also be possible for streams to leave interior nodes and directly reach gateways. This implies that intra–Netstation routing cannot be hidden at the network level. It also implies that the exterior and interior LANs be link–layer equivalent.

Assume that external LAN gateways are informed of the source route to each netstation's supervisory node at boot time. For connection–oriented traffic, a route to a specific node within a Netstation can be associated with the {host#, port#} pair. As a part of connection set–up, the supervisory node would inform the gateway of the {host#, port#}:{node source route} mapping. This becomes a logical extension to the gateway routing table and reflects the many–homed nature of a Netstation. The gateway would use that information when encapsulating incoming packets for that connection. The supervisory node must also as a part of set–up inform its node of the {host#, port#}:{gateway source route} mapping.

Once both the gateway and destination node have these mappings, connection data may flow without requiring further direct involvement of the supervisory node until the connection is closed. When a connection is closed a node would remove its mapping and inform its supervisory node. The supervisory node would then direct the gateway to remove the mapping for that connection.

For traffic exchanged between nodes within a network, no gateway involvement is necessary. As a part of connection set–up, the source and destination supervisory nodes would exchange {host#, port#}:{node source route} mappings. When a connection is closed each node would remove its mapping and inform its supervisory node.

A less efficient but simpler mechanism is to treat each Netstation as if it had only one address, that of its supervisory node, and route all traffic through it. The supervisory node would maintain the {host#, port#}:{node source route} mappings, functioning as a mini–gateway. This has the disadvantage of recentralizing the processing of external traffic. It creates a possible bottleneck and unnecessarily increases the supervisory node's overhead.

Within the network, access to a node requires only knowledge of the source route to it. This strongly suggests that a 'permission' be associated with packets that arrive at interior nodes. Packets without the correct permission would be discarded. The ability to issue these permissions should be inherent in each Netstation's supervisory node.

6.b. *Network Growth and Installation*

The architectural is very flexible. Adding a new node to a netstation requires only the connecting of that node's channel(s) to unused channel(s) on the internal LAN. Discovering the topology of the internal

LAN would occur at boot–time. While channel interfaces are functionally equivalent at the physical link layer, nodes themselves may of course be heterogeneous.

The crossbar that interconnects Netstations would normally be located in a wiring closet. Cables would branch from it to individual Netstations and other crossbars. One or more nodes of a Netstation could be connected directly to the cluster. Multiple Netstations could be chained together to share a single connection to the crossbar.

The flexibility of physical topology in combination with the favorable scaling property allows a Netstation to grow in a loosely bounded and dynamic manner. This is not a property associated with most architectures today. Some care must still be taken when adding nodes to a topology to avoid introducing bottlenecks.

6.c. *Multiprocessor Communication*

By making each processor of a multiprocessor computer be a node on the internal LAN, access to subsets of processors becomes a matter of publicizing routes to the subset. This eases some problems associated with partitioning multiprocessors. Remote users would be able to directly access those processing nodes in their partition and those nodes would have direct access to the external network.

7. Conclusions

Several potential communications bottlenecks in current workstation architecture are removed. This is accomplished by using a gigabit point–to–point internal LAN to distribute communications resources throughout the workstation architecture. Major data producing or consuming devices within the workstation are attached as nodes on this LAN. By analogy they become network–mapped devices. This organization is called a Netstation Architecture.

Netstation Architecture speeds protocol processing through parallelism, by using link–layer routing hardware to partially subsume dispatching at higher protocol layers, and by executing protocol software within each network interface. The processing functions within each node's protocol processor can be dynamically tailored to match its requirements. As the display node example presented earlier suggests, adapting the protocol processor for a specific application can provide great enhancement in performance.

Netstation Architecture blurs the distinction between addressable subsystems that are internal and those that are external. When the internal and external LANs use equivalent link–layer technology, internal devices are network addressable and directly accessible externally as well as internally. The boundary of a Netstation can be indistinct and can change dynamically. This architecture is especially well suited to communications–intensive applications.

Special thanks to J. Postel, D. Cohen, D. Hollingworth, K. Korner and others who discussed this paper with me. Thanks also to the editor for pointing out portions which were not clear.

8. References

[1] Partridge, C.
How Slow is One Gigabit Per Second
Computer Communication Review, Vol. 20, No. 1, January 1990.

[2] Jain, N., Schwartz, M., Bashkow, T. R.
Transport Protocol Processing at GBPS Rates
Proceedings of Sigcomm–90, pp. 188–199.

[3] Kanakia, H., Cheriton, D.R.
The VMP Network Adapter Board (NAB):
High–Performance Network Communication for Multiprocessors
Proceedings of Sigcomm–88, pp. 175–187.

[4] Arnould, E., Bitz, F., Cooper, E., Kung, H. T., Sansom, R., Steenkiste, P.
The Design of Nectar:
A Network Backplane for Heterogeneous Multicomputers
Proceedings of the Third International Conference on Architectural Support for Programming Languages and Operating Systems, ACM 1989, pp. 205–216.

[5] Clark, D. D., Tennenhouse, D. L.
Architectural Considerations for a New Generation of Protocols
Proceedings of Sigcomm–90, pp. 200–208.

[6] Schroeder, M. D., Birrel, A. D., et al.
Autonet: a High–speed, Self–configuring Local Area Network Using Point–to–point Links
Digital Equipment Corporation, SRC Research Report 59, April 30 1990.

[7] Athas, W. C., Seitz, C. L.
Multicomputers: Message–Passing Concurrent Computers
IEEE Computer, pp. 9–24, August 1988.

[8] SCI – Scalable Coherent Interface
Draft Report P1596: Section 1 / D0.85. IEEE.

[9] Seitz, C. L.
Concurrent Architectures
Chapter 1 in *VLSI and Parallel Computation*, Ed. Suaya, R., Birtwistle, G.
Morgan and Kaufmann, 1990.

[10] Cheriton, D. R.
Sirpent$_{TM}$: A High–Performance Internetworking Approach
Proceedings of Sigcomm–89, pp. 158–169.

[11] *Submicron Systems Architecture Project Semiannual Technical Report*
California Institute of Technology, Computer Science Caltech–CS–TR–90–05.

[12] Dally, W. J., Seitz, C. L.
Deadlock–Free Message Routing in Multiprocessor Interconnection Networks
IEEE Transactions on Computers, Vol. C–36, No. 5, May 1987.

[13] Kermani, P., Kleinrock, L.
Virtual Cut–Through: A New Computer Communication Switching Technique
Computer Networks, 3(4), pp. 267–286, September 1979.

[14] Flaig, C. M.
VLSI Mesh Routing Systems
California Institute of Technology, Computer Science 5241:TR:87, May 1987.

[15] Cohen, D.
Personal Communication, May 1991.