IMPLEMENTATION OF A THREE-STAGE BANYAN-BASED ARCHITECTURE WITH INPUT AND OUTPUT BUFFERS FOR LARGE FAST PACKET SWITCHES

Fabio M. Chiussi and Fouad A. Tobagi

Technical Report No. CSL-TR-93-577

June 1993

This work has been supported in part by NASA under grant NAGW-419, by NSF under grant NCR-9016032, by Pacific Bell and by the AT&T Foundation.

Implementation of a Three-Stage Banyan-Based Architecture with Input and Output Buffers for Large Fast Packet Switches

Fabio M. Chiussi and Fouad A. Tobagi

Technical Report: CSL-TR-93-577

June 1993

Computer Systems Laboratory
Departments of Electrical Engineering and Computer Science
Stanford University
Stanford, CA 943054055

Abstract

Fast packet switching, also referred to as Asynchronous Transfer Mode (ATM), has emerged as the most appropriate switching technique for future Broadband Integrated Services Digital Networks (B-ISDN). A three-stage banyan-based switch architecture with input and output buffers has been recently described [Chi93]. Such architecture, also referred to as the Memory/Space/Memory (MSM) switching fabric, is capable of meeting the challenges posed by a successful deployment of B-ISDN; namely, it is made nonblocking with low complexity, and is scalable to large sizes (>1000 input/output ports); it supports a wide diversity of traffic patterns, including highly-bursty traffic; it maintains packet sequence, is self-routing, and is simple to operate.

In this paper, we investigate implementation issues pertaining to the MSM, and demonstrate its feasibility at the sizes of interest, at both 155.52 Mb/s and 622.08 Mb/s ATM standard data rates. For this purpose, we have designed and simulated at the required speeds the critical circuit components of a 1024 x 1024 MSM using a BiCMOS sea-of-gates on a 0.8-μm technology, and studied how system synchronization can be achieved. We estimate that the largest achievable switch size using current VLSI technologies is 4096 x 4096 at the 155.52 Mb/s data rate, and 1024 x 1024 at the 622.08 Mb/s data rate. Finally, we compare the complexity of the MSM switch with that of other fast packer switching architectures, and show that, due to its low chip count and simpler synchronization requirements, the MSM may be built in larger sizes than other architectures.

Key Words and Phrases: fast packet switching, broadband integrated services digital networks, asynchronous transfer mode, high-speed switching systems, banyan interconnection networks, VLSI design, BiCMOS, BiNMOS, Sea-of-Gates.

1. Introduction

Fast packet switching, also referred to as Asynchronous Transfer Mode (ATM), has emerged as the most appropriate switching technique to handle the high data rates and the wide diversity of traffic requirements envisioned in Broadband Integrated Services Digital Networks (B-ISDN). The ATM standard specifies 53-byte fixed-size packets (referred to as cells) comprising 48 bytes of payload and 5 bytes of control information, and line speeds of 155.52 Mb/s and 622.08 Mb/s [CCI90].

ATM switches capable of meeting the challenges posed by a successful deployment of B-ISDN must be designed and implemented. Such switches should be nonblocking and capable of handling the highly-bursty traffic conditions that future anticipated applications will generate; they should maintain packets belonging to each call in their original order; they should be scalable to the large sizes expected when B-ISDN becomes widely deployed; accordingly, their complexity should be as low as possible; finally, they should be simple to operate; namely, their architecture should facilitate the determination of whether or not a call can be accepted, and the assignment of a route to a call [Chi93].

Several basic architectural designs of ATM switches have been proposed, and many among them fully prototyped [JSA91]. They typically fall into three categories; namely ***shared-memory*** [Kuw89,Koz91 ,Sho91], ***shared-medium*** [Suz89], and *space-**division*** [Hua84,Tur88,Tob91], each having its features and shortcomings [Tob90a]. Regardless of the particular architectural design used, however, given an implementation technology and its constraints, the size of a switch is bound to be limited. (For the prototypes so far realized, the largest-size switch has been 32 **x** 32 or 64 **x** 64 depending on the architecture used.) In order to build larger sizes, several switching modules (of the same or of different types) are combined in a multi-module configuration, which consists of a row of output switches, each handling a group of output lines (e.g., 32 or 64 lines), and

a routing network which routes packets from the input lines to the appropriate output switching module. As with the basic switch architectures, several multi-module switch architectures have been conceived [Suz89,Koz91,Sho91,Ban91,Eng89,Wan92].

A three-stage banyan-based architecture with input and output buffers, which meets the challenges of B-ISDN, has been recently described [Chi92,Chi93]. This architecture, also referred to as the Memory/Space/Memory (MSM) switching fabric, is made nonblocking with low complexity, and is scalable to large sizes; it supports a wide diversity of traffic patterns, including highly-bursty traffic; it maintains packet sequence, is self-routing, and is simple to operate. The MSM features a routing fabric that uses as a building block the banyan routing network (which is self-routing and amenable for VLSI implementation in sizes up to 64 x 64 on a single chip [Mar90,Tob91]); this routing fabric consists of multiple banyan routing networks placed in parallel, with input controllers that dispatch the packets to the banyans in such a way as to achieve efficient utilization of the banyans'. By providing a small amount of buffering in the input components, the number of banyans needed can be reduced to the minimum possible. Packet sequence is maintained by using first-come-first-served service discipline in dispatching packets. The arrangement of the banyans in parallel results in an architecture which allows input and output components to be shared by many lines, and the number of banyans to be increased to that needed to accommodate the traffic, thus increasing the overall size of the switch beyond the achievable size of the banyan networks.

In this paper, we discuss the implementation of the MSM. The MSM offers important advantages at the implementation level in terms of achievable sizes and data rates, owing to:

---

1 **It is important to note that the use of banyans in parallel with input and output buffers has originally been proposed in [New88b] in the context of unslotted switching systems, and then described in [Sar91] for a slotted switch; a complete characterization of this configuration in terms of its performance under various traffic conditions, including highly-bursty traffic, and a study of its scalability issues has been presented in [Chi93].**

i) a two-phase operation of the banyans (a routing phase and a data-transfer phase), which allows to use different clock rates in each phase to meet different circuit requirements, thus making it possible to sustain high internal data rates (622.08 Mb/s instead of only 155.52 Mb/s) with currently available technologies.

ii) easier synchronization of the various components (banyans, input controllers, output buffers), which allows to build a large switching fabric (> 1000 input/ output ports), and operate it at 622.08 Mb/s.

The objective of our implementation work on the MSM switching fabric is threefold.

i) Study implementation issues in the MSM, and demonstrate the feasibility of the switch at both 155.52 Mb/s and 622.08 Mb/s ATM data rates. For this purpose, we have designed and simulated at the required speeds the critical circuit components of a 1024 x 1024 MSM using a BiCMOS (BiNMOS) sea-of-gates [ElG89] on a 0.8-μm technology [ElD89,ElD90], and studied how system synchronization can be achieved.

ii) Estimate the largest achievable switch size using current VLSI technologies. Based on our implementation experience and on the analysis of existing realizations of circuitries performing functionalities similar to those needed in the MSM, we estimate that, by using full-custom design and state-of-the-art VLSI technologies, an MSM of size 8192 x 8192 is feasible at the 155.52 Mb/s data rate, and an MSM of size 1024 x 1024 is the largest feasible switch at 622.08 Mb/s.

iii) Compare the complexity of the MSM switch with that of other architectures. Specifically, we show that, for a given size, the chip count of the MSM is significantly lower than that of multistage configurations of nonblocking modules. We also show that, due to its low chip count, simpler synchronization

requirements, and self-routing operation, the MSM may be built in larger sizes than other architectures.

This paper is organized as follows. In Section 2, we describe the architecture and operation of the Memory/Space/Memory switching fabric. In Section 3, we discuss its implementation issues, describe our design, and estimate the achievable switch size with current VLSI technologies; we then compare the complexity of the MSM with that of other architectures.

2. The Memory/Space/Memory Switching Fabric

2.1. Basic Configuration

a) Architecture

In its basic configuration, shown in Fig. 1, the Memory/Space/Memory (MSM) switching fabric consists of K banyans placed in parallel, along with input controllers with buffer capabilities, one for each of the N inputs, which dispatch the incoming packets to the banyans, and output buffers connected to the corresponding outputs of each banyan. In order to minimize the number of banyans necessary to achieve a desired packet loss rate, the input controllers must dispatch the packets to the banyans so as to maximize the utilization of the banyan networks. This can be achieved by dispatching the packets to the banyans in sequence (referred to ***as sequential dispatching*** in the following). The operation of the switch is assumed to be slotted. In every time slot, each input controller which has packet(s) available in its corresponding input buffer dispatches a packet to the first banyan in the sequence; packets are then routed by the banyan; successful packets proceed to the output buffers, and unsuccessful packets are dispatched to the second banyan. The operation proceeds similarly for all the banyans; packets that are still unsuccessful after the last banyan has been attempted are stored in the input buffers for further processing in the next slot. By organizing the input buffers as FIFO's, the sequence of the packets is maintained.

For given switch size and traffic characteristics, the number of packets lost in the input buffers is a function of the number of banyan networks K and the size (in packets) of the input buffers B_I. The packet loss rate in the MSM with sequential dispatching, under uniform traffic at full load, is shown in Fig. 2, for $N = 32$. A minimum of 3 banyans in parallel is necessary to achieve low packet loss rates ($< 10^{-6}$) with small input buffer

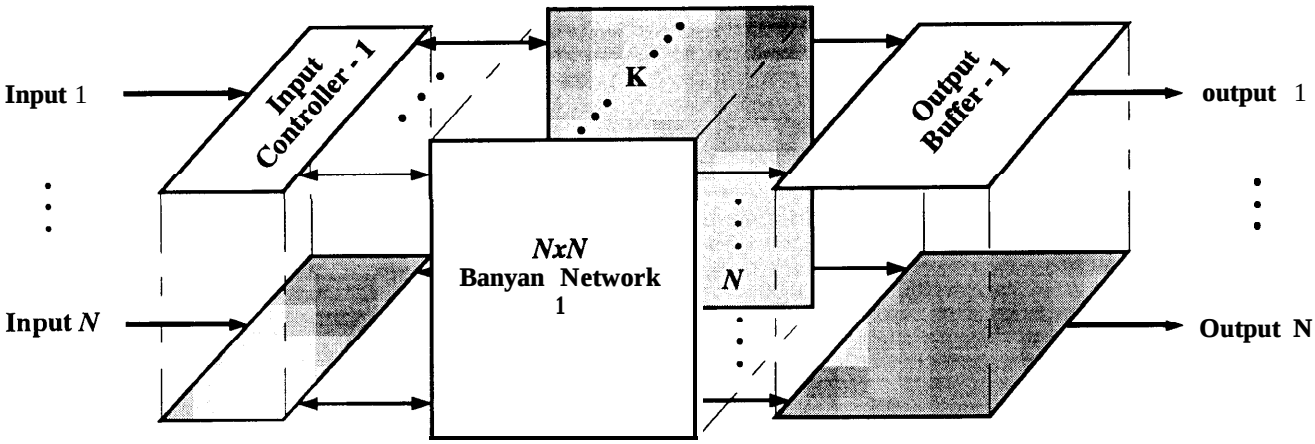


Fig. 1. The MSM switching fabric: basic configuration.

sizes, under uniform traffic at full load. This minimum of 3 networks in parallel is easily anticipated, since the throughput of a single banyan network of size N = 32 at full load is 0.4, making the total capacity with three networks about 1.2. The 20% extra capacity available explains the relatively small buffer sizes needed. (More detailed simulation results are reported in [Chi93].)

b) Switch Operation

We now describe in more detail the operation of the banyans. In our discussion, we consider the switch operation to be synchronous. All incoming lines have the same transmission capacity, packets have fixed size L (namely, we consider 53-byte ATM packets), and the time axis is slotted with slot size T equal to the transmission time of a packet on a line. In general, however, incoming packets from different input lines are not synchronized with each other, nor with the internal system clock. We begin here by assuming that packets on different input lines have been synchronized and aligned with the system slot boundaries. In Section 3.4 below, we discuss how this synchronization

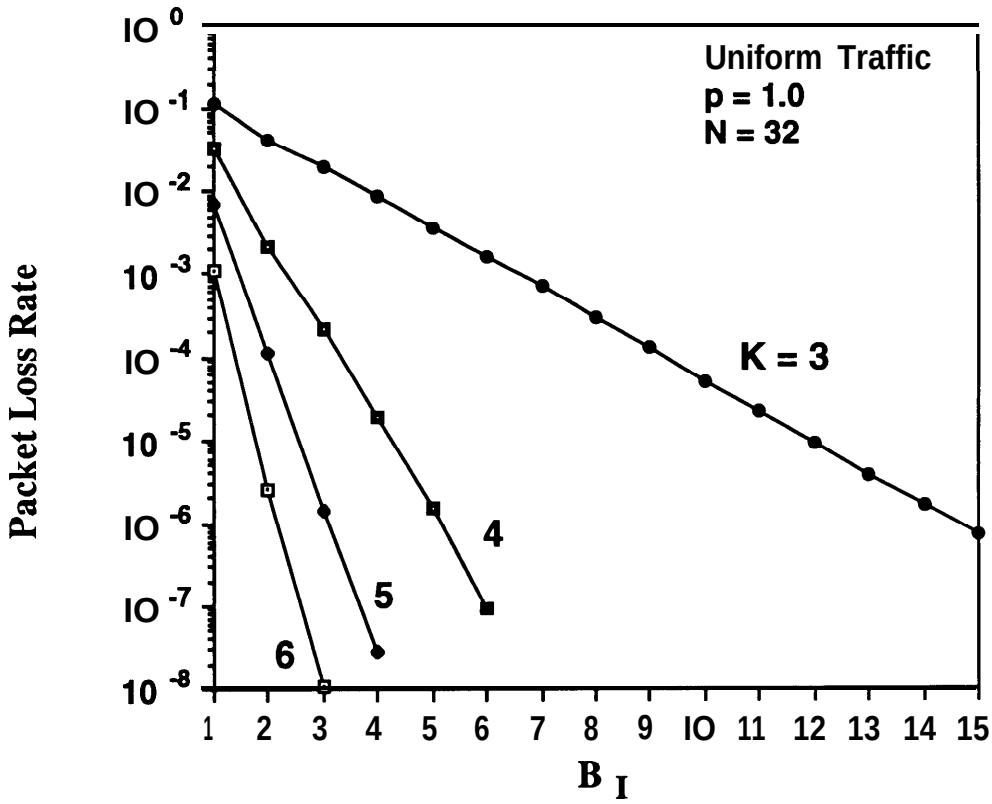


Fig. 2. Packet loss rate in the input buffers of the MSM switching fabric with input buffers and sequential dispatching, under uniform traffic at full load, A4 = 1, N = 32 (obtained by simulation).

may be achieved within the controllers and the banyan networks, thus obviating the need for external aligners.

As in any ATM switch [JSA91], for each packet, prior to entering the MSM switching fabric, header-conversion circuits generate the local switching header used by the switching fabric, based on the information in the ATM header. The local header in the MSM comprises three fields:

- Activity Bit ***a*** : it indicates whether the current slot contains a packet ***(a*** = 1), or not ***(a = 0).***

- Priority Field P : this field is optional and is used only if multiple priority classes exist. It comprises $q = \log Q$ bits, where Q is the number of external priority levels.
- Address Field D: This field contains the address of the output port $d_1, d_2, \ldots . d_n$ (where $n = \log N$, and d_1 is the most significant bit); it is inferred from the virtual circuit information in the ATM header.

Once the local header is generated, packets are sent to the switching fabric, stored in the corresponding input buffers, and then dispatched to the banyans. To perform the dispatching algorithm, in every slot the operation of a banyan is divided in two phases:

1) ***Route Set-up Phase.*** During this phase, the local switching header portions of the packets (referred in the following simply as ***headers)*** are routed through the banyan, establishing the paths in the interconnection network. At any stage s in the banyan, the proper setting of a switching element is a function of two bits in each header present at its inputs, namely a and d_s (if external-priority service is in effect, then the setting is also a function of the priority fields in the headers). An active header (identified by $a = 1$) arriving at a switching element requests to be routed to the upper output if $d_s = 0$, and to the lower output if $d_s = 1$. Anytime there is a conflict between two active headers at a switching element ***(i.e.,*** d_s-bits are equal for the two headers), one of the two headers is routed properly, while the other is forced in the "wrong" direction and has its activity bit reset (clearly, this conflict resolution follows a similar idea to that used in the TBSF). The selection of the "winner" can be random, but more practical deterministic selection algorithms can be adopted with no effect on performance. A non-active header (identified by $a = 0$), ***i.e.,*** either a misrouted header or an empty slot, is routed accordingly to the current state of the switching element (determined by the other header, if active, or by the state of the switching element in the previous slot).

Once the headers reach the outputs of the banyan network, the outcome of the route set-up phase ***(i.e.,*** which headers have been properly routed by the network) is described by the activity bit of the headers: headers with $a = 1$ at the output of the network have been successfully routed; headers with $a = 0$ have been misrouted (or represent empty slots). The outcome of the route set-up phase is fed back to each controller at the input by having the switching elements in the last stage of the banyan network transmit backward the activity bit of each header, all the way to the input controllers, through the paths that have just been set-up.

2) Packet Transmission Phase. Each input controller examines the feedback signal and determines whether the header has been routed properly or not. If the header has been successful (feedback signal = 1**),** the controller transmits the packet ***(i.e.,*** the ATM header and payload) to the output buffers.

With sequential dispatching, at the end of the route set-up phase, if an input controller determines that the header currently in process has been properly routed by a banyan, it transmits the corresponding packet to that banyan, and dispatches the next header in the input queue to the remaining banyans in the sequence. If a header has been unsuccessful in a banyan, it is simply dispatched to the following one. Packets which are unsuccessful after the last banyan has been attempted, remain in the input buffers and are processed in the following time slot. Up to K packets can be routed from each input to the outputs in a time slot.

The two-phase operation brings important benefits at the implementation level. They are discussed in detail in Section 3.

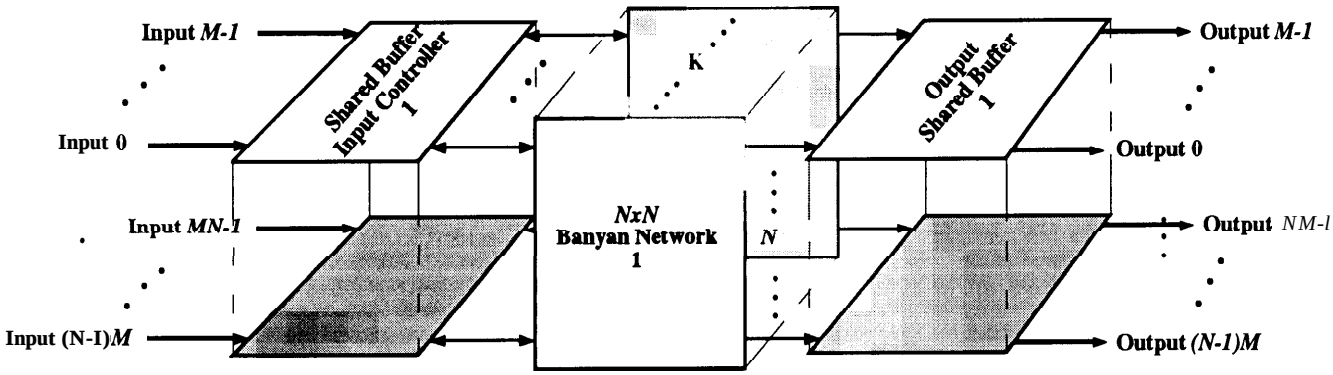


Fig. 3. The MSM switching fabric for large fast packet switches.

2.2. Large Size MSM Switching Modules

a) Architecture

In order to increase the size of the MSM switch beyond the achievable size of the banyan network, we observe that, given their relatively simple functionality and small buffer requirements, the input components actually have the capacity to handle more traffic than that coming from one input line and to control a number of banyans larger than that needed to sustain the traffic from a single line; similarly, the output components have the capacity to handle more traffic than that destined to a single output line. Therefore input and output components can be shared by several (say M) inputs and outputs, respectively, making the total size of the switching fabric equal to $M \cdot$ N. The resulting configuration is shown in Fig. 3. The switch operates as follows: in each input controller, packets arriving at its M input lines are first multiplexed into a shared buffer; packets are then sequentially dispatched to the banyans and routed to their requested output shared buffers; finally, in each output component, packets are switched to their requested output links selected among the M output ports served by the buffer. In this case, the first log, N bits of the address field D in the local switching header are used for routing the packets in

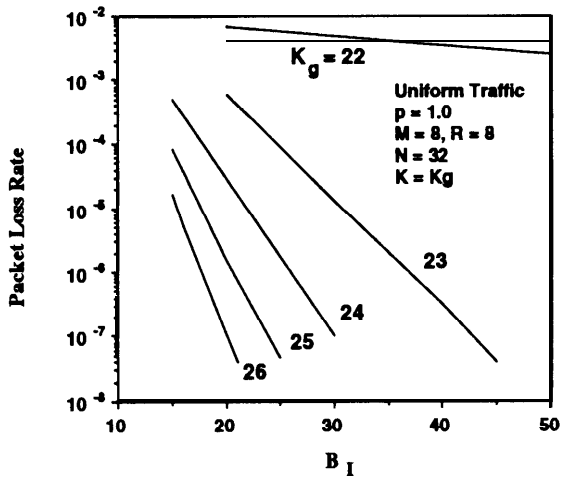


Fig. 4. Packet loss rate in the input buffers of a 256 **x** 256 MSM switch, $M = 8$, $N = 32$, under uniform traffic conditions, $p = 1$ (obtained by simulation).

the banyan networks, and the remaining log, M bits in the address field specify the desired output port among the M ports served by the corresponding output buffer "switch".

Since the routing capacity offered by the banyans in parallel must be adequate to sustain the aggregate traffic from the M links entering each input controller, the required K to maintain a given packet loss rate in the input buffers increases with M. As a rule of thumb, for $N = 32$ and $N = 64$, 3 banyans for each of the input ports sharing an input controller are needed to achieve very low packet loss rates in the input buffers with relatively small sizes of the input shared buffers B_I, under uniform traffic at full load (again, this rule of thumb is easily anticipated, since the throughput of a banyan network with $N = 32$ and $N = 64$ at full load is 0.40 and 0.36, respectively, making the total capacity with three networks larger than 1, and the total capacity with $3M$ networks larger than M). For example, the packet loss rate in the input buffers of a 256 **x** 256 switch with $M = 8$ and $N = 32$, under uniform traffic at full load, is depicted in Fig. 4. With $K = 24$ banyans, $B_I = 30$ packet buffers are needed to achieve a packet loss probability equal to 10^{-7}. If more banyans are used, smaller buffer sizes suffice for the same packet loss rate.

In the MSM switching fabric, the number of input lines that can be fed to an input controller and the number of output lines that can be served by an output shared buffer are limited by four factors:

i) the bandwidth requirements of input and output memories, respectively;

ii) the functionality to be performed in the input controller and in the output buffer, respectively;

iii) the buffer requirements B_I and B_O, respectively, necessary to guarantee a desired packet loss rate in the fabric; and

iv) the number of dispatching operations to different banyans that can be completed within a slot duration.

Here, we briefly discuss the nature and extent of these limitations in the input and output components.

The memory bandwidth in the input components must be adequate to sustain a flow of M packets into the memory and K packets from the memory per slot; the memory bandwidth is therefore equal to $(M + K)V$, where V is the line speed. However, we have show in [Chi93] that it can be reduced to $(2M$ + l)V with no noticeable degradation in performance. The memory bandwidth in the output components is equal to $(M + K)V$.

In the input components, with sequential dispatching and related FIFO operation, the shared memory is a relatively simple sequentially-accessed memory. The memory control must be capable of synchronizing and multiplexing M incoming packets into the memory, of retrieving up to K packets from the memory per slot (again, it has been shown in [Chi93] that the maximum number of retrieved packets per slot can be limited to M + 1 with no observable performance degradation), and of delivering them to the corresponding banyans in which paths have been set up. The functionality of the input controllers has to

include the implementation of the sequential dispatching algorithm to orchestrate the operation of the banyan networks. Given the simplicity of the algorithm, however, this limiting factor is not as important as synchronization, bandwidth and memory access requirements.

In contrast, the output buffer is a shared-memory switch serving M ports, and therefore is a randomly-accessed memory in which M linked lists have to be managed. The memory control logic has to synchronize and enqueue in the requested queues up to K incoming packets, and retrieve from the memory and transmit to the proper output ports M outgoing packets per slot.

The buffer requirements are fairly modest in the input controllers under any traffic condition of interest, as shown in more detail in the following section. On the contrary, since the MSM achieves output buffering, if the switch is designed to sustain bursty traffic, the local buffer requirements in the output components are large, thus becoming an important concern.

Finally, the objective of maintaining packet sequence poses an additional limitation on the number of lines served by input and output components, since the sequential scanning of all the K banyans must be completed within a slot duration T in order to maintain FIFO operation of the input buffers. Therefore, the maximum number of banyans that can be placed in parallel is:

$$K_{max} = T / T_s = \frac{T_s + T_t}{T_s} \qquad (1)$$

where T_s is the duration of the set-up phase and T_t is the duration of the transmission phase in a banyan. Thus, the limitation on K_{max} comes from the achievable clock rate during the set-up phase.

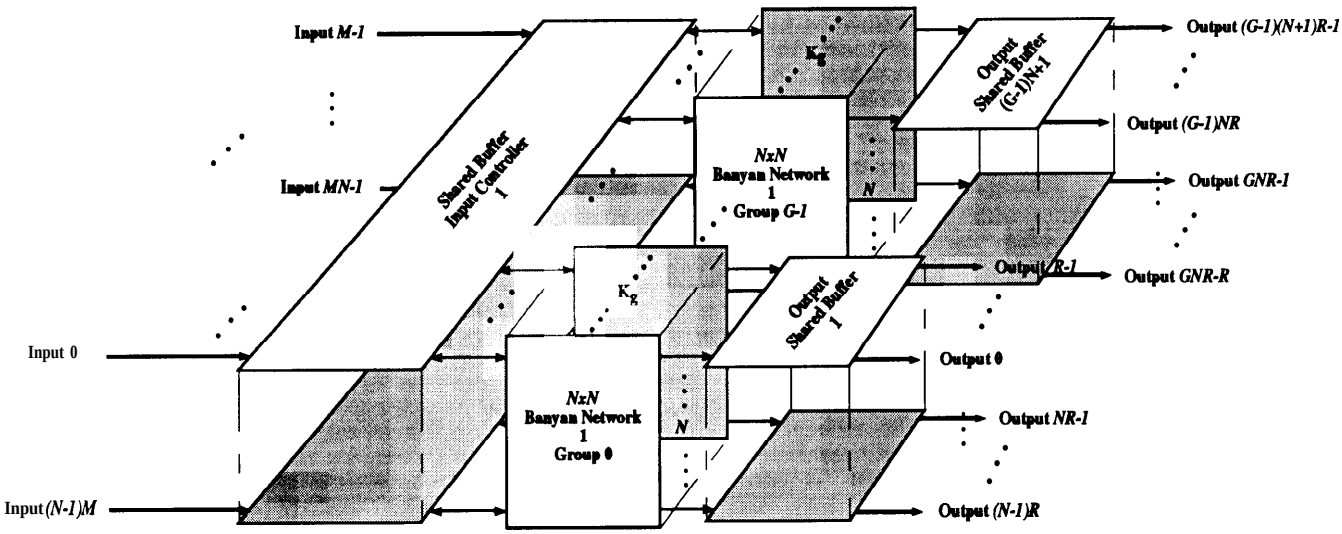


Fig. 5. The MSM switching fabric: general configuration, $M/R = G > 1$.

This discussion suggests that, as far as the limitations on bandwidth, functionality and buffer requirements are concerned, the achievable number of input lines M in a controller may be larger than the achievable number of output lines R in an output buffer. In the configuration of Fig. 3, however, for a given N, the more stringent limitations on R would determine the achievable switch size; furthermore, in this configuration, we note that, since K is a function of M, the limitation on K imposed by the requirement to complete the dispatching to all the banyans within a time slot directly translates into a limitation on M (and on R).

To further increase the maximum achievable switch size, we have therefore proposed the general configuration for the MSM, shown in Fig. 5. In this configuration, each input controller receives M input lines and is connected to G groups of K_g banyans each (the total number of banyans is still defined as $K = GK_g$). In each group, each of the corresponding output buffers has R output ports, where $R = M/G$. With $G > 1$, the input controller first selects the desired output group to which each of the packet is destined

(specified by the first log, G bits in the address field of the local switching header) and enqueues the packet in the corresponding input queue. Then the operation proceeds as described above, by applying the dispatching algorithm to each of the G groups of K_g banyans. Packets are routed to their requested output shared buffer, and then switched to the requested output port selected among the R ports served by the buffer. In this general case, as a rule of thumb, in each group, 3 banyans for each of the R output ports connected to an output shared buffer are necessary to achieve low packet loss rates in the input buffers with small buffer sizes. In general, for increasing values of G, with constant total number of banyans K, the required input buffer size to achieve a desired packet loss rate increases slightly; alternatively, more banyans have to be used. For example, the packet loss rate in the input buffers of the 256 x 256 switch with M = 8 and N = 32, described above for R = 1, is depicted in Fig. 6 for different R. With $R = 4$, for $K = 2K_g$ = 24 banyans (*i.e.*, with two groups of K_g = 12 banyans each), 44 packet buffers are needed to guarantee a 10^{-7} packet loss rate (rather than 30 buffers in the case with R = 1). (More detailed simulation results are reported in [Chi93].)

In the general configuration, the bandwidth of the output shared memories is reduced to $(K_g + R)V$. The output components have only R output ports, thus reducing the complexity of the memory management and the local buffer requirements in each component. In the input controllers, the shared memory accommodating the G input queues has still bandwidth $(M + K)V$ (it can be reduced to $(2M + G)V$ with no observable degradation in performance [Chi93]). If the bandwidth is not manageable, completely partitioned memories for each group can be used, each with bandwidth $(M + K_g)V$ (which can be limited to $(M + R + 1)V$); it is shown in [Chi93] that, by using partitioned memories, the increase in total required input buffer size is not significant. Also the limitation on the number of banyans in parallel coming from the requirement to complete the scanning of all the banyans within a time slot is significantly ameliorated since, given

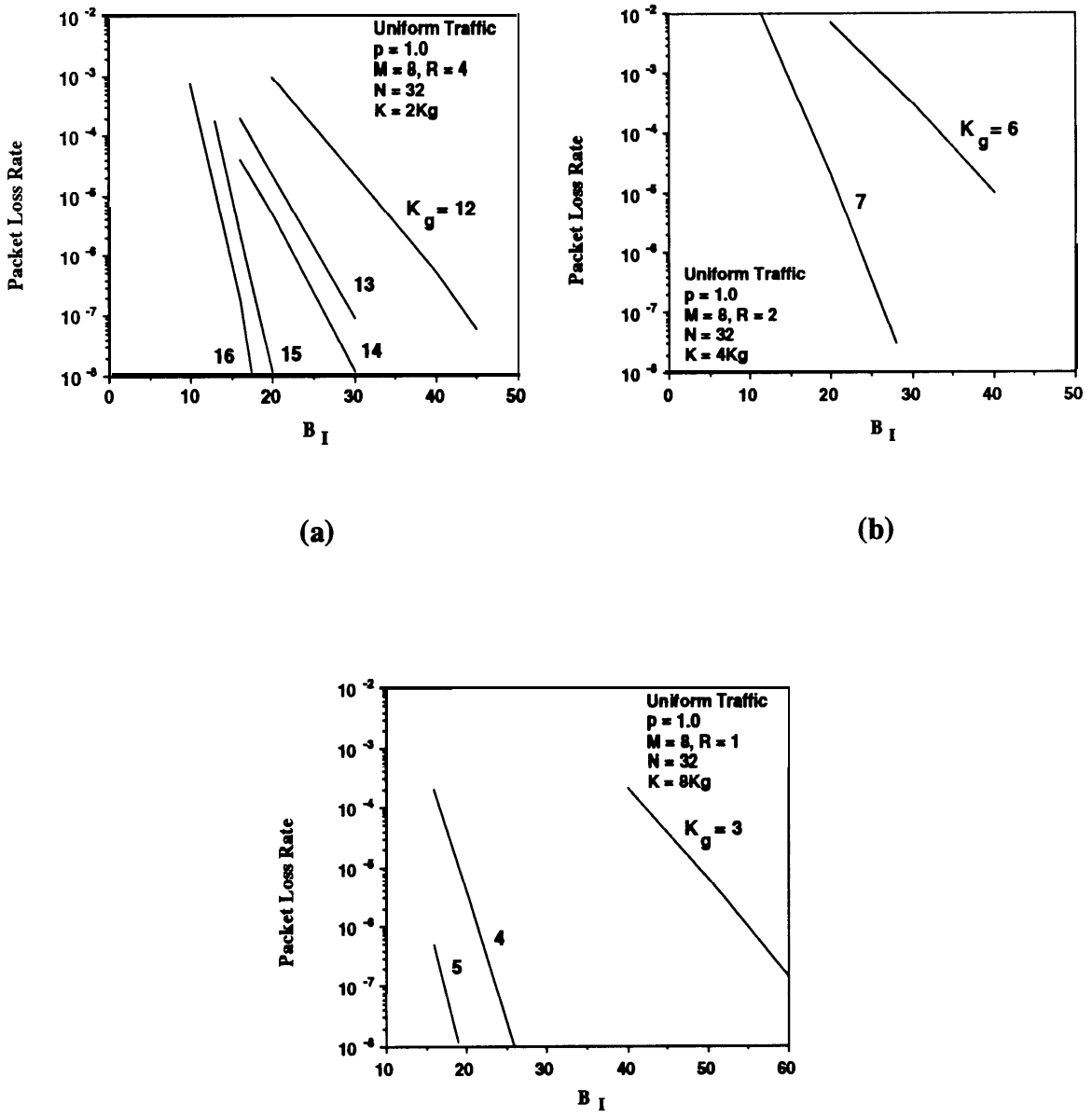


Fig. 6. Packet loss rate in the input buffers of a 256 x 256 MSM switch, G > 1, under uniform traffic conditions, $p = 1, N = 32$; a) $M = 8, R = 4$; b) $M = 8, R = 2$; c) $M = 8$, $R = 1$ (obtained by simulation).

that the dispatching algorithm is performed on each group in parallel, only K_g set-up phases (as opposed to K set-up phases as in the previous configuration) have to be completed in a time slot. With this configuration, therefore, we can individually adjust the design parameters M, R, N, G, K_g, B_I and B_O to accommodate the technology constraints and achieve the desired size and performance.

b) Correlated Traffic Patterns

Since the throughput of a banyan network is very sensitive to the input traffic pattern, one possible cause of concern in the MSM switching fabric is the fact that the packets, during their sequential scanning of the banyans, enter every banyan in the sequence from the same input. Under traffic patterns exhibiting correlation among packets, either in the space domain (e.g., ***communities-of-interest*** traffic pattern [Tur88,Tob91,Chi93]) or in the time domain ***(e.g., bursty*** traffic pattern), this may lead to recurrent conflicts between the same packets maintained over consecutive banyans and/or several slots, and cause degradation in throughput. Consequently, more banyans may have to be provided to achieve low packet loss rates [Chi93].

Providing more banyan, although possible, is not desirable. In fact, as mentioned above, there is a limit on the number of banyans that can be used in parallel with FCFS operation of the input buffers ***(i.e.,*** maintaining the sequence of the packets); thus, a larger number of banyans translates into a stricter limitation on ***M,*** and consequently on the switch size (of course, providing more banyans also increases the required amount of resources and complicates the control of the banyans).

An effective solution to overcome the throughput degradation under correlated traffic patterns is to randomize the input patterns of different banyans, so as to reduce the likelihood of persistent "bad patterns" and recurrent conflicts. One possibility is to provide a ***full randomization network*** in front of each banyan, which provides randomization over

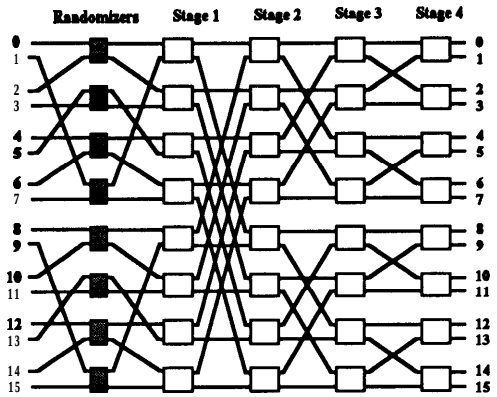


Fig. '7. Example of a banyan networks augmented with randomizers by pairs, N = 16.

all inputs of the banyan [Tur88]. The complexity of the resulting concatenation of the randomization network and the banyan network is twice the complexity of the banyan network [Chi93]. In particular, the number of stages is doubled; consequently, in the MSM, the duration of the route set-up phase T_s in each routing network is also doubled. Since the limitation on the maximum number of banyan networks that can be placed in parallel is $K_{max} = T / T_s$, doubling T_s halves K_{max}, and again translates into a stricter limitation on M.

In [Chi93], we show that most of the benefits of a full randomization network are obtained with minimum additional hardware complexity by using a ***single-stage*** randomization network which randomizes packets only between properly selected pairs of inputs. Since such a network consists of a single stage, the impact on T_s (and consequently on K_{max}) resulting from its addition is minimal. Furthermore, we note that in practice, given the typical aspect ratios of VLSI designs of banyan networks, such additional stage can be usually fitted rather easily on the same chip of the banyan network (thus, the additional hardware complexity resulting from the use of a single-stage randomization network is negligible). A procedure to construct the single-stage randomization network (also referred to as ***randomizers*** by ***pairs)*** is presented in [Chi93]. An example of the configuration resulting from the addition of the randomizers by pairs

in front of a banyan network is shown in Fig. 7, for N = 16. Under correlated traffic patterns, the addition of randomizers by pairs in front of the banyans effectively breaks the correlation among packets by distributing the traffic in the networks, and eliminates throughput degradation [Chi93]. From a practical point of view, the single-stage randomization network can be simply implemented by adding, in front of each banyan network, an extra stage of switching elements with the activity-bit-resetting capability disabled, and by providing an additional randomly-generated bit in the address field of the headers to control them.

c) The MSM Output Buffers

Packet loss in the MSM occurs also in the output buffers. For a given switch size, the number of packets lost in the output buffers is a function of the output buffer size B_O (in packets) and the traffic pattern. The output-buffer requirements are also highly dependent on the degree of memory sharing in the buffers.

The output-buffer requirements in an output buffer switch under uniform traffic conditions have been extensively studied by several authors [Hlu87,Hlu88,Eck88], for both completely-partitioned and shared memories. Under these traffic conditions, the required buffer sizes are relatively small. For example, at loads as high as 0.95, 110 packet buffers per port are sufficient to guarantee packet loss rates lower than 10^{-6}. Assuming 53-byte ATM cells, this translates in memory sizes of less than 50 Kbit. The buffer requirements decrease rapidly as the load decreases. At 0.9 load, 5 1 packet buffers are ***necessary*** to achieve 10^{-6} loss rate; at 0.8 load, 28 buffers are required for the same loss rate.

With $R > 1$, each output buffer is shared among R ports. The required buffer size per port decreases significantly as the memory is shared among more and more ports [Hlu87,Hlu88]. For example, the buffer requirements per output port in a 256×256

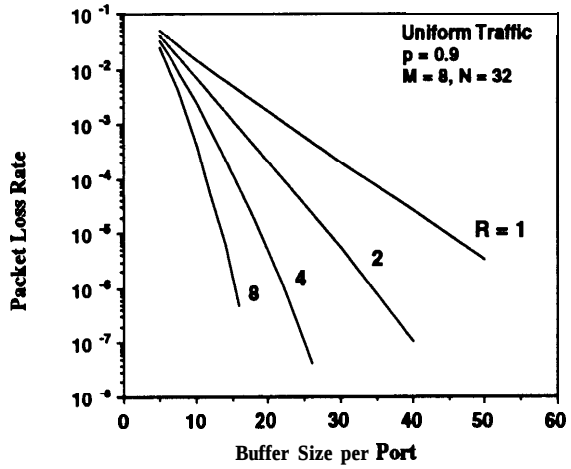
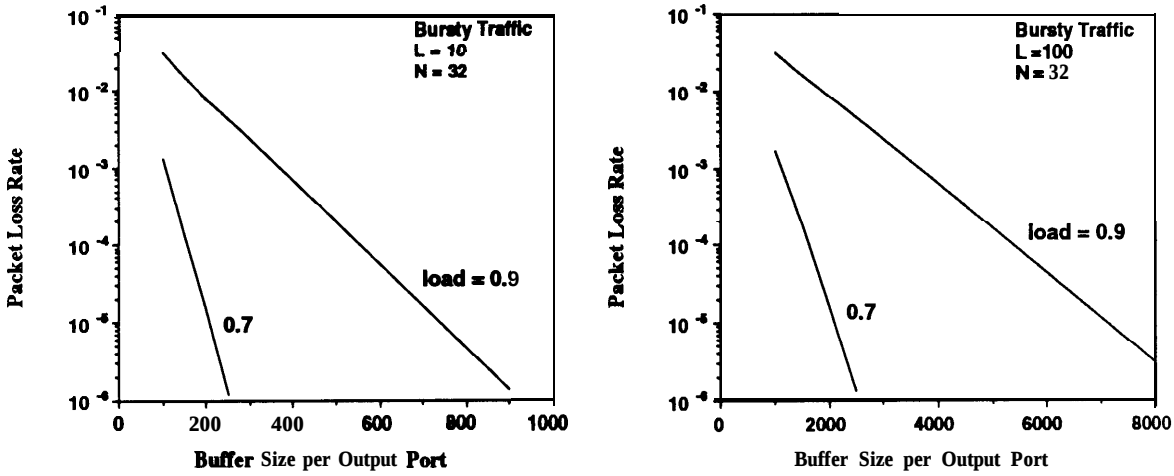


Fig. 8. Packet loss rate in the output buffers of a 256 x 256 MSM, M = 8, under uniform traffic conditions, for different R, p = 0.9 (obtained by simulation).

MSM with M = 8 and N = 32, obtained by simulation, are shown in Fig. 8 for R ranging from 1 to 8, at load p = 0.9; 56 packet buffers are necessary to achieve a packet loss rate below 10^{-6} with completely-partitioned buffers (R = 1); a substantial reduction in buffer size per port is attained for R = 2; in this case, 35 buffers per port suffice for the same packet loss rate. The reduction continues for larger R, albeit it is progressively of a lesser extent; for R = 4, 23 packet buffers per port, and for R = 8, 15 buffers per port are required for a loss rate below 10". In general, in each output buffer component, the buffer requirements to accommodate uniform traffic are fairly small, for both completely-partitioned and shared buffers. Assuming ATM cells, with R = 1, a memory of less than 25 Kbit is required in each single-port output buffer to sustain a 0.9 load; for R = 8, a total memory of 50 Kbit in each 8-port output buffer component is necessary at the same load.

In a switch achieving output buffering, under bursty traffic, the size of the output buffers required to guarantee a desired packet loss rate is substantially larger than that needed under uniform traffic conditions [End90,Lie90,Chi93]. For example, considering a bursty traffic model in which each input generates geometrically distributed bursts of average length L (within a burst, packets destined to the same output arrive continuously



(a) (b)

Fig. 9. Packet loss rate in the output buffers of an MSM, under bursty traffic, for 0.9 and 0.7 load, N = 32; a) L = 10; b) L = 100 (obtained by simulation).

in consecutive time slot), with the output requests of the bursts uniformly distributed over all destinations, the packet loss rate in the output buffers of a 32 x 32 MSM with $M = R =$ 1, at 0.9 and 0.7 load, obtained by simulation, is shown in Fig. 9.(a) and Fig. 9.(b), for average burst lengths L = 10 and L = 100, respectively. At 0.9 load, for L = 10, more than 920 packet buffers per port are necessary to achieve a loss rate below 10^{-6} ; the increase in buffer size is quite dramatic respect to the 51 packet buffers required under uniform traffic conditions at the same load. The required buffer size for a desired packet loss rate increases approximately linearly with the average burst size. For L = 100, 9000 buffers are necessary for a 10" loss rate. Also under bursty traffic conditions, memory sharing substantially decreases the required output buffer size per port to meet a given packet loss rate. For example, in a 128 $\times$128 MSM with M = R = 4, with L = 10, at 0.9 load, 440 packet buffers per port suffice for 10^{-7} loss rate [Chi93]. The increase in buffer requirements with the burst size is still approximately linear; with L = 100, 4600 buffers per port are necessary for the same packet loss rate. In each output-buffer component, the local buffer requirements to sustain bursty traffic conditions are clearly quite large. With

ATM cells, in the 32 x 32 switch, the required memory size in each single-port output-buffer component is approximately 4 Mbit, to accommodate bursts with average size of 100 packets at 0.9 load. In the 128 x 128 switch, each four-port output component requires a memory of about 8 Mbit to sustain the same traffic.

3. Implementation of the Memory/Space/Memory Switching Fabric

As described in the previous section, the MSM switching fabric consists of several instances of three basic components: the banyan-network component, the input-controller component, and the output-buffer component. In the following, for each of these components, we:

i) describe its functionality;

ii) identify implementation objectives and issues pertaining to the component;

iii) describe our implementation;

iv) estimate the achievable size of the component with current VLSI technologies.

We then address how such components can be synchronized together in order to build the complete switching system, and describe the circuitry that must be added to each component in order to make system synchronization possible.

3.1. The Banyan Network Component

3.1.1. Functionality

The functionality and operation of the banyan network has been described in Section 2.1 above. We recall that, in every slot, the operation is divided in two phases:

i) a route set-up phase, in which the header portions of the packets are routed through the banyan, establishing the paths in the interconnection network; at the end of this phase, a feedback signal consisting of the activity bit of the headers once they reach the outputs of the network is sent back to each controller at the input; and

ii) a transmission phase, in which the packets corresponding to successful headers are transmitted to the output buffers through the paths that have been set-up.

It is important to note that in the MSM the operation of the banyan during the route set-up phase is fully synchronous; namely, in order to perform the switching algorithm, all headers are bit-synchronized and aligned ***(i.e.,*** their starting times are synchronized) with one another at the input of the network, and their synchronization and alignment maintained at each stage in the network. In contrast, during the transmission phase, the paths have already been set-up, bit streams can flow "freely" to the outputs, and therefore alignment of the packets is not necessary. (Internally to the network, even bit-synchronization of the streams is not required.) However, synchronization between connected pairs of inputs and outputs in the banyan must be maintained.

From an implementation point of view, the two-phase operation of the banyans brings two important benefits at the chip-design level.

i) The two-phase operation implies that different clock rates may be used in each phase. During the route set-up phase, the logic in charge of the switching algorithm and the requirement of keeping the header synchronized and aligned impose the critical constraints on the clock rate. On the contrary, during the packet transmission phase, no logic is involved and synchronization is looser. This means that the hardware constraints during different phases may differ substantially, and the maximum achievable clock rate in the route set-up phase may be lower than the maximum achievable clock rate in the transmission phase (for example, circuit simulations show that, using state-of-the-art CMOS or BiCMOS technology [ElD89], the sustainable clock rate in the route set-up phase may be four times lower than the clock rate in the transmission phase). Using different clock rates is useful in two respects: a) it allows to support the 622.08 Mb/s data rate; in fact, even with advanced BiCMOS technology it would not be possible to achieve a 622 MHz clock rate during the route set-up

phase (while clock rates as high as 800 MHz can be achieved in the transmission phase); and b) at lower data rates ***(i.e.,*** 155.52 Mb/s) it allows to simplify the switching element and reduce its area; in fact, by using a lower clock rate in the route set-up phase, the synchronization of clock and data signals both within the switching elements and from stage to stage is simpler, and can be achieved just by matching the signals' delays, (rather than by using circuitry to preserve synchronization), thus translating into a smaller design.

ii) With the two-phase operation, the state of the switching elements is set prior to the transmission of the packets. Thus, within each switching element, the path followed by the packets does not include the storage elements occupied by the control bits needed to perform the switching algorithm. This reduces considerably dynamic power dissipation internally to the chip.

Finally, we recall that randomizer by pairs (consisting of a single stage of switching elements whose state is randomly set) are added (on the same chip) in front of the banyan networks. Therefore, the routing network ***(i.e.,*** randomizers plus banyan network) comprises three different types of switching elements: a) switching elements to be used in all but the last stage of the banyan network; b) switching elements to be used in the last stage of the banyan network, capable of generating and transmitting back the feedback signal; and c) switching elements to be used in the single-stage randomizers. In the following, we show that all these switching elements can actually be obtained by simple variations of a single basic design.

3.1.2. Implementation Objectives and Issues

The objectives in the implementation of the banyan network component are four:

i) to implement a routing network of the largest size possible on a single chip;

ii) to support the desired operating data rates (155.52 Mb/s and 622.08 Mb/s);

iii) to limit power dissipation; and

iv) to perform the switching algorithm during the route set-up phase incurring a small latency per switching element.

The last objective is motivated by the fact that, as noted in Section 2.2 above, the number of banyans in parallel Kg that is used in a group must satisfy:

$$K_g \leq T / T_s = \frac{T_s + T_t}{T_s} \qquad (1)$$

where T is the duration of the time slot, T_s the duration of the set-up phase and T_t the duration of the transmission phase; from condition (1), $T_s \leq T / K_g$; thus, given a certain number of banyans K_g that have to be used in parallel to provide the required routing capacity, a small latency per switching element translates into a lower required clock rate in the route set-up phase; in fact, in general, T_s is given by:

$$T_s = (n + 1) \cdot \frac{l_s}{R_s} + (f + a_s)\frac{1}{R_s} \qquad (2)$$

where $n + 1$ is the number of stages ($n =$ log, N stages in the banyan, plus one stage for the randomizers), l_s is the latency per stage (in bits) incurred to perform the switching algorithm, R_s is the clock rate during the route set-up phase, f is the duration (in bits) of the feedback phase, and a, is the duration of any additional period of time that may be necessary for signal synchronization and for changing the flow of signals from forward direction ***(i. e.,*** from inputs to outputs, during set-up) to reverse direction ***(i.e.,*** from outputs to inputs, during feedback), as explained in more detail below.

Conversely, we note that, for a given latency per switching element, the required clock rates during set-up and transmission phases depend on the number of banyans Kg that have to be scanned in a slot. In fact, from condition (1), the larger the number of

banyans that have to be scanned in a slot is, the smaller T_s is, and the higher the required clock rate during the set-up phase is. (Similarly, note that the smaller T_s is, the larger T_t is, and the lower the required clock rate during the transmission phase is; the duration T_t of the transmission phase is:

$$T_t = (424 + r + 1 + a_t)\frac{1}{R_t} \qquad (3)$$

where 424 is the number of bits in an ATM cell, r = log, R = log, (M / G) is the number of bits in the address to be used in the output buffer, one bit is the activity bit, and a_t is the duration (in bits) of a synchronization phase at the beginning of the transmission phase, and R_t is the clock rate during the transmission phase.)

From a design point of view, since the number of banyans that have to be scanned in a slot, for a given M, depends on the choice of the design parameter G, by properly choosing G we can impose required clock rates in the two phases of operations that are achievable with the available technology.

As a practical example of how the choice of G can be made taking into account the technology constraints on the achievable clock rates in the banyans, we consider our realization of the banyan network for a 1024 x 1024 MSM switch (M = 16, N = 64) for both 155.52 Mb/s and 620.08 Mb/s data rates.

At 149.76 Mb/s data rate (corresponding to the 155.52 Mb/s raw ATM data rate), the slot duration T is 2.83 μs. Assuming2 f = 2 bits and a_s = 4 bits, from equation (2) we have T_s = 13 / R_s. We also assume a, = 2 bits. Then, for different G, the required number of banyans in parallel per group, the corresponding maximum duration of the route set-up phase T_s (derived from condition (1)), and the required (minimum) clock rate R_s in the set-up phase and R_t in the transmission phase (derived from (2) and (3), respectively) are:

---

2 **In reality, *f*, a,, and a, depend on the clock rate, as explained below; thus, these assumptions should be verified once the clock rates have been** derived, **and** the **procedure** iterated.

G	K_g	T_s	R_s	R_t
1	***50***	***56.6*** ns	229.58 Mb/s	155.34 Mb/s
2	26	108.9 ns	119.38 Mb/s	157.95 Mb/s
4	13	217.8 ns	59.69 Mb/s	164.15 Mb/s

It is evident from these examples that the clock rate in the set-up phase is highly dependent on G; thus, indeed, by properly choosing G, we can adjust the required clock rates so that they can be achieved with the available technology. In our realization, although the clock rate required with G = 1 is within the limits of the BiCMOS technology adopted, we have preferred a more conservative approach, and chosen G = 2. The lower clock rate required in the set-up phase with G = 2 has allowed us to relax the specifications of the logic design and achieve synchronization more easily, leading to simpler and more robust design. We also note that the clock rates required with G = 4 are comfortably achievable even by conventional CMOS technology.

At 599.04 Mb/s data rate (corresponding to the 622.08 Mb/s raw ATM data rate), the slot duration T is 0.7 1 μs. Assuming f = 2 and a_s = 7, we obtain $T_s = 16 / R_s$. We also assume a, = 8 bits. In this case, for different G, the minimum required clock rates in the two phases of operation are:

G	K_g	T_s	R_s	R_t
1	***50***	14.2 ns	1130.26 Mb/s	630.0 1 Mb/s
2	26	27.2 ns	***587.74*** Mb/s	***640.63*** Mb/s
4	13	***54.4*** ns	293.87 Mb/s	665.79 Mb/s
8	8	***88.4*** ns	180.84 Mb/s	700.76 Mb/s

The clock rates in the set-up phase required with G = 1 and G = 2 are not achievable with our technology. The clock rate required with G= 4 is at the edge of what can be obtained with the technology. Given the experimental nature of our realization, we have chosen G

= 4 and adopted an aggressive design approach to meet the speed requirements. However, as noted below, the design tolerances that we were able to achieve would not be sufficient in a design destined to in-field use. A more conservative and robust choice would be G = 8, leading to less strict synchronization requirements and wider tolerances in the design.

3.1.3. Chip Design

We have designed a chip realizing a 64×64 banyan network augmented with randomizcrs by pairs. Two versions of this chip have been designed and simulated, one for the 155.52 Mb/s ATM data rate and the other for the 622.08 Mb/s ATM data rate. They differ in that the design for the lower data rate has been kept simple and compact, while the design for the higher data rate is slightly larger and more complex, since it requires some additional circuitry in order to meet the tighter circuit specification.

a) 155.52 Mb/s Design — Chip Layout and Global Signals

The chip layout of the version for the 155.52 Mb/s fits on a 3.7 lmm x 4.95mm silicon area. The chip receives four external control signals: SYST_CLK, providing the system clock, SYST-SLOT, defining the system slot boundaries, RESET, used to force switching elements and synchronization circuits into a know state at the beginning of operation, and ENABLE, used to enable the operation of the component. The chip also receives a low-speed clock LCLK (119.38 Mb/s) and a high-speed clock HCLK (157.95 Mb/s), to be used in set-up and transmission phases, respectively. From the external signal SYST-SLOT, an internal global signal CLKFRM is generated and used to define the local slot boundaries throughout the chip. Similarly, from RESET, an internal global signal CLR_STATUS is generated. From CLKFRM and the clock information, a global signal FWD is generated to control the direction of the data flow on chip ***(i.e.,*** **FWD** = 1, forward direction, data flow from inputs to outputs during path-establishment and

transmission phase; FWD = 0, reverse direction, data flow from outputs to inputs during feedback phase).

Here, we describe the banyan network circuit assuming that, at the beginning of the route set-up phase, headers are synchronized and aligned with one another. In Section 3.4 below, we discuss how synchronization and alignment of the headers is achieved (in the banyan-network component) during an alignment phase immediately preceding the route set-up phase.

b) 155.52 Mb/s Design — 2 x 2 Switching Element

The objective of achieving a l-bit latency ***(i.e.,*** the minimum possible latency) per stage in performing the switching algorithm imposes that the logic in each switching element must be capable of processing the address bit d_s, decide the state of the switching element and transmit the activity bit to the following stage in a single bit duration (resulting in the time diagram illustrated in Fig. 10). In order to achieve this objective, the decision logic must be streamlined. A block diagram of the 2 x 2 switching element is depicted in Fig. 11. The area of the switching element is $340\mu m \times 130\mu m$.

A detailed time diagram of the signal behavior in two 2 x 2 switching elements in consecutive stages is shown in Fig. 12. In the first of the switching elements, at the beginning of the route set-up phase, the activity bits a of the headers arrive at the two inputs of the switching element (denoted as **InputA** and **InputB** in the figure). As CLKFRM goes high (the rising edge of CLKFRM is guaranteed to arrive after the activity bits, but before the latching edge of LCLK), the local counter is activated, to time the decision process. The internal signal **DEC1** goes high, so that the activity bits can proceed to the DECISION LOGIC, where they arc latched; the internal signal TRANS goes low, separating the inputs of the switching element from the outputs; note that the internal signal DEC2 is low, and the output of the pass transistor **P1** is not actively driven, but

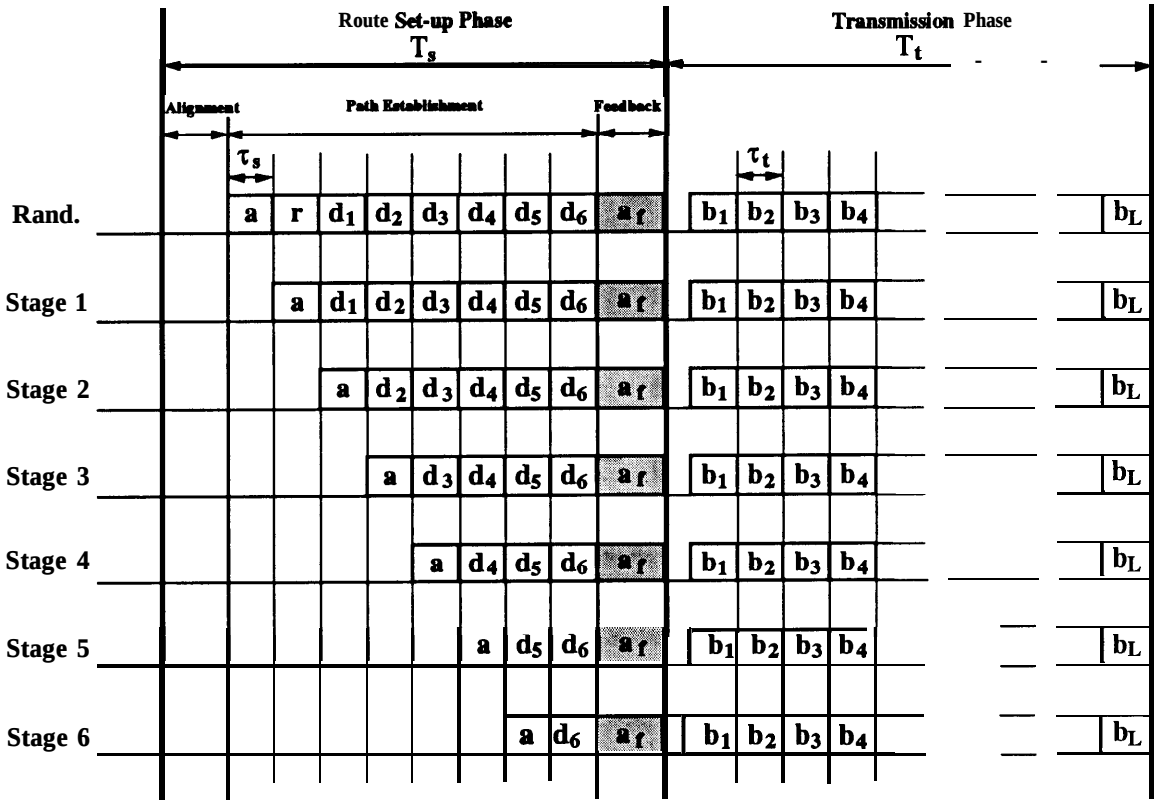


Fig. 10. Time diagram of the internal operation of the banyan network.

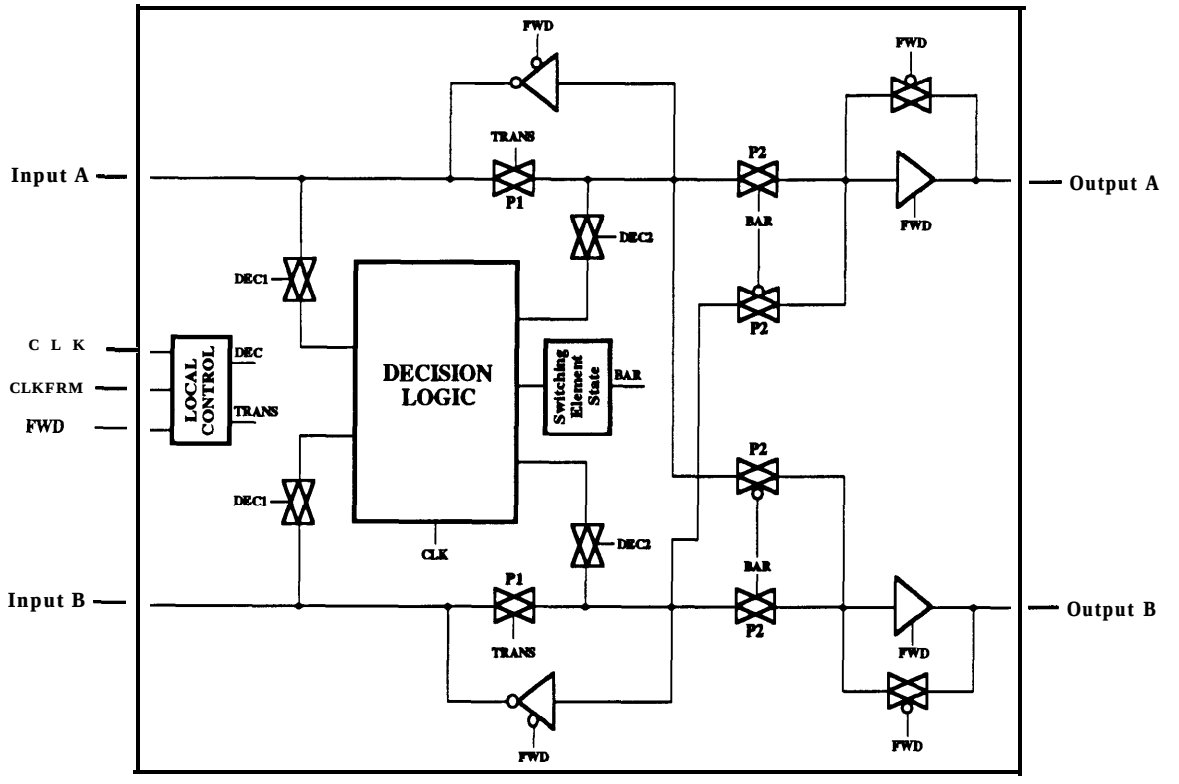


Fig. 11. Block diagram of a 2 x 2 switching element in the MSM banyan chip.

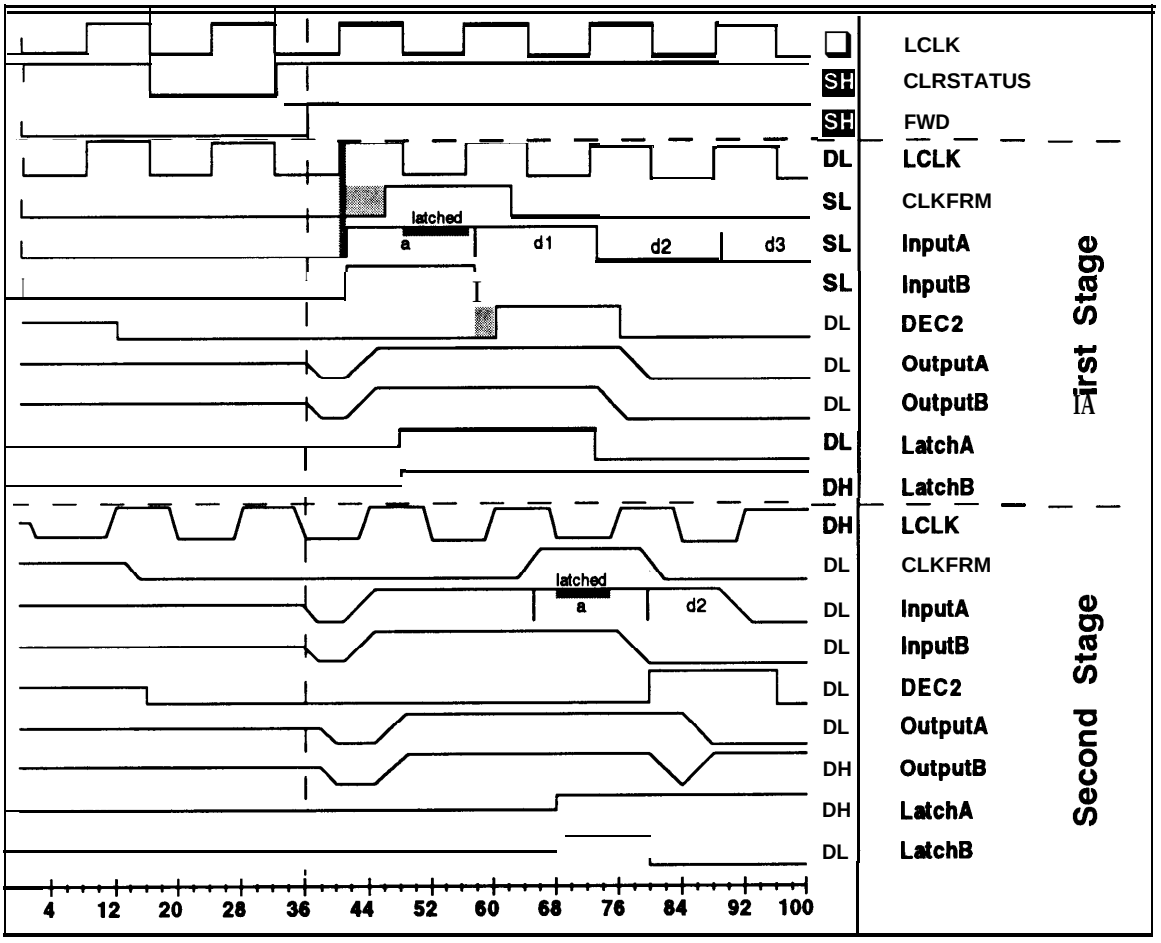


Fig. 12. Signal behavior during the decision process in a 2 x 2 switching element.

remains charged at the value it had before the falling edge of TRANS. During the following bit duration, the address bits d_s in the headers arrive at the switch and, since DEC1 is high, proceed to the DECISION LOGIC (in the case of Fig. 12, the header arriving at input A requests to be routed to output B, and the header arriving at input B requests to be routed to output A). The state of the switching element is decided, based on the activity bits and on the address bits, and latched in a storage element for the whole slot duration; the signal BAR is set accordingly (BAR = 1 for bar state, BAR = 0 for cross state), and the signal DEC2 goes high, enabling the activity bits to reach the outputs of the switching element and be transmitted to the following stage. Clearly, given the latency involved from the arrival of the address bit until the decision is completed, the activity bit does not occupy the whole bit duration. The requirement, however, is that the activity bit reaches the following stage before the latching edge of LCLK in that stage (it should be noted that LCLK is distributed by each switching element to the corresponding switching element in the following stage, and is also delayed while it traverses the switching element, thus making this requirement easier to meet).

In case of contention between two headers, conflict resolution is performed in order to select a winner, which is then properly routed; the activity bit of the loosing header is reset, and the header is misrouted. Note that a random selection of the winner would require the inclusion of pseudo-random-number generator circuits at each switching element (an addition which proves to be costly). Instead, in our realization of the banyan network, we set the state of the switching element to be the complement of the state assumed in the previous slot. Computer simulation has shown that this procedure is performance-wise equivalent to a random conflict-resolution algorithm for all traffic patterns of interest.

As the transmission of the activity bit is completed, DEC1 and DEC2 go low, disabling the decision logic, and TRANS goes high, allowing the remaining bits in the

header to pass through the switching element. CLKFRM is delayed by one bit and transmitted to the following stage to time the decision process in that stage. During the transmission of the feedback signal, FWD goes down, enabling the reverse path in the switching element. Finally, during the transmission phase, TRANS and FWD are high, enabling forward transmission of the data over the path that has been set-up.

The basic design of the switching element has to be slightly modified to realize the switching element to be used in the last stage of the banyan network. Specifically, the drivers in charge of the reverse path must be directly controlled by the latches in the decision logic where the activity bits (which have to be transmitted back to the input controllers) are stored. The switching element to be used in the randomization stage is also easily obtained from the basic design by removing the capability of resetting the activity bits in case of conflict. The randomization stage is operated by appending a randomly-generated bit in front of the destination address in the headers (this is done in the input controllers, as described in the following section).

We observe that in the MSM, the local switching address does not need to be conserved within the banyans (i.e., an address bit, once used in a switching element, can be dropped), since it is available in the input controllers. In contrast, in the Tandem Banyan Switching Fabric (TBSF) [Tob90b,Tob91], which uses a series rather than a parallel arrangement of the banyans, the switching address has to be cyclically shifted in each switching element, since it has to be used by every banyan in the series. In addition, the switching algorithm in the MSM is simpler than in the TBSF, since the conflict bit is not needed. These two factors translate into considerable area reduction (about 60%) in the switching element in the MSM, with respect to the switching element in the TBSF, implemented in the same technology [Chi9 1 ,Tob9 1]. The complexity of the switching element in the MSM is comparable to that of a switching element in the banyan network of the Batcher-Banyan switch [Mar90], although the latter does not necessitate feedback

or conflict-resolution circuitry (specifically, the switching element in the MSM banyan network, designed on our experimental, area-inefficient 0.8-μm BiCMOS gate array, shows only a 30% area penalty compared with a custom-designed, on a 1.2-μm CMOS technology, switching element in the Batcher-Banyan).

c) 155.52 Mb/s Design — Banyan Network and Circuit-Simulation Results

Interconnections between switching elements at different stages are bidirectionally driven by tri-state drivers, since route set-up, feedback, and transmission are "half-duplexed" on the same interconnection lines (the connections between banyans and input controllers are also bidirectionally driven).

Since the on-chip data flow has a preferred direction from left to right (and from right to left during feedback), skews between clock and data signals can be minimized by accurately laying out the clock lines, so that they closely follow the data paths. The low-speed clock, used during the routing phase, is distributed to each row of switching elements, and the delay that it incurs while traversing a switching element is matched as closely as possible with the delay incurred by the data signals. Clearly, with the sea-of-gates approach used, matching data and clock has required a cell library specifically designed for this chip. The high-speed clock, used during the transmission phase, is distributed only to the driver circuits at the inputs and outputs of the banyan. This significantly reduces the capacitive load on the clock lines and facilitates the distribution of the high-speed clock. As far as power distribution is concerned, a mesh topology is used, as necessary to accommodate the dynamic power requirements at high data rates.

To further illustrate the operation of the banyan network, we consider a specific example in which three headers are dispatched to the banyan network chip. The signal behavior at the input of the banyan network and at two specific switching elements in the banyan is illustrated in Fig. 13. Three headers are dispatched to the banyan: *banyan*[1]

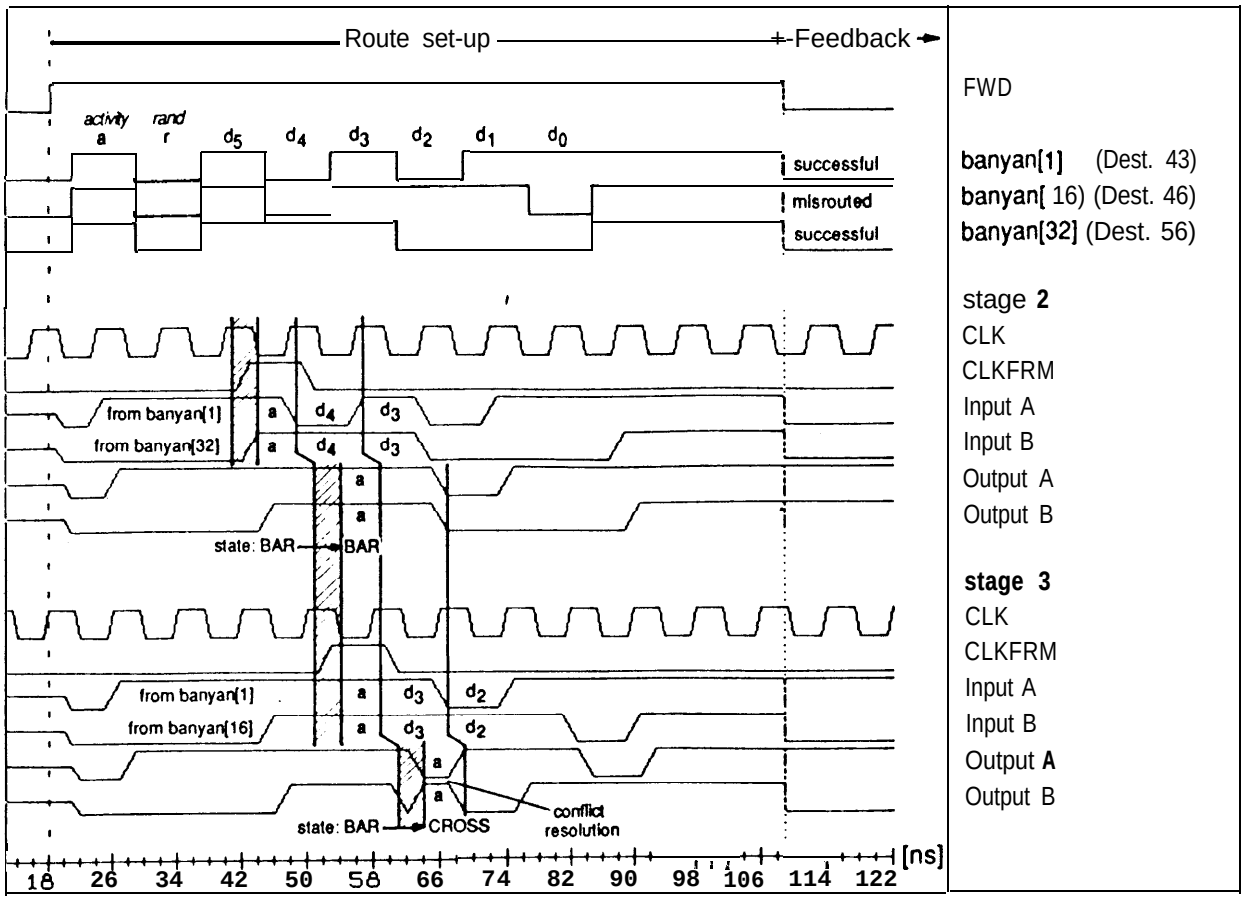


Fig. 13. Example of the signal behavior at the input of the banyan chip and at various switching element in the banyan network.

arrives at input 1 of the banyan and is destined to output 43; *banyan*[16] arrives at input 16 and is destined to output 46; *banyan*[32] arrives at input 32 and is destined to output 56. Headers are routed through the banyan. As shown in the figure, *banyan*[1] and *banyan*[32] reach the same switching element in the second stage of the banyan network, and are successfully routed to output A and output B of the switching element, respectively. At a switching element in the following stage, *banyan*[1] collides with *banyan*[16]. The former is properly routed to output B of the switching element, the latter is misrouted and its activity bit reset to 0. Then the headers proceed to the output of the banyan. During the feedback phase, the outcome of the routing phase ***(i.e.,*** the activity bits of the headers at the end of the route set-up phase) are transmitted back to the input controllers.

Given the relatively low complexity of our design, we were able to simulate the whole 64 x 64 banyan network chip both at the behavioral and at the circuit level, at the speeds of interest, using ADEPT. Furthermore, we were able to extract the whole layout, and use SPICE to simulate the operation of the whole chip in a selected number of critical cases (in fact, given the structure of the banyan network, by submitting properly selected subsets of headers, critical paths and worst-case combinations of events in the circuit can be easily isolated). The maximum simulated operating speed, at nominal conditions, were 220 MHz during the route set-up phase, and more than 600 MHz during the transmission phase. Such maximum speeds well above the required operating speeds ensure wide design tolerances to accommodate variations in temperature and process parameters. However, these operating speeds are not adequate for the design for the 622.08 Mb/s.

d) 622.08 Mb/s Design

In the design for the 622.08 Mb/s, to achieve the required operating speeds of 293.87 MHz during the route set-up phase and 665.80 MHz during the transmission

phase, the following two modifications must be introduced in the design of the 2 x 2 switching element.

i) Referring to the design of Fig. 11, the objective of implementing the feedback path in a way that is very simple and compact has led to the use of the series connection of pass transistors P1 an P2. This solution, although perfectly adequate for the design for the 155.52 data rate, obviously degrades the voltage level on the node between the two transistors, since such node is not actively driven, and limits the achievable clock rate during transmission. To solve this problem, an active driver must be introduced between P1 and P2; such driver must be tri-state, in order not to interfere during the feedback phase.

ii) In the design for the lower data rate, it was possible to control the skews of clock and data signals simply by matching their delays. Clearly, such a solution is only adequate up to a certain clock rate. To achieve higher clock rates in the route set-up phase, clock and data must be re-synchronized at each stage.

With these modifications, the maximum simulated operating speeds (at nominal conditions) were 310 MHz during the route set-up phase, and more than 800 MHz during the transmission phase. As mentioned above, in this case the design tolerances for the set-up phase are rather narrow, and a choice of G = 8 (with corresponding clock rates of 180.84 Mb/s and 700.76 Mb/s during set-up and transmission phase, respectively) would allow a more robust design. Both these modifications increase the area of the switching element. As a result, the area of the switching element for the 622.08 Mb/s data rate is $490\mu m \times 130\mu m$, a 40% increase with respect to the switching element for the lower data rate.

3.1.4. Achievable Size with Current VLSI Technologies

At the high operating speeds of interest, in order to achieve stage-to-stage synchronization without complex re-synchronization of the signals, it is certainly highly desirable that the banyan network is implemented on a single chip. Here, we estimate the maximum size N of the banyan network that is achievable using current VLSI technology. The constraints on the achievable size are determined by area and pin limitations, and by the synchronization requirements at the operating speeds of interest ***(i.e.,*** stage-to-stage synchronization internally to the banyan, and synchronization of the banyan network with N input components and N output components).

We have described the implementation of a 64 x 64 banyan network on a single chip for both 155.52 Mb/s and 622.08 Mb/s data rates using a high-performance, yet relatively area-inefficient gate-array on a rather small silicon area. From this experience, as far as area is concerned, we estimate that a full-custom 128 x 128 network is certainly implementable on a single chip using current submicron technologies. At the high operating speeds of interest, the limitation on pins stems from power dissipation in the I/O's (which may be CMOS or ECL at 155.52 Mb/s, and are ECL at 622.08 Mb/s), rather than from the number of pins that can be accommodated on a chip. At 155.52 Mb/s, we estimate that the power dissipation in the I/O's of a 128 x 128 network is comfortably manageable (especially given that CMOS I/O's may be used). In contrast, at 622.08 Mb/s, the power dissipation in the ECL I/O's would be prohibitive. As far as synchronization is concerned, based on our experience, we estimate that at 155.52 Mb/s synchronization of a 128 x 128 network can be achieved; it is our opinion, however, that synchronization of a 256 x 256 network would be problematic. Note that, at 622.08 Mb/s, not only power dissipation on the I/O's would prevent the realization of a 128 x 128 network, but also synchronization in such a network would be difficult. In conclusion, we estimate that the maximum sizes

of banyan networks achievable with current technology are 128 x 128 at 155.52 Mb/s and *64* x *64* at 622.08 Mb/s.

3.2. The Input Controller Component

3.2.1. Functionality and Design Issues

We recall from Section 2.2 above that, in a input controller with G = 1 receiving M input lines and controlling K banyans, the required functionality includes multiplexing up to M incoming packets per slot into the common memory, performing the dispatching algorithm, and retrieving and transmitting to the banyans up to K packets per slot (however, we have shown in [Chi93] that the number of packets transmitted to the banyans per slot can be limited to $M + 1$ without visible degradation in performance). With input lines of bandwidth V, the required memory bandwidth is $(M + K) \cdot V$ (again, it has been shown in [Chi93] that it can be reduced to $(2M + 1) \cdot V$ without noticeable effect on performance).

In a input controller with G > 1, the functionality must also include switching of each incoming packet into its requested group queue.

a) G = 1

A block diagram of an input controller for G = 1, with M input lines, and controlling K banyans, is shown in Fig. 14. In order to achieve the required memory bandwidth, serial-to-parallel (S/P) conversion of the bit streams is necessary to multiplex incoming packets into the common memory. (Similarly, parallel-to-serial (P/S) conversion is required to retrieve packets from the memory.) At the beginning of each slot, the local switching header at the top of the input queue is retrieved from the memory; if no header is available, a dummy header with activity bit a = 0 is generated. The *select-banyan* circuit selects the first banyan in the sequence, and the header is dispatched to the

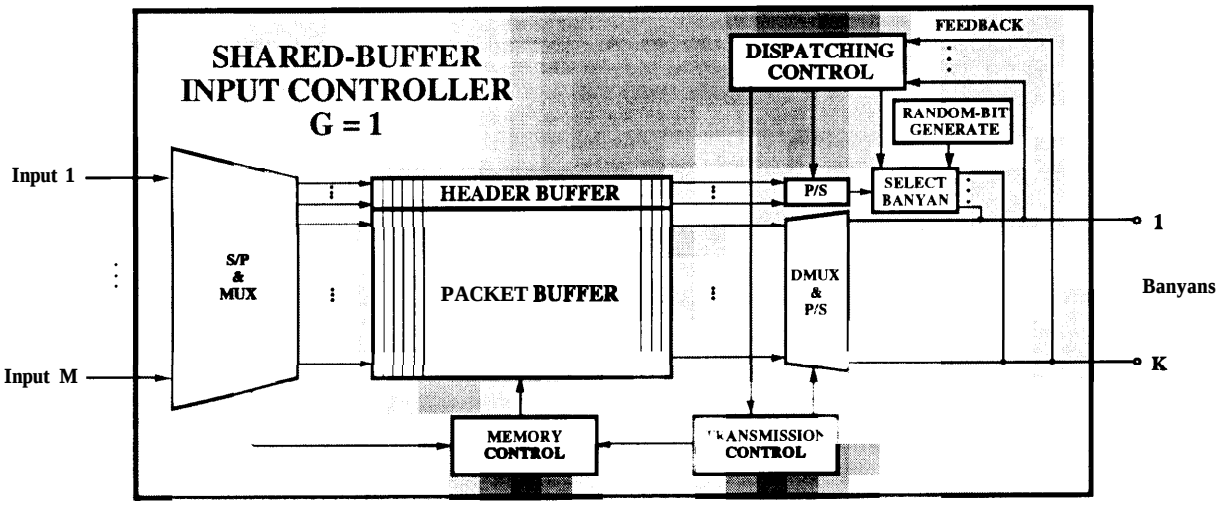


Fig. 14. Block diagram of the input component for G = 1.

banyan; during dispatching, a randomly-generated bit is appended in front of the destination address to be used to control the randomizers by pairs in front of the banyan network. Upon reception of the feedback signal (i.e., the value of the activity bit of the header after the path-establishment phase in the banyan has been completed), the ***dispatching-control*** circuit determines whether the header has been properly routed or not in the banyan. If the header has been misrouted, the dispatching-control circuit signals to the select-banyan circuit to dispatch the header to the following banyan. If the last banyan in the sequence has been attempted, the header is submitted to the first banyan at the beginning of the following slot. If the header has been successful, the dispatching-control circuit signals the successful dispatching to the select-banyan circuit, so that the following header in the queue, which has meanwhile been retrieved from the memory and prepared in serial form, is submitted to the following banyan; the dispatching control circuit also signals the successful dispatching to the ***transmission-control*** circuit, which is thereafter in charge of the transmission of the packet corresponding to the successful header. The transmission-

control circuit manages the demultiplexing from the memory, P/S conversion, and transmission to the banyans of up to ***K*** successful packets per slot.

As discussed in detail in the following, the major design issues in the input controller are: i) the design of the S/P converters and multiplexing of the packets into the memory, and ii) the design of the P/S converters and demultiplexing of the packets from the memory.

The common memory is sequentially accessed, given the FIFO operation of the input queue; thus, memory control simply requires two pointers, sequentially incremented (and eventually cyclically rounded) after each memory access, to store the current write and read memory addresses. The two pointers are compared in order to detect a buffer overflow. If the buffer is full, the write operation of the packets into the memory is not performed, and packets are discarded. In this way, packets in the memory are given priority over incoming packets; of course, the reverse could be easily implemented as well. If there is no incoming packet on an input line in a given slot, the write operation corresponding to that line is not performed.

b) G > 1

With G **>** 1, the main design issue is whether the queues corresponding to each of the G groups of banyans share a common memory, or are accommodated in separate memories.

In the former case, the memory is randomly accessed and G linked lists of packets, one per each group queue, must be maintained within the memory, making memory management considerably more complex than in the case of sequentially-accessed memory. The required memory bandwidth is (M + ***K) . V,*** as in the case of G = 1. A block diagram of the input component with group queues sharing a common memory is depicted in Fig. 15. The linked lists are formed by keeping the address of the packets at the head and at

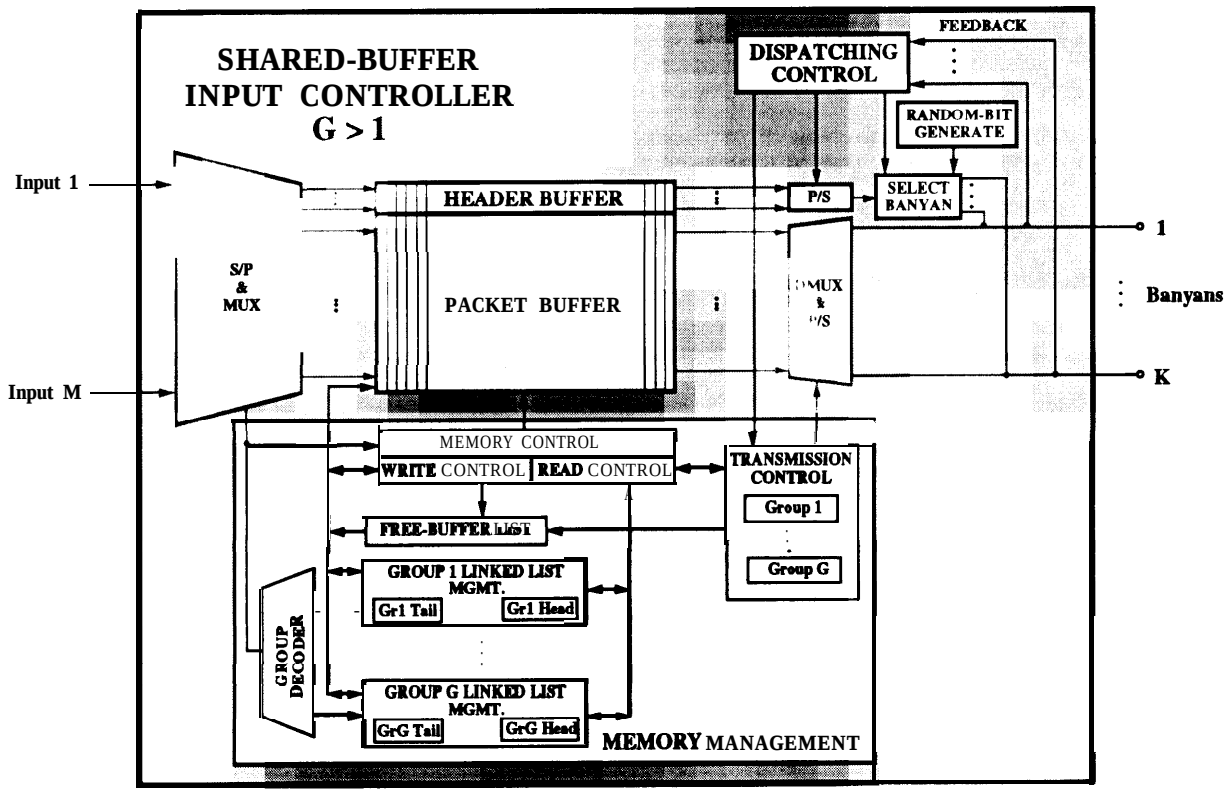


Fig. 15. Block diagram of the input component for G > 1, with group queues sharing a common memory.

the tail of the lists in the ***memory-management*** circuit, and by storing, together with each packet, a pointer to the next packet in its list. For each incoming packet, the ***group decoder*** selects the linked list in which the packet has to be inserted, specified by the first log, G bits in the destination address of the packets. The address of an idle memory location is retrieved from the ***free-buffer list,*** and the packet is stored in that location by the ***memory-control*** circuit. Simultaneously, in order to append the new packet at the tail of the desired linked list, the corresponding ***linked-list-management*** circuit updates the address chain by having the pointer of the packet currently at the tail of the linked list point to the new packet, and by storing the address of the new packet as the new tail of the list. To read packets from the queue, the address of the packet at the head of each group queue is fed to the memory-control circuit, which controls demultiplexing and P/S

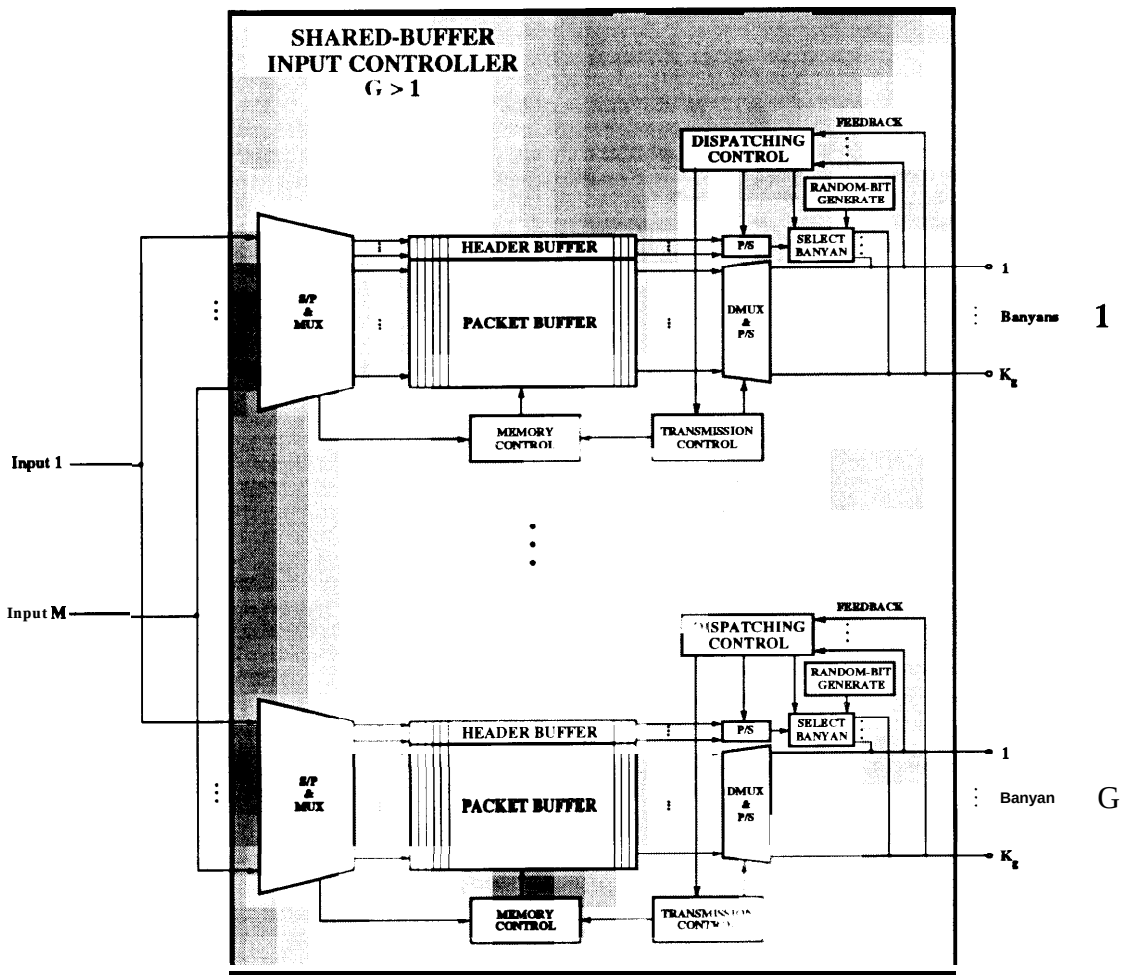


Fig. 16. Block diagram of the input component for G > 1, with group queues accommodated in separate memories.

conversion of the selected packets. Then, dispatching and transmission is performed in parallel for each group. If a packet is successfully dispatched, the corresponding linked list is updated and the next packet in the list is read from the queue. Once the transmission of a packet is completed, the address of its memory location is returned to the free-buffer list.

Alternatively, the G queues can be accommodated in separate memories; in this case, the circuitry described for G = 1 is simply replicated for each of the G groups, as depicted in Fig. 16. Each group handles K_g banyan networks. Packets arriving at the M

inputs are broadcast to all G queues; then, in each S/P converter, it is assessed whether or not each incoming packet has to be written into the corresponding memory. With separate memories for each queue, each memory is sequentially accessed; thus, the memory control is simple. Furthermore, the required memory bandwidth in each memory is reduced to $(M + K_s)$. **V.** Finally, this organization is very convenient in case the component has to be partitioned into several chips, since controllers for different groups can be placed on separate chips, with no interaction between chips other than the broadcasting of the packets.

From this discussion, it is evident that separate memories for each of the group queues are, in general, a more convenient design choice than a single memory shared among all queues. In fact, given the relatively small values of G that are of practical interest, and given that the required buffer sixes in the input components are rather small with both shared and partitioned memories **[Chi93],** the simpler operation, reduced memory bandwidth, and ease of partitioning of the latter clearly outweighs the advantage in buffer requirements of the former.

3.2.2. Chip Design

We have designed the critical circuit components of an input controller with $M = 16$ input lines, to be used in a 1024 x 1024 MSM switch. Circuits for both the 155.52 Mb/s and the 622.08 Mb/s data rates have been designed.

Since our objective in this implementation exercise was to demonstrate the feasibility of the input controller, we have used for this design the same 0.8-μm BiCMOS sea-of-gates adopted for the banyan network chip. This experimental sea-of-gates features a small die-size and quite limited dimensions of the cell array (namely, 218 x 140 cells), so that the buffer memory and the required circuitry for multiplexing and demultiplexing packets cannot be fitted in an actual chip. In addition, the sea-of-gates approach is not a

suitable design choice for the type of circuitry necessary in the input component. For example, an SRAM memory cell implemented with this sea-of-gates has an area about 7 times larger than a comparable memory cell implemented on a custom 0.8-μm CMOS technology used in a shared-memory switch [Koz91]. Furthermore, given the constraints imposed by the sea-of-gates, matching the pitches of S/P and P/S converters with those of memory, sensing and decoding is problematic, and leads to inefficient area utilization. Despite these limitations of the adopted technology, we have nevertheless used it to design and simulate at the desired speeds the critical circuitries of the input component (namely, the multiplexing and S/P conversion of M lines, the memory and its control, and the P/S conversion and demultiplexing to handle K_g banyans). In fact, in terms of electrical behavior, the severe area penalty incurred with respect to a more suitable custom design approach makes this design a worst-case scenario; thus, such an implementation exercise is a good indication of the feasibility of the component. In order to estimate the silicon area and the number of chips required to realize an actual input component, we have analyzed the specifications of currently available custom technologies that have been used in shared-memory switches (which require similar types of functionalities) [Koz9 1,Sho91]; specifically, we estimate that the input component for M = 16 can be accommodated on a single chip.

Prior to enter the input controller, each packet is processed by a header-conversion circuit which, based on the virtual circuit information in the ATM header, generates a local switching header and appends it in front of the packet. In order to deal with "easy numbers[3]" in the input controller, a 3-byte local header is actually appended to the 53-byte ATM cell.

---

[3] **It is common practice, in the design of this kind of circuits, to avoid to choose values of the design parameters that are prime numbers. Whenever possible, values that are powers of 2, or that form simple ratios with one another are preferred. Indeed, these small adjustments of the values of the design parameters often lead to simpler design of the multiplexing and demultiplexing circuits and simpler organization of the memory, as it will become evident in the sequel.**

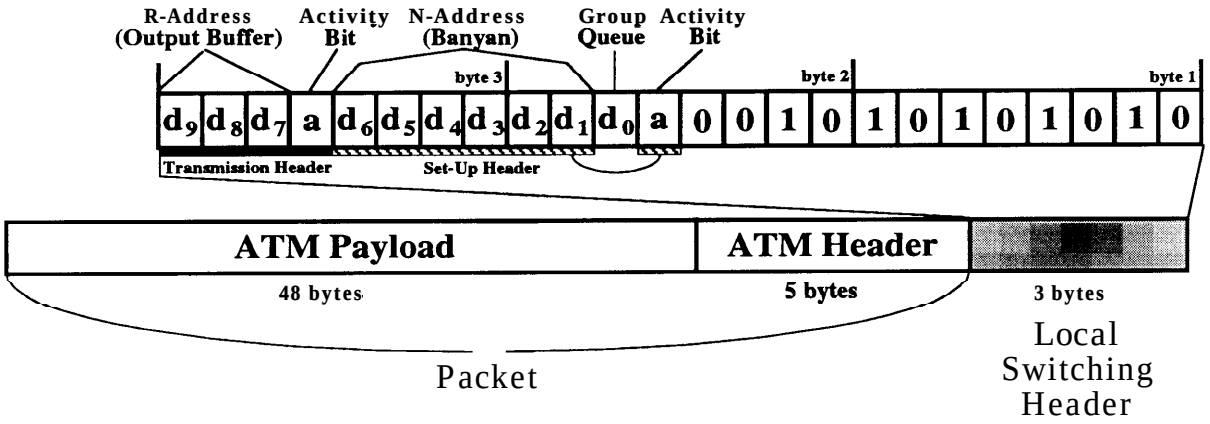


Fig. 17. Local switching header format for the 155.52 Mb/s design.

a) 155.52 Mb/s Design — Chip Organization and Floorplan

In the design for the 155.52 Mb/s data rate, as depicted in Fig. 17, the local header comprises 1 address bit ***(i.e.,*** log, G, where G = 2, as derived in the previous section) to select the desired group queue, a 7-bit ***(i.e.,*** log, N address bits + 1 activity bit) header that is used to set-up the paths in the banyans, and a 4-bit header that is sent in the transmission phase through the banyan, consisting of 3 address bits that are used once the desired output buffer is reached to select among the R = 8 ports served by the buffer, plus 1 activity bit. All the remaining bits in the header are padded with a sequence of zeroes and ones, to facilitate slot synchronization. Due to the addition of the local switching header, the rate of the input lines is 158.24 Mb/s (= (56 / 53) x 149.76Mb / s).

In our design for the 155.52 Mb/s ATM data rate, we have used G = 2, partitioned memories to accommodate the two group queues, and K_g = 26 banyans per group; with these parameters, the required buffer size to achieve a packet loss rate in the input buffer below 10" under bursty traffic with L = 100, at 0.9 load, is 90 packet buffers per queue. Maximum parallelism in the memory access, equal to 448 (= 56 x 8) is used. Again, to deal

with "easy numbers" in the design, although there are only 26 banyans per group, we perform 28 memory read operations per slot, each 448-bit wide (indeed, note that 56 x 8 / 28 = 16 is a simple ratio). Since 16 write operations (note: 56 x 8 / 16 = 28) are also necessary, a 64.3-ns cycle-time memory is required (= 448 / ((16 + 28) x 158.24 x 10^6)). With the adopted BiCMOS sea-of-gates, for the relatively small memory sizes of interest, we have actually designed a 25-ns cycle-time memory (which is indeed required to meet the specifications of the design for the 622.08 Mb/s, as described below). With such memory, which is considerably faster than what needed in this design, a lower degree of parallelism in the memory access could be used. Given our purpose of investigating the feasibility of the component, however, we have kept maximum parallelism in the memory access, and used this memory at 64-ns cycle time.

The floor-plan that we have used for the input-controller chip is depicted in Fig. 18. In this case, no circuitry is shared between the two group queues. For each queue, the memory has been arranged, as often done, in four banks of $L / 4 \times B$ cells, where L = 448 is the length of a packet plus the local header, and B is the buffer size per queue, which has been chosen equal to 128 (an "easy number" larger than 90). Each memory bank stores 14 bytes ***(i.e.,*** one quarter of a packet plus local header, as indicated in Fig. 18) and consists of 7 blocks of 128Word x 16bit RAM. The memory uses a conventional 6-transistor CMOS SRAM cell, realized in one sea-of-gates cell. BiNMOS logic, because of its superior driving capabilities, is used in row decoding and in the drivers of row and column lines. Actually, with the available memory cell and BiNMOS logic to drive row and column lines, larger memory blocks could be used in the design for the 155.52 Mb/s data rate, thus reducing the area overhead in the memory. However, our intention to reuse the same memory organization in the design for the 622.08 Mb/s data rate made us arrange the memory to meet the specifications of that data rate, resulting in smaller memory blocks than strictly necessary.

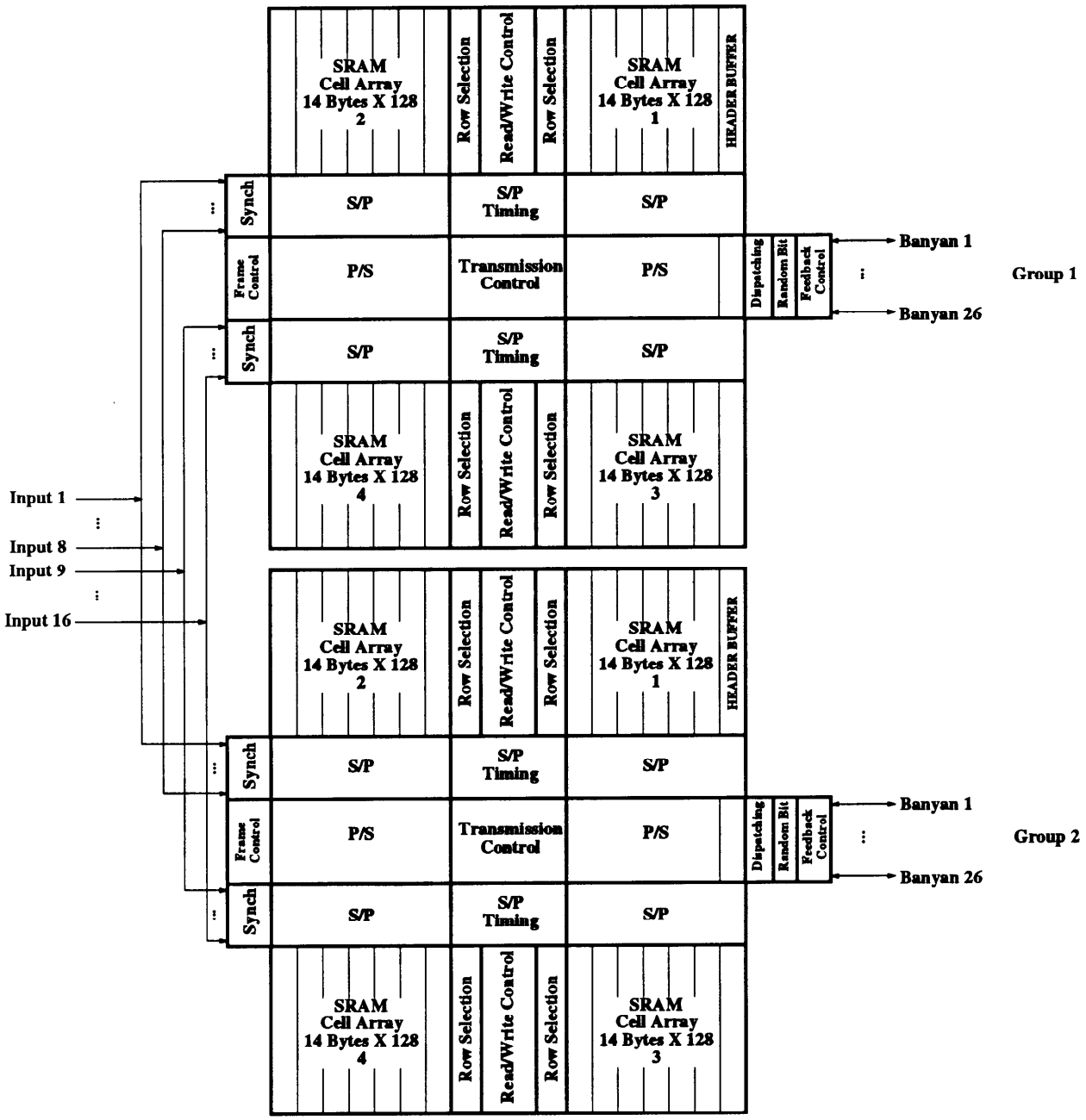


Fig. 18. Floorplan of the input-controller chip for $M = 16$, $G = 2$, $K_g = 26$.

It should be noted that alternative organizations to the completely separate circuitries for each group queues shown in Fig. 18 may be used. In particular, the S/P converters may be shared by the two queues. A 4-bank arrangement of the memory can still be used, with each bank divided in two parts, each storing one quarter of the packets for one of the queues (clearly, each bank, since it contains both queues, must have storage capacity to accommodate twice the number of words than in the previous case). Note that, in this case, the resulting larger memory blocks impose slightly stricter circuit specifications.

b) 155.52 Mb/s Design — S/P & MUX

A critical issue in the input component is the design of the S/P conversion and multiplexing of the packets into the memory. In fact, the synchronization requirements in the S/P converters constitute one of the major limitations on the number of lines A4 that an input controller can receive. Here, we discuss the design of S/P conversion and multiplexing (S/P & MUX) assuming that the incoming packets arrive at the S/P converters already synchronized with the local clock in the input controller and aligned with the local slot boundaries. In Section 5.4 below, we show that synchronization and alignment of the packets can be actually performed within the S/P converters.

For each input, the S/P converter, as depicted in Fig. 19, consists of a row of latches, with input and output enables, connected to a data line on which bits belonging to incoming packets are propagated. During each bit duration, a specific latch is selected to receive the incoming data bit. The output of each latch is connected to a corresponding column line in the memory; to write the bits in parallel into the memory, several latches are enabled simultaneously and their contents are transferred to the memory via the column lines.

Two interrelated problems must be dealt with in the design of S/P & MUX: area and synchronization. In fact, the S/P converters tend to be large, and contribute significantly

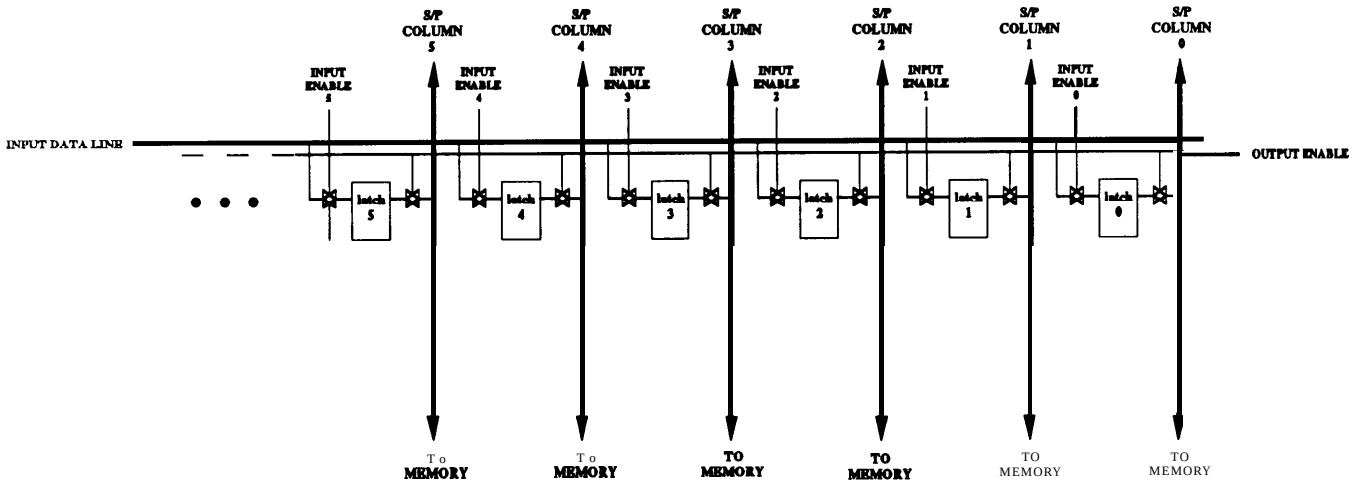


Fig. 19. S/P converter for an input line.

to the total silicon area required by the input component; furthermore, the S/P converters must operate synchronously with one another and with the memory; however, due to their large area, strict synchronization of several converters is difficult and eventually becomes a limitation on the number of S/P converters (and therefore on the number of input lines M) that can be used. For these reasons, it is important to minimize their area. This is accomplished by reducing the number of latches necessary for each input line.

Since packets arrive back to back on the input lines, if a packet is simply inserted in the S/P converter and then transferred into the memory in a single operation once it is fully received, then $2L$ latches per line are necessary to guarantee that an incoming bit does not overwrite the corresponding bit of the previous packet before the latter can be saved in the memory. Fortunately, S/P conversion and multiplexing can be accomplished using less than L latches per input line by properly:

i) managing the multiplexing of the packets in the memory,

ii) scheduling the write operations in the memory, and

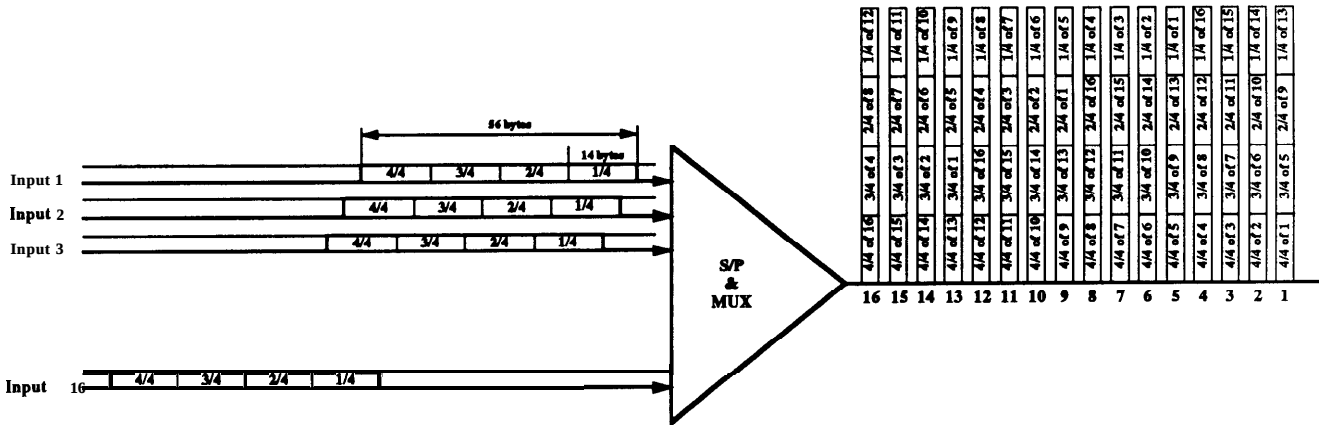


Fig. 20. Operation of S/P & MUX, 155.52 Mb/s data rate.

iii) offsetting the arrival times of the packets on the input lines.

Specifically, by managing the S/P & MUX circuitry as shown in Fig. 20, only $L/2$ latches per input line are needed. In this scheme of operation, the output of the S/P & MUX is 448-bit parallel data that consist of four blocks, each containing one quarter of four different packets. During a slot, 16 such parallel-data outputs are generated, and 16 corresponding write operations into the memory must be scheduled in such a way that, for each input line, the first quarter of a packet is written into the memory before the third quarter of the packet arrives, and the second quarter of a packet is written prior to the arrival of the fourth quarter; similarly, the third quarter is written in memory before the arrival of the first quarter of the next packet, and the fourth quarter is written prior to the arrival of the second quarter of the next packet.

The scheduling of write and read operations in the memory is constrained by the memory cycle time, and by the fact that read operations must be timed as needed to prepare the data streams for the set-up and transmission phases in the banyans. In general, in order to satisfy the desired scheduling requirements of the write operations, it is

necessary to space the write operations as uniformly as possible during a slot duration, in a way that is compatible with the required timing of the read operations. In our case, the starting times of set-up and transmission phases in each banyan are shown in Fig. 21, and a corresponding suitable scheduling of write and read operations with a 64-ns cycle-time memory is reported in Fig. 22. With this scheduling, in order to satisfy the scheduling requirements of the multiplexing, the arrival times of packets in consecutive input lines are offset by 24 bits. (Note that guaranteeing that packets are offset by a fixed number of bits is as difficult as guaranteeing that packets are aligned with one another.)

Referring to the floorplan of Fig. 18, since the first and third quarter of the packets, which are stored in memory banks that are vertically aligned with each other, are written simultaneously (and similarly the second and fourth quarter), each latch in the S/P converters must be connected to distinct column lines for the top and bottom half of the memory, and the column line corresponding to the desired portion of the memory selected at each write.

Further reductions in the required number of latches per line are more problematic, due to the constraints in scheduling the memory operations, and would necessitate to use a memory with a shorter cycle-time; in addition, further reductions in the number of latches would make the organization of the memory access more difficult. In general, the scheme of operation for the S/P & MUX circuitry described here can be tailored to most designs. However, there may exist some combinations of input data rate, memory cycle-time, number of write operations, number of read operations, and timing of the phases in the K_g banyans for which the scheme is not feasible with only $L/2$ latches in the S/P converter for each input line. In this rare unfortunate cases, either some of these parameters are modified, or the multiplexing scheme requires L latches per line.

In the S/P converters, during each bit duration, the proper latch for every input line must be selected. Provided that incoming data streams are closely synchronized and

write	completed @ (s)	read	completed @ (s)
		1	6.400E-08
		2	1.280E-07
1	1.920E-07		
		3	2.560E-07
		4	3.200E-07
2	3.840E-07		
		5	4.480E-07
		6	5.120E-07
3	5.760E-07		
		7	6.400E-07
4	7.040E-07		
		8	7.680E-07
		9	8.320E-07
5	8.960E-07		
		10	9.600E-07
		11	1.024E-06
6	1.088E-06		
		12	1.152E-06
		13	1.216E-06
7	1.280E-06		
		14	1.344E-06
8	1.408E-06		
		15	1.472E-06
		16	1.536 E-06
9	1.600E-06		
		17	1.664E-06
		18	1.728E-06
10	1.792E-06		
		19	1.856E-06
		20	1.920E-06
11	1.984E-06		
		21	2.048E-06
12	2.112E-06		
		22	2.176E-06
		23	2.240E-06
13	2.304E-06		
		24	2.368E-06
		25	2.432E-06
14	2.496E-06		
		26	2.560E-06
		27	2.624E-06
15	2.688E-06		
		28	2.752E-06
16	2.816E-06		

Fig. 21. Starting times of set-up and transmission phases in the banyans, 155.52 Mb/s data rate; times are in seconds, measured from the start of a time slot in the first banyan.

banyan	start setup (s)	start transm. (s)
1	0.000E+00	1.089E-07
2	1.089E-07	2.178E-07
3	2.178E-07	3.267E-07
4	3.267E-07	4.356E-07
5	4.356E-07	5.445E-07
6	5.445E-07	6.534E-07
7	6.534E-07	7.622E-07
0	7.622E-07	8.711 E-07
9	8.711 E-07	9.800E-07
10	9.800E-07	1.089E-06
11	1.089E-06	1.198E-06
12	1.198E-06	1.307E-06
13	1.307E-06	1.416E-06
14	1.416E-06	1.524E-06
15	1.524E-06	1.633E-06
16	1.633E-06	1.742E-06
17	1.742E-06	1.851 E-06
10	1.851 E-06	1.960E-06
19	1.960E-06	2.069E-06
20	2.069E-06	2.178E-06
21	2.178E-06	2.287E-06
22	2.287E-06	2.396E-06
23	2.396E-06	2.505E-06
24	2.505E-06	2.613E-06
25	2.613E-06	2.722E-06
26	2.722E-06	2.831 E-06

Fig. 22. Scheduling of memory read and write operations, 155.52 Mb/s data rate; times are in seconds, measured from the start of a time slot in the first banyan.

precisely offset by the desired amount of bits, two different approaches may be used. One possibility is to create S/P COLUMN ENABLE control lines by connecting together the INPUT ENABLE's of latches on different rows that have to be activated at the same time, and then select the proper S/P COLUMN ENABLE during each bit duration; since packets are offset by 24 bits, also the **S/P** COLUMN ENABLE's are connected to latches which are offset by the same amount, so that the area occupied by such interconnections is quite large. For the same reason, S/P COLUMN ENABLE lines tend to be long; hence, they are difficult to drive, and the skews on the enables in latches in different rows and positions are not easily controllable.

An alternative approach, which we have adopted in our design, is to time each S/P converter independently, by sending the enable signal to the first latch of the converter when the first bit of a packet in the corresponding input line must be received, and then propagating the enable signal through a delay line ***(i.e.,*** a row of flip-flops) so that it reaches each latch at the correct time. With this configuration, the S/P converters may also be used to adjust the offset of the packets, as explained in Section 5.4 below.

Note that, in order to properly enable a write operation in the memory, the first activity bit in the local header (see Fig. 17) is also sent to the memory control circuit to signal whether or not the packet in the current slot is active; the bit that specifies to which group queue the packet is destined is not stored in the S/P converter, but it is used by the memory control to determine whether an active packet must be stored in the memory, or is destined to a different group queue. Finally, we note that the switching header to be used in the set-up phase is stored in contiguous bits in the first block of the first memory bank, and the header to be sent in the transmission phase is stored adjacent to the packet, as shown in Fig. 23.

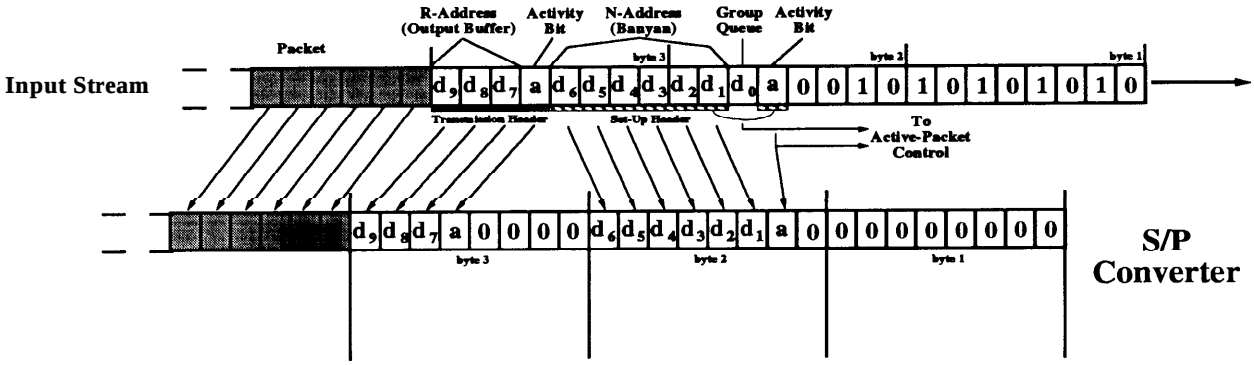


Fig. 23. Insertion of incoming data in S/P converter.

c) 155.52 Mb/s Design — DMUX & P/S and Dispatching

In each slot, up to K packets and local headers may be read from the memory. Demultiplexing of packets from the memory and P/S conversion (DMUX & P/S) is accomplished by means of P/S converters, which have a structure similar to that of the S/P converters. Each P/S converter consists of a row of latches with input and output enables; the inputs of the latches are connected to corresponding column lines in the memory. By enabling several latches simultaneously, bits are written in parallel into the latches from the memory. The outputs of the latches are connected to a common output data line; to read the content of the P/S converter in serial format, during each bit duration a specific latch is selected and its content transferred to the output data line. A P/S converter for every banyan network is provided.

The P/S converters present design problems similar to those encountered in the S/P converters. Also in this case, therefore, it is important to minimize the area occupied by the P/S converters by reducing the number of latches per output line. By demultiplexing the packets with a scheme of operation similar to the one described above for multiplexing, the number of required latches per output line is $L/2$. The operation of the DMUX & P/S

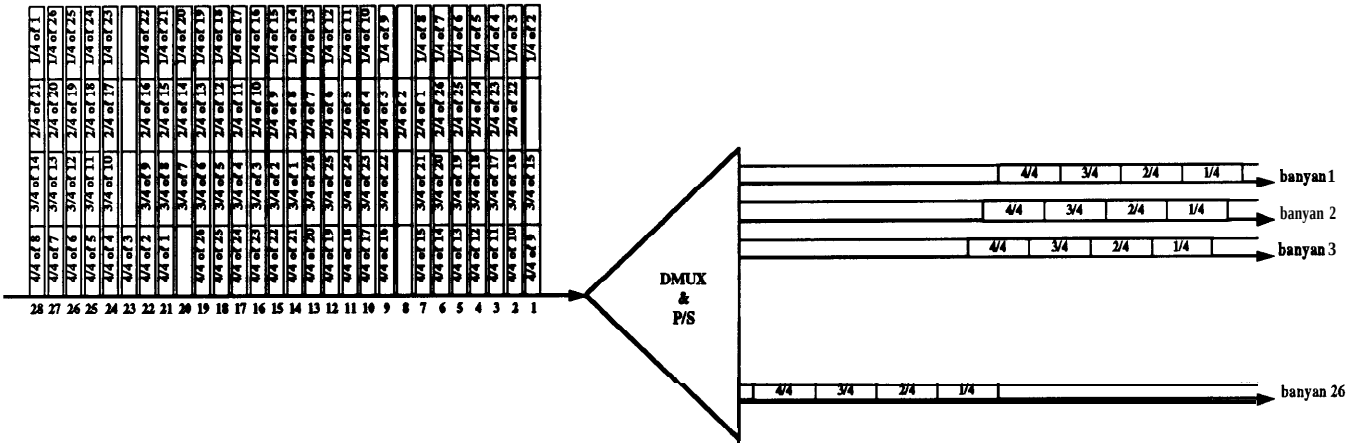


Fig. 24. Operation of DMUX & P/S, 155.52 Mb/s data rate.

circuit is shown in Fig. 24. During a slot, 28 read operations are performed, each consisting of 448-bit parallel data. Each parallel data consists of four blocks, containing one quarter of four different packets (note that, since the number of read operations is larger than the number of banyans, some of the blocks are empty). The read operations are scheduled in such a way that, for each output line, the first quarter of a packet is written into the P/S converter corresponding to a given banyan before dispatching in that banyan is started, but after the third quarter of the previous packet has been transmitted through the banyan; similarly for the other quarters of the packets. Also in this case, since there are $L/2$ latches per line, each latch in the P/S converters must be connected to two column lines, corresponding to the top and bottom half of the memory.

The operation of DMUX & P/S must also handle the fact that in general packets may be unsuccessful in some banyans. The first portion of the packets must be extracted from the memory and prepared in serial form in each P/S converter before the set-up phase in the corresponding banyan is completed. In our design, as soon as a controller determines that a packet is unsuccessful in a banyan, the remaining read operations

pertaining to the corresponding P/S converter are disabled for that slot, and the first quarter of the packet is copied to the P/S converter corresponding to the next banyan (of course, the read operation from the memory pertaining to such P/S converter is disabled to make sure that the packet is not overwritten); this is achieved during the set-up phase in the following banyan, so that if dispatching in that banyan is successful, the packet is ready for transmission. (Note that scheduling of the read operations in each P/S converter is rigorously maintained.) Since there are K_g P/S converters, this solution of maintaining a fixed one-to-one relation between converters and banyans, and copying the data as necessary from one converter to another, is simpler than transferring the data from the memory into a P/S converter, and providing circuitry to select to which banyan that converter should be actually connected during a slot.

In contrast, only one of the headers used to set up the paths in the banyans is active at any given time; furthermore, in case an header is unsuccessful in a banyan, it must be immediately delivered to the next banyan. Thus, a copy operation from one converter to another would add unnecessary delay in dispatching. Accordingly, we use only two P/S converters to serialize the headers, with circuitry to select to which banyan network they should be connected. One converter contains the active header and the other contains the following header in the queue, which must be ready for dispatching in case the first header is successful.

During every dispatching operation, a random bit to control the randomizers by pairs in the banyans is generated by sampling with the local clock the state of a l-bit counter that is incremented by an asynchronous clock (indeed, it can be shown that the sequence of samples obtained in this way is a random sequence). The random bit is appended in front of the destination address in the header during its transmission to the banyan.

The feedback signal f from the banyan networks is reshaped and used directly to control dispatching and transmission. Specifically, iff = 1 (successful header), dispatching of the next header to the next banyan and transmission of the current packet to the current banyan is enabled; if f = 0 (misrouted header), dispatching of the header to the next banyan, and copy of the first quarter of the packet from the P/S converter corresponding to the current banyan to the P/S converter corresponding to the next banyan is enabled.

d) 622.08 Mb/s Design

In our design for the 622.08 Mb/s ATM data rate, we have used G = 4. Accordingly, the local header comprises 2 address bits to select the desired group queue, a 7-bit header that is used to set-up the paths in the banyans, and a 3-bit header that is sent in the transmission phase through the banyan, consisting of 2 address bits (R = 4), plus 1 activity bit. Due to the addition of the local switching header, the rate of the input lines is 632.95 Mb/s (= (56 / 53) x 599.04 x 10^6).

In our design we have used partitioned memories to accommodate the group queues, and K_g = 13 banyans per group; with these parameters, the required buffer size to achieve a packet loss rate in the input buffer below 10" under bursty traffic with L = 100, at 0.9 load, is less than 60 packet buffers per queue. The organization of the group queues is similar to the one used in the design for the lower data rate, with complete separation of the circuitry for each queue. As in the previous case, the memory has been arranged in four banks of L / 4 x B cells, where B has been chosen equal to 64 (an "easy number" larger than 60); each memory bank consists of 7 blocks.

Since 16 write operations per slot are necessary, with our 25-ns cycle-time memory, by using maximum parallelism in the memory access, a maximum of 12 read operations per slot can be performed (= (448 / (25 x 10^{-9} x 632.95 x 10^6)) - 16). Fortunately, we have

shown above that as long as the output bandwidth of the memory is greater than $R + 1$ packets per slot, no degradation in performance is observed.

By using the same scheme of operation of the S/P & MUX circuitry described above for the lower data rate, the S/P converters can be designed using only $L/2$ latches per input line. In this case, the starting times of set-up and transmission phases in each banyan are shown in Fig. 25, and a corresponding suitable scheduling of write and read operations with the 25-ns cycle-time memory is reported in Fig. 26 (again, the arrival times of packets in consecutive input lines are offset by 24 bits).

As in the design for the lower data rate, a P/S converter for every banyan network is provided to achieve demultiplexing of packets from the memory and P/S conversion. The scheme of operation of the DMUX & P/S circuit is shown in Fig. 27; during a slot, 12 read operations are performed. Since the number of read operations is smaller than the number of banyans, data is copied in the P/S converter corresponding to the last banyan in the sequence (and a packet is dispatched to that banyan) only if at least one of the dispatching operations in the previous banyans has been unsuccessful in a given slot.

3.2.3. Achievable Size with Current VLSI Technologies

The limitations on the number of input lines M that a input controller can accommodate come from the required memory bandwidth of the input shared-buffer and from the functionality that the input controller must provide.

As far as memory bandwidth is concerned, at the 155.52 Mb/s ATM data rate, for example, by using a currently available 20-ns cycle-time memory [Sho91], with maximum parallelism in the memory access, the maximum achievable M *with* $G = 1$ *(i.e.,* $M = R$*)* is 35 (in fact, recall that the required memory bandwidth is $(M + K) \cdot V$); with $G = M$ *(i.e.,* $R = 1$*),* using partitioned memories for each group queue, the maximum achievable M is 141

write	completed @ (s)	read	completed @ (s)
		1	2.500E-08
1	5.000E-08		
		2	7.500E-08
2	1.000E-07		
		3	1.250E-07
3	1.500E-07		
4	1.750E-07		
		4	2.000E-07
5	2.250E-07		
		5	2.500E-07
6	2.750E-07		
		6	3.000E-07
7	3.250E-07		
8	3.500E-07		
		7	3.750E-07
9	4.000E-07		
		0	4.250E-07
10	4.500E-07		
		9	4.750E-07
11	5.000E-07		
		10	5.250E-07
12	5.500E-07		
		11	5.750E-07
13	6.000E-07		
		12	6.250E-07
14	6.500E-07		
15	6.750E-07		
16	7.000E-07		

Fig. 25. Starting times of set-up and transmission phases in the banyans, 622.08 Mb/s data rate; times are in seconds, measured from the start of a time slot in the first banyan.

banyan	start setup (s)	start transm. (s)
1	0.000E+00	5.445E-08
2	5.445E-08	1.089E-07
3	1.089E-07	1.633E-07
4	1.633E-07	2.178E-07
5	2.178E-07	2.722E-07
6	2.722E-07	3.267E-07
7	3.267E-07	3.811 E-07
8	3.811 E-07	4.356E-07
9	4.356E-07	4.900E-07
10	4.900E-07	5.445E-07
11	5.445E-07	5.989E-07
12	5.989E-07	6.534E-07
13	6.534E-07	7.078E-07

Fig. 26. Scheduling of memory read and write operations, 622.08 Mb/s data rate; times are in seconds, measured from the start of a time slot in the first banyan.

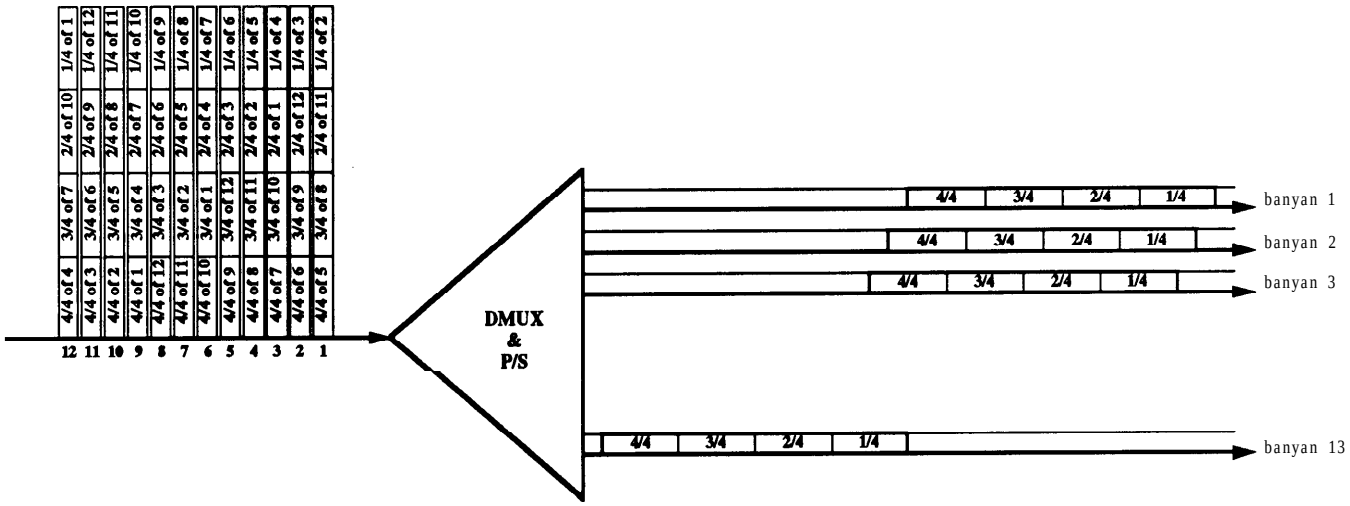


Fig. 27. Operation of DMUX & P/S, 622.08 Mb/s data rate.

(recall that, with G > 1 the required memory bandwidth in each memory is $(M + K_g) \cdot V$). Similarly, at the 622.08 Mb/s ATM raw data rate (corresponding to 599.04 Mb/s input data rate for the switch), by using the same 20-ns cycle-time memory, with G = 1 the maximum achievable M is only 8; with G = M, the maximum achievable M becomes 32.

The main limitation on the maximum achievable M comes from the multiplexing of packets arriving on the M lines into the common memory, which has to be performed regardless of G. We have described above our design of the multiplexing of 16 lines into a common memory at both 155.52 Mb/s and 622.08 Mb/s. From this experience, and by examining the specifications of multiplexing of 32 lines in existing shared-memory switches [Koz91], we conclude that, at 155.52 Mb/s, M = 32 is certainly feasible. It is our opinion that also M = 64 is within reach of currently available VLSI technology. At 622.08 Mb/s, we conservatively estimate that M = 16 is the largest number of lines that can be multiplexed into the common memory.

The required silicon area is determined primarily by the buffer requirements and by the circuitry necessary to perform S/P conversion and P/S conversion. We recall that, in general, the required size of the input buffer is relatively small. For example, with $M = 16$, $G = 2$, and $K_g = 26$, the required buffer size to achieve a packet loss rate below 10^{-6} under uniform traffic at full load is only 60 packet buffers; under bursty traffic with average burst size equal to 100 packets at 0.9 load is well below 200 packet buffers. Considering that the memory can be a conventional CMOS RAM, and analyzing the specifications of existing shared-memory switches [Sho91,Koz91] to evaluate the required area for S/P and P/S conversion and memory access, we estimate that an input component with $M = 16$ can be implemented on a single chip. To realize $M = 32$, $G \geq 4$ may be necessary to satisfy condition (1). With G = 4, a 50-ns memory is required to provide the necessary bandwidth. With $M = 32$, the buffer requirements and the area required by the memory access are probably too large for the whole component to be accommodated on a single chip, and two chips may be required.

3.3. The Output Buffer Component

3.3.1. Functionality

The output buffer is essentially a $K_g \times R$ shared-memory switch. Its architecture and functionality is therefore conceptually similar to the architecture of the input controller with $G > 1$ and group queues sharing a common memory, which has been discussed above and depicted in Fig. 15. In this case, the memory is randomly accessed, and the memory management is more complex since R linked lists, one per each output port, must be maintained. K_g incoming packets per slot have to be multiplexed in the memory. The required memory bandwidth is $(R + K_g) \cdot V$, where $K_g \cong 3R$. Packets arriving from consecutive banyans are offset by the duration of the set-up phase, which in the design for 155.52 Mb/s corresponds to 18 bit durations of the high-speed clock used during

transmission, and in the design for the 622.08 Mb/s corresponds to 37 bit durations. (In reality, as explained in the next section, due to the synchronization scheme used, the offset between consecutive packets is not strictly precise, and some adjustment is required.)

3.3.2. Achievable Size with Current VLSI Technologies

The limitations on the number of output lines R that a output component can have come from the functionality that must be provided (in particular, multiplexing of K_g incoming packets per slot into the common memory, and memory management), from the required memory bandwidth of the output shared-buffer, and from area constraints.

As far as bandwidth is concerned, by using a 20-ns cycle-time memory, the largest achievable R is 35 at 155.52 Mb/s, and 8 at 622.08 Mb/s. In the output components, the required area is determined primarily by the buffer requirements, by the circuitry necessary to perform S/P and P/S conversion for the memory access, and by the memory control.

An 8 x 8 shared-memory switch implemented on a single chip has been reported [Sho91]; in this realization, the total memory size is 256 packet buffers, which is adequate to sustain uniform traffic at 0.9 load. Based on the specifications of this realization, we estimate that an output buffer component with $R = 8$, with the buffer sized to sustain uniform traffic, can be implemented on a single chip at 155.52 Mb/s. $R = 16$ may also be realizable on a single chip, using a more compact memory technology such as the one used in [Koz91].

If the switch is designed to sustain bursty traffic, however, the buffer requirements are quite large, and the output buffer component must be partitioned over several chips. In this case, a bit-sliced organization of the buffer must be adopted. Bit-slicing brings some additional complexity in the design, since S/P conversion and multiplexing of the K_g incoming packets must be accomplished in two steps. First, packets have to be

synchronized and converted from serial to S-bit wide parallel format (where S is the number of slices used). Then, each of the S bit streams is fed to a different slice, and further converted from serial to the desired parallel format for the memory access. The feasibility of this buffer organization has been demonstrated, for 32 incoming lines, in existing shared-memory switches [Koz91]. Based on the specifications of this realization, we conservatively estimate that the maximum achievable size of the output component is 8 at 155.52 Mb/s, and 4 at 622.08 Mb/s. For example, under bursty traffic with average burst length equal to 10 packets at 0.9 load, assuming a total switch size equal to 1024, with R = 8, about 140 packet buffers per output port are necessary to achieve a packet loss rate below 10^{-6}. Based on available technologies, we estimate that such a component can be accommodated on three chips.

It is interesting to contrast the partitioning of input and output components over multiple chips. In the former case, since the memory is sequentially accessed, the most convenient way to partition the system is to completely separate the queues in different chips. Given the rather moderate buffer requirements at the input, the penalty in buffer size resulting from the use of partitioned memories rather than shared memories is not significant, and certainly more than offset by the added simplicity in design; in addition, full duplication of circuitry for each queue (e.g.. S/P converters) is affordable. Quite differently, the design tradeoff in the output buffers of the MSM between the beneficial effect of buffer sharing on the total memory requirements, and the complexity involved to implement a shared buffer is not as simple. If shared-memory is used, bit-slicing is required to partition the component into multiple chips. Alternatively, completely-partitioned memories could be used, at the cost of increasing significantly the total required buffer size. In this case, the output component essentially becomes a shared-medium switch, rather than a shared memory.

3.4. System Synchronization

The Memory/Space/Memory switching fabric is a synchronous system, and its operation is slotted. Packet alignment and synchronization control over the whole switching fabric represent two critical tasks in the synchronous design of fast packet switches, and constitute a primary source of limitation on the achievable switch size. Hence, in order to build a large switch, it is important to relax synchronization requirements as much as possible, and still achieve synchronous operation of the switch. In particular, with a large number of I/O lines, alignment of all the incoming packets and strict synchronization of all the bit streams with one another throughout the switch may not be feasible. Thus, a synchronization scheme that reduces the number of packets that have to be aligned together and the number of streams that have to be synchronized simultaneously is instrumental for the feasibility of the switch.

In the MSM, a global system clock and a signal defining the system's slot boundaries are distributed to every component in the switching fabric. However, the components are just loosely synchronized with one another, due to phase differences between clocks in different components located physically apart from each other; namely, in general, the signals in different components may ***only be mesochronous*** [Mes90] (i.e., their frequencies are equal, but their phases may differ). For similar reasons, the slot boundaries in different components are not necessarily aligned with one another. In the MSM, packet synchronization and alignment is accomplished by means of a two-step scheme: first, in each input controller, M incoming packets are synchronized with one another; then, in each banyan, N incoming packets are synchronized together.

3.4.1. Synchronization in the Input Component

For each input ***line, an input-port interface*** placed prior to the switching fabric terminates the incoming fiber, regenerates the incoming signal, appends a local switching

header to the packet, and sends it to the switching fabric. The global system clock and the signal defining the system's slot boundaries are distributed to the input-port interfaces. In each input-port interface, incoming signals are synchronized with the local clock and slot boundaries; however, since the input-port interfaces are also loosely synchronized with one another and with the switching fabric, signals exiting different input-port interfaces are not necessarily synchronous with one another, nor packets are precisely offset with each other by the precise amount required to perform their multiplexing into memory (described in Section 3.2.2 above).

In the first step of the synchronization scheme, in each input controller, the incoming packets from the corresponding M input lines are synchronized with the local clock and slot boundaries. Given the relatively small values of M, synchronization and alignment ***(i.e.,*** adjustment of the offset between packets arriving on consecutive lines) of the M input data streams may be achieved in different ways. One possibility is to closely synchronize the M input-port interfaces feeding an input component with one another and with the input component itself, and offset the input data streams by the precise number of bits in the input-port interfaces. The feasibility of this synchronization scheme has been demonstrated in existing shared-memory switches with 32 input lines at 155.52 Mb/s and 8 input lines at 622.08 Mb/s [Koz91]; thus, it is certainly a viable solution for the realization of the input component.

In our design, however, we have used a more general approach, which only requires that the input-port interfaces are loosely synchronized; in this solution, incoming signals are synchronized with the local clock using a limited amount of elastic storage placed in front of the S/P converters. Since the incoming signals are mesochronous (because, as we have said, they are generated from a global system clock distributed to all interfaces and input components), their phases with respect to the sampling clock used to write them in the elastic memory are indeterminate; thus, to perform the writing operation,

we need to guarantee that the latching edge of the clock does not overlap with the transition periods of the signals. In the design for the 155.52 Mb/s data rate, this is achieved by writing the incoming signals in the elastic memory using one of two clocks, in opposition of phase with each other. For each incoming signal, the selection of which one of the two clocks can sample the signal is performed using the sequence of ones and zeroes padded at the beginning of the local switching header. (Note that, since at this data rate the set-up and holding times of the elastic storage elements are small with respect to the bit duration, it is always guaranteed that one of the two clocks can be used for sampling.) Once the clock has been selected, the incoming bits are written into the elastic memory, and then read from it using the local clock. (Note that, in order to avoid metastable states in the storage elements, we must make sure that read and write operations do not overlap.) Circuit simulation shows that only 2 bits of elastic memory per line are needed. In the design for the 622.08 Mb/s data rate, the rate of the incoming signals is so high that the bit duration is not much larger than set-up and holding times of the storage elements; in this case, four sampling clocks with phases spaced by 90% are necessary to guarantee that there is always one of them that can correctly sample the incoming signals. After the incoming signals have been synchronized with the local clock, they are sent to the S/P converters, where their offset is adjusted.

Given that incoming signals are loosely synchronized, they are of course also not precisely offset by the required number of bits. The input-port interfaces use their local slot boundaries to offset their outgoing packets by the desired amount (note that, since the slot boundaries in different input-port interfaces are also not perfectly aligned with one another, the incoming packets in the input component, even if they were bit-synchronized, would not be necessarily offset with each other by the exact required number of bits). Without loss of generality, we can assume that the global signal defining the system slot boundaries is distributed to the various components in such a way that, after bit

synchronization in the input controllers, the difference between actual and desired offset between two packets arriving on consecutive lines (and between the beginning of packets arriving on the first input line and the slot boundary in the input controller) is bounded by a relatively small number of bits; namely, we have supposed that the value of such a difference is between -3 and +8 bits (a rather wide range even at 622.08 Mb/s). Then, the offset can be easily adjusted during the insertion of the packets in the S/P converters, by activating each S/P converter only when the first activity bit in the local header is sensed.

3.4.2. Synchronization in the Banyan Network Component

In the second step of the synchronization scheme, packets are synchronized at the inputs of the banyan networks. It is important to note that strict synchronization and alignment in the banyan networks is required only among up to N headers during the route set-up phase (recall that during this phase a lower clock rate may be used, thus facilitating strict synchronization). Similarly to the input component, a limited amount of elastic storage is provided in front of the banyan network in order to achieve close synchronization and alignment of the headers (in fact, also in this case the incoming signal are mesochronous). As far as synchronization is concerned, the operation of the elastic memory is similar to that of the elastic memory in the input components. At the beginning of each slot, all input lines are forced to zero. Then, each input controller sends one bit equal to one, which is sensed by the input circuitry in the banyan and used to select which one of the sampling clocks can be used to write the incoming signal into the elastic memory. Since the operation of the elastic memory for each input line starts only when the first bit of the incoming header is sensed, the alignment of the headers is easily performed within the elastic memories. (In fact, similarly to the input component, without loss of generality, we can assume that the global signal defining the system slot boundaries is distributed to the input controllers and banyan networks in such a way that, after bit synchronization, the misalignment among packets is bounded by a relatively small number of bits.) Once

the headers are stored in the elastic memories, all the elastic memories are read simultaneously using the internal clock, and the headers are sent to the banyan network.

At the beginning of the transmission phase, the paths in the banyans between inputs and requested outputs are fully established, and the data streams can flow independently from each other, provided that synchronization is maintained between pairs of connected inputs and outputs in the banyan. However, bit-synchronization of the input stream with the internal clock must be achieved. Strict alignment of the packets in this phase is not required, and the output driver circuits can individually self-synchronize with the beginning of the transmitted packets. A similar circuit to the one used in the route set-up phase can be used. In the banyans, each output is then individually synchronized with the corresponding output buffer. With this synchronization scheme, strict synchronization of input components, banyan networks, and output buffers with one another is avoided.

Finally, as far as system synchronization is concerned, we also observe that, in contrast with the TBSF, in the MSM no lines are split, and no loops in the synchronization scheme are introduced by recirculation. In a three-dimensional configuration of the MSM, such as the one reported in Fig. 3 above, all chip-to-chip interconnections may have the same lengths and no crossovers between lines are needed. All these are facilitating factors in system synchronization.

3.5. Comparison with Other Architectures

In this section, we compare the MSM with other architectures for large fast packet switches in terms of chip count and achievable switch size (for a complete review of existing architectures for large switches the reader is referred to [Chi93]). For simplicity, we only consider switches operating at the 155.52 Mb/s data rate, since the majority of the existing prototypes has been designed for this data rate.

As described above, we can implement a 1024 x 1024 MSM by selecting $M = 16$, $N = 64$, $R = 8$, and using two groups of 26 banyans each. We have seen above that, in this case, the input controller can be accommodated on a single chip; the banyan network can be implemented on a single chip; if the switch is designed to sustain bursty traffic with average burst size $L = 10$ packets, at 0.9 load, the output-buffer component requires three chips. The total chip count is therefore 500 chips (= 64 + 26 x 2 + 64 x 2 x 3). We have also estimated that the largest achievable size for the MSM at 155.52 Mb/s using current VLSI technology is 8192 x 8192. Such size is obtained by selecting $M = 64$, $N = 128$, $R = 8$, and using four groups of 27 banyans each. We have estimated that the input controller requires 4 chips, the banyan network a single chip, and the output-buffer component 3 chips. The total chip count is 2156 chips (= 128 x 4 + 27 x 4 + 128 x 4 x 3).

We first compare the MSM with multi-module configurations using non-blocking (and thus buffered) modules in the routing network [Koz91, Sho91, Ban91]. (See [Chi93] for a review of these configurations.) In this discussion, we only consider configurations that use switching modules of the shared-memory type, and assume that the available memory technology features a 22-ns cycle-time; consequently, the maximum available memory bandwidth, using maximum parallelism in the memory access equal to 424 ($=53\times 8$) bits, is $128 \cdot V$ (128 = 424 / (20 x 10^{-9} x 149.76 x 10^{6})), where Vis the line rate, equal to 149.76 Mb/s (corresponding to the 155.52 Mb/s ATM raw data rate).

Considering a ***Benes*** configuration with ***packet-by-packet*** routing, we can build a nonblocking 1024 x 1024 switch by using 3 stages of 32 modules of size 32 x 32, with the internal links running at the same speed of the external links [Chi93]. In this case, in order to sustain bursty traffic with average burst length equal to 10 packets at 0.9 load, about 110 packet buffers per port are necessary in each switching module; we estimate that such switching module can be implemented using 9 chips. Thus, the total chip count for the Benes configuration is 864 chips (= $3 \times 32 \times 9$). With the available memory

technology, the largest-size switching module that can be realized is 64 x 64 (recall that a switching module of size ***n*** requires a memory bandwidth equal to ***2nV);*** in such a module, more than 100 packet buffers per port have to be provided to support the desired traffic conditions; we estimate that 17 chips are required to implement the module. A Benes configuration of 3 stages of 64 modules of size 64×64 has a size equal to 4096 x 4096, for a total chip count of 3264 chips (= 3 x 64 x 17). It is important to note that, using packet-by-packet routing in a multi-module architecture of buffered modules, the packet sequence is not maintained, and additional means have to be provided to restore the original order of the packets [Chi93].

In a three-stage Benes configuration, by using ***virtual-circuit-by-virtual-circuit*** (VC-by-VC) routing [Chi93], the packet sequence is maintained, but the internal links must be run at three times the speed of the external links to make the fabric nonblocking [Mel89]. In this case, with the available memory technology, if we only consider module sizes that are powers of two, the largest module that can be implemented is 16 x 16 (in fact, the modules in the internal stage of the Benes require a memory bandwidth equal to $3 \times 2 \times 16 \cdot V$, where ***V*** is the speed of the external links); in these modules, about 125 packet buffers per port should be provided to support the desired traffic conditions; we estimate that such modules are realizable on 5 chips. The size of a three-stage Benes configuration using such module is 256 x 256, for a total chip count equal to 240 chips. With no restriction to module sizes that are powers of two, the largest achievable size for a switching module is 21 x 21; we estimate that it can be accommodated on 6 chips; the size of a three-stage Benes based on this module is 441 x 441, with a total chip count of 378 chips (= 3 x 21 x 6). Using a Benes configuration with more than three stages, with VC-by-VC routing, requires to run the internal links at even higher speeds to make the fabric nonblocking (for example, in a five-stage Benes, the internal links must be run at

4.63 times the speed of the external links [Mel89]), thus further restricting the achievable size of the switching modules.

In a three-stage *Clos* topology with *VC-by-VC* routing, $m = 2n$ (where n is the number of inputs in each module in the first stage) modules must be used in the intermediate stage, and the internal links must be run at twice the speed of the external links, to make the fabric nonblocking [Mel89]. Considering first only module sizes that are powers of two, with the available memory technology, the largest achievable sizes for the modules are: 16×32 for the modules in the first stage (the required memory bandwidth is $(16 + 32 \times 2) \cdot V$), which can be accommodated on 5 chips; 32×32 for the modules in the second stage (the required memory bandwidth is $32 \times 2 \times 2 \cdot V$), which can be implemented on 9 chips; and 32×16 for the modules in the third stage, which can be accommodated on 5 chips. A three-stage Clos network using these modules consists of a first stage of 32 modules of size 16×32, a second stage of 32 modules of size 32×32, and a third stage of 32 modules of size 32×16, for a switch size equal to 512×512 and a total chip count equal to 608 chips ($= 32 \times 5 + 32 \times 9 + 32 \times 5$). With no restriction to module sizes that are powers of two, the largest achievable sizes are: 25×50 for the modules in the first stage (required memory bandwidth equal to $(25 + 50 \times 2) \cdot V$), realizable on 8 chips; 32×32 for the modules in the second stage (required memory bandwidth is $32 \times 2 \times 2 \cdot V$), realizable on 9 chips; and 50×25 for the modules in the third stage, realizable on 8 chips. A three-stage Clos network using these modules consists of a first stage of 32 modules, a second stage of 50 modules, and a third stage of 32 modules, for a switch size equal to 800×800, and a total chip count equal to 962 chips ($= 32 \times 8 + 50 \times 9 + 32 \times 8$).

In summary, the MSM has a lower chip count and can be built in larger sizes than multi-module configurations based on nonblocking modules. The advantage with respect to configurations using packet-by-packet routing is substantial (e.g., for a 1024×1024

switch, the number of chips in the MSM is 500 rather than 864 in a Benes configuration of nonblocking modules, and the largest achievable size for the MSM is 8192 x 8192, as opposed to 4096 x 4096 for the multi-module configuration). Note that the difference in chip count increases with the switch size (e.g., a 8192 x 8192 MSM only requires 2156 chips, as opposed to a 4096 x 4096 multi-module configuration of nonblocking modules, which requires 3264 chips). Furthermore, the multi-module configurations of nonblocking modules using packet-by-packet routing also require means to restore the original order of the packets. The advantage of the MSM in terms of chip count and achievable switch size is even more dramatic with respect to configurations using VC-by-VC routing (e.g., the largest size for the multi-module configuration of nonblocking modules is 800 x 800, only one tenth of the achievable size for the MSM).

We then compare the MSM with multi-module configurations based on a bufferless routing fabric [Chi93]. *In case* of *the Growable Switch* [Eng89], assuming an expansion factor *m/n* = 2.5, and considering shared-memory output modules, the largest output switching module that can be achieved using a 22-ns cycle-time memory is 80 x 32 (the required memory bandwidth is (80 + *32)V);* such a module, with sufficient buffer capacity to support bursty traffic with average burst length equal to 10 packets, at 0.9 load, can be accommodated on 12 chips. A corresponding 32 x 80 routing module in the first stage can be implemented on 2 chips. We estimate that the maximum number of minislots to perform the routing algorithm (corresponding to the number of input and output modules) that can be accommodated in a time slot at this data rate is 32. Accordingly, 32 x 32 crossbars are required in the intermediate stage; each crossbar can be implemented on a single chip. The largest achievable Growable Switch consists therefore of a first stage of 32 routing modules of size 32 x 80, a second stage of 80 crossbars of size 32 x 32, and a third stage of 32 output switching modules of size 80 x 32, for a switch size equal to 1024 x 1024, and a total chip count equal to 528 (= 32 x 2 + 80 + 32 x 12).

For the ***Group Banyan*** switch [Wan92], we assume an expansion factor $m/n = 2$ and estimate that the largest size of a binary-grouping module that can be implemented on a single chip is 64 x 64 (= $m \times m$); consequently, $n = 32$. Each output switch has size 64 x 32, and can be implemented on 11 chips with sufficient buffer capacity to support the desired traffic conditions. For a 1024 x 1024 switch, the routing fabric consists of 5 stages of 32 binary-grouping modules. The total chip count is therefore equal to 512 (= 32 x 5 + 32 x 11). It should be noted, however, that the synchronization of many stages of binary-grouping modules may be problematic, thus limiting the achievable switch size.

In summary, the MSM has basically the same chip count of other multi-module configurations based on bufferless routing fabrics; however, the achievable size of the MSM may be larger than that of other architectures.

3.6. Summary

Given the high speeds at which fast packet switches have to operate, an essential element in the design of a fast packet switching architecture is the demonstration of its feasibility at the sizes and speeds of interest. We have studied the feasibility of the Memory/Space/Memory switching fabric by designing and simulating the critical circuit components of a 1024 x 1024 MSM switch at both 155.52 Mb/s and 622.08 Mb/s ATM standard rates, using a BiCMOS sea-of-gates on a 0.8-μm technology. Specifically, we have designed a banyan network of size 64, and shown the feasibility of input controllers with 16 input lines and of output buffer components with 8 output lines, capable of handling the number of banyan networks needed to accommodate the traffic offered to the switch. Finally, we have shown how synchronization of all the components constituting the switch can be achieved.

We have also estimated the largest achievable switch size using current VLSI technology, and concluded that a 8 192 x 8192 switch, with $N = 128$, $M = 64$ and $R = 8$, is

feasible at the 155.52 Mb/s ATM data rate. At the 622.08 Mb/s ATM data rate, a 1024 x 1024 MSM switching fabric, with $N = 64$, $M = 16$, and $R = 4$, may be the largest switch that can be realized with available technology.

References

[Ban91] T. R. Banniza *et al.*, "Design and Technology Aspects of VLSI's for ATM Switches," *IEEE Jour. Select. Areas Comm.*, SAC-9, pp. 1255- 1264, Oct. 199 1.

[CCI90] CCITT Study Group XVIII, Revised Draft Recommendation I.21 1, Geneva, May 9-25, 1990.

[Chi91] F. M. Chiussi, H. Amano, and F. A. Tobagi, "A 0.8-μm BiCMOS Implementation of the Tandem Banyan Fast Packet Switch," *Proc. CZCC 91,* paper 3.3, San Diego, CA, May 1991.

[Chi92] F. M. Chiussi and F. A. Tobagi, "A Hybrid Shared-Memory/Space-Division Architecture for Large Fast Packet Switches," *Proc. ICC* '92, paper 332.5, Chicago, IL, June 1992.

[Chi93] F. M. Chiussi and F. A Tobagi, "Performance of a Three-Stage Banyan-Based Architecture with Input and Output Buffers for Large Fast Packet Switches," *Tech. Rep. No. CSL-TR-93-573,* Stanford University, Stanford, CA, June 1993.

[Eck88] A. E. Eckberg and T.-C. Hou, "Effects of Output Buffer Sharing on Buffer Requirements in an ATDM Packet Switch," *Proc. INFOCOM '88,* paper 5A.4, New Orleans, Louisiana, March 1988.

[ElD89] M. El-Diwany *et al.*, "An Advanced BiCMOS Process Utilizing Ultra-Thin Silicon Epitaxy Over Arsenic Buried Layers," *Proc. 1989 Int. El. Dev. Meet.,* paper 9.7, Washington, December 3-6, 1989.

[ElD90] M. El-Diwany *et al.*, "Low Voltage Performance of An Advanced CMOS/ BiCMOS Technology Featuring 18 GHz Bipolar fT and Sub-70ps CMOS Gate Delays," *Proc. 1990 Znt. El. Dev. Meet, San* Francisco, CA, December 9-12, 1990.

[ElG89] A. El Gamal, J. L. Kouloheris, D. How, and M. Morf, "BiNMOS: a Basic Cell for BICMOS Sea-of Gates," *Proc. CICC* 89, paper 8.3, San Diego, CA, May 15-18, 1989.

[End90] N. Endo *et al.*, "Traffic Characteristics Evaluation of a Shared Buffer ATM Switch," *Proc. GLOBECOM '90,* paper 905.1, San Diego, CA, December 1990.

[Eng89] K. Y. Eng, M. J. Karol, and Y. S. Yeh, "A Growable Packet (ATM) Switch Architecture: Design Principles and Applications," *Proc. GLOBECOM '89,* paper 32.2, Dallas, TX, Nov. 1989.

[Hlu87] M. Hluchyj and M. Karol, "Queueing in High-Performance Packet Switching," *IEEE Jour. Select. Areas Commun.,* SAC-5, 8 , pp. 1274-1292, Oct. 1987.

[Hlu88] M. Hluchyj and M. Karol, "Queueing in Space-Division Packet Switching," *Proc. INFOCOM '88,* paper 4A.3, New Orleans, Louisiana, March 1988.

[Hua84] A. Huang and S. Knauer, "Starlite: A Wideband Digital Switch," *Proc. GLOBECOM '84,* pp. 121-125, Atlanta, GA, Nov. 1984.

[JSA91] Special Issue on Large Scale ATM Switching Systems for B-ISDN, eds. W. E. Stephens *et al., IEEE Jour. Select. Areas Comm.,* SAC-9, Oct. 199 1.

[Koz91] T. Kozaki *et al.,* "32 ×32 Shared Buffer Type ATM Switch VLSI's for B-ISDN's," *IEEE Jour. Select. Areas Comm.,* SAC-9, pp. 1239-1247, Oct. 1991.

[Kuw89] H. Kuwahara *et al.,* "A Shared Buffer Memory Switch for an ATM Exchange," *Proc. ICC '89,* paper 4.4, Boston, MA, June 1989.

[Lie90] S. C. Liew, "Performance of Input-Buffered and Output-Buffered ATM Switches Under Bursty Traffic: Simulation Study", *Proc. GLOBECOM '89,* paper 905.2, San Diego, CA, Dec. 1990.

[Mar90] W. S. Marcus, "A CMOS Batcher and Banyan Chip Set for B-ISDN Packet Switching," *IEEE Jour. Solid-Stute Circ., 25,* pp. 1426-143 1, Dec. 1990.

[Mel89] R. Melen and J. S. Turner, "Nonblocking Networks for Fast Packet Switching," *Proc. INFOCOM '89,* pp. 548- 557, Ottawa, Ont., Canada, 1989.

[Mes90] D. G. Messeschmitt, "Synchronization in Digital System Design," *IEEE Jour. Select. Areas Comm.,* SAC-B, pp. 1404-1419, Oct. 1990.

[New88] P. Newman, "A Fast Packet Switch for the Integrated Services Backbone Network," *IEEE Jour. Select. Areas Comm.,* SAC-6, pp. 1468- 1479, Dec. 1988.

[Sar91] K. W. Sarkies, 'The Bypass Queue in Fast Packet Switching," *IEEE Trans. Commun., 39,* pp. 766-774, May 1991.

[Sho91] Y. Shobatake *et al.,* "A One-Chip Scalable 8 x 8 ATM Switch LSI Employing Shared Buffer Architecture," *IEEE Jour. Select. Areas Comm.,* SAC-9, pp. 1248-1254, Oct. 1991.

[Sho91] Y. Shobatake ***et al.***, "A One-Chip Scalable 8 x 8 ATM Switch LSI Employing Shared Buffer Architecture," ***IEEE Jour. Select. Areas Comm.,*** SAC-9, pp. 1248-1254, Oct. 1991.

[Suz89] H. Suzuki ***et al.***, "Output-Buffer Switch Architecture for Asynchronous Transfer Mode," ***Proc. ICC*** '89, paper 4.1, Boston, MA, June 1989.

[Tob90a] F. A. Tobagi, "Fast Packet Switch Architectures for Broadband Integrated Services Digital Networks," ***IEEE Proceedings,*** vol. 78, 1, Jan. 1990.

[Tob90b] F. A. Tobagi and T. C. Kwok, "Fast Packet Switch Architectures and the Tandem Banyan Switching Fabric," ***Proc. NATO Workshop on Architecture and Performance Issues of High-Capacity Local and Metropolitan Area Networks,*** Sophia Antipolis, France, June 25-27, 1990.

[Tob91] F. A. Tobagi, T. Kwok, and F. M. Chiussi, "Architecture, Performance and Implementation of the Tandem Banyan Fast Packet Switch," ***IEEE Jour. Select. Areas Comm.***, SAC-9, pp. 1173-1193, Oct. 1991.

[Tur88] J. S. Turner, "Design of a Broadcast Packet Switching Network," ***IEEE Trans. Commun., 36,*** pp. 734-743, June 1988.

[Wan92] W. Wang, "Architectural Design and Performance Analysis of Large-Scale Asynchronous Transfer Mode (ATM) Switches," ***Ph.D. Thesis,*** Stanford University, Stanford, CA, Aug. 1992.

[Wu80] C.-L. Wu and T.-Y. Feng, "On a Class of Multistage Interconnection Networks," ***IEEE Trans. Computers,*** pp.694-702, Aug. 1980.