October 1987

Report No. STAN-CS-87-1185
Also Numbered CSL-TR-87-346

Blazenet: A Phototonic Implementable Wide-Area Network

by

Zygmunt Haas and David R. Cheriton

Department of Computer Science

Stanford University
Stanford, CA 94305

Blazenet: A Photonic Implementable Wide-Area Network

Zygmunt Haas
Electrical **Engineering** *Department*
Stanford University, CA 94305

David R. Cheriton
Computer Science Department
Stanford University, CA 94305

October 5, 1987

Abstract

High-performance wide-area networks are required to interconnect clusters of computers connected by local area and metropolitan area networks. Optical fiber technology provides long distance channels in the multi-gigabit per second range. The challenge is to provide switching nodes that handle these data rates with minimum delay, and at a reasonable cost.

In this paper, we describe a packet switching network, christened *Blazenet*[1], that pro vides low delay and has minimal memory requirements. It can be extended to support multicast and priority delivery. Such a network can revolutionize the opportunities for distributed command and control, information and resources sharing, real-time conferencing, and wide-area parallel computation, to mention but a few applications.

This work was sponsored in part by the Defense Advanced Research Projects Agency under contract N00039-86-K-0431, by the Digital Equipment Corporation, by ATT Information Systems and by Bell Northern Research.

[1] The name *Blazenet* refers to the use of lasers with fiber optics as well as the boomerang aspect of the returning packets that cannot proceed onwards, i.e. Boomerang Laser network. It also refers to the notion of a packet "blazing" a route through the network, and the speed at which it does so.

1 Introduction and motivation

The potential of computer communication is, at present, severely handicapped by the poor performance of wide-area networks. The geographically dispersed clusters of machines operated by military, commercial, government, and research organizations are information and resource "islands" that limit the efficiency, capability, and responsiveness of these organizations. Distributed environments and more performance-demanding applications will characterize future wide-area communication, requiring wide-area networks that are matched in delay and bandwidth to the performance and requirements of local-area and metropolitan-area networks.

Optical fiber provides a long distance channel technology that makes this goal feasible. Transmission rates of gigabits per second with bit error rate on the order of 10^{-9} over tens of kilometers are already achieved today [6,7,9]. Fibers are being installed extensively [4], replacing twisted pairs and coaxial cables, and bringing with them the benefit of very high bandwidth, two or three orders of magnitude higher than that of the existing networks. The challenge is to provide switching nodes that handle these high data rates with minimal delay and at a reasonable cost.

Optical switching and processing of the optical transmission opens new dimensions in future networking. Photonic implementation, as opposed to a conventional electronic implementation, offers increased switching speeds [15,9]. In addition a network built totally out of optical components is more

immune to electromagnetic interference and electromagnetic pulse and provides more secure transmission. Since the technology of optical devices is still in its infancy, a network design based on simple node design is highly desirable.

In this paper, we describe ***Blazenet,*** **a** packet-switched network based on optical fiber and high-performance switching nodes. Three key ideas behind ***Blazenet*** design are: source routing, packet **loopback** on blockage, and photonic implementation made possible by the first two. Fiber loops that constitute ***Blazenet*** links provide the temporary storage for blocked packets in transit, thus using the storage inherently present in the links. ***Blaxenet*** is presented here as a wide-area network. We see it as a backbone network, whose nodes are gateways to other networks. Nevertheless, the concept of ***Blazenet*** is easily applied to smaller networks and, with some constraints on transmission rate and on maximum packet size, even to local area networks.

Section 2 describes ***Blazenet*** design, addressing the issues of packet-switching and traffic congestion. Section 3 presents **a** detailed switching node design. Section 4 shows the expected ***Blazenet's*** performance results determined by simulation. Section 5 discusses some extended features that can be incorporated into ***Blazenet's*** design, such as priority, multicast, and the loop reservation scheme. Section 6 presents some higher layers issues that have direct implication on ***Blazenet's op*** eration. The final Section 7 summarizes our conclusions about the design and the **implications.**

2 *Blazenet* design

A ***Blazenet*** is composed of **a** set of switching nodes interconnected by **point-to-point** logical links formed by the fiber loops. The hosts and gateways on the periphery of the network act as sources and sinks for the network traffic. Packets generated by hosts are passed to the switching nodes to which they are connected. The packets are then forwarded from node to node until they arrive at the switching node connected to the destination host, removed by the switching node and passed to the destination **host.** An example of a four node ***Blazenet*** is shown in Figure 1.

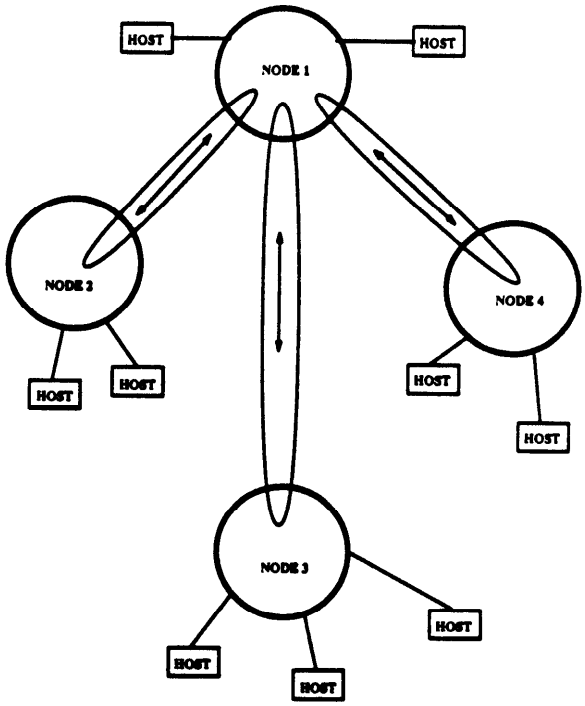

Figure 1: A four node ***Blazenet*** configuration

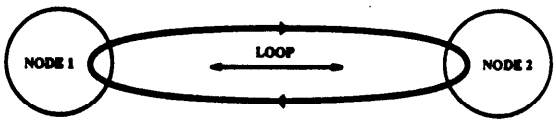

Figure ***2:*** A ***Blazenet*** loop

A loop, shown in Figure 2, is built of two **point-to-point** logical links. In such a configuration each bidirectional link connecting two adjacent nodes is replaced by **a** single-loop. A number of loops can be multiplexed on a single fiber (if the fiber provides enough capacity) by ***Wavelength Division Multiplexing*** technique, for example.

Figure ***3:*** The ***Blazenet*** packet format

The *Blazenet* packet format, shown in Figure 3, is composed of two delimiting synchronization fields (*syncs*), a header, and a data portion. Within the header, the *token* identifies whether the packet is a blocked packet (set *token)* or not (reset *token),* the *loopcount* limits the packet lifetime within the network (as described later), and the *hop-selects* dictate the hop-by-hop route for the packet to reach its destination. The data portion contains higher level protocol data and can, optionally, be protected by a checksum.

2.1 Source route packet switching

Blazenet uses source routing. Each packet contains a sequence of *hop-selects,* specified by the source host. The *hop-selects* represent the switching operations to be taken in the sequence of nodes along the packet path through the network from its source to its destination. Each *hop-select* field indicates the output link on which the packet is to be forwarded for that hop. When a packet arrives at a switching node with its *token* reset, the first *hop-select* field in the packet is examined to determine the next output link for the packet. If that output link is available for transmission of a new packet, the first *hop-select* field is zeroed and the packet is immediately routed to the available output link. The zeroing of the first *hop-select* field during the forwarding process means that the first non-zero *hop-select* field in the packet always represents the current hop selection. A packet with a set *token* arriving at a switching node is simply left on the loop to be returned to the blocking node, after the *token* field is reset.

This design has several advantages. First, because of the simple logic required to make the hop selection, it is feasible to perform the switching function at gigabit per second data rates. In particular, no table lookup is required for the switching decision. Second, the delay for switching in a node is limited to the time required to interpret the packet header, check the availability of the output link, and perform the actual switching operation (if the output link is available). Since the extra delay introduced by a switching node is estimated to be a few tens of bits, this delay is only a small fraction of the propagation delay of a link in a wide-area network. Finally, the simplicity of the node logic suggests that a photonics implementation is feasible.

2.2 Handling packet blockage

A packet is called blocked if it arrives at a switching node when the next output link is unavailable. A blocked packet is routed back to the previous switching node on the reverse link of the loop that the packet arrived on. Upon its arrival at the previous switching node, the returned packet is sent again to arrive at the blocking switching node one round-trip time after its first arrival at this node. Thus, the loop effectively provides short-term storage of the packet, causing the packet to reappear at the blocking switching node a short time later. The *loopcount* field is decremented and examined each time a packet is returned. When *loopcount* reaches zero, the packet is removed from the network, preventing a packet from indefinitely looping within the network under failure or very heavy load conditions.

This approach to handling blockage has several advantages. First, *Blazenet* dramatically reduces the average packet delay through a loaded network and increases the network capacity, compared with a design in which the packet is simply dropped when blocked at the outgoing link, referred to here as a *Lossy* network. When a packet is dropped in a *Lossy* network, it has to be retransmitted by the source after some timeout, at least one round-trip time long. Since the probability of a packet being blocked increases with path length, as does the network investment in the blocked packet, dropping the packet seriously degrades the network performance under load for wide-area networks with realistic diameter.

Second, the design does not require memory in the switching node of the size and speed required to store all blocked packets, such as would be needed for a conventional *Store-and-Forward* design. Several megabytes of memory operating at 1 Gbps would increase the

cost of the switching nodes and make their realization in optics less attractive (at least in the near future), The combination of the high data rates, the wide-area span of the links, and the low-cost of the fiber makes this form of storage attractive. For example, a 100 km link (= 200 km loop) operating at 1 Gbps can store nearly 1 Mbit or 125 packets of 1 kbyte each.

Finally, the **loopback** technique exerts back pressure on the link over which the packet was received, because the loop is then less available for new packets to be forwarded on it. In the extreme, this back pressure extends back from the point of contention to one or more packet sources. Besides alerting the packet source of congestion, the back pressure provides fast feedback to the source routing mechanism, allowing it to react quickly to network load and topological changes.

A potential disadvantage arises when the link between switching nodes is very long, since the round-trip delay on the loop may be excessive. We **avoid** this problem by including **loopback** support in the optical repeaters that are required every few tens of kilometers on **a** fiber optic link. Thus, **a** packet that is blocked at **a** switching node is looped back either to the previous switching node or to the previous **repeater**, whichever is closer. If, for example, the distance between **adjacent** switching nodes is 100 km, the round-trip delay is approximately 1 **msec.** Because **a** ***Blazenet*** switching node includes the regeneration function between the input **and** output ports, it can be used as a **repeater**, thereby **automatically** supporting the **loopback** function. (In this use, only two input and two output loops are used.) The network is then built from one type of interconnection component, rather than two. By using such a design, packets can loop at intermediate loops on a long link, reducing its delay through the network (assuming the congestion clears before the packet is dropped). Consequently, the packet is delayed in time units corresponding to the round trip time on the intermediate loop rather than that for the entire link. Another improvement consists of designing the last loop shorter than other loops on the link. Consequently, blockage at low-load operation has smaller effect on the packet delay.

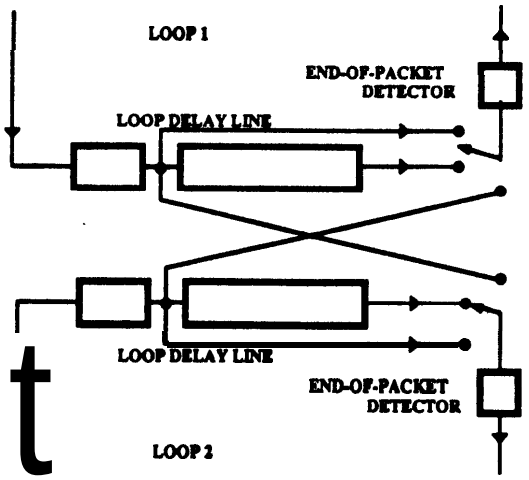

Figure 4: The switching node design

3 Switching node design

A ***Blazenet*** switching node can be implemented as simple interconnection of a number of photonic components. We assume the availability of fast switching devices capable of switching within **a** small fraction of duration of a header bit, once the switching command has been initiated. Such devices exist for at least as fast as 3 **GHz** operation speed. Slower devices can be employed for lower cost by maintaining an adequate inter-packet gap.

Figure 4 shows the block design of a ***Blazenet*** switching node. Each ***Loop*** has a Delay Line, build out of a piece of fiber. The ***Delay Line*** is long enough to contain a packet header, a maximum sized packet, and the number of bits corresponding to the time the control logic requires to do the actual switching. Figure 5 shows the signals extracted from the transmission entering the ***Delay Line*** and the corresponding timing. The signal extraction is initiated by the ***new-packet*** signal generated by **a** pattern detection circuit, which search for the ***sync*** pattern. Upon ***sync*** detection, the circuit raises the ***new-packet*** line, indicating to the Control a new packet arrival. At this time the Control looks for the values of the ***token,*** the ***loop-count,*** and the ***hop-***

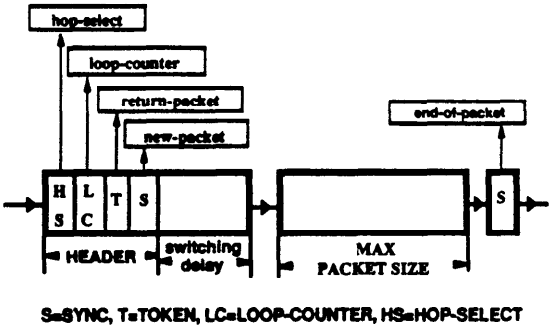

Figure 5: Timing of ***Delay Line*** signals

select signals. The indication that a packet leaves the ***Delay Line*** is provided to the Control by the ***end-of-packet*** line of the input or the output loop, depending if the packet is forwarded or blocked. The switching decision is performed during the period of time named "switching delay," at the end of which a switching command is issued.

The ***Delay Line*** is considered to be free if it does not contain a packet or any part of a packet. By using the two signals ***new-packet*** and ***end-of-packet*** the Control can uniquely determine the status of the ***Delay Line.***

When a packet is to be forwarded to a loop the availability of this loop (i.e, if no connection of any other loop to this loop already exists) and the availability of its Delay ***Line*** are checked. If both are available, the packet from the input loop is clocked onto the output loop. If, on the other hand, the output loop and/or its ***Delay Line*** are busy, the packet is blocked and returned by being clocked out on the loop it came on, after its ***token is*** set. In case more than one packet tries to enter a specific loop, only one packet wins (the one with the higher priority, or one chosen randomly in case of equal priorities), and the other(s) are clocked out on their loops. Upon its arrival to the other end of the loop, the blocked packet is clocked into the corresponding ***Delay Line,*** blocking access to this loop for any new arrival. When the packet reaches the end of the ***Delay Line*** it is clocked out onto the loop it came on, after the ***token*** is reset. (Another possible implementation is to check the loop availability upon reception of a returned packet. Only in

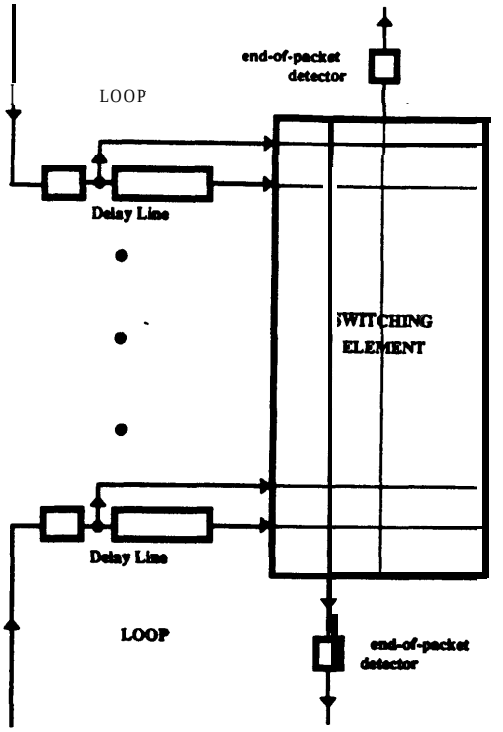

Figure 6: The switching node design using a switching matrix

the case the loop is busy (forwarding another packet), the returned packet is entered into the ***Delay Line.*** A packet is, however, directly clocked out if the loop is found free upon the returned packet arrival. This improvement has the advantage of including an additional ***Delay Line*** delay only in the case the loop is busy. On the other hand, this implementation has a disadvantage of complicating the switching process and therefore, the Control itself.

The Switching Element is capable of connecting each one of its inputs to any one of its outputs. The Switching Element can be designed in several ways. One such a possibility is to use a switching matrix, as shown on Figure 6. In this case maximal connectivity can be achieved.

The Control performs the actual routing decisions based on the signals that indicate the status of the loops. The signals entering the Control are shown in Figure 7. The routing algorithm takes into account the following parameters:

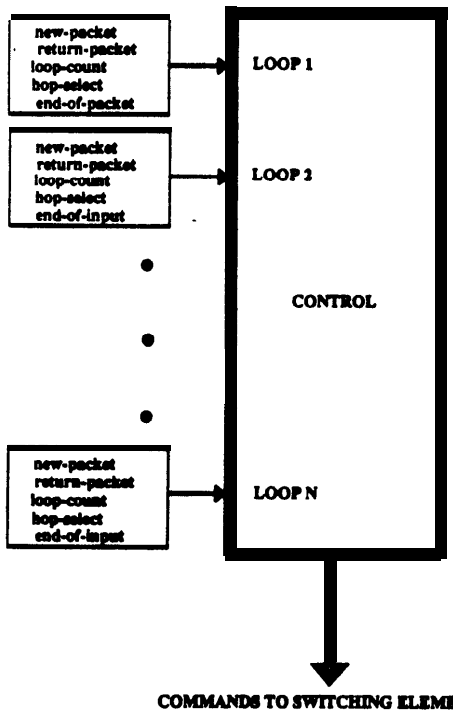

Figure 7: The switching node control signals

1. input packet destination,
2. availability of the output loop and its ***Delay line***
3. priority of the packet (as explained in Section 5.1).

In general, a packet is either completely forwarded or returned. However, in some cases where extraordinary priority is needed, it may be necessary to abort transmission of a packet currently being forwarded. A simple mechanism can be incorporated into the design such that upon reception of a high priority packet, the packet is immediately forwarded on the appropriate loop.

The input traffic from a host connected to the switching node is switched in a similar way the traffic from any loop is. The main difference is in the indication of an available packet. An indication line, from a host to the Control, continues to show the presence of a packet until it is forwarded. No returning of an incoming packet is ever performed.

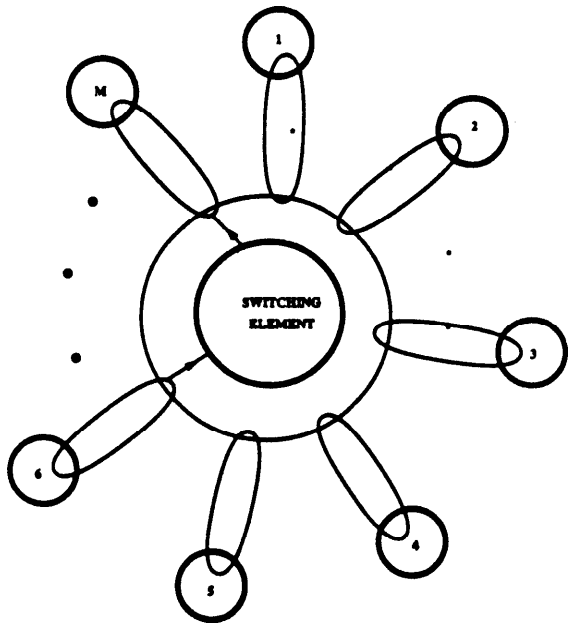

Figure 8: ***Star*** topology

The output traffic destined to a host connected to the switching node is received on one of the outputs of the Switching Element and passed to the appropriate host. A host is assumed to be always ready to accept its traffic. If the host is unavailable, the packets are simply discarded. Therefore, the major difference between the through traffic and the exiting traffic is that the latter is never returned. The reason for not returning exiting blocked traffic is to avoid situations in which the network can possibly be blocked because of a host malfunction.

If the speed of the lines increases to the value where bit recognition of the header becomes a problem, representing a header bit by several actual bits will allow more time for decoding. Thus, by keeping the header bit "duration" constant, the problem of header bits recognition is essentially independent of the actual line speed.

4 *Blazenet* performance

Average packet delay through the network as the function of the network throughput is the main performance criterion we chose to con-

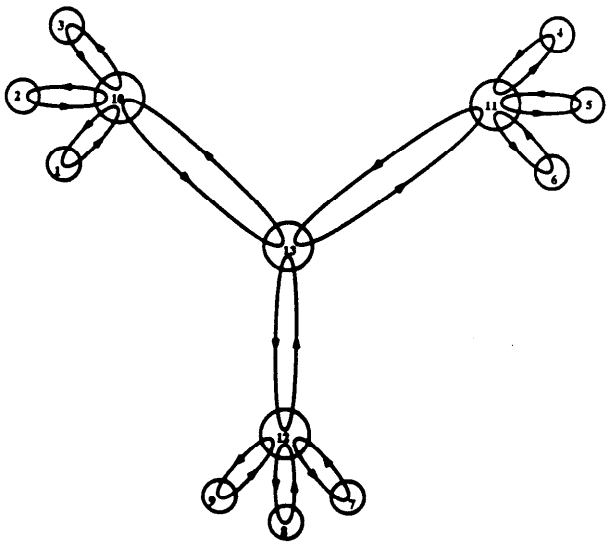

Figure *9: **Star-of-Stars*** topology

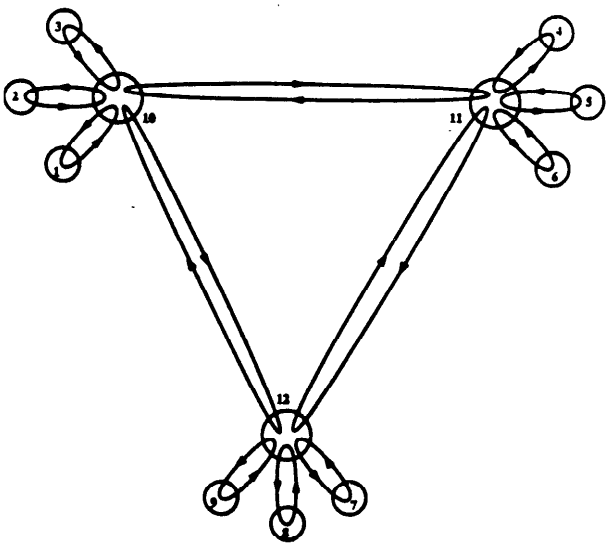

Figure 10: ***Triangle-of-Star*** topology

sider in this paper. Average packet delay is the period of time from when the packet is passed to the network until it is delivered to its destination averaged over all packets entering the network, and includes the queuing time at the network entrances.

To evaluate ***Blazenet*** performance, a general event-driven simulation program was developed. The simulation enabled us to evaluate ***Blazenet*** performance, as well as to compare ***Blazenet*** performance with the idealistic case of nonblocking network, that is with the propagation delay only. We also compared ***Blazenet*** performance with the case of the ***Lossy*** approach. The following graphs show ***Blazenet*** performance for several different network architectures and different packet sizes. In all of the examples we assume that the **traffic** matrix is symmetric, the link **capacity** is 1 Gbps, and the links are all equal and approximately 100 km long. We also assume infinite ***loop-counter*** value, and equal priority of all packets.

Blazenet performance was evaluated for packet sizes of 5 kbit and 10 Kbit. These values represent reasonable trade-off between the long ***Dela*** y ***Line*** for large packet size and excessive header (Blazenet and high-level pro **tocol)** overhead of small packets. For example, the combination of a Blazenet, VMTP and IP headers could total 100 bytes, requiring a 10 kbit packet to put the overhead under 10%. On the other hand, 5 kbit and 10 kbit correspond to 1 km and ***2*** km ***Delay Line*** on 1 Gbps link (or transimssion times of 5μsec and 10μsec) respectively, thus representing a feasible design.

In the Section 5.5 the double-loop ***Blazenet*** is introduced. In the following graphs we include the performance of the double-loop approach for comparison and later reference.

The first example is of the ***Star*** topology with 5 inputs. General ***Star*** topology with ***M*** inputs is shown in Figure 8. The delays as a function of network throughput, evaluated for single- and double-loop configurations, are presented in Figure 11. The propagation delay through the network is also shown for comparison.

The second example is the ***Star-of-Stars*** topology, shown in Figure 9. The simulation results are presented in Figure 12.

The final case ***is*** of the ***Triangle-of-Stars*** topology shown in Figure 10, with corresponding results in Figure 13.

The comparison of ***Blazenet*** performance with the ***Lossy*** for ***Star-of-Stars*** network is shown in Figure 14. In the ***Lossy*** network case it is assumed that a blocked packet is retransmitted immediately after a single round trip

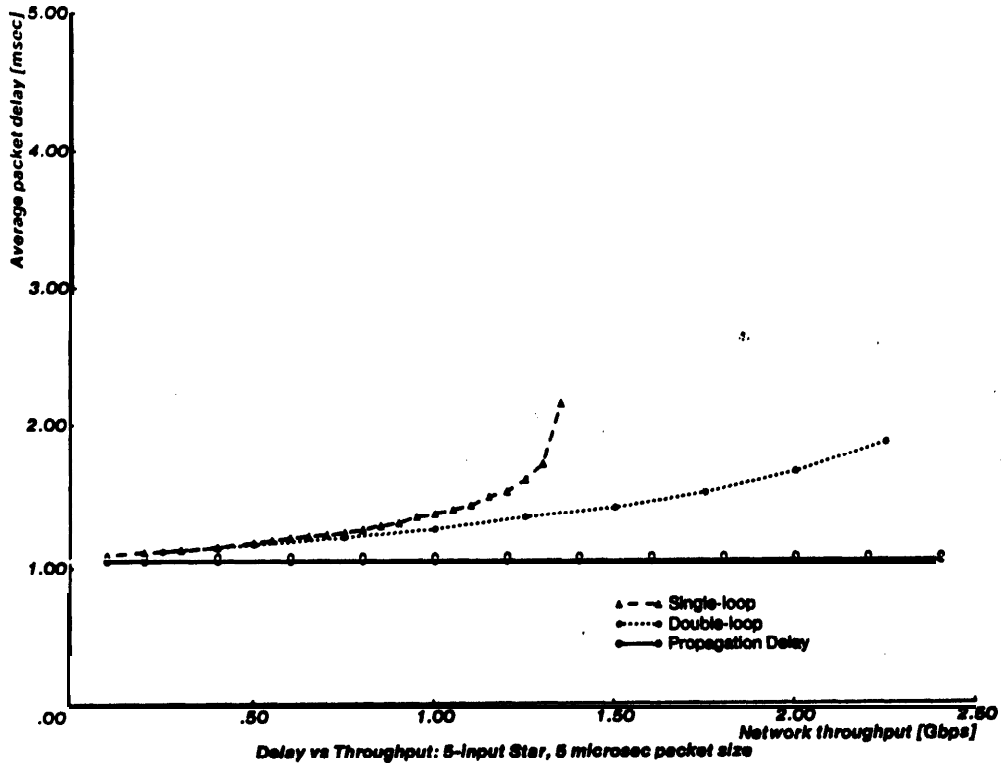

Figure 11: Delay of ***Star Blazenet***

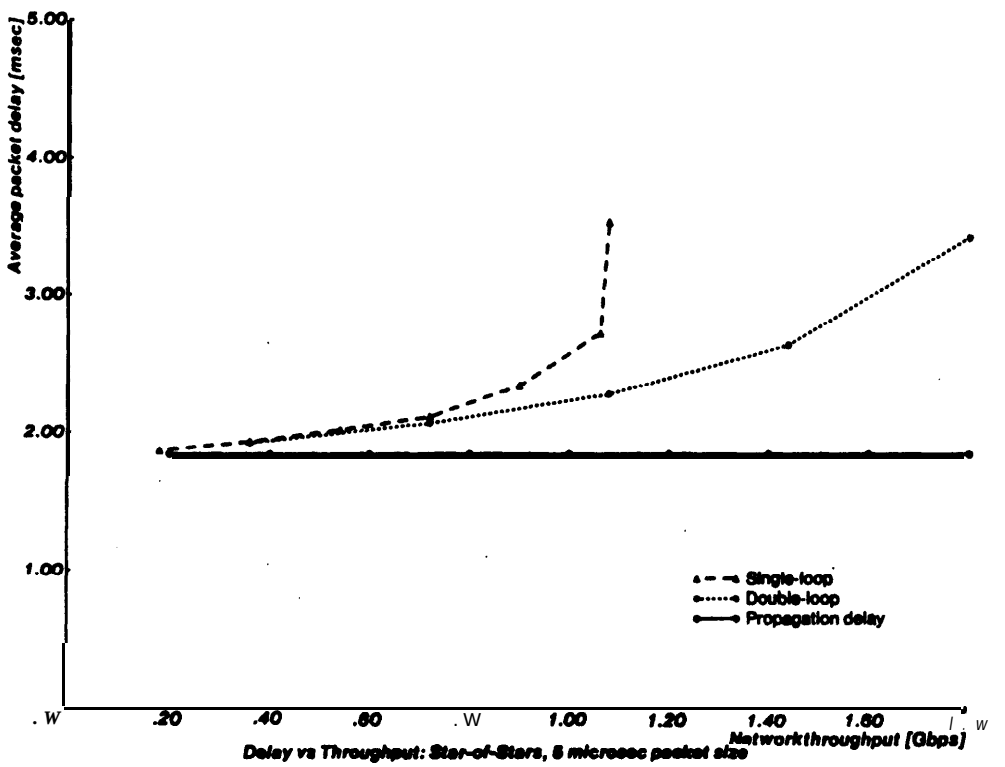

Figure 12: Delay of ***Star-of-Stars Blazenet***

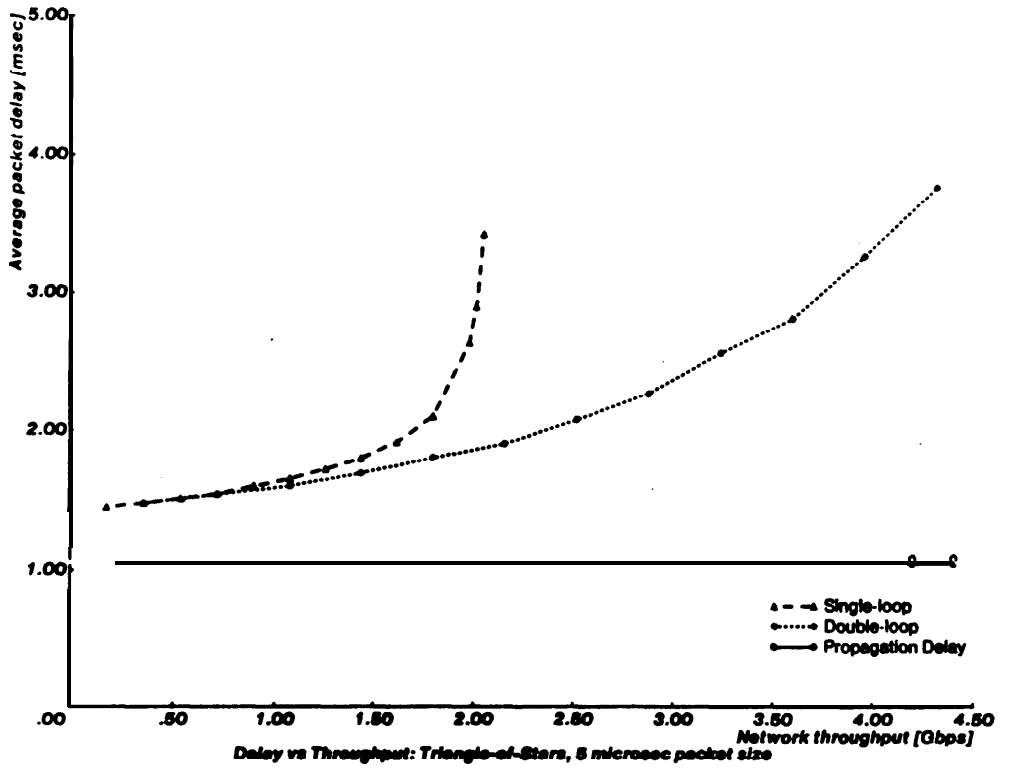

Figure 13: Delay *of Triangle-of-Star Blazenet*

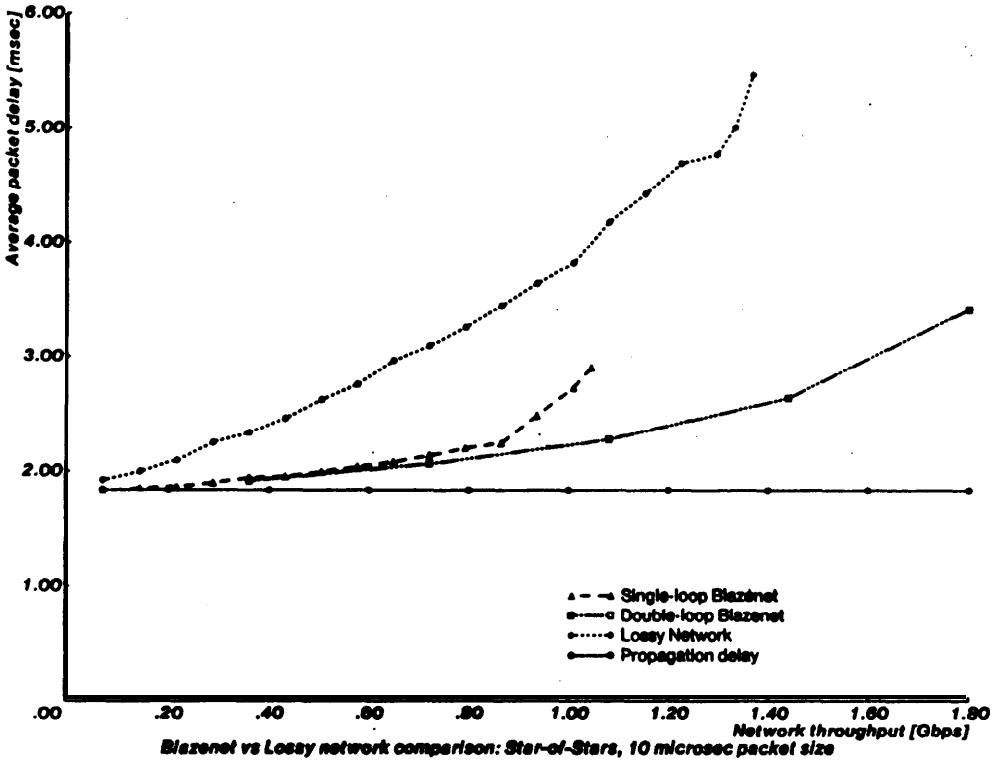

Figure 14: Performance of ***Lossy*** network and ***Blazenet***

delay between the source and the destination without any processing overhead, thus favoring the ***Lossy case.*** Also, the small network span somewhat favor5 the ***Lossy*** approach in this **comparison,** since the ***Blazenet*** advantages are emphasized in network5 with large average path length.

From these and other simulation results we conclude that the penalty in delay paid by ***Blazenet*** as **oppposed** to the ideal case of **non-**blocking network **is** in the range of **a** few tens of percents at lower load, for which the network **is** assumed to be designed. In addition ***Blazenet*** experience5 considerable shorter delay than the ***Lossy*** approach. Double-loop ***Blazenet*** doe5 not have excessive delay and decrease in the network throughput for heavy-load operation ("Aloha-like" behavior), behavior that single-loop configuration experience. (We believe that the **future** networks will offer very high bandwidth, leaving the network to op erate in the low-load. For example, consider **a** ***Blazenet*** connecting **a** collection of 10 **Mbps** Ethernets operating at 10% utilization. Assume further that 25% of an Ethernet traffic is to be transferred on the backbone ***Blazenet,*** built of links of ten 1 Gbps fiber5 each link. Thus, as many as 4000 Etherneta can coexist on ***Blazenet*** utilizing the network only in **10%,** utilization that represents low-load condition.) Consequently, we consider the **single-** and double-loop ***Blazenet*** as an attractive future high-performance network design.

5 Extended features

The Section 3 presented the basic ***Blazenet*** and its node design. This Section expands on the basic node design to include some of the more sophisticated features: priority traffic, limiting of the life-time of a packet, broadcast and **mul-**ticast, and network monitoring. Besides providing very important services to the network users, these features increase the strength of the network to cope with abnormal situations, increasing, therefore, the network reliability.

Figure 15: Modified packet format

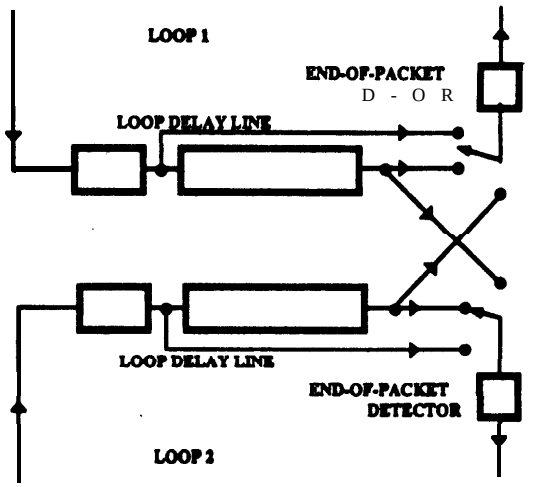

Figure 16: Modified node design

5.1 Priority traffic

The implementation of priority **traffic** in ***Blazenet*** can be accomplished in two **ways:** by including a priority field in the packet format, or by giving preference to some traffic during the forwarding process. The former approach is considered first. The packet format with the ***priority*** field is presented in Figure 15.

The priority traffic implementation **is** achieved by delaying the forwarding of a packet by **a** period equal to the transmission time of **a** maximum packet length. At the end of this period, the packet with the highest priority is forwarded, while other packets (if any) are looped back. The hardware of the basic ***Blazenet*** node design has to be modified in order to accommodate **this** additional functionality. The main adjustment is to include a packet detector circuit within the ***Delay Line*** that initiate the ***packet-ready*** signal. The modified node design is shown in Figure 16 and the modified ***Delay Line*** in Figure 17.

A packet that is clocked into a ***Delay Line*** and has not reached the ***packet-ready*** point is

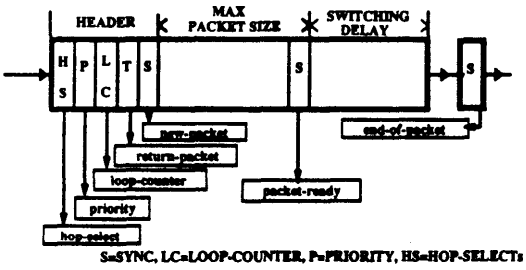

Figure 17: Modified ***Delay Line*** structure

called an active packet. The set of active packets at any point in time is the set of packets competing on the loops.

The new packet arriving on a ***Loop*** is clocked into the ***Delay*** **Line.** After its ***main-header*** (composed of the fields: ***sync, token, loop-counter, priority,*** and the ***hop-selects)*** are received, the Control is notified of the packet arrival and the packet ***main-header*** information is passed to the Control. The Control gathers all such information from all the ***Delay*** **Lines.** When a packet is shifted to the ***packet-ready*** point in the ***Delay Line,*** the decision is taken whether the **packet** will be forwarded or looped back. The decision is made according to the following algorithm:

IF (***(priority*** $\geq$ ***priority*** of all active packets with the same ***hop-select)***
AND (no transmission in progress)
AND (destination ***Delay Line is*** free))
THEN **forward** the **packet**
ELSE **loop** the **packet back;**

The forwarding or blocking (namely the switching) operations are, otherwise, done as before.

The essence of the above procedure is that, by delaying all packets by one packet length (i.e, one packet look-ahead), the priorities of all the relevant packets can be gathered and the correct decision about which packet to forward can be made. Therefore, the only difference in this modified version of ***Blazenet is*** the instant in time when the Control decision is made.

Using the above scheme, the delay of a forwarded packet is increased by the transmission time of a packet of the maximum size. However, this time is negligible compared with the propagation delay encountered by a packet on a link in wide-area network. (For example, for 10 kbit packets on 1 Gbps ***Blazenet*** the additional delay is only 10 μ**sec,** delay that is small compared to 500 μ**sec** that is the propagation time of a 100 km link.) The total delay is, therefore, essentially unaffected by this hardware modification.

Another way to implement priority **traffic** in ***Blazenet*** is to give preferences to some traffic during the forwarding process. One such possibility is to prefer always the traffic coming from the hosts connected to the node over all the other traffic. Such a mechanism is useful for coping with temporary **traffic** surges from the node's hosts. However, although this ap **proach** lowers the delay of the preferred traffic, the mean packet delay in the whole network is increased. Therefore, in order to insure fairness, usage of such a mechanism should be restricted.

The preferences given to some **traffic** can be based on other criteria. Traffic arriving on some loops (for example, traffic coming from congested areas) may be given higher priority in the forwarding process. The preferences criteria can be based on various network parameters and can be adjustable in time, as the network load and topology change.

5.2 Limiting packet life-time

The network needs to limit packet life-time because of three reasons: to eliminate erroneous traffic to exist in the network and interfere with valid traffic, to discard real-time **traffic** that could not be delivered on time and became obsolete, and to avoid wraparound of packet sequence numbers in high-level protocols.

In ***Blarenet,*** the ***loop-counter*** provides the mechanism for limiting the life-time of packets within the network. The ***loop-counter is*** decreased each time a packet is blocked and returned. When the ***loop-counter*** reaches zero, the packet is discarded. The ***loop-counter*** represents, therefore, the maximum number of

times a packet can loopback. The value of the **loop-counter is** set by the source host, **according** to packet type and time limitations for the packet delivery.

Unless the loops are of equal length, the ***loop-counter*** mechanism does not provide an accurate mean for limiting **a** packet life-time within the network. In the case the loops are of unequal length, using the minimum loop length of the packet path for the calculation of the ***loop-counter*** can be an **adequate approach**. Assign n to represent the refractive index of the fiber, l_{min} the minimum loop length of the packet **path (=** twice the distance between the **adjacent** switching nodes), l_i the length of the i^{th} loop, h number of hops on the packet **path (=** number of switching nodes on the **path - 1),** t_{min} minimum life-time of **a** packet in the network and c the speed of light in vacuum. Therefore, the value of the ***loop-counter*** can be calculated using:

$$loopcounter \geq \frac{c \cdot t_{min}}{n \cdot l_{min}} - \frac{1}{2 \cdot l_{min}} \cdot \sum_{i=1}^{h} l_i$$

Another approach would be to use some weighted **average** of the loops lengths of the packet path, l_{avg}. In this **case** the ***loop-counter*** is calculated by the same formula **as** above, substituting l_{avg} for l_{min}.

If the general repeater/switching node design is used, **all** network loops are of equal length, l (possibly with the exception of the last loop hitting the node), and the calculation of the required value of the ***loop-counter*** becomes:

$$loop\text{-}counter \geq \frac{c \cdot t_{min}}{n \cdot l} - \frac{h}{2}$$

By imposing some minimum value on the packet life-time, t_{min}, the packet will not be discarded **because** of its lifetime expiration for at least this period of time. This is advantageous in situations where we are more concerned with the possibility of discarding still valid packet, than with the possibility of an obsolete packet living in the network or even being passed to the destination. When we are in the opposite situation, namely when we are more concerned with the access load created by an obsolete traffic than of the possibility of discarding valid traffic, we should use some maximum permissible value for the packet life-time, t_{max}, **instead.** (The above formulas continue to be valid in this case, with the substitution of maximum loop length l_{max} for the l_{min} and reversing the unequality sign.) Note, that by manipulating the current value of the packet life-time, the network can regulate its load. Of course, such a manipulation is justified in some special circumstances and for traffic that does not require reliable transport through the network.

The implementation of the ***loop-counter***

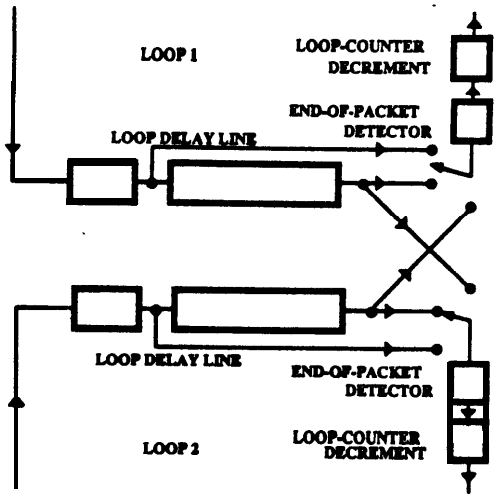

Figure 18: Node design with ***loop-counter*** implementation

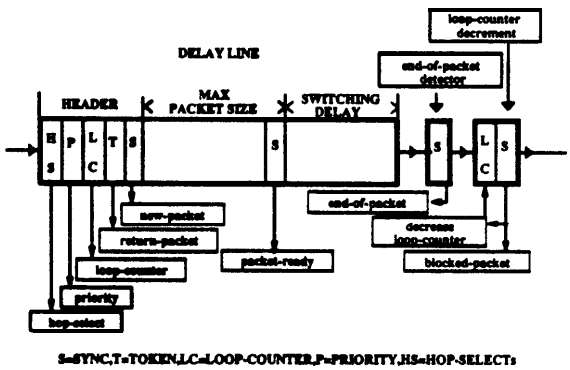

Figure 19: ***Delay Line*** with ***loop-counter implementation***

mechanism includes a decrement circuit. This circuit, as well as the circuit that tests the value of the loop-counter, operate on returned packets only. No action is necessary when the packet is forwarded. The modified node design that includes the implementation of the ***loop-counter*** is presented in Figure 18. The structure of a ***Delay Line*** is shown in Figure 19. While the packet enters **a *Delay* Line** the ***loop-counter*** is checked by the Control. In case the value of the ***loop-counter*** is zero, the packet is discarded by connecting the output of the ***Delay Line*** to the ground. The other possibility is to pass the discarded packet to a host performing the function of the switching node monitoring (named Monitor).

As a blocked packet is clocked out of the ***Delay Line,*** the ***blocked-pocket*** circuit detects the ***sync*** and the ***token*** of the packet which initiate a ***decrease-loop-counter*** signal if the ***token is*** set. The delay between the ***blocked-packet*** and the ***decrease-loop-counter*** circuits are exactly of such a distance that when the ***blocked-packet signal*** is raised, the ***loop-counter*** is received by the ***decrease-loop-counter*** circuit. The operation is, therefore, fully autonomous, not requiring any intervention of the Control.

When the ***loop-counter*** is represented by a binary number, the hardware needed to implement the decrement may be difficult to implement. A somewhat easier solution may be to use a bit pattern **as a *loop-counter.*** In this scheme the ***loop-counter*** is composed of a string of l's, equal in number to the required value of the ***loop-counter.*** Each decrement of the ***loop-counter*** consists now of resetting one such bit. Zero is detected by having all zero pattern. This scheme has the disadvantage of providing unnecessary long ***loop-counter*** field. Fortunately, the maximum value of the ***loop counter*** is expected to be small. Consequently, the ease of implementation justifies the bit wastage.

Yet another approach to the ***loop-counter*** usage is to provide a special ***loop-counter*** per each hop. In this case, instead of the ***hop-***selects fields, the packet header contains fields composed of ***hop-selects*** and ***loop-countens.*** The advantage of this scheme is the possibility of an exact calculation of the packet's life time, as well as the possibility of selectively limiting the delay of each of the loops on the packet's path.

5.3 Broadcast and multicast

Multicast refers to sending a packet to multiple destinations by a single transmission from the packet source.

Routing of multicast packets on ***Blazenet*** is achieved by a tree-like forwarding path, where the source is the root and the destinations are the leaves. A multicast packet is forwarded as a single packet up to the point where it is split to two or more packets forwarded on different links. The split packets can also be multicast packets, in which case each one is split again at some subsequent node.

A multicast packet address is, in fact, a mapping of this tree graph to a linear notation. The linear notation consists of a list of ***hop-selects*** while searching the tree in the following way: visit the leftmost unvisited son of the current node, if any, whose subtree contains at least one destination. Each ***hop-select*** consists now of two subfields: the ***level-indicator*** and the ***output-number.*** The ***level-indicator*** indicates the level of the current node in the whole tree, while the output-number is the number of the loop the packet has to be forwarded on (in the current node). The ***level-indicator*** is actually the hop distance of the current node from the source. Figure 20 shows the ***hop-select*** structure incorporating the above changes.

Upon packet arrival at a switching node, the requested loops are checked for availability and the packet is split to all these requested output loops that are available, if any. The packet is also returned carrying the addressing information of all the blocked outputs, if at least one output loop is unavailable.

While a multicast packet is split within a switching node, the new generated packets carry the addressing representation of the relevant subtree only. The address field is, therefore, divided **among** the new generated packets, whereas the ***syncs,*** the ***token,*** the ***loop-counter,*** the ***priority,*** and the ***data*** portion of the packet are replicated within each one of the new packets. The replication is performed

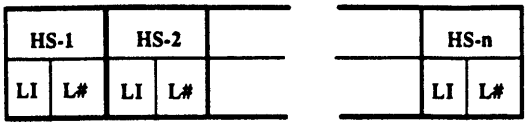

Figure 20: Modified ***hop-select*** structure for multicast delivery

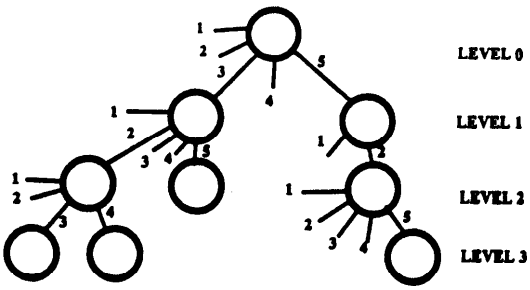

Figure 21: Tree graph of the multicast example

by connecting the input loop to more then one output loops. The division of the address field is performed by replicating the whole address field in each one of the new packets and erasing the irrelevant portion of the address field in any one of the new packets.

The following example clarify the multicast addressing structure. Assume a single packet is to be multicasted to four destinations. The corresponding tree graph is shown in Figure 21. The initial address field is presented in Figure 22. The first number of each ***hop-select*** represents the level indicator and the second one the output loop number. The first path is composed of the following sequence of ***hop-selects: 3, 2, 3, 0.*** The second: 3, 2, 4, 0. The third: 3, 5, 0. The fourth: 5, 2, 5, 0. A ***hop-select*** of the last forwarding node on the packet path (the destination node), is by definition 0. Therefore, **all** the **paths** end with ***hop-select*** equal to 0.

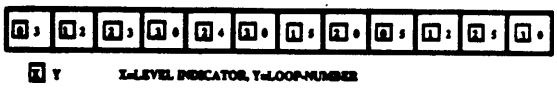

Figure 22: Initial address field for the **multi**-cast example

When the packet in the example arrives at the first node, it is split into two packets: one to be multicast to destinations: 1, **2, 3,** and the second to be unicast to destination 4. The second packet is forwarded to its destination along the route: 2, 5, 0, whereas the first packet, when arrived to the second node on its path, is split once more. One of the new packets goes on output line number 2, the other is forwarded directly to its destination on output line number 5.

The address adjustment for multicast packet is more complicated than for unicast case, because of the necessity of splitting the address field. The Control looks for the ***level-*** indicator ***in*** the first ***hop-select.*** The whole address is then split into as many pieces as there are ***hop-selects*** with the same value of the first ***level-indicutor.*** The division of the packet address filed into pieces is performed by breaking the address field on the boundary of ***hop-selects*** with **values** of ***level-indicator*** equal to the value of the ***Jewel-indicutor*** of the first ***hop-select.*** Each new packet carries one such piece and is then forwarded according to the first ***hop-select.*** During the forwarding process the first ***hopselect*** is erased. The address field of the new packet is, therefore, composed of only the relevant sub-tree.

Another possible addressing scheme for **multicast** on ***Blazenet*** is the usage of **a** single ***hop-select*** field to indicate multiple output connection. In this scheme ***M*** bits are used for each route, each bit for one of the ***M*** possible output loops. A bit is set if the packet has to be forwarded on the corresponding output loop. In this scheme, as in the previous one, the nested structure of the various paths realize the multicast delivery. This scheme is more efficient in the case of multicast to many destinations, however, the control has to be able to create the ***hop-select*** of the returned packet containing the indication of the blocked loops. Thus, this scheme requires more complex Control design. Consequently, the preferred solution depends on the implementation requirements. Figure 23 shows such a representation for the above multicast example.

It is to be noted that in both addressing

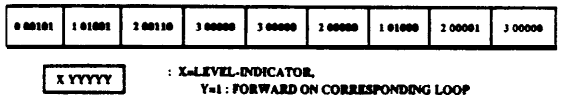

Figure 23: Address field for the multicast example using bit representation.

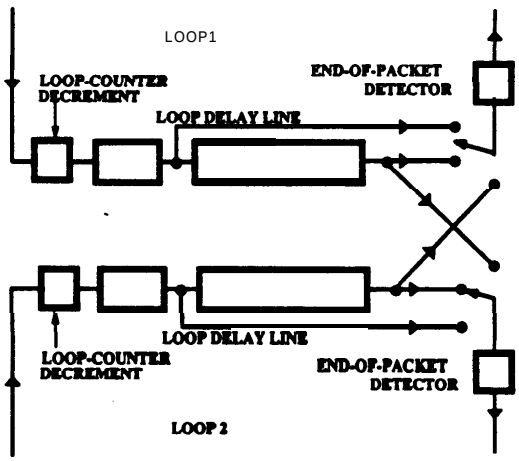

Figure *24:* The ***Bluzenet*** node design, as modified to include flooding.

schemes the packets created by splitting the original packet have unused gaps in the address field. Moreover, even in the unicast case, erasing the used ***hop-selects*** creates gaps. Although it is possible to eliminate these gaps, the cost of the gaps is insignificant, since typically the **header** is only **a** small portion of the whole packet.

Broadcast is **a** special case of multicast, where a packet is to be transmitted to all the possible network destinations. Broadcast can be implemented on ***Bluzenet*** in two different ways: by using the multicast mechanism with address of all the network destinations, or by a flooding approach.

Flooding can be implemented by dedicating a specific ***hop-select*** value to instruct the forwarding nodes to forward the packet on all its loops (possibly with the exception of the loop directed to the node the packet comes from). The first ***hop-select*** does not need to be erased in the forwarding process. By slightly modifying the treatment of the ***loop-counter*** in the switching nodes, the damping

Figure 25: Test packet structure

of the flooding process is guaranteed. The ***loop-counter*** modification consists of decreasing the ***loop-counter*** value each time a packet is received by **a** switching node, whenever the packet is blocked or successfully forwarded. This modification requires placing the ***loop-counter*** decrement unit before the switch of the ***Delay Line,*** as shown in Figure 24. In order to make the packet reception by **all** the network nodes possible, the value of the ***loop-counter*** should be specified to the maximum path length from the packet source to any network destination with some reasonable addition for packet loopbacking. Using this flooding mechanism a broadcasted packet can be **received** more than once. Consequently, higher layers protocols must discard the duplicated packets. Flooding can be used to cope with abnormal network behavior and to increase the network reliability.

5.4 Monitoring the network

Topological and load condition changes in the network require constant adjustments of routing tables and forwarding policies within hosts. The source routing used ***in Blurenet*** gives an opportunity to easily acquire the network changes in a distributed manner. This is performed by hosts named Monitors, possibly one for each switching node, that perform the network data collection operation. Each Monitor initiates tests for link availability and link load condition. These tests are performed by sending packets by the Monitor back to itself over **specific** paths in the network, using source routing. Packets sent through an **unoperative** link are not delivered back to the Monitor. By analyzing information from many network paths, the Monitor can detect incremental changes in the network load and network topology (i.e, availability of **a** specific link).

A test packet structure is shown in

Figure 25. In order to avoid confusion, the *test-nr* field differentiates between various tests (that can be performed concurrently) **and** the *path-nr* field uniquely identifies the specific path under the test. The *input-time* records the time the packet was entered into the network and serves for calculation of the packet delay through the specific path.

In the following discussion we assume that the network changes are incremental, that is the probability of a failure of more than one link or node between any two tests is negligible. Therefore, we can assume that at any time the Monitor ignorance of the network's status is at most a state of one variable.

The tests are performed in the following manner. Each Monitor sents packets over the network to cover all the network links. If a packet does not return, more tests are initiated in **order** to determine which link on the missing packet path is down. The intersection of all the missing packets paths' gives the unoperative link (to remind, there is only one unoperative link, if any). However, in the case **a** link does not have **an alternative,** the link failure cannot be uniquely identified.

If a Monitor decides on a link being **unoperative**, it passes this information to **all** the other hosts connected to the Monitor's switching node and causes changes in their routing tables. Later, from time to time, the Monitor might reissue some tests to check if the status of an unoperative link has changed.

The **same** approach **can** be used in order to locate the areas of congestion in the network. However, more sophisticated algorithms must be used in order to analyze the packets' delays and to evaluate the state of the congestion of **a** specific link or group of links. **A** useful assumption is that **a** link's load does not change rapidly. This assumption can be justified by the fact that the network is of the high-throughput characteristic. Therefore, the influence of **a** single event that might have a dramatic effect on a link's load in a **low**-throughput networks is decreased by the **high**-capacity of high-speed networks. **Also,** the fact that the networks are of **a** mesh topology, contributes to the smoothing effect.

Conclusions from these tests serve as **a** factor in determining the forwarding policies and alternate routing schemes. It should be pointed out that these tests should be implemented in such a way that they do not significantly contribute to the network load. This is accomplished by cleverly designing the tests, so to decrease their number, by performing them with proper frequency, and by using small test packet size.

The Monitor, besides continuously determining the state of the network, can be assign other tasks. One such **a** task may be to serve as **a** collector of discarded packets. Packet that is discarded in the switching node the Monitor is connected to, is passed to the Monitor. An additional field in the *Blazenet* packet format might instruct the Monitor on the necessity of announcing the source of the packet about the discarding operation. The Monitor examines the discarded packet and may initiate **a** special NAK packet back to the packet's source. The NAK packet is composed of the discarded packet header and the reason why the packet was discarded.

The Monitor, as presented in this subsection, serves as a network tool to cope with network malfunctions and with network abnormal behavior. By performing such **a** function, the Monitor turns *Blazenet* into an immune communication channel, unloading some processing burden from the network interfaces, and increasing the reliability of the whole communication process.

5.5 Alternative design choices

One alternative of *Blazenet* implementation to that presented is to use slotted loops. Yet another variation is achieved by using **double**-loop configuration for the bi-directional transmission. Consider first the slotted version.

In the slotted version, the loops are divided into slots of the packet size (in case of variable packet size, the slots are of the maximum packet size). Packets can be inserted only into empty slots, indicated by some bit within the packet format. By appropriately delaying each input traffic, slots' arrivals to a switching node are synchronized, so that all the packets arrive at the same time. Packet can be, then, **inter-**

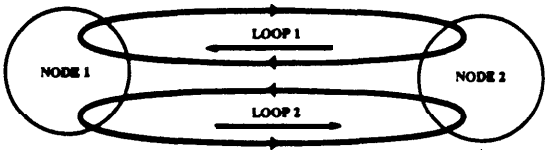

Figure 26: A double-loop configuration

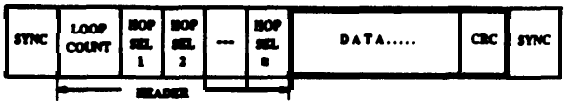

Figure 27: The packet format for the double-loop *Blazene t*

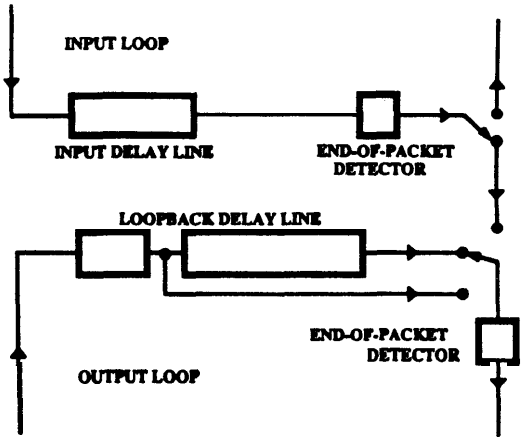

Figure 28: The double-loop switching node design

changed between the slots of the various loops. The slotted version has some advantage in performance over the non-slotted approach. Nevertheless, the required slot synchronization is a serious disadvantage of the slotted version. Also, in a network with **a** variable packet size the usage of the maximum packet length as the slot size may be of some disadvantage.

In the double-loop version of the network two loops replace a bidirectional link of a conventional network. Such a configuration is presented in Figure 26. Lower portion of loop 1 serves transmission from node 2 to node 1, while transmission from node 1 to node 2 uses lower portion of loop 2. Blocked transmission is returned on the upper portion of the loop it came on. No indication of a packet being a returned packet is necessary in the double-loop case, since the usage of the upper portion of a loop indicates that the packet is **a** blocked one. The modified packet format is shown in Figure 27.

Figure 28 shows the modified block design of a ***Blazenet*** switching node to accommodate the double-loop configuration. Each ***Input Loop*** has a Delay Line ***(Input Delay Line),*** build out of a piece of fiber. The ***Input Delay Line*** **is** long enough to contain the leading ***sync,*** the ***loopcount,*** the ***hop-selects,*** and the number of bits corresponding to the time for the control logic to do the actual switching. Figure 29 shows the signals extracted from the transmission entering the ***Input Delay*** **Line** and the corresponding timing. Upon ***sync*** detection, ***a*** pattern detection circuit raises the ***new-packet*** line, indicating to the Control ***a*** new packet arrival. At this time the Control looks for the values of the ***loop-count*** and the ***hop-select*** signals. The indication that a packet leaves the ***Input Delay Line is*** provided to the Control by the ***end-of-input*** line.

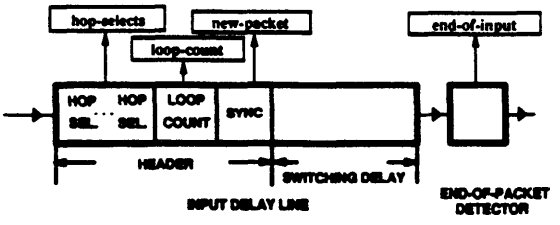

Figure 29: The ***Input Delay Line*** signals and their timing

Each ***Output Loop*** also has its own Delay Line, the ***Loopback Delay Line.*** The ***Loopbuck Delay*** Line must be the length of the maximum packet size plus the switching delay. The signals extracted from the information within the ***Loopback Delay Line*** and their timing are presented in Figure 30. Upon detection of the leading ***sync*** pattern of a returned packet, the ***return-packet*** signal is raised. This indicates the occupation of the ***Loopback Dela y Line.*** Similar circuit, positioned at the end of the ***Loopback Delay Line,*** scans for the trailing sync. Detection of the trailing ***sync*** by this circuit initiates the ***end-of-output*** signal, indicat-

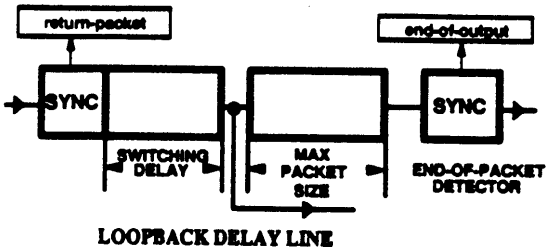

Figure *30:* The ***Loopback Delay*** **Line** signals and their timing

ing when **a** packet leaves the ***Loopback Delay Line.*** Using the two signals, the Control can uniquely decide on the ***Loopback Delay Line*** state.

The process of forwarding **a** packet in the double-loop configuration is similar to that of the single-loop case with proper differentiation between the ***Input Deluy Line*** **and** ***Loopback Deluy Line*** functions.

The main advantage of the single-loop over the double-loop version is in reduction of hardware: fibers, transmitters, receivers, etc. The double-loop version is simpler to implement, possesses some reliability advantages, has lower delay and stable throughput under **heavy-load.**

Blazenet **variations can** be combined. Thus, single-loop and double-loop ***Blazenet*** can operate on slotted or unslotted loops. In this paper we concentrate on the non-slotted single-loop version.

5.6 Support for stream traffic

As showed in [2], the ***virtual partial cut-through*** switching method is the preferred switching technique in the ***transactional*** environment. ***Bluzenet is*** not an exact implementation of any conventional ***cut-through*** technique. The reason is that even though ***Blazenet*** does not store an unblocked packet, a blocked packet is stored for longer time than necessary due to the fact that storage of the loops can be "accessed" only at discrete instances. Simulation results, presented in Section 4 shows that delay degradation is not significant (especially at low-load). If the last loop on each link is of the length of a single packet size, the implementation tends to resemble more the ***full virtual c&through*** technique. Although, the **storage is** still "accessed" at discrete instances in this case, the access is more frequent, leaving the gap between the end of the current transmitted packet and the "stored" packet of the maximum size of one packet length.

As pointed out **in [3],** for **wide-area** networks operating under low-load conditions, there is marginal gain to have **a** single very high-speed channel of capacity larger then some threshold **value** (which is in the range of Gbps). It is more advantageous to have multiple parallel high-speed channels (loops ***in Bluzenet*** case) operating at the **capacity** of the threshold. This **arrangement** has also the advantage of providing an increased network reliability.

In order to improve ***Bluzenet*** performance for transmissions that are more of the ***stream*** type, we present a different switching scheme that can be incorporated into the ***Blazenet*** design, and that can integrate ***streum*** traffic without the excessive overhead of conventional circuit-switching, yet **capable** of dedicating a path through the network. For reference we **call** this scheme ***Loop-Switching.*** The basic idea is to reserve **a** loop for the duration of the whole message. This is done in the following manner: a host generating traffic injects its packets into a loop. Upon its arrival at the next switching node, the stream of these packets try to reserve the next loop on the path. If the loop is unavailable, some of the first packets might be returned, the following ones will be forwarded on the now-reserved loop. **A** reserved loop is dedicated for the transmission until there is no arriving packet to be forwarded on the loop for a period of at least one round trip time of the loop. At this point the end of the **message** is declared and the loop is freed for other connection. Thus, the packets of the message diffuse through the network reserving the path for itself. ***Loop-Switching*** assumes existence of many parallelly operating loops, forming a single link. The usage of the scheme is justified only for ***stream*** transmissions that occupy the channel for duration of the order of the propagation delay

of a single hop. Also, special indication must be included in the packets header to indicate that the ***Loop-Switching*** service is required. We note the ease of integration of ***stream*** and ***bursty*** traffic when the ***Loop-Switching*** technique is optionally provided.

6 Higher layers

Since ***Bluzenet*** does not provide error detection on the ***header*** portion of a packet, packets, besides the possibility of being lost or discarded, can also arrive at a wrong destination. The data portion of the packet can, optionally, have error protection. However, if the information of the packet destination is embedded into the data portion of the packet, the transport layer discovers packet misdelivery and discards the erroneous packet. Packets discarded by the destination, packets discarded by the network, and packets lost while in transit are retransmitted by the source of the packet after a NAK is received or after some time-out expired.

Packets are also not guaranteed to arrive in the order they are sent into the network, since a formerly sent packet can be blocked and arrive after a later sent packet, which does not undergo blockage. Also here, transport layer has to take care of the packet reordering, creating a transparent service for the end-to-end communication.

Bluxenet provides some limited flow control on the physical layer. When the load increases the loops become more populated and less traffic can be inserted into the loops. Thus, the flow control is performed by the back-pressure that propagates from the point of congestion to the entrances of the network. This flow control is basically at the ***hop level*** and in a limited sense also at the ***entry-to-exit level. Bluzenet*** does not support higher level flow control.

Time stamps, done by the ***loop-counter*** mechanism, have two major roles: to sup port real-time traffic delivery and to avoid erroneous infinite traffic circulation. In more sophisticated application, the priority of the packet can be varied according to the value of the time stamp. The priority of packet with smaller residue life time will be increased. Transport layer, session layer and possibly even the application layer may play role in the determination of the value of the ***loop counter.***

7 Summary and conclusions

In this paper we have presented ***Bluxenet,*** a wide-area high-speed packet-switched network suitable for fiber optics implementation. We have discussed ***Bluzenet*** architecture, ***Bluzenet*** operation, ***Blurenet*** switching node design, ***Bluzenet*** performance, and ***Blazenet*** extended features.

Closer look on ***Bluzenet*** reveals some of the network's salient properties: high-speed switching, the lack of conventional memories, good behavior under traffic load, flow control by the back-pressure mechanism, priority traffic, multicast delivery, and the possibility of photonic implementation. Specifically, ***Bluzenet*** provides switching of multi-gigabit per second data rates, low delay, and good behavior under load.

The use of source routing allows each switching node to make switching decisions on the fly, minimizing the switching delay. The use of a loopback channel, which effectively stores packets that are blocked at the switch, minimizes the packet loss under load without requiring additional memory within the switch. Simulation results indicate that the ***Blurenet*** performance is comparable for low-load operation to the ideal case of a nonblocking network, and is much better than that of the ***Lossy*** networks.

Finally, the simplicity of the switching node as a result of the use of source routing, and the absence of switching buffer memory makes it feasible to realize the switching node through the use of photonics. Photonics makes the switching node more immune than electronics to electromagnetic monitoring or interference. It also provides greater performance and reliability, especially as photonic technology matures ([16]).

The importance of ***Blazenet*** extends beyond the mere fact of the existence of another com-

munication network design. The ***Blusenet*** concept demonstrates the feasibility of packet-switching in high speed networks. In the other words, the ***Bluzenet*** design shows that it is not necessary to resort to circuit-switching to handle the data rates made possible by **optical** fiber. In fact, when computer traffic has to be carried, packet-switching has some crucial advantages over circuit-switching, advantages that are emphasized in high-speed networks. Consequently, ***Bluxenet*** provides the basis for packet-switched, high-speed networks designs.

In general, the ***Bluzenet*** switching node is a simple, universal, high-performance component suitable for optical implementation, providing low delay and high bandwidth communication, with support for multicast, priority traffic, and real-time traffic, features that appear essential for the next generation of communication **applications.**

We *see Bluzenet* as a representative of a future class of networks that behave as passive "light pipes" for **data,** offering high throughput, low delay, and high reliability. With the introduction of this class of wide-area networks, we expect that the computer interfaces, rather than the networks, will appear to be the performance and functionality bottlenecks of the communication process. However, further research **and** development are required to make this perception of the future a reality. In particular, an actual photonic realization of the ***Blazenet*** concept is of great importance. Today's state of the art in photonic switching permits such a realization only to a limited degree ([16]). Nevertheless, this limited realization **can** serve as a first step towards a future **all-photonic** communication network.

8 Bibliography

References

[1] **Z. Haas** and D. R. Cheriton, ***"Blazenet:*** A High-Performance Wide-Area Packet-Switched Network Using Optical Fibers," in Proceedings of the IEEE Pacific RIM Conference on Communication, Computers and Signal Processing, Victoria, B.C., June 45, 1987.

[2] **Z. Haas,** "Packet-Switching in Future High-Performance Wide-Area Networks," Ph.D. dissertation, Electrical Engineering Dept., Stanford University, in preparation.

[3] **Z. Haas** and D. R. Cheriton, "A Case for Packet-Switching in High-Performance Wide-Area Networks," in Proceedings of SIGCOMM '87 Workshop, Stove VT, Aug 11-13, 1987.

[4] **A.** Bellman, "Switching Architectures Towards the Nineties," in Proceedings of the International Seminar on ***Digital Communications, New Directions in Switching und Networks,*** Ziirich, Switzerland, March 11-13, 1986.

[5] **J.** S. Turner, "New Directions in Communications," in Proceedings of the International Seminar on ***Digital Communications, New Directions in Switching and Networks,*** Zurich, Switzerland, March 11-13, 1986.

[6] **L.** C. Blank ***et al.,*** "120-Gbit . km lightwave system experiments using 1.478-μm and 1.52-μm distributed feedback lasers," in ***Proceedings of the Conference on Optical Fiber Communication,*** 1985, pp. 86-87.

[7] **S. R.** Nagel, "Optical Fiber-the Expanding Medium," ***IEEE Communications Magazine,*** April 87, vol. 25, **No.4,** pp. 33-43.

[8] **L.** Thylén, "High Speed, Wide Bandwidth Digital Switching and Communications Utilizing Guided Wave Optics," in Proceedings of the International Seminar on ***Digital Communications,*** **New Directions** in Switching ***and Networks,*** Zurich, Switzerland, March 11-13, 1986.

[9] **F.** Guterl and G. Zorpette, "Fiber optics: poised to displace satellites," ***IEEE Spectrum,*** August 1985.

[10] M. Aoki, T. Uchiyama, S. Tonami, A. **Hayakawa** and H. Ichikawa, "Protocol Processing for High-Speed Packet Switching Systems," in Proceedings of the International Seminar on ***Digital Communications, New Directions*** **in** ***Switching and Networks,*** Zurich, Switzerland, March 11-13, 1986.

[11] J. S. Turner, "Design of a Broadcast Packet Switching Network," Washington University, Computer Science Department, technical report WUCS-85-4, March 1985.

[12] **M.** Sakaguchi and K. Kaede, "Optical Switching Devices Technologies," in Proceedings of the International Seminar on ***Digitul Communicutions, New Directions in Switching and Networks,*** Zurich, Switzerland, March 11-13, 1986.

[13] R. **I. MacDonald,** "Optoelectronic Switching," in Proceedings of the International **Seminar** on ***Digital Communications, New Directions in Switching and Networks,*** Zurich, Switzerland, March 11-13, 1986

[14] H. S. **Hinton,** "Photonic Switching Using Direct **ional** Couplers," ***IEEE Communication Magazine,*** **vol.25, No.5,** May **1987.**

[15] P. R. Prucnal, D. J. Blumenthal and P. A. Perrier, "Photonic Switch with Optically Self-Routed Bit Switching," ***IEEE Communication Magazine,*** **vol.25, No.5,** May 1987.

[16] P. R. Prucnal, "All-optical ultra-fast networks," ***SPIE Fiber Telecommunications and Computer Networks,*** Vo1.715, 1986.

[17] S.D. Personick "Photonic Switching: Technology and Applications," ***IEEE Communication Magazine,*** **vol.25, No.5,** May 1987.

[18] Trudy E. Bell. "Optical Computing: ***A Field*** **in** ***Flux,"*** IEEE Spectrum, August 1986, volume 23, number 8.

[19] Ira Jacobs. ***"Design Considerations*** for ***Long-Haul*** Lightwave ***Systems,"*** IEEE Journal on Selected Areas in Communications, December 1986, volume SAC-4, number 9.

[20] **A. Tanenbaum.** ***"Computer $N_e t_{wor} k_s$,"*** Englewood Cliffs: Prentice Hall, 1981.

[21] **P.** Kermani and L. Kleinrock, "Virtual Cut-Through: A New Computer Communication Switching Technique," ***COMPUTER NETWORKS,*** 1979

[22] W. J. Dally and C. L. Seitz, "The Torus Routing Chip," ***Journal of Distributed Computing,*** vol. 1, no. 3, 1986

[23] P. Green, ***ed. "Computer Network Architecture and Protocols," New*** York: Plenum Press, 1982.

[24] H.Ikeman, ***et al. "High-Speed Network uses Fiber Optics,"*** Electronics Week, **Oct** 22, 1984