ABSTRACT

Emerging B-ISDN applications are accelerating the demand for high-speed and high-performance fiber optic local and metropolitan area networks (LANs and MANs). Most of the networks proposed for such an environment are based on either a folded bus or dual unidirectional bus technology, primarily due to the unidirectional nature of the fiber optic medium, their cost effectiveness, and their ability to carry asynchronous transfer mode (ATM) cells. Consequently, the design of the medium access control (MAC) protocols is the most crucial aspect for these networks since the decisions made at this level will determine the capabilities and the hardware cost of the network. In this article, the author proposes a novel MAC-layer protocol, called Optical Reservation Multiple Access (ORMA), which is suitable for data traffic on fiber optic folded bus MANs and LANs. Its main feature is an efficient optical reservation technique to allow stations to reserve transmission slots, which is attractive for use in fiber optic networks. Unlike conventional protocols, where reservation is mostly done using software, ORMA performs its reservation using simple and effective optical circuits. Complete descriptions of the ORMA protocol and the associated nodes are given. Then the performance of ORMA is investigated as a function of the throughput, mean packet delay, and fairness among stations using analytical and simulation results. Our protocol is shown to achieve high throughput and small transmission delays while preserving the fairness of the whole network. Moreover, it compares favorably with related state-of-the-art networks. Thus, the ORMA protocol lends itself well to future-generation high-speed MANs and LANs.

ORMA: A High-Performance MAC Protocol for Fiber-Optic LANs/MANs

Mounir Hamdi, Hong Kong University of Science and Technology

Data and telecommunications are presently experiencing an evolutionary change. Technological progress promises the deployment of optical and electronic components with nearly unlimited transmission capacity and multiples of the processing speed of current systems. These advances facilitate the support of numerous new services by simultaneously giving access to a huge number of users. Concurrent to this evolution, an increased interest in real-time and multimedia applications can be seen in engineering, medicine, manufacturing, and entertainment. As a result, there is an increasing demand for high-speed, high-performance fiber optic local and metropolitan area networks (LANs and MANs) to replace today's dedicated communications systems by supporting a wide range of applications.

Most of the networks proposed for this new broadband integrated services digital networking (B-ISDN) environment are based either on folded or dual unidirectional fiber optic bus technology [1, 2]. With this design choice comes the challenge of designing medium access control (MAC) protocols for these networks. As a result, researchers are actively developing MAC protocols that will provide the performance and services expected of next-generation LANs and MANs. One reason for the importance of this research is that all higher-layer services are built on the fundamental packet transfer service, which is provided by the MAC sublayer, and it is the MAC protocol that determines the characteristics of this fundamental service. Hence, improvements to MAC services result in improved system performance, while the provision of new MAC services means that new applications can be developed. Using optical fiber as a transmission medium, which is projected to have speeds beyond several gigabits per second and span distances of hundreds of kilometers, imposes more strict constraints on message/packet/slot processing times at the intermediate nodes, thereby providing a new challenge to MAC-level protocol designers.

In particular, a MAC protocol for high-speed LANs/MANs should possess several basic characteristics. First, it must be simple enough to be implemented directly in hardware so that it can exploit the bandwidth capacity offered by optical fiber. Second, it must be fair; that is, the throughput of a station should be independent of its location within the network. In addition, a bursty station should not be served to the detriment of other stations. In other words, the MAC protocol should not allow a single station to use all the available bandwidth inadvertently. However, the MAC protocol should be flexible enough to allocate bandwidth according to the applications' demands. Third, the throughput of the MAC protocol must be independent of the network's size and transmission rate. This item is related to the well known *a-parameter*.[1] Any MAC protocol which is sensitive to this parameter limits its usefulness to networks up to a certain size and rate. Fourth, the access delay of the stations should be bounded. This ensures that a process will always be able to transmit at regular intervals regardless of the current network load. This is especially important for real-time applications such as voice and/or video transmission. Finally, the performance (i.e., access delay and throughput) of the MAC protocol should be predictable. The presence of unpredictability requires more resources to be allocated at the stations by the users to be able to cope with the unexpected, and increases the complexity of the MAC protocol.

This research work is supported in part by a grant from the Hong Kong Research Grant Council RGC/HKUST 619/94E.

[1] *The normalized propagation delay, a, is defined in the literature as the ratio of the propagation delay to the mean packet transmission time.*

0163-6804/97/$10.00 © 1997 IEEE

Our aim in this article is to introduce and evaluate the design of a new MAC protocol, called *Optical Reservation Multiple Access* (ORMA), which is especially suited to fiber optic LANs/MANs and can satisfy the performance requirements described above. The ORMA protocol is based on an efficient explicit reservation mechanism using novel hardware solutions. The originality of the protocol lies in the decoupling of the reservation and transmission cycles to achieve high performance and simplicity in the protocol.

This article is organized as follows. In the following section, we overview state-of-the-art MAC protocols for high-speed networks. The section after that introduces the architecture of the ORMA high-speed network. We then describe the ORMA MAC protocol and its implementation details. The performance of the ORMA protocol is discussed, describing both an analytical model and results obtained from extensive simulation. In the final section, we give some concluding remarks.

Alternative Approaches

Recently, several new MAC protocols for high-speed LANs and MANs have been proposed with the aim of meeting some or all of the requirements listed earlier, including distributed queue dual bus (DQDB) [3], P_i-Persistent [4], Load-Controlled Scheduling of Traffic (LOCOST) [5], and Cyclic Reservation Multiple Access (CRMA) [6]. Most of these protocols are based on either a folded or dual unidirectional bus topology, primarily due to the unidirectional nature of the fiber optic medium. Furthermore, they use fixed-size transmission slots generated by headend nodes for data transmission.

The DQDB MAC-level protocol has been accepted as the IEEE 802.6 standard for high-speed MAN networks [3]. The underlying network of the DQDB is a dual bus consisting of two unidirectional slotted buses, A and B, operating in opposite directions. The slots are generated by the headend node of each bus. Every node receives and transmits on both buses, so bus selection is based on the destination. Reservations for transmissions on bus A are made on bus B via requests and vice versa. The DQDB protocol reserves a slot on bus A via a request bit in a slot on bus B. Access to the bus is controlled by request and countdown counters. The request counter keeps track of the current free-slot requirements of downstream nodes, whereas the content of the countdown counter indicates the number of free slots to be passed before the node's own transmission takes place. Transmissions are scheduled one at a time. Scheduling is done by transferring the contents of the request counter to the countdown counter, resetting the request counter, and initiating the transmission of a request on the reverse bus. The purpose of this distributed queuing is to obtain, or at least approximate, a single view of a first-in first-out (FIFO) queue for each pending transmission in all active nodes across the network.

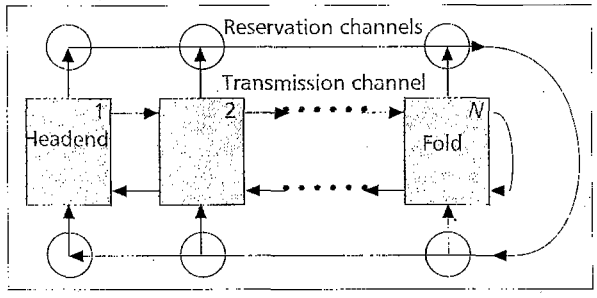
DQDB has been designed to improve channel utilization. It can achieve full channel utilization; however, some problems with unfairness in the protocol have been identified [7, 8]. Specifically, it has been shown that a station will receive better service (in terms of higher throughput, lower mean message delay, or both) if it is located at the head of the DQDB bus; and over both buses combined, the station will perform better if it is located at either end of the network. This skewedness becomes more prominent when the network's end-to-end propagation delay becomes significantly greater than the packet transmission time. To overcome the unfairness problem, numerous modifications to the basic DQDB protocol have been proposed [7, 9]. However, while these improvements solve, to some extent, the unfairness problem of DQDB, they render the resulting protocols more complex. This, in turn, can have a big effect on the performance of the protocol and, more important, on the hardware cost of the stations' interfaces to the channels.

Under the P_i-Persistent protocol, a ready station persists in its attempts to transmit its packet in an empty slot with probability P_i until the transmission is complete. In order to increase the channel utilization and be fair to all stations, each individual station needs to modify its P_i based on channel activity, the estimated number of active stations, and their traffic loads [4]. Using the LOCOST protocol, every station measures the traffic intensity of the channels and then, based on this measurement, determines its transmission rate until the next measurement is made [5]. The main idea in the last two approaches is requiring each station to monitor the traffic on the channels and, based on the statistics observed, throttle its transmission rate accordingly. They can indeed improve the fairness of the protocol; however, there are two potential problems associated with these schemes. First, each station adjusts its transmission rate according to the estimated traffic load. It is very likely that the estimated load is different from the real load; therefore, high channel utilization may not be achieved. Second, it may take a longer period of time for the system to reach a completely fair state. It is also possible, especially when the traffic fluctuates dynamically, that a completely fair state may never be reached. Unfortunately, these deficiencies generally do not show in the simulation analysis, as mentioned and analyzed in [10, 11]. Finally, monitoring the traffic load of the channel, and in the case of the P_i-Persistent protocol monitoring the traffic load of individual channels on top of that, can have a big effect on the cost of the interface boards between the stations and the network.

The cyclic reservation MAC protocols attempt to solve the unfairness of the DQDB-type protocol and the instability of the statistics-based protocols through explicit reservation mechanisms whereby stations have to reserve transmission slots in advance in order to gain access to the channel [11]. Among the many proposed cyclic reservation protocols, CRMA has received the most attention [6, 12] because it can achieve high channel utilization while guaranteeing fairness among stations. In CRMA, the stations access the bus according to cycles of slots. Each cycle is explicitly numbered by an integer, the *cycle number*. The lengths of the cycles are not fixed and are a function of station demands. The stations reserve slots in each cycle, and the headend node generates a cycle sufficiently long to satisfy these reservations. The reservation and generation of cycles are based on two access commands called `RESERVE` and `START`, respectively. These commands are issued by the headend node and have the cycle number as an argument. Each `RESERVE` command also has the *cycle length* as an argument.

A 1.13 Gb/s prototype network using the CRMA protocol has already been built by IBM Zurich and is fully operational [13]. However, the CRMA protocol has some problems of its own. First, CRMA requires more complex structures at both the nodes and the headend station. As a result, the cost of an interface board between a station and the network can be almost as costly as the stations themselves, as shown in [13]. Second, the reservations made by the nodes are uncertain and must be confirmed by the headend station to be valid. They can be rejected; in such a case, a retry is necessary by the nodes.[2] Consequently, nodes have to maintain three message

[2] *A reservation slot contains the reservations made by a number of nodes. When a reject slot is unused by the headend station, the outcome is that the reservation slots present on the bus are invalidated altogether.*



■ **Figure 1.** *The ORMA network architecture.*

queues, namely the confirmed reservation queue (CRQ), tentative reservation queue (TRQ), and entry/reentry queue (ERQ). The headend station also maintains two queues, the global reservation queue (GRQ) and the elasticity buffer, which is also a queue despite its name. Multiple reservation slots coexist on the bus at any given instant, and the number of control slots is six (*Reserve, Confirm, Start, Reject, Unused,* and *Noop*). Furthermore, since the reservation slots and data slots are multiplexed onto the same channel, the generation of a high number of reservation slots will decrease channel utilization. On the other hand, the generation of a small number of reservation slots may not be able to keep up with the stations' needs for reservations.

Our aim in this article is to design a new MAC protocol for high-speed LANs and MANs that can retain the nice characteristics of the CRMA protocol while solving its existing problems. In addition, the associated hardware cost of this protocol should be minimized to justify its cost-effective implementation. Toward achieving this goal, we propose a new MAC protocol, Optical Reservation Multiple Access (ORMA). The ORMA protocol uses novel and simple hardware solutions for cycle reservation which can be overlapped with the data transmission by employing separate reservation channels. This new protocol has been evaluated analytically and by using extensive discrete-event simulations, and has been shown to achieve full channel utilization. Furthermore, it is a fair protocol; that is, all stations have equal opportunity to access the transmission medium. Also, the network achieves a small average transmission delay which makes it suitable for multimedia and real-time applications.

ORMA Network Architecture

The proposed ORMA protocol is suitable for high-speed data transmission on a folded bus, like the CRMA protocol. The architecture for an ORMA network is depicted in Fig. 1. Similar to most folded bus networks, there are two special nodes in the network, a *headend* node and a *fold* node. The fold node divides the bus into an *out-bound* segment and an *in-bound* segment. An attached node (station) uses the out-bound segment to transmit messages to destination nodes, and the in-bound segment to receive messages from the various stations in the network. The message lengths are fixed, and the transmission and reception of messages are performed using fixed-size slots. These slots are generated by the headend node.

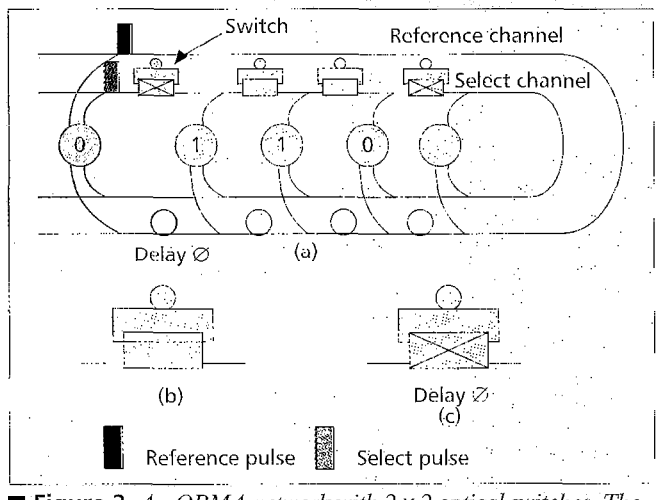
In an ORMA network, there are three separate channels. The first channel, the *data channel,* is used by the attached nodes for exclusively sending and receiving messages. The second channel, the *reference channel,* and the third channel, the *select channel,* are employed as *reservation channels* for the nodes to request transmission slots for their ready messages in the coming network cycles. The three channels may use three physically separate channels. On the other hand, they can employ wave-division multiplexing (WDM) to partition the high bandwidth of the optical fiber into three subchannels, where one subchannel would be used for data transmission and the other two for reservation requests. The former method is a simpler solution since it avoids the technical challenges of WDM (e.g., tuning devices, channel interference). The advantages and disadvantages of these two schemes in terms of performance and hardware cost are beyond the scope of this article. For more details, the interested reader is referred to [14].

The most crucial task in a reservation-based protocol is how to efficiently collect reservation information from all the attached stations so that the headend node generates the appropriate number of transmission slots. If this task is done efficiently, we expect to have an efficient network; otherwise, it can have a detrimental effect on performance. For example, slot reservation is performed in a CRMA network in the following way. Periodically, transmission slots would contain a `Reserve` command embedded into the `command` field of a transmission slot. First, a station has to recognize this command in order to perform reservation. Moreover, it has to increment a `reservation` field in the transmission slot so that the headend node knows how many transmission slots have been reserved. This reservation example has many drawbacks:

- A station must be capable of decoding and recognizing commands (e.g., `Reserve`).
- A station should read and write into specific fields inside a transmission slot.
- The `command` and `reservation` fields occupy valuable bit space within a transmission slot.

As a result, these factors have a tremendous effect on the performance and hardware cost of the CRMA network [13]. The ORMA network is being proposed to solve these problems. This is done by avoiding `Reserve` commands and eliminating the process of reading and writing from and to a `reservation` field.

The ORMA network uses optical conditional delays to allow the reservation of transmission slots by the stations through employing 2 x 2 optical switches, as illustrated in Fig. 2. This is related to the strategy adopted for designing multiprocessor systems [15]. The nodes in an ORMA network of size N are *indexed* (addressed) from left to right, starting with index 0 (headend node) and ending with index $N - 1$ (the fold node). In the ORMA network, each switch $S(i)$ is controlled by station $P(i)$. If all the switches are set to straight, an optical signal incurs the same propagation delay on both the refer-



■ **Figure 2.** *An ORMA network with 2 x 2 optical switches. The propagation delay of an optical signal from* P(i) *to* P(i + *1) on the select channel is* τ *if the switch* S(i) *is set to straight as shown in (b), and* $\tau + \Delta$ *if the switch* S(i) *is set to cross, as shown in (c).*

ence channel and the select channel between any two stations i and j in the ORMA network. However, an additional time delay equal to Δ can be introduced on the select channel by setting switch $S(i)$ to cross, as shown in Fig. 2c. In other words, when $S(i)$ is set to straight, it takes a time τ for an optical signal to propagate from $P(i)$ to $P(i + 1)$ on the select channel, while when $S(i)$ is set to cross, such propagation will take a time $\tau + \Delta$.

Reservation is performed on an ORMA network as follows. At the beginning of each reservation cycle, each attached station, $P(i)$, sets its switch, $S(i)$, to straight if it wants to reserve a transmission slot in the coming transmission cycle; otherwise, the switch stays in its default setting, cross. At the same time, the headend node injects two pulses on the reference channel and select channel, respectively. These two pulses will *coincide*, thus producing a double-height pulse at the station whose index i is equal to the number of reservations made by all stations in the ORMA network. By properly adjusting the detecting threshold of the detector at station i, this double-height pulse can be detected, thereby addressing station i. Thereafter, this station will send its index to the headend node to inform it about the number of reservations made and hence the length of the transmission cycle. The major thrust of this reservation scheme is to determine optically (i.e., with no electronic intervention) and as fast as possible the total number of reservations made by all the stations in a reservation cycle. Again, one of the main reasons for adopting this hardware solution to perform reservation is to avoid forcing the stations to decode, read from, and write into the reservation fields, a process which can delay the transmission rate of the reservation process at each node considerably [1, 13].

Now, we formally present this hardware scheme, *algorithm reservation*, which determines the number of reservations made by all attached stations in an ORMA network during a reservation cycle. We assume that a station, $P(i)$, can set its switch, $S(i)$, to straight by injecting a binary bit 1 into the switch. Hence, if a station, $P(i)$, wants to perform reservation in a particular reservation cycle, it stores a binary bit 1 in a reservation register, a_i. Otherwise, it stores a binary bit 0 in a_i, indicating that it does not wish to perform reservation. Consequently, the addition of the contents of all a_is is equal to the total number of reservations made during that reservation cycle. At the start of each reservation cycle, each station $P(i)$ injects the content of its register, a_i, into its attached switch $S(i)$. Then a *reference* pulse and select pulse are inserted simultaneously on the reference and select channels, respectively, by station $P(0)$ (headend node). If

$$\sigma = \sum_{i=0}^{n-1} a_i,$$

then σ delays (each delay equal to Δ) will be removed from the select channel such that the reference and select pulses coincide at station $P(\sigma)$.

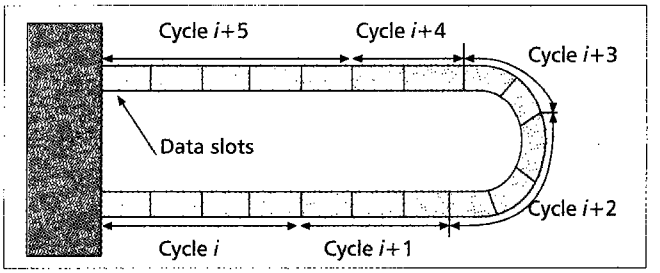
Algorithm Reservation

Input: a binary sequence $a_i = 0$ or 1. Initially a_i is stored at $P(i)$.

Output: $\sigma = \sum_{i=0}^{n-1} a_i$.

(1) $P(i)$ sets $S(i)$ to straight if $a_i = 1$, cross if $a_i = 0$.

(2) At time 0, $P(0)$ injects reference pulse and select pulse signals. If $P(j)$ is selected, then the sum is $\sigma = j$. That is, index j of the station that sees the coincidence of the reference pulse and the select pulse gives the sum σ.



■ **Figure 3.** *Data slots and transmission cycles in an ORMA network. Each cycle contains a variable number of slots depending on the reservation made.*

As an example, Fig. 2 shows the switch settings for the ORMA network when stations $P(1)$ and $P(2)$ want to perform reservation. In this case, the pulse coincidence occurs at station $P(2)$.

Proposition — Using the hardware scheme *algorithm reservation*, index j of the station that sees the coincidence of the reference and select pulses is equal to the sum

$$\sigma = \sum_{i=0}^{n-1} a_i.$$

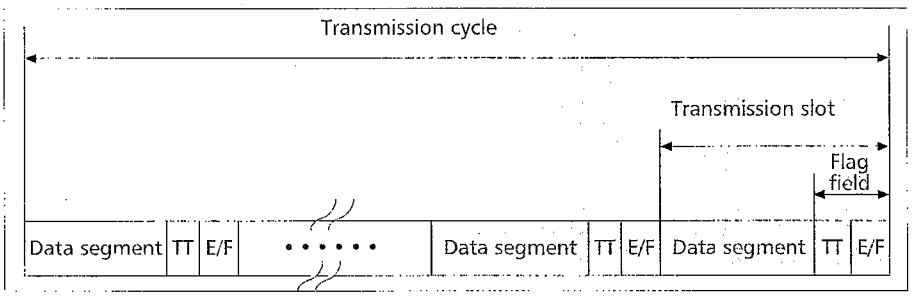
Proof — Since both the reference and select pulses are injected simultaneously on the reference and select channels, respectively, at time $\tau = 0$ (the beginning of a reservation cycle), the time at which the reference pulse arrives at station j (on the lower part of the reference channel) is $t_{r,j} = (n + 1)\tau + (n - j)(\tau + \Delta)$. Let σ be the number of 1s in the binary sequence a_i. Then σ switches will be set to straight. The time at which the select pulse arrives at station j (on the lower part of the select channel) is $t_{s,j} = (n + 1)\tau + (n - s)\Delta + (n - j)\tau$. Let $t_{r,j} = t_{s,j}$; one obtains $\sigma = j$. ■

Realize that it is implicitly assumed in algorithm reservation that the stations of the ORMA network are equally spaced on the folded bus; that is, the physical distance between any two stations in an ORMA network is the same. However, this condition can be relaxed and the same functionality of the above algorithm still achieved [14].

The index of the station where the coincident pulse occurs corresponds to exactly the number of reservation requests made by all stations in the ORMA network. Thus, we have a very efficient hardware scheme that would give us the exact number of reservation requests during a transmission cycle. As a result, the nodes access the channel according to cycles. A cycle consists of a variable number of slots of fixed size, as illustrated in Fig. 3. These cycles represent the payload capacity reserved in previous reservation cycles. A low level of reservations results in short cycles, whereas a high level of reservations results in accordingly long cycles.

This reservation method has many significant characteristics:

- The ORMA reservation scheme gives us the exact number of reservation requests during a transmission cycle, not an estimation such as those adopted by many recently proposed protocols [1, 10]. Moreover, it is done using simple and efficient hardware solutions, unlike traditional solutions involving reading and writing, which take a relatively long time to process and require more hardware to be implemented.
- Because of the separation of the reservation channels and the data channel, and coupled with the fact that the reservation process is very fast, any node can have almost instantaneous access to transmission slots whenever they desire to transmit. As a result, the performance of an ORMA network can be extremely high. Furthermore, it adds more flexibility to the MAC protocol, which would make it suitable for integrated services since the reservation and transmission processes are independent of each other.



■ **Figure 4.** *The transmission cycle and slot format of the ORMA protocol. Each cycle contains a variable number of slots, and each slot contains a data segment and a control segment.*

- The ORMA protocol achieves total fairness among all the stations regardless of their positions on the folded bus. That is, each station gets the same opportunity to perform reservation on any transmission cycle regardless of its position in the network.
- By using separate channels for transmission and reservation, the MAC protocol becomes quite simple. This, in turn, makes its hardware implementation simple and cost-effective.

THE ORMA PROTOCOL

The ORMA protocol manipulates two asynchronous transmissions: message and reservation transmissions. The message transmission is performed using fixed-size transmission slots similar to those employed by other MAC protocols such as DQDB and CRMA [1], and is shown in Fig. 4. Each transmission slot consists of two fields. One field is a 2-bit flag. The most significant bit (E/F bit) indicates whether the slot is empty or full; the other bit, which is a train-tail (TT), indicates if the given slot is the last slot in a transmission cycle. The E/F bit equal to 1 indicates that the slot is full; if equal to 0 it indicates that the slot is empty. The TT bit equal to 1 indicates that the given slot is the last slot in the transmission cycle; if equal to 0 it indicates that the slot is not the last slot in the transmission cycle. The remaining field is the data field where the packet is to be loaded, which includes the source and destination addresses.

We use *reservation cycle* to mean the period of time between the initiation of select and reference pulses by the headend station and its reception of the total number of reservations made from the in-bound segment of the bus. A *transmission cycle* denotes the time needed to produce a sequence of transmission slots equal to the number of reservation requests as determined by the index of the station where pulse coincidence occurred plus their propagation time along the folded bus. The sequence of slots generated during one transmission cycle forms a *transmission train*. Each transmission train has a format similar to that of Fig. 4. All the slots in a train have the TT bit set equal to 0 except the last one, where TT is set equal to 1 to indicate the train-tail to the nodes.

RESERVATION CYCLE

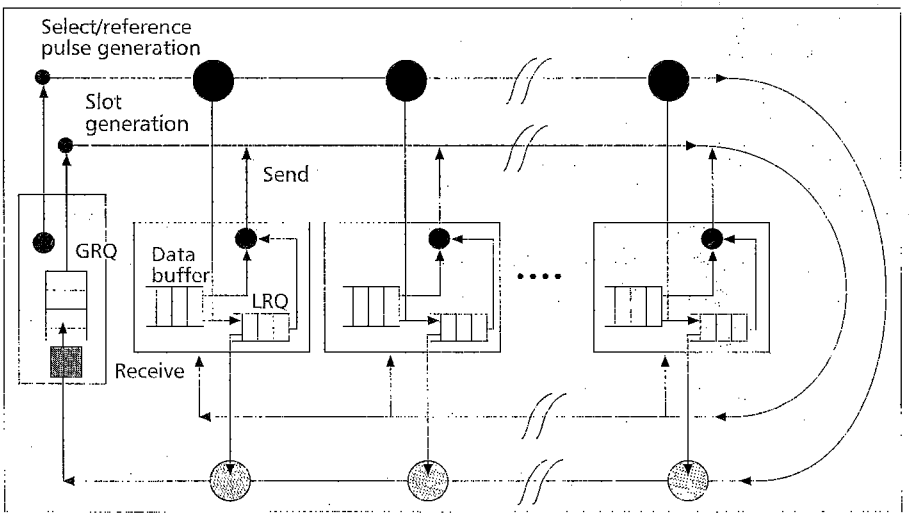
The headend node could simultaneously generate the select pulse and reference pulse once the previous reservation result has been received from the inbound segment of the bus. However, in order to decrease the reservation cycle time, the select and reference pulses are generated in a pipeline fashion without having the headend node wait for a reservation cycle to complete. When the headend station receives the final reservation result (the index of the station where pulse coincidence occurred) from the in-bound segment of the bus, it determines the exact number of slots that should be produced for the corresponding transmission cycle.

The reservation cycle is obviously much shorter than the transmission cycle, one main factor in the efficiency of the ORMA protocol. Furthermore, each transmission slot will require additional delays at each station for address and E/F bit inspection. Consequently, we must have a way of buffering reservation requests in the stations themselves, and we also should have the capability to buffer the results of the reservation cycles in the headend node. Hence, in an ORMA network we should have two types of queues. The first type is the local request queue (LRQ) on each node, which is used to buffer the reservation requests generated by the corresponding node. The other type is the global request queue (GRQ) on the headend node, where the reservation results of each reservation cycle are stored. Figure 5 shows the relationship between these queues.

When a select pulse passes by the switch of a node, the switch enters the *Enable* state. At this time, if the node has a reservation request, a YES flag is queued in the LRQ of this node. If the node has no requests, a NO flag is inserted in the LRQ. Hence, each node inserts an item into its LRQ in every reservation cycle. At the end of each reservation cycle, the reservation result arrives at the headend node. The headend finds the number of reservation requests by all nodes and puts the corresponding number into its GRQ, where that same number of slots would be generated to the nodes. Thereafter, the headend can start sending the reference and select pulses for the next reservation cycle. Consequently, the length of the GRQ is equal to that of each LRQ.

DATA TRANSMISSION CYCLE

During each transmission cycle, the headend node generates a number of transmission slots equal to the number obtained during the reservation cycle, which is stored in the GRQ. Fur-

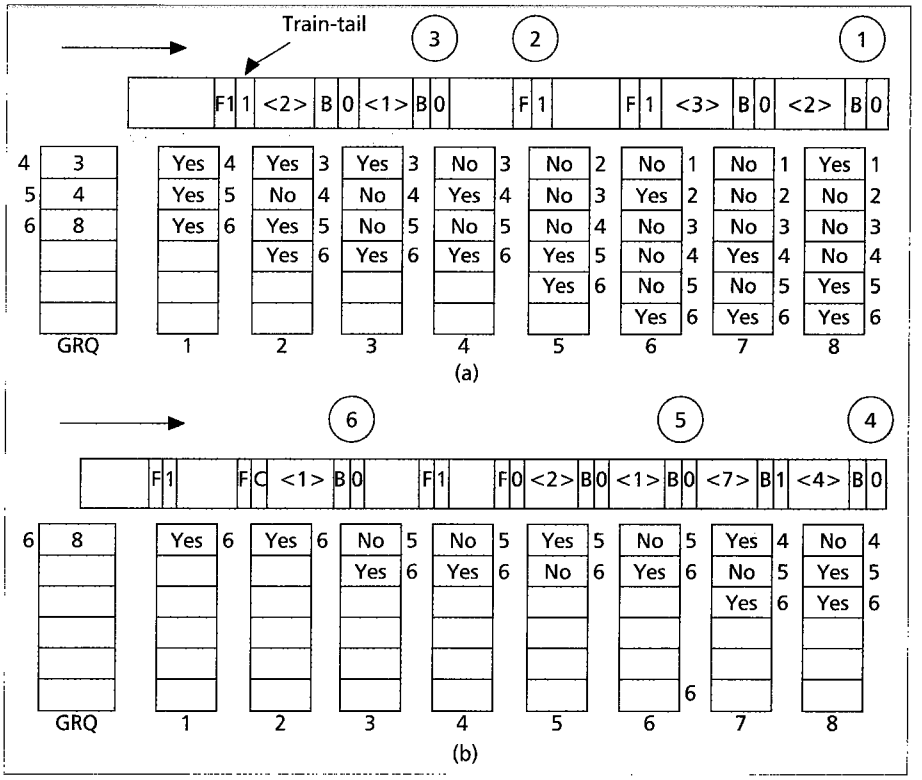


■ **Figure 5.** *The local request queues and global request queues in ORMA which are used to store transmission requests and reservation requests, respectively.*

thermore, the headend node removes that value from its GRQ. This value is simply equal to the number of YES items in the head of all LRQs of the nodes. When a transmission slot passes by a node, the node checks the flag bit of the slot if the head of its LRQ is a YES flag. If the slot is empty (i.e., E/F bit = 0), the node inserts its packet into the data segment field of the slot. If the head of the node's LRQ has a NO flag, the slot passes to the next node with no modification.

When the train-tail (determined by checking the TT bit of each slot) transmission slot passes by, the item in the head of the LRQ is dequeued. The dequeuing operation is carried out by all nodes, including those with items in the head of their LRQ equal to a NO. Figure 6 shows an example operation of ORMA. In Fig. 6a, three transmission cycles are shown (numbered 1, 2, and 3). Transmission cycle 1 contains three transmission slots. Its first two slots have been loaded by node 2 and node 3, respectively, and its third slot will be used by node 8. The number in angled brackets indicates the source node number of this loaded slot. Transmission cycle 2 has only one transmission slot since only node 6 has a reservation request in that reservation cycle. Hence, there is only one YES flag in all LRQs with respect to transmission cycle 2. It can be seen that the number of slots in a transmission cycle (train) is equal to the number of YES flags in that transmission cycle. The GRQ of the headend station has three items which correspond to transmission cycles 4, 5 and 6 since these slots have not been generated yet. The numbers contained in the GRQ are equal to the number of YES flags in all LRQ with respect to each transmission cycle (as indicated by the number marked on the right side of each LRQ).

In Fig. 6b, which is a later snapshot of the network operation, all slots of trains 4 and 5 and part of train 6 have been generated by the headend node. The first slot of train 4 has left the out-bound segment of the folded bus and taken out the slot loaded at node 1. The second and third slots of train 4 have been occupied by nodes 4 and 7, respectively. Note that the snapshot time is the time when node 7 is loading its packet onto the third slot of train 4 and the corresponding item in LRQ of node 7 has not been dequeued yet. It can be seen that all the slots of train 4 have been occupied, and there are no related YES flags left in LRQs afterward. Train 5 has two free slots which will be used by nodes 5 and 7. Train 6 will have eight slots since all nodes have submitted reservation requests in this case. As a result, all the slots generated by the headend node will be utilized by the stations.



■ **Figure 6.** *ORMA data transmission example illustrating how the states of the LRQ and GRQ change.*

THE PERFORMANCE OF THE ORMA PROTOCOL

In this section, we describe the performance of the ORMA protocol. The performance results have been obtained through the use of both an analytical model and and a discrete-event simulator. The simulator is more flexible and can be used to investigate the protocol under a wide variety of situations, whereas the analytical model has been used to confirm the behavior of the simulator.

MODELING OF THE ORMA PROTOCOL

We assume that the ORMA network consists of n stations and the system load is evenly distributed across all stations. Single packets arrive at each station, and their interarrival times are distributed according to a Poisson distribution. The measure of interest is the packet waiting time at each station. When a packet is generated by a station, it has to wait in the station's LRQ to raise a reservation request by setting the attached switch to the appropriate position. Afterwards, the packet has to wait until the time when its transmission request is processed by the headend node. A typical space-time diagram of an arbitrary packet transmission employing the ORMA protocol is shown in Fig. 7.

In this figure, the vertical axis denotes the position of the station in the folded bus, and the horizontal axis denotes the time stages of the packet transmission. The various parameters in the figure are defined as follows:

t_0: an arbitrary tagged packet is generated at node i.
t_1: get a chance to reserve.
t_2: the reservation pulses propagate through the headend node.
t_3: the tagged transmission cycle is started.
t_{4a}: the first slot in the transmission cycle passes by node i.
t_4: the reserved slot passes by node i.
t_{5a}: the first slot in the transmission cycle comes to node j.
t_5: the tagged slot comes to node j, the destination.

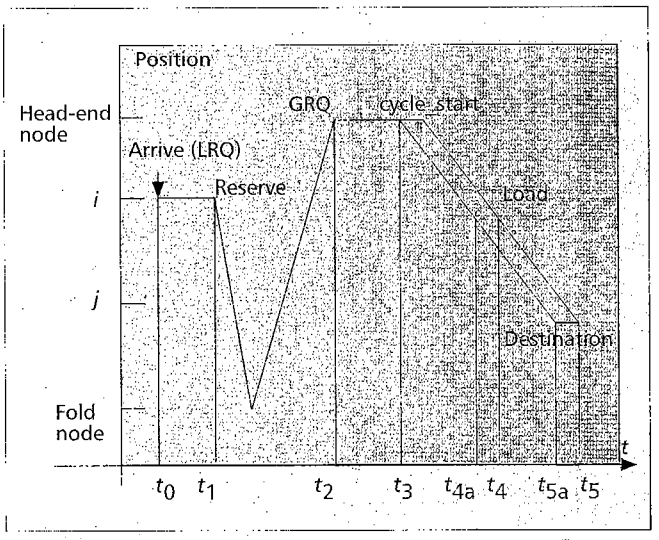
We can see that for this arbitrary packet, the total waiting time, T_w, which is $t_5 - t_0$, can be divided into four parts:

- Waiting time in LRQ: $T_{LRQ} = t_1 - t_0$.
- Waiting time in GRQ: $T_{GRQ} = t_3 - t_2$.
- Reservation train cycle time: $T_{rp} = t_2 - t_1$.
- Data slot cycle time: $T_{dp} = t_5 - t_3$.

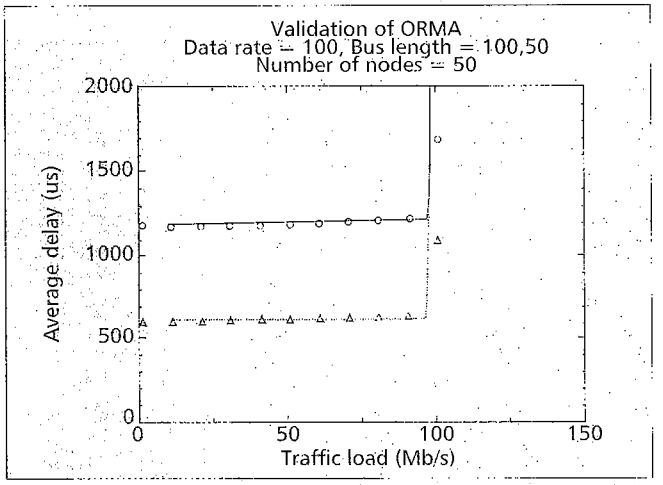
Thus, the mean packet delay can be given by

$$E[Tw] = E[T_{LRQ}] + E[T_{GRQ}] + E[T_{rp}] + E[T_{dp}]$$

The packets arrive in the ORMA network according to a



■ **Figure 7.** *Time-space chart of ORMA transmission, illustrating the components of the ORMA MAC delay.*



■ **Figure 8.** *The ORMA delay using our analytic model and our discrete-event simulation.*

Poisson distribution with mean λ and are uniformly distributed among all the nodes in the network. Therefore, the mean arrival rate at each node is $\lambda_i = \lambda/n$.

In an ORMA network, the frequency of generating select and reference pulses for reservation is fixed and equal to $\mu_r = R_r$, where R_r is the transmission rate of the reservation channels (i.e., select and reference channels). The LRQ is a FIFO queue with Poisson arrival rate of mean λ_i and constant service rate μ_r, so it can be represented by an M/G/1 model [12, 16]. According to [12, 16], the expected number of packets in the LRQ can be given by

$$E[N_{LRQ}] = \rho + \rho^2 \cdot \frac{1+C_b^2}{2(1-\rho)}$$

where $\rho = \lambda_i/\mu_r$, $C_b^2 = \mu_r\sigma_r^2$, and σ_r^2 is the standard variance of service time distribution. Because the service frequency is constant, $\sigma_r^2 = 0$ and $C_b^2 = 0$. Therefore,

$$E[N_{LRQ}] = \rho + \frac{\rho^2}{2(1-\rho)}$$

By Little's theorem [16], the expected waiting time in the LRQ is given by

$$E[T_{LRQ}] = \frac{E[N_{LRQ}]}{\lambda_i} = \frac{2\rho-\rho^2}{2\lambda_i(1-\rho)}$$

Now let us consider the operation of the GRQ. The number of reservation requests is stored in the GRQ. We can imagine that each item in the GRQ corresponds to the reservation of one transmission slot. Hence, the arrival rate of items in the GRQ is the sum of individual requests of all stations (i.e., the sum of the output of every LRQ). Since we have assumed that the input to each LRQ has a Poisson distribution with a mean λ_i, and all LRQs are identical and independent, then, under the condition of steady state of the overall system, the total request rate is given by $\lambda = n\lambda_i$. This is exactly the system arrival rate. Henceforth, we can also consider the distribution of the items arriving to the GRQ to be Poisson. Consequently, the process of the GRQ can be modeled by an M/G/1, where the service frequency is also fixed, but now becomes $\mu_s = R_d/s$, where s is the length of a transmission slot in bits, and R_d is the data rate of the transmission channel. By the very similar method used in the analysis of the LRQ, we can obtain the average waiting time in the GRQ as follows:

$$E[T_{GRQ}] = \frac{E[N_{GRQ}]}{\lambda} = \frac{2\varphi-\varphi^2}{2\lambda(1-\varphi)}$$

where $\varphi = \dfrac{\lambda}{\mu_s}$.

Next we come to the cycle time (transmission time + propagation delay) of the reservation process and transmission slots. Let L be the length of the network; then the round-trip length of the bus is $2L$. Let C denote the speed of light in a guided transmission medium. Then the cycle time of the reservation request made by the first station (which is closest to the headend station or is the headend station itself) is $1/R_r + 2L/C$, while the cycle time of the furthest node is $1/R_r + L/C$. For simplicity we can assume the n stations to be equally spaced along the ORMA network; then the mean cycle time of the reservation requests is given by

$$E[T_{rp}] = \frac{1}{R_r} + \frac{3L}{2C} = \frac{1}{\mu_r} + \frac{3L}{2C}$$

The transmission slots are also generated by the headend station and transmitted through the folded bus. Each packet to be transmitted is loaded on the in-bound segment of the folded bus and received on the out-bound segment of the folded bus. Moreover, the slot occupation in a transmission train is ordered in a FIFO manner, so each packet has to wait for its own reserved slot to pass by to be transmitted. That is, after the GRQ has processed a request, the corresponding packet has to wait until its reserved slot passes by, at which time the packet is loaded onto that slot to be delivered to the destination station. Again, we assume that the destination stations are equally spaced along the out-bound segment of the folded bus; then the mean cycle time of the transmission slots is given by

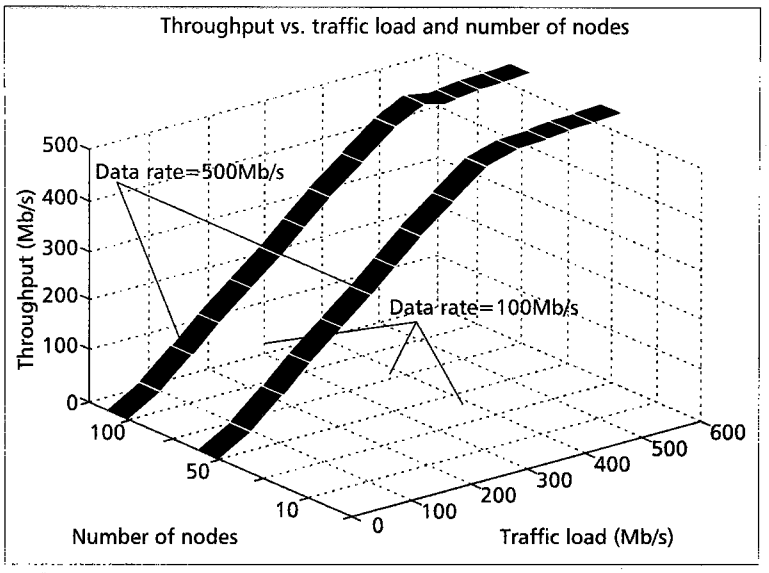
$$E[T_{dp}] = \frac{s}{R_s} + \frac{L}{C} + \frac{L}{2C} = \frac{s}{R_s} + \frac{3L}{2C} = \frac{1}{\mu_s} + \frac{3L}{2C}$$

By combining all the results above, we consequently obtain the mean packet delay of an ORMA protocol as follows:

$$E[T_w] = E[T_{LRQ}] + E[T_{GRQ}] + E[T_{rp}] + E[T_{dp}]$$

$$= \frac{2\rho-\rho^2}{2\lambda_i(1-\rho)} + \frac{2\varphi-\varphi^2}{2\lambda(1-\varphi)} + \left(\frac{1}{\mu_r} + \frac{3L}{2C}\right) + \left(\frac{1}{\mu_s} + \frac{3L}{2C}\right)$$

$$= \frac{4\mu_r n - 3\lambda}{2\mu_r(\mu_r n - \lambda)} + \frac{4\mu_s - 3\lambda}{2\mu_s(\mu_s - \lambda)} + \frac{3L}{C}$$



■ **Figure 9.** *The effect of the network length and number of stations on the throughput of an ORMA network with a data rate of 0.5 Gb/s and 100 Mb/s.*

Figure 8 shows the mean packet waiting time of the ORMA protocol as a function of the traffic load using the above analytical model. It is assumed that the data rate is 100 Mb/s, the network length is either 50 km or 100 km, and the number of attached stations is 50. Furthermore, we assume that $R_r = R_d$. Moreover, an identical ORMA network with the above characteristics has been simulated using a discrete-event simulator. Each simulation has been run for over 30 million slots; the results are not gathered until the first million slots, and are plotted in Fig. 8. As can be seen, the simulated and analytical results match closely.

Simulation of the ORMA Protocol

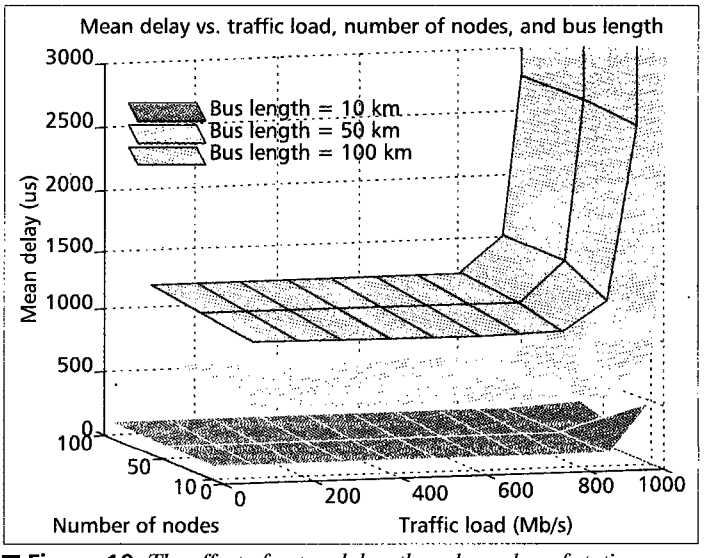
In this section, we present the simulation results and compare the performance of the ORMA protocol with that of the CRMA protocol. The CRMA protocol has been chosen because it has received considerable attention over the past few years and is well understood by many researchers. Every simulation was run for more than 30 million slots, and the results were not accumulated until the first million slots had passed to avoid unusual behavior during startup. All simulations assume the following:

- The stations are equally spaced along the folded bus.
- The load of the network is evenly distributed among the stations, and the packet arrival rate at each station is a Poisson process.
- The transmission slot size is 53 bytes (ATM size).
- The insertion delay for each station is 8 bit times.
- The network size is either 50 or 100 stations.
- The network length is either 50 km or 100 km.
- The data rate is either 100 Mb/s, 500 Mb/s, or 1 Gb/s.

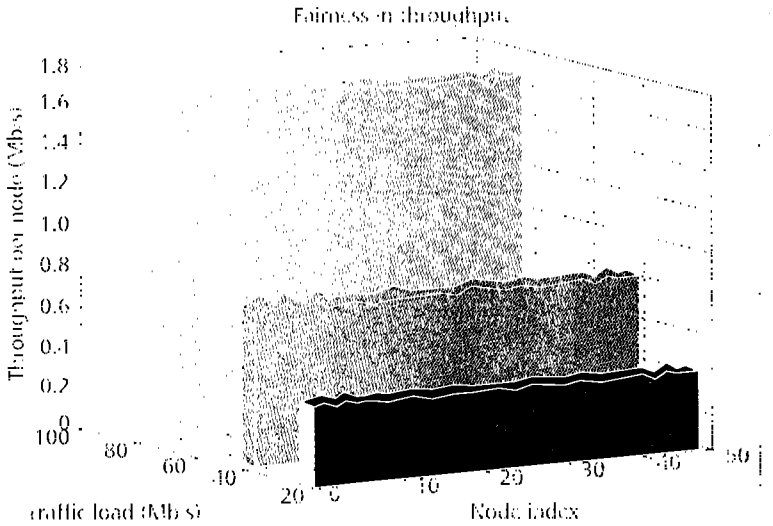
The ORMA Network Throughput — First, we investigate how variations in network size and data rates affect the performance of the ORMA protocol as measured by its system throughput. The throughput (network utilization) of the ORMA network is defined as the amount of user information — including message header bits, but excluding the fields used for packet reservation — successfully transmitted on the network. Figure 9 shows the throughput of the ORMA protocol as a function of the number of attached stations, traffic loads, and input data rates. The throughput seems to be totally independent of the size, length, and data rate of the ORMA network. This is a desirable characteristic of MAC protocols; that is, their performance should not be dependent on the size of, or technology used for, a specific network. The ORMA protocol is equally good for a wide range of networking environments. Moreover, since no packet collision can occur in the transmission channel, the network throughput does not decrease as the input or offered loads increase. In fact, as long as the input rate exceeds the transmission channel rate, the throughput is almost directly proportional to the input load.

The ORMA Network Delay — Here, we investigate the average packet delay of the ORMA protocol as the total network length and number of stations are varied, as shown in Fig. 10. The data rate, D, of the ORMA network is set to 1 Gb/s, the length of the ORMA network, L, is set either to 10 km, 50 km, or 100 km, and the number of attached stations, N, is either 10, 50, or 100. From Fig. 10, we can see that the average packet delay is reasonably independent of the number of attached stations, and is slightly affected by the length of the network (which is unavoidable). Moreover, the mean delay becomes very large only when the load of the ORMA network exceeds 100 percent. Finally, we can note that an average delay of around 0.6 ms for a network of length 50 km and around 1.2 ms for a network of length 100 km seems to satisfy the quality of service needed by most real-time and multimedia applications where a small bounded delay of around a few tens to hundreds of milliseconds is essential.

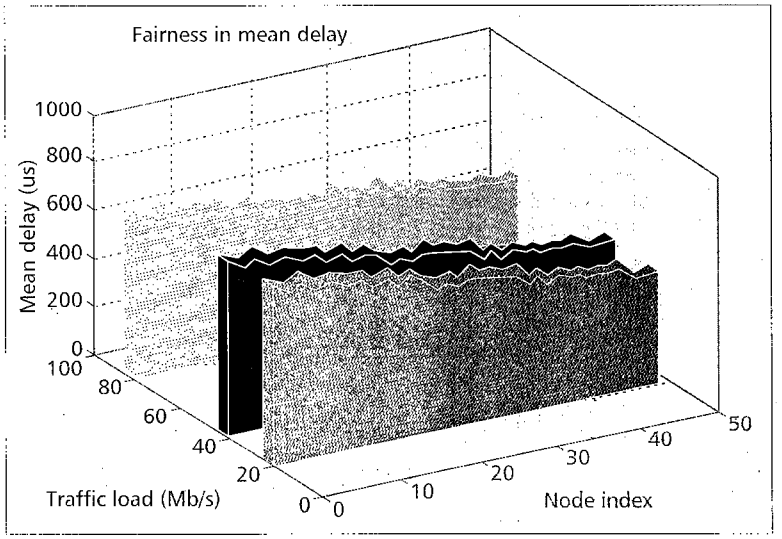
Fairness of the ORMA Protocol — One of the most important characteristics of a MAC protocol is its *fairness*. A MAC protocol must be fair; that is, the throughput and average packet delay of a station must be independent of its location within the network. Furthermore, a bursty station should not be served to the detriment of others, and the protocol should not allow a station to usurp the available bandwidth capacity inadvertently. This does not mean that bursty stations should be assigned lower priority, since every station is entitled to use a reasonable portion of the available bandwidth. However, a bursty station should not be granted additional bandwidth



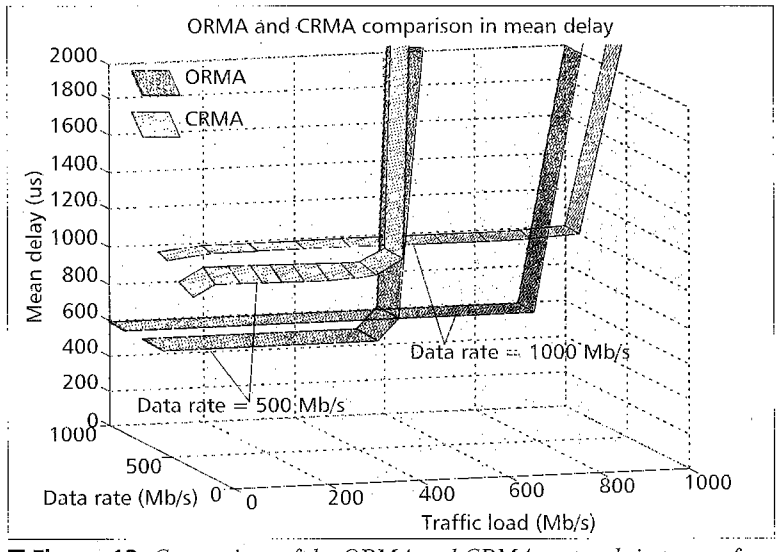
■ **Figure 10.** *The effect of network length and number of stations on the delay of the ORMA protocol for a data rate of 1 Gb/s.*



■ **Figure 11.** *The fraction of throughput allocated to each station of an ORMA network. Data rate = 100 mb/s, bus length = 50 km.*



■ **Figure 12.** *The average packet delay of each individual station in an ORMA network for different network configurations.*



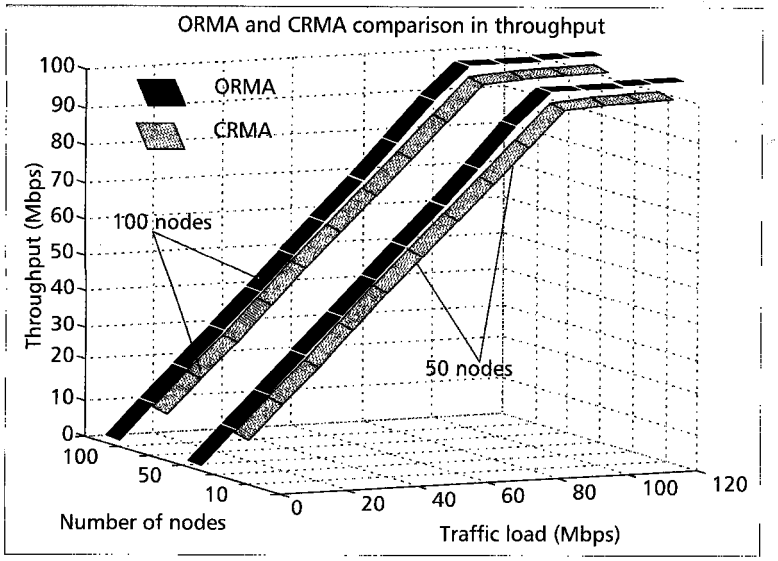
■ **Figure 13.** *Comparison of the ORMA and CRMA protocols in terms of average packet delay.*

stolen from moderately or lightly loaded stations. Also, in the presence of multiple bursty stations, available idle bandwidth should be distributed among them evenly.

The ORMA protocol follows the above description of a fair protocol. This can be seen from the fact that during each reservation cycle, each station has the chance to reserve one or more slots, which would be available to it in the corresponding transmission cycle no matter how many other stations are making reservations. Moreover, the ORMA protocol does not suffer any degradation in throughput to have a fair network. Also, in ORMA, even when there is only a single bursty node with a lot of packets to send, the headend node can still produce successive slots to meet the needs of that node. Figure 11 illustrates the fairness of the ORMA protocol by showing the fraction of throughput allocated to each station as a function of its index (i.e., position in the folded bus). As can be seen, the fraction of throughput allocated to each station is independent of its position on the folded bus and is almost the same for various data rates.

Figure 12 illustrates the fairness of the ORMA protocol by showing the average packet delay of each station. The small variation seen in the average delay of individual stations is mainly due to the average propagation delay incurred during the transmission of a packet. This propagation delay unavoidably depends on the position of the station on the in-bound segment of the channel and is totally *independent* of the MAC protocol used (e.g., ORMA protocol). Fortunately, the variation of propagation delay can easily be bounded given the length of the network.

Comparison of the ORMA and CRMA Protocols — In this section, we compare the performance of the ORMA and CRMA protocols in terms of average packet delay and throughput. The CRMA protocol has been chosen because it has received considerable attention over the past few years, and is well understood by many researchers. Moreover, the two protocols have many things in common (e.g., explicit reservation schemes, folded bus). Figure 13 compares the average packet delay of both protocols for various network sizes and lengths. As can be seen from the figure, the average packet delay of the ORMA protocol is lower than that of the CRMA protocol under all parameters considered. This is mainly due to the fact that the frequency of the reservation process in the ORMA protocol is much higher than the frequency of the `Reserve` commands of the CRMA protocol, which are embedded in the transmission slots. Figure 14 compares the throughputs of the ORMA and CRMA protocols. Recall that the throughput is defined as the amount of user information — including message header bits, but excluding the fields used for packet reservation — successfully transmitted on the network. The throughput of the ORMA protocol is higher than that of the CRMA protocol because the size of the fields (overhead) used for packet reservation in the CRMA protocol is bigger than that in the ORMA protocol [13].



■ **Figure 14.** *Comparison of the ORMA protocol and the CRMA protocol in terms of network throughput.*

CONCLUSION

In this article, we proposed a new high-speed metropolitan and local area network protocol named ORMA. The main feature of this protocol is a simple and fast reservation scheme that is attractive for use in high-speed fiber optic networks. Unlike traditional protocols, where arithmetic processing is generally needed, which tends to slow down the transmission of reservation slots and degrade the performance of the protocol, the reservation in an ORMA protocol is performed using simple and efficient hardware circuits. We have evaluated the performance of ORMA in terms of average packet delay, throughput, and fairness using an analytic model and discrete event simulation. It is shown that the performance of the ORMA protocol is independent of the size of the network and the data transmission rate. Furthermore, the ORMA protocol is shown to be a fair protocol, unlike networks such as DQDB.

Computer simulations further indicate that the average packet delay of the ORMA is sufficiently small and can be bounded (to almost a constant). This makes ORMA useful in almost all practical real-time and multimedia applications. Moreover, it was shown that the average packet delay of the ORMA network is lower than that of state-of-the-art networks such as the CRMA network. The throughput of the ORMA was shown to increase linearly with the input data rate of the network up to its channel data rate. This is higher than that of the CRMA network since the CRMA reservation fields are larger in size. Given the performance and architecture of this protocol, ORMA seems to be an appropriate protocol for future high-speed networks operating at data rates exceeding 1 Gb/s.

REFERENCES

[1] D. J. G. Mestdagh, *Fundamentals of Multiaccess Optical Fiber Networks*, Artech House, 1995.

[2] C. Partridge, *Gigabit Networking*, Reading, MA: Addison Wesley, 1993.

[3] IEEE Std. 802.6-1990, "Distributed Queue Dual Bus (DQDB) Subnetwork of a Metropolitan Area Network (MAN)," IEEE, July 1991.

[4] B. Mukerjee *et al.*, "Dynamic Control and Accuracy of the Pi-Persistent Protocol Using Channel Feedback," *IEEE Trans. Commun.*, vol. 39, no. 6, June 1991, pp. 887–98.

[5] J. O. Limb, "Load-Controlled Scheduling of Traffic on High-Speed Metropolitan Area Networks," *IEEE Trans. Commun.*, vol. 37, Nov. 1989, pp. 1144–50.

[6] M. M. Nassehi, "CRMA: An Access Scheme for High-Speed LAN and MANs," *Proc. IEEE ICC*, 1990, pp. 115–21.

[7] M. Conti, E. Gregori, and L. Lenzini, "A Methodological Approach to an Extensive Analysis of DQDB Performance and Fairness," *IEEE JSAC*, 1991, pp. 76–87.

[8] M. Conti, E. Gregori, and L. Lenzini, "DQDB/FBS: A Fair MAC Protocol Stemming from DQDB FairnessAanalysis," *Proc. 2nd IEEE Wksp. Future Trends of Distributed Comp. Sys.*, 1990, pp. 152–59.

[9] F. Borgonovo *et al.*, "FQDB: A Fair Multisegment MAC Protocol for Dual Bus Networks," *IEEE JSAC*, 1993, pp. 1240–48.

[10] M. A. Marsan, C. Casetti, and G. Panizzardi, "On the Performance of Topologies and Access Protocols for High-Speed LANs and MANs," *Comp. Networks and ISDN Sys.*, 1994, pp. 873–93.

[11] H. R. V. As, "Media Access Techniques: The Evolution toward Terabit/s LANs and MANs," *Comp. Networks and ISDN Sys.*, 1994, pp. 603–56.

[12] P. T. Gia and R. Dittmann, "A Discrete-Time Analysis of the CRMA Protocol," *Perf. Eval.*, 1992, pp. 185–200.

[13] E. A. Zurfluh, *et al.*, "The IBM Zurich Research Laboratory's 1.13 Gb/s LAN/MAN Prototype," *Comp. Networks and ISDN Sys.*, 1993, pp. 163–83.

[14] M. Hamdi and L. Wang, "Performance of ORMA under a Wave Division Multiplexing Environment," CS-HKUST-35 Tech. Rep., 1995.

[15] C. Qiao and R. G. Melhem, "Time-Division Optical Communications in Multiprocessor Arrays," *IEEE Trans. Comps.*, May 1993, pp. 577–90.

[16] P. G. Harrison and N. M. Patel, *Performance Modelling of Communication Networks and Computer Architectures*, Reading, MA: Addison-Wesley, 1993.

BIOGRAPHY

MOUNIR HAMDI [M] received the B.Sc. degree with distinction in electrical engineering from the University of Southwestern Louisiana in 1985, and M.Sc. and Ph.D. degrees in electrical engineering from the University of Pittsburgh in 1987 and 1991, respectively. While at the University of Pittsburgh, he was a research fellow involved with various research projects on interconnection networks, high-speed communication, parallel algorithms, switching theory, and computer vision. In 1991 he joined the Computer Science Department at Hong Kong University of Science and Technology as an assistant professor. His main areas of research are high-speed networks, ATM packet switching architectures, wireless networking, and parallel computing. Dr. Hamdi has published over 60 papers on these areas in various journals and conference proceedings. He co-founded and co-chairs the International Workshop on High-Speed Network Computing, is on the editorial board of *IEEE Communications Magazine*, and has been on the program committees of various international conferences. Dr. Hamdi is a member of the ACM.