Architectural Choices in Large Scale ATM Switches

Jonathan Turner and Naoaki Yamanaka

WUCS 97-21

May 1, 1997

Department of Computer Science
Campus Box 1045
Washington University
One Brookings Drive
St. Louis, MO 63130-4899

ABSTRACT

The rapid development of Asynchronous Transfer Mode technology in the last 10-15 years has stimulated renewed interest in the design and analysis of switching systems, leading to new ideas for system designs and new insights into the performance and evaluation of such systems. As ATM moves closer to realizing the vision of ubiquitous broadband ISDN services, the design of switching systems takes on growing importance. This paper seeks to clarify the key architectural issues for ATM switching system design and provides a survey of the current state-of-the-art.

Architectural Choices in Large Scale ATM Switches

Jonathan Turner
jst@cs.wustl.edu
Washington University

and

Naoaki Yamanaka
yamanaka@nttsl.nslab.ntt.co.jp
NTT Research Labs, Musashino

ABSTRACT

The rapid development of Asynchronous Transfer Mode technology in the last 10-15 years has stimulated renewed interest in the design and analysis of switching systems, leading to new ideas for system designs and new insights into the performance and evaluation of such systems. As ATM moves closer to realizing the vision of ubiquitous broadband ISDN services, the design of switching systems takes on growing importance. This paper seeks to clarify the key architectural issues for ATM switching system design and provides a survey of the current state-of-the-art.

1. Intoduction

Switching systems are central components in communications networks and serve two central purposes. First, they allow a reduction in overall network costs by reducing the number and/or cost of the transmission links required to enable a given population of users to communicate. Second, they enable heterogeneity among terminals and transmission links, by providing a variety of interface types. ATM networks have introduced a range of new issues into switching system design, but the principal ones arise from two key features of ATM. The first is its support of multi-rate virtual circuits with statistical multiplexing for efficient transmission of multimedia and data traffic. The second is its support for multicast virtual circuits which enable efficient transmission of information from a source to many receivers. At the same time, switching systems for ATM, must address the fundamental issues of scalability, reliability and cost-effectiveness. The pressure of the competing requirements on switching systems, together with new opportunities created by advances in underlying technologies, have helped to spark a remarkable period of creativity in switching system design.

As the title of the paper suggests, we attach considerable importance to the issue of scalability in ATM switch design. While one can build communication networks of arbitrary size using switching systems of limited capacity, it turns out to be far more cost-effective to use switching systems that have sufficient capacity to handle traffic from tens or hundreds of thousands of individual users. As ATM networks are more widely deployed and approach the sort of ubiquitous usage that telephone networks enjoy, there will be a compelling need for high capacity systems. Just as telephone switches require capacities of 1-10 Gb/s to provide cost/effective service to large user populations, ATM switches supporting applications using thousands of times the bandwidth of traditional voice services, will need

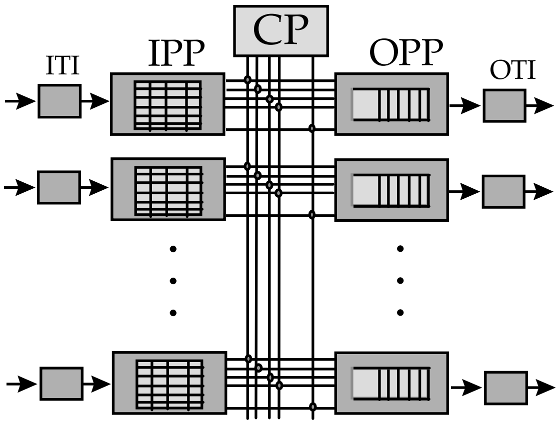

Figure 1. Simple Bus-Based Switch

terabit per second capacities. This in turn demands switch architectures with linear or near-linear scaling characteristics.

There have been several prior surveys of ATM switching that provide useful insights into this field. The papers by Ahmadi and Denzel [AHM89] and Tobagi [TOB90] provide good coverage of the state-of-the-art as of the 1990 time frame. The paper by Zegura [ZEG93] introduces a systematic method for comparing architectural alternatives in a quantitative fashion and uses it to demonstrate substantial differences in the cost/performance of different switch architectures. In addition, the book by DePrycker [DEP95] includes a survey of a representative set of ATM switching system designs and the collection of papers by Dhas, Konangi and Sreetharan [DHA91] provides a good introduction to the literature.

This paper seeks to place the key architectural issues raised by the design of ATM switching systems into perspective. While we do not provide a comprehensive survey, we have sought to capture the major ideas that have emerged from the switching research community in the last decade, and to fit them into a framework that facilitates the comparison of different system designs. We group systems into three major categories: single stage switching systems, buffered multistage systems and unbuffered multistage systems. The succeeding sections of the paper are organized by these categories and within each section, a variety of sub-categories are detailed.

2. Single Stage Switching Systems

Most ATM switches today use one of several single stage switching techniques. Single stage switches are relatively simple, but are limited in the number of ports and total throughput that they can support effectively. This section surveys the major categories of single stage switches and discusses the key design and performance issues.

2.1. Buses

The most common form of switching system is built around a simple passive bus, as illustrated in Figure 1. Arriving cells first pass through an *Input Transmission Interface* (ITI), which converts the (typically optical signal) to a parallel electronic form. This involves recovering timing information from the encoded signal and presenting the decoded signal to the subsequent components. The *Input Port Processor* (IPP) implements the input processing functions at the ATM level. This typically involves synchronizing the arriving data stream to the internal timing of the switch, checking for ATM header errors and performing a routing table lookup based on the arriving cell’s VPI and VCI. Given this information, the lookup yields an outgoing port number and a new VPI and VCI which replace the original values of these fields as the cell is forwarded on the outgoing link. To transfer data to outputs, IPPs contend for access to the bus (using one of a variety of bus-contention techniques) and then transmit cells on the bus in parallel form, using all w signal lines. Along with the received data, the IPP transmits the number of the output port on which the cell is to be forwarded. The *Output Port Processors* (OPP) compare the output port numbers in cells observed on the bus to their own addresses and buffer matching cells in a queue prior to transmission. The OPP may implement a simple fifo, a collection of several fifos with differing priorities or per virtual circuit queues. It may also implement one or more congestion

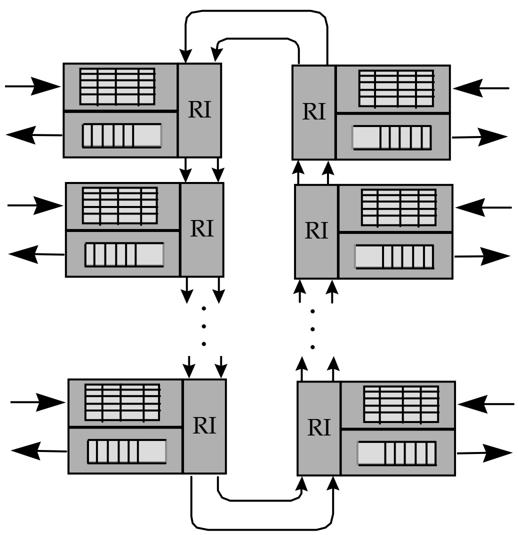

Figure 2. Ring-Based Switch

control mechanisms to improve queuing performance in the presence of bursty data traffic. The *Output Transmission Interfaces* (OTI) encode data for transmission and convert it to optical form for transmission to distant switches and terminals.

In addition to the data path components emphasized in Figure 1 a switch must also have a *Control Processor* (CP) which configures the IPP routing tables in response to user requests. The CP is typically implemented as a general-purpose processor with the appropriate software to perform terminal-to-switch and switch-to-switch signaling, in addition to switch control and maintenance functions. In some systems, the CP is connected directly to the bus and may also have a separate control bus connecting it to the memory implementing each of the routing tables. Alternatively, the CP may be connected to the system through one of its external ports, and communicate control information using ATM cells carried on a designated VPI/VCI. This requires that the IPPs and OPPs have the ability to interpret and respond to control cells appropriately. In larger switching systems, multiple control processors may be required. The most straightforward approach in this case is use a conventional shared memory multiprocessor.

To provide non-blocking communication, a bus supporting n ports, operating at a data rate of R bits per second each, must provide a bandwidth of at least Rn bits per second. If the clock frequency used for the bus is r Hz, the bus width, w, must be at least Rn/r. For example, a system supporting 16 OC-3 links with an internal clock rate of 40 MHz, requires a bus width of about 64 bits. Notice that as the number of ports in a system increases, both the number of ports connecting to the bus and the width of the bus must increase. This yields a quadratic growth characteristic, making it uneconomical to implement very large systems. Another problem with bus systems is that as the number of ports connecting to a bus increases, the capacitive loading on the signal lines grows, reducing the maximum clock frequency that can be achieved (that is r shrinks with the capacitive loading). This means that the bus width must grow faster than the port count in order to maintain sufficient bus bandwidth. There are techniques to reduce or eliminate the impact of this capacitive loading effect, but they are more complex to implement.

For smaller capacity switches (up to a few Gb/s), the bus-based architecture is very common. A typical commercial system is the ForeRunner ASX-200 made by Fore Systems [FOR97]. TranSwitch produces and sells an integrated circuit that implements the IPP and OPP functions and the required bus-interface logic for a bus-based switch. A detailed description can be found in reference [TRA96].

2.2. Rings

Figure 2 shows a single stage system implemented using a ring. This system contains the same components as the bus system described above, but replaces the bus with a ring-based interconnect. Each IPP and OPP has an associated *Ring Interface* (RI), which is responsible for inserting data onto the ring and removing data from the ring. The simplest ring protocol for ATM switches uses a time slotted approach in which cells are sent on the ring synchronously during specified time slots, and a busy/idle bit is used to indicate the availability or unavailability of a given time slot. An IPP with a cell to send waits for the start of a time slot in which the busy/idle bit indicates that the time slot is available, changes the value of the bit to make the time slot busy and then transmits the cell within the time slot. OPPs compare

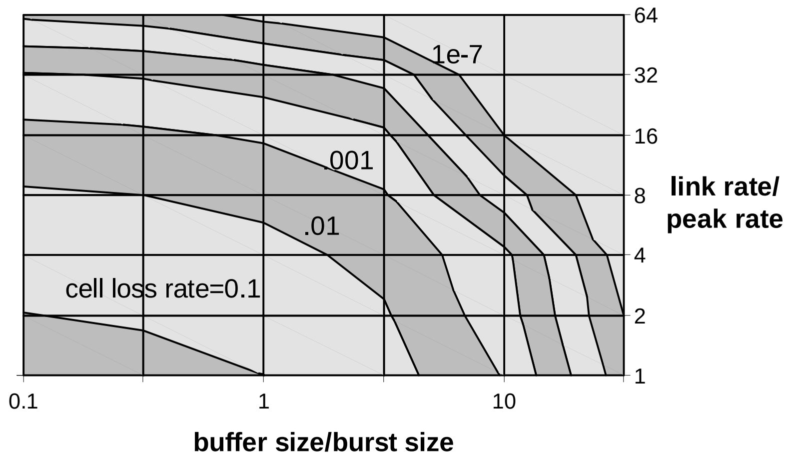

Figure 3. Contour Plot of Cell Loss Rate
(peak-to-avg=10, offered load=0.6)

the output port number of transmitted cells to their own address and copy matching cells off the ring, changing the value of the busy/idle bit to indicate that the slot is now available.

Ring systems, like buses, have quadratic growth characteristics, but unlike buses, they are not further limited by capacitive loading effects. Because all data transmission is point-to-point, one can typically operate rings with higher clock frequencies than bus systems implemented using the same technology. This can become a significant factor in systems with large numbers of ports. Rings do introduce some additional latency, relative to buses, but for switching applications these latencies are typically fairly modest. For example, a ring supporting 16 OC-3 links with a ring clock rate of 80 MHz and a latency of two clock ticks per ring interface would have a latency of under one microsecond, which is about one third the time required to transmit the cell on the external OC-3 link.

While rings are popular in local area networks and have advantages over buses in switch design, they are not often used for this purpose. One system that does use a ring for switching is described in [TAK87].

2.3. Queuing Issues in Single Stage Switches

ATM networks are designed to handle a variety of different types of applications, including applications with time-varying data rates. For applications in which the peak data rate is much larger than the average data rate, statistical multiplexing is used to share the link bandwidth most efficiently among the competing data streams. In single stage systems based on buses and rings, the queuing occurs primarily at the OPPs. The amount of memory needed for efficient statistical multiplexing is highly dependent on the ratio of the link rate to the peak transmission rate of individual streams. If this ratio is high enough (say 20-to-1 or more), small buffers are sufficient. If the ratio is small, the total buffer requirement at an OPP is approximately 10 times the size of the average sustained data burst. Note, by burst size we mean the size of an application-level burst of data, not the size of a transport level packet. For an application such as interactive, high resolution image retrieval, burst sizes of 1 MB are typical, leading to required buffer capacities of up to 10 MB, if the data transfer rates are high, relative to the link rates.

Figure 3 is a contour plot showing how the cell loss rate varies as a function of the ratio of link rate to peak virtual circuit rate, and as a function of the ratio of burst size to buffer size. This plot was produced by simulation of on-off bursty traffic with exponentially distributed holding times in the on and off states and an average traffic intensity equal to 60% of the link's capacity. Similar plots can be produced for different holding time distributions, such as a "heavy-tailed" distributions like the Pareto distribution. The effect of changing to a Pareto distribution is to shift the place where the contours intercept the horizontal axis to the right. There is no change in where the contours intercept the vertical axis.

For systems in which large buffers are needed to obtain efficient statistical multiplexing, delay can become a significant factor. This leads to a requirement for multiple priority levels, to allow real-time traffic with limited rate variability to obtain access to the link without substantial queuing delays. As a result, switches will often have two or more service class specific queues in the OPP. Some systems go further and provide per virtual circuit queues to ensure fair access to the output link for competing bursty data sources. This adds some complexity to the output port processor but can be important for good performance when the ratio of link rate to peak virtual circuit rate is small. This is most often the case when the link rates are relatively low (150 Mb/s or less). Higher bandwidth links can often get by with smaller amounts of buffering and less sophisticated buffering techniques.

2.4. Shared Memory

In bus or ring based switches in which large queuing memories are provided at the output ports to ensure good statistical multiplexing performance, one observes that the average memory utilization is usually quite low. This suggests that there is an opportunity for reducing the overall system cost by sharing the memory among the various output ports, instead of dedicating it to each output. This can be done by introducing a large shared memory connected to the bus or ring containing per-link or per-VC queues implemented as linked lists. Because this implies that cells must pass first from the input ports to the shared memory and from there to the outputs, the bus/ring bandwidth must be doubled for nonblocking switching and the memory bandwidth must be at least as large as the bus/ring bandwidth. While this can be difficult to achieve in larger, high speed systems, the cost-reduction available through the sharing of the memory can be substantial. In typical cases, the amount of memory needed in a shared buffer system is just two to three times larger than the amount needed for a single output port in the case of dedicated buffering. So in a 16 port OC-3 switch, we could reduce the buffer memory requirement by a factor of 5 to 8, at the cost of increasing the parallelism of the bus/ring from 64 to 128 and using a memory subsystem with an aggregate bandwidth of at least 600 MB/s.

MMC Networks makes a set of chips that can be used for shared memory switches. A detailed architectural description of a shared memory switch using these devices can be found in [MMC96].

2.5. Crossbars

The last major class of single stage switch is the crossbar, showing in Figure 4. A crossbar comprises an array of *crosspoints*, which when closed, connect a horizontal data path to a vertical data path, thereby creating a connection between an input port and an output port. Crossbar-based systems can be significantly less expensive than bus or ring systems with equivalent performance because the crossbar allows multiple data transfers to take place simultaneously between disjoint input/output pairs. This means that each IPP (OPP) need only have the ability to send (receive) data at the rate of its input (output) link times a small constant (called the *speed advantage*), while in a bus or ring system, it must be able to send (receive) data at n times the link rate. The overall bandwidth of the crossbar must be at least as large as the comparable bus or ring, but each IPP and OPP need not interface directly at the full system bandwidth, allowing the overall system cost to be significantly reduced.

The speed advantage required in a crossbar system depends on the nature of the traffic and the desired queuing behavior. If the queuing is to be done primarily at the outputs, the speed advantage must be large enough so that contention for outputs, caused by the natural random timing of cell arrivals, does not cause too many cells to accumulate at input ports. For bursty traffic with peak rates that are small

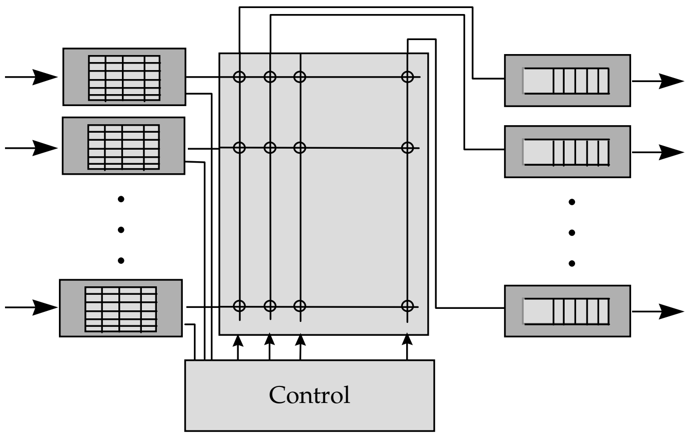

Figure 4. Crossbar-based Switch

relative to the link rates, a modest speed advantage (say 2) is sufficient. For larger peak rates, the speed advantage may need to be increased to prevent significant queuing at the input ports. References [GEN94, DOI92, YAM94] describe crossbar-based switches that use a speed advantage to reduce or eliminate output contention. The *knockout switch* described in [YEH87] also addresses output contention by increasing the bandwidth of the data path from the crossbar to the OPPs. In this case however, there is no central controller. Instead, there is a knockout concentrator for each output, which accepts cells from any of the inputs and concentrates them onto a smaller number of outputs (typically 8), which in turn feed into the output buffer in the OPP.

Some switch designers seek to avoid the requirement for an output speedup by providing cell buffers at each crosspoint of the crossbar. If sufficient buffering is provided in the crossbar, head-of-line blocking can be avoided in the IPPs and the buffers storing cells for any particular output compete for that output's bandwidth, independently of all other outputs. Examples of such designs can be found in [NOJ87, OKI97, TOM93, TOM95, TOM95, TOM92]. Because this approach dramatically increases the complexity of the crosspoints, the number of which grows quadratically with the number of switch ports, it becomes prohibitively expensive (in comparison to alternative designs) for all but the smallest configurations.

Another way to avoid the need for a speed advantage in the crossbar is to queue at the input ports instead of at the outputs. However, to achieve efficient system operation, one cannot use a single fifo input queue, since this leads to head-of-line blocking. There are several alternatives to a single input queue. The most effective is to maintain a separate linked-list queue for each output port, or each virtual circuit. Another benefit of input buffering is that it does allow a certain amount of buffer sharing, reducing the memory requirements, relative to a system using output queuing or crosspoint queueing. Examples of designs of this sort can be found in [LOC95, KAT96a, KAT96b, MCK96a, MCK96b, SCH91].

The control of crossbar systems is significantly more complex than for bus or ring systems. Typically, IPPs signal to a central crossbar controller, which output or outputs they have cells for, and the controller matches inputs to outputs so as to maximize the number of cells that can be switched in the upcoming data transfer cycle, configures the crosspoints appropriately and informs each selected IPP of which output it has been connected to. The IPPs may include a priority for each requested output (typically these are dynamic priorities that depend on the number of cells that each IPP has waiting for the requested output). In this case, the controller seeks to maximize the sum of the priorities of the cells selected for transmission. The problem that must be solved by the controller is a maximum matching problem. While exact algorithms for both the weighted and unweighted versions of the problem are known, they are computationally too expensive to use in most switching applications. Consequently, various approximate matching algorithms are generally substituted. While these produce good results in typical situations, they can produce matchings that are significantly sub-optimal in the worst-case.

2.6. Multicasting in Single Stage Switches

An important feature of ATM networks is their ability to support multicast virtual circuits for communication among groups of arbitrary size. A simple one-to-many multicast enables data transmission from a source to many receivers. Many-to-many multicasts are also possible, although not currently supported by standards. For switches with a moderate number of ports, multicast can be implemented by including a *bit vector* in the routing tables in the IPP, with one bit for each output port. When an IPP transfers a cell on the bus or ring, it includes the bit vector with the cell. OPP_i, upon receiving the cell, checks bit *i* of the bit vector and copies the cell to its output queue if the bit is set. Otherwise it ignores the cell. This approach, while very simple, has the drawback that it results in every copy of the cell having the same VPI/VCI combination. This complicates the switch control software considerably and can lead to call blocking due to clashes on the VPI/VCI values. This can be corrected by adding routing tables to each of the output ports, containing new VPI/VCI values for the outgoing cells. Typically, these tables are indexed using a *Multicast Identifier* obtained from the IPP routing table. In this case, each output routing table contains an entry for every multicast identifier, including those for which it does not actually forward cells. This leads to inefficient use of memory (potentially, this approach uses nearly *n* times the memory required for point-to-point switching), which can be avoided by using a *Content Addressable Memory* (CAM) instead of a simple lookup table.

In shared memory switches, arriving multicast cells must be included in the linked list output queues for all the output links on which the cell is to be sent. When a cell arrives at an IPP, it is transferred on the bus or ring to the shared buffer, copied into memory and then a pointer to the stored cell is included in the linked list for each of the outputs which is to get a copy. The translation from multicast identifier to outgoing VPI/VCI can be done when the cell is read from each output queue on which it is placed. Again, this can be done with either a simple lookup table or a CAM.

For crossbar switches, multicast forwarding raises some more difficult issues. First, the crossbar controller must be able to accept requests to send a single cell to an arbitrary subset of the outputs rather than to just a single output. This requires that an IPP with a multicast cell to send provide a bit vector indicating which outputs it needs. If two contending multicast cells have some but not all outputs in common, efficient system operation requires immediate forwarding of as many copies as can be sent, without waiting for those that must be held back due to the conflict. This in turn, means that IPPs must keep track of which outputs a given multicast cell has been forwarded to, so that subsequent transmission attempts target only the ones that have yet to receive a copy. In systems using input queuing, queue management for multicast connections can be complex, if one seeks to prevent a busy output port from slowing the transmission of copies of multicast cells to less busy ports.

2.7. Scaling Up Using Port Multiplexing

One way to build switching systems handling larger numbers of external links is to multiplex multiple external links onto a single higher speed link and then use a single stage switch whose ports operate at the higher speed. This approach is used in a number of ATM switches, which are designed for external links of 150 Mb/s but which internally use single stage switches capable of data rates of 600 Mb/s. One example is the Fore Systems ASX 200 [FOR96]. For any given circuit technology, there is a limit to the applicability of this technique, but within its range of applicability it offers the most cost-effective way to scale up to larger sizes.

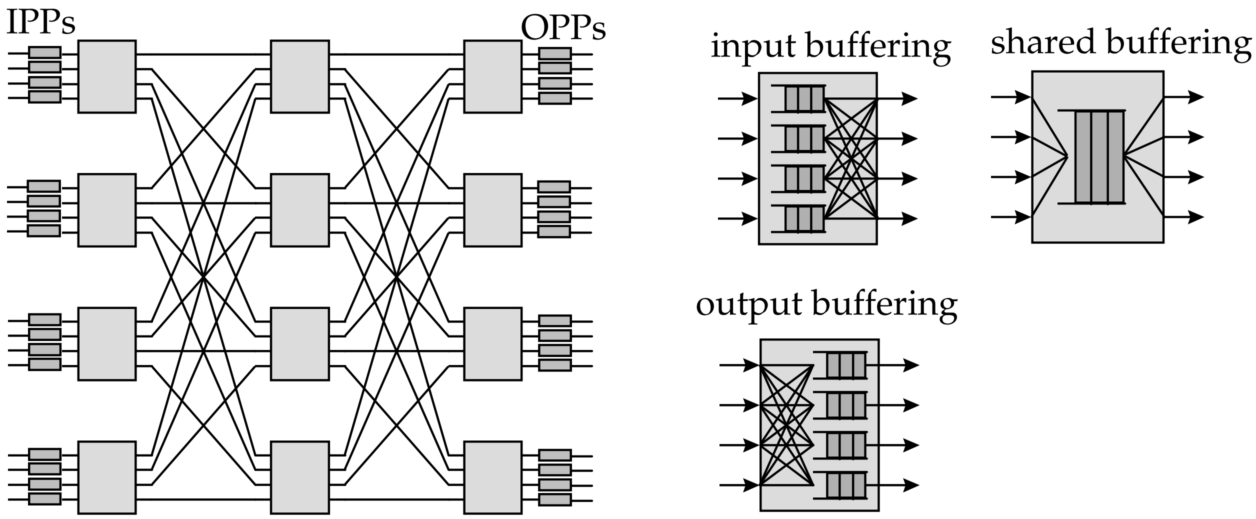

Figure 5. Buffered Multistage Networks

3. Buffered Multistage Switches

Single stage switching techniques are inherently limited by their quadratic complexity growth. While the use of a higher speed core to handle lower speed ports provides some relief, for any given technology, there comes a point where greater amounts of parallelism are needed to obtain higher throughput. For systems implemented using modern CMOS integrated circuits, buffered multistage networks are among the most attractive. An example of a buffered multistage network is shown in Figure 5. Each of the *switch elements* used to construct the network can store some (usually modest) number of cells in its internal buffers. Queues can be located on the input side, output side or can be shared. There are two main classes of systems; those that use *dynamic routing* and those that use *static routing*. In systems that use dynamic routing, each cell is routed independently, spreading the traffic as evenly as possible among all the different paths between any given pair of input and output ports. In static routing systems, all cells in a given virtual circuit follow the same path through the multistage network. Static routing networks maintain cell ordering directly, but require explicit path selection and are far more subject to virtual circuit blocking. Dynamic routing networks require the use of some other mechanism to restore cell ordering, after cells emerge from the interconnection network. Typically this involves some form of resequencing buffer.

3.1. Dynamic Routing

Interconnection networks based on dynamic routing are among the most cost-effective alternatives for large switches. Perhaps the most popular topology for dynamic routing networks is the Benes network, $B_{n,d}$ (the subscript n is the number of input and output ports that the interconnection network has, the subscript d is the number of inputs and outputs of the switch elements used to construct the network). The Benes network is constructed recursively. For n=d, it consists of a single d input, d output switch element. For n=d^2, it has three stages, with d switch elements in each stage and a single link connecting each pair of switches in consecutive stages (the network in Figure 5 is a Benes network with n=16 and d=4). When used to implement an ATM interconnection network with dynamic routing, the first stage distributes arriving cells across all switches in the middle stage to balance the load. The second and third stages route cells to the output, using an output port number inserted in the cell header by the input port processor. This output port number is interpreted as a pair of base d digits, reducing the routing process to a straightforward selection of one of *d* outputs in each of the last two stages. The Benes network topology generalizes naturally to five, seven or more stages. To construct a network with $n=d^k$, one first constructs *d* networks with $n=d^{k-1}$; these subnetworks form the central stages of the network being constructed. The central subnetworks are then preceded by a first stage with d^{k-1} switch elements, each of which is linked to all of the central subnetworks. The central networks are similarly followed by a last stage with d^{k-1} switch elements, each of which is linked to all of the central subnetworks. The resulting network has $2k$-1 stages, altogether. The first k-1 of these stages distribute cells to balance the overall system load, while the last *k* route cells using successive base *d* digits of the cells' output port numbers.

The Benes network is particularly attractive for dynamic routing networks because under dynamic routing, the average load on its internal links is guaranteed to be no larger than the maximum load on its inputs or outputs. This means that excellent performance can be obtained with internal link bandwidths that are not be substantially larger than those of the external links. As with a single stage crossbar, some speed advantage may be needed to handle bursty traffic with high peak data rates. Because the traffic distribution reduces the effect of high peak rates on the Benes network's internal links, it is usually sufficient to provide this speed advantage only at the very last stage of the network, on the data paths joining the last stage switch elements with the OPPs.

One drawback of the Benes network is that it must be expanded in fairly large increments (factors of *d*). When *d* is larger than 2, this can be a substantial constraint. Fortunately, the topology can be easily generalized to allow expansion in smaller steps. Moreover, with some care in the design of the switch elements, these networks can be designed to allow expansion while the system is in operation.

One issue with networks that use dynamic routing is that cells can get out of order as they pass through the system. This requires the addition of a *resequencing buffer* to each OPP, to restore the correct cell ordering. While a variety of resequencer designs are possible, the simplest approach is to use time-based resequencing in which cells are time-stamped when they enter the interconnection network and the OPPs use the time stamp information to determine the time cells have spent in the interconnection network, allowing them to reorder the arriving cells back into the correct sequence. Cells that pass through the network quickly are delayed in the resequencing buffer in order to give "slower" cells an opportunity to catch up. This means that the resequencing buffer does introduce some added delay, increasing the minimum latency of the system, but simultaneously reducing the cell delay variation. For systems with up to about 1000 ports, a resequencing buffer of between 50 and 100 cells can be sufficient to reduce the probability of out-of-sequence cells to very low levels.

Systems using buffered multistage networks with dynamic routing have been built by Alcatel [DEP87a, DEP87b, DEP90, HEN90, HEN92], CSELT [BOS96, COL94, CLA94, LIC96] and Washington University [CHA97, TUR88, TUR94].

3.2. Static Routing

Static routing networks operate similarly to dynamic routing networks, except that all cells in a given virtual circuit are constrained to follow the same path. The routing lookup performed in IPPs of static routing networks, inserts a *path specification* into the cell headers of arriving cells, rather than just output port numbers. The successive digits of a cell's path specification are used to select output ports in the sequence of switch elements that the cell passes through. A variety of interconnection topologies can be used, including the Benes network. One of the most popular alternatives is a three stage *Clos network* which is similar to the three stage Benes network but uses asymmetric switch elements in the first and third stages. In particular, the switch elements in the first stage have *d* inputs and *r* outputs, while those in the third stage have r inputs and d outputs. The r middle stage switches have an equal number of inputs and outputs, typically d, but not necessarily.

When a new virtual circuit is to be added from some input port x to some output port y of a static routing network, the control processor managing the switching system must find some path through the network with sufficient unused bandwidth on each of its links to accommodate the new virtual circuit. For the three stage Clos network, this can require checking r pairs of links, to determine if they can accommodate the addition of the new virtual circuit. If none of the potential paths has enough unused capacity, the virtual circuit request is *blocked*.

If one abstracts the bandwidth requirements of a virtual circuit as a single scalar value, one can derive conditions under which a network is *nonblocking*. For the three stage Clos network, blocking can be avoided if the following inequality is satisfied.

$$r > 2\left\lceil \frac{\beta d - B}{1 - B} \right\rceil - 2$$

In this expression, β is the ratio of the bandwidth of the external links to the bandwidth of the internal data paths (the reciprocal of the speed advantage) and B is the ratio of the maximum rate of any single virtual circuit to the bandwidth of the internal data paths. (For example, a system with 2.4 Gb/s external interfaces and internal data paths of 4.8 Gb/s will have $\beta=1/2$. If the users whose virtual circuits are routed through this switching system all have access links that operate at 150 Mb/s, then B=1/32.) If β and B are both ½, the number of middle stage switches needed to avoid blocking is 2d-3.

Similar results can be obtained for networks with more than three stages. In particular, a Benes network is nonblocking for static routing if

$$(1/\beta) \geq (1 - 2/d)(B/\beta) + (2/d)(1 + (d-1)(k-1))$$

where $k = \log_d n$ and n is the number of ports. In general, the speed advantage needed to make the Benes network nonblocking is between k and 2k-1 and approaches (B/b)+2k-2 for larger values of d. This contrasts sharply with the case of dynamic routing which requires no speed advantage to make the network nonblocking, although, as discussed in the next sub-section, some speed advantage (generally much smaller) is needed for good queuing performance in both cases. In systems that are already operating at the maximum clock rate that the given technology can support, increasing speed advantage must be obtained by increasing the amount of parallelism in the system (typically by increasing data path widths). This leads to a proportional increase in cost, making it advantageous to keep the required speed advantage as low as possible.

CNET [DEV88] designed and implemented the first ATM switch using static routing. Similar systems were later produced by NEC [SUZ89], Fujitsu [YOS96], Hitachi [KOZ91, KUW89] and IBM [DEN91]. Virtual circuit blocking analyses for this class of systems can be found in [MEL89, MEL93].

3.3. Queuing Performance of Buffered Multistage Switches

The queuing performance of buffered multistage interconnection networks is the second key performance issue for this class of systems. The design parameters that have the most impact on performance are the speed advantage, the dimensions of the switch elements, their queue organization and queue size and the presence or absence of inter-stage flow control. In general, large switch elements with shared queues and inter-stage flow control give the best performance for a given total amount of buffering. For uniform random traffic, a system with shared buffer switch elements with eight or more inputs and outputs and four buffer slots per input and output, can handle fully loaded input and output links if the interconnection network has a speed advantage of about 1.25 or more.

There has been only limited study of the performance of this class of systems for time-dependent traffic, and the models used for these analyses are not completely realistic. However, there are some general observations that can be made. Systems using a Benes network with dynamic routing can generally handle any time-dependent traffic for which the instantaneous load does not exceed the bandwidth of the interfaces between the last stage switch elements and the OPPs. Consequently, if one can ensure that such overloads occur very rarely or not at all, one can be assured of excellent queuing performance with only a small speed advantage. Determining the probability than an interface is overloaded at any random instant is easy to determine for a wide variety of multiplexed data streams with time-dependent behavior, using direct numerical convolution of the individual sources' rate distributions. In general, overload probabilities are small, even for average external link loads close to 1, when the ratio of the peak bandwidths of individual virtual circuits is small relative to the interface rate. Where this condition is not met, one must increase the speed advantage at the output interface to reduce the overload probability to acceptably small levels. However, it is not necessary to increase the speed advantage throughout the interconnection network. One can eliminate the possibility of overloading the interface between the last stage and the OPP by implementing per virtual circuit flow control mechanisms between the OPPs and the IPPs. This adds significant complexity to the IPP and OPP, although the memory needed for the flow control state information and IPP buffering need not be impractically large in absolute terms.

For networks that use static routing, the queuing performance in the presence of time-dependent traffic is somewhat more problematical. In these systems, congestion can occur on any inter-stage link in the interconnection network. There are essentially three ways to address the congestion issue. First, one can ensure that it occurs rarely by limiting both the average total traffic on any link and/or the ratio of the link rate to the peak rates of individual virtual circuits. The second way to address congestion is to provide sufficient buffering at each switch element to handle typical data bursts. For virtual circuits whose peak rates are large, the buffer sizes must be at least comparable to the size of average length bursts. For large buffers, one must have separate queues for different classes of traffic, in order to prevent real-time traffic from being excessively delayed by bursty data traffic. The third approach to controlling congestion in multistage networks using static routing is to employ more sophisticated flow control mechanisms between stages. The most effective approach is to have a separate queue for each virtual circuit that passes through the given switch element. In this approach, a switch element with a congested output link signals to its upstream neighbors, blocking new cell transmissions for all virtual circuits using the congested link. These signals can be propagated backward through successive stages of the network to the IPPs, where larger buffers are typically available for storing bursts.

References [BIA93, COL96, GIA94, JEN83, LUC94, PAT94, SZY89, TUR93b] provide a sampling of the literature on the queuing performance of buffered multistage networks.

3.4. Multicasting in Buffered Multistage Switches

Multistage networks using dynamic or static routing can be extended in a fairly direct way to support multicast communication. For systems that are not too large, or that involve only small multicast connections, the set of all outputs for a cell can be encoded and placed in the cell header at an IPP and used by switch elements to guide the process of routing and copying cells to the required outputs. In general, multicasts for any collection of output sets that can be conveniently encoded using a modest number of bits can be handled in this way. For greatest flexibility, systems need to be capable of labeling different output copies of a given multicast cell with distinct VPI/VCI values. This requires that IPPs insert Multicast Identifiers (MI) into cell headers when cells belonging to multicast virtual circuits arrive at input ports. These are used by OPPs to select the proper outgoing VPI/VCI values from multicast routing tables.

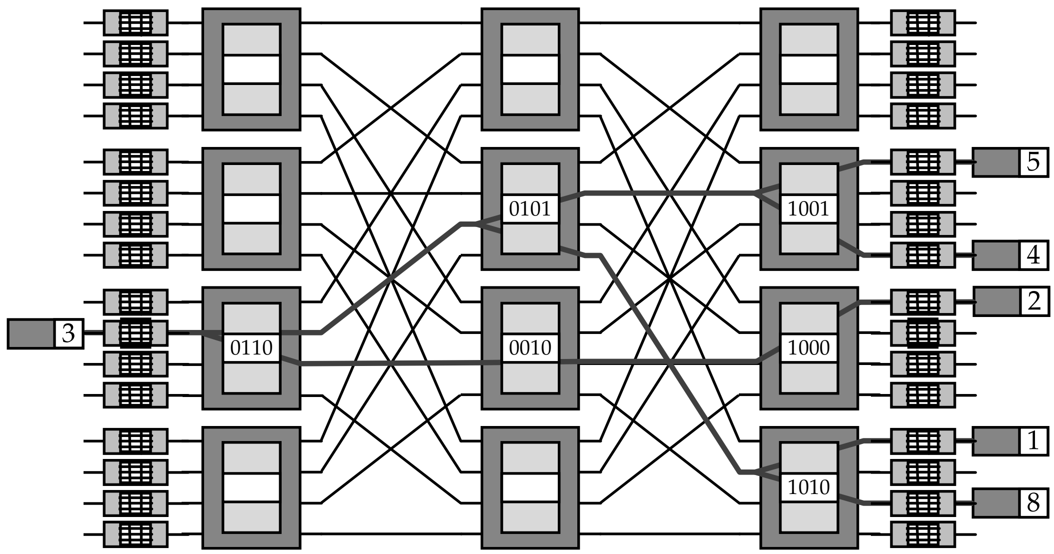

Figure 6. Multicasting in Buffered Multistage Networks

For large systems that must be able to support arbitrary multicasts, a more general technique is needed. For dynamic networks, one method is to augment switch elements with multicast routing tables, indexed by the Multicast Identifier (MI). Switch elements use MIs to select routing table entries that specify the set of switch element outputs an arriving cell should be forwarded to. If a Benes interconnection network is used, routing tables are needed in the last k stages of a $2k$-1 stage network. The first k-1 stages need only do dynamic load balancing, as in the case of point-to-point cells. Because dynamic routing networks can route cells belonging to a given multicast virtual circuit through any part of the interconnection network, the routing information must be replicated multiple times. This redundancy increases the routing table memory requirements significantly and limits the number of distinct multicast virtual circuits that it is practical to implement using this approach. The system developed by Alcatel [DEP87a, DEP87b, DEP90, HEN90, HEN92] uses this method.

Similar techniques can be used for static routing networks, as shown in Figure 6. In this case routing tables are needed at every stage of a multistage network. As in the case of dynamic routing networks, an MI can be used to determine the set of output ports a given multicast cell should be forwarded to. Alternatively, the routing table entries may be selected using arriving cells' VPI/VCI values, and the entries can specify both the set of outputs to receive copies and the VPI/VCI values to be assigned to the outgoing copies on each of that switch element's outgoing links. This latter approach can require fewer total routing table entries in large configurations but is somewhat more complex to manage.

For static routing networks, the introduction of multicast can have a large impact on virtual circuit blocking performance. The best possible blocking performance is obtained in these networks when the branching required for multicast connections occurs as close as possible to the OPPs. This minimizes the use of interconnection bandwidth by multicast connections. Unfortunately, under worst-case conditions, endpoint additions and deletions can lead to a situation in which many multicast virtual circuits branch at early stages, making it more difficult to add new connections or augment existing multicast connections. To prevent blocking, one must increase the interconnection network's speed advantage by a large factor. For the Clos network, the following inequality must be satisfied to make the network nonblocking.

$$r > \left\lceil \min\{r, n/d\} \frac{\beta d - B}{1 - B} \right\rceil + \left\lceil \frac{\beta d - B}{1 - B} \right\rceil - 2$$

A speed advantage of $(1/d)\left(n + (1 + (d-1)(k-1)) - (B/\beta)\right)$ is needed to make the Benes network nonblocking. In some situations, lower costs can be realized by sacrificing the strict nonblocking property. Networks that are *reroutably nonblocking* allow any new virtual circuit to be setup without rearranging any existing ones, but may require rerouting a given multicast virtual circuit when adding a new endpoint to it. Networks satisfying this property require much smaller speed advantages than strictly nonblocking networks and may be acceptable in many practical situations.

For both static and dynamic routing networks, the amount of memory needed for multicast routing can become excessive, particularly in larger system configurations. For this reason, alternative approaches have been developed that can reduce the total amount of routing memory required and simplify the operational aspects of the system. One class of approaches uses a two part interconnection network in which the first part (the *copy network*) produces copies of multicast cells, while the latter part (the *routing network*) routes the copies to the desired outputs. Multicast routing tables are positioned between these two sections, but are not needed at every switch element. See [TUR88] for an example of a system of this type. One way to implement this class of systems is to have IPPs insert an MI and a pair of copy network output numbers in the headers of cells belonging to multicast virtual circuits. The copy network uses the pair of specified output numbers to control the copy process, delivering copies to all output ports in the range bounded by the specified pair of outputs. When cells arrive at the multicast routing table between the copy and routing networks, the MI is used to select a table entry specifying the OPP that a particular copy is to be forwarded to and the VPI/VCI to be included with the cell when it is forwarded on the outgoing link. In the case of dynamic routing networks, the copies of different multicast cells in the same virtual circuit can be sent to different sets of copy network outputs in order to balance the load across all the copy network outputs. Load balancing makes it possible to build systems with a smaller internal speed advantage than would otherwise be required, but comes at the expense of larger numbers of multicast routing table entries than would be needed without load balancing. One can trade off these two factors to provide the best combination for a given system configuration.

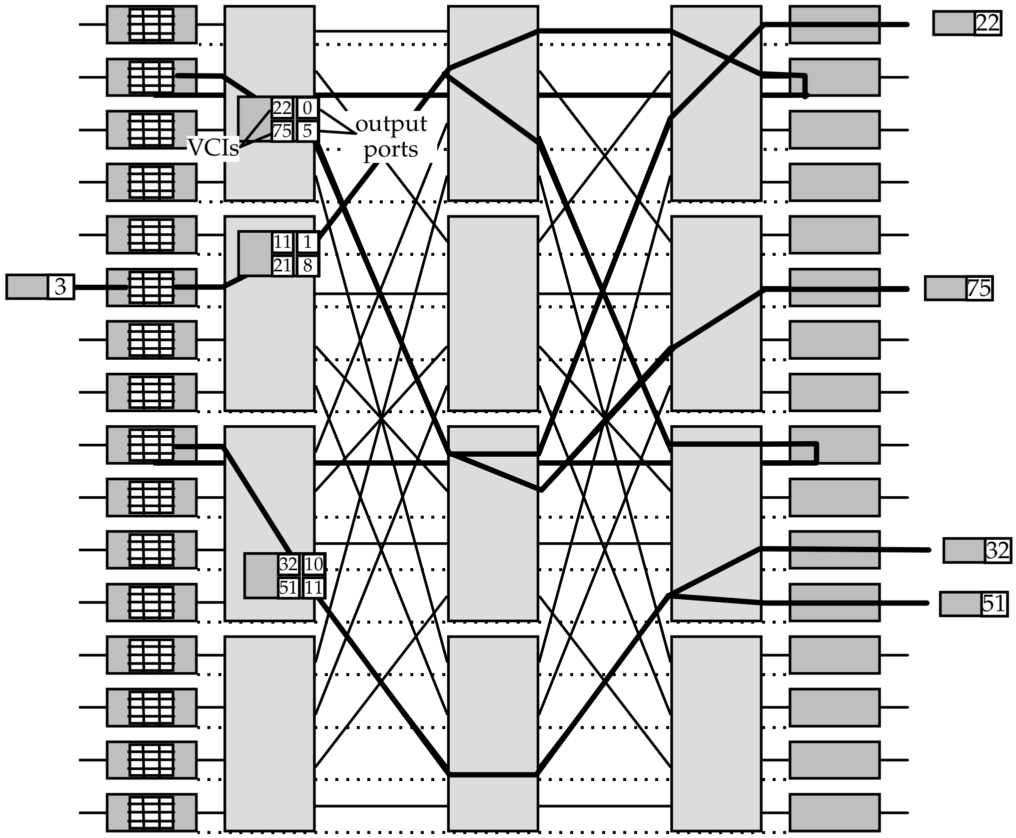

Figure 7. Multicasting Using Cell Recycling

In some situations, one can reduce the cost of the system by combining the functions of the copy and routing networks into a single network that is capable of doing both functions. In such a system, cells belonging to point-to-point virtual circuits are routed through the network to the desired output in a single pass, exactly as in a system supporting only point-to-point virtual circuits. Cells belonging to multicast virtual circuits are handled in two passes through the network. The required number of copies is produced on the first pass, and then the copies are sent through the network again, with each copy forwarded to its required output link. This requires a *recycling data path* between each OPP and its corresponding IPP. Multicast cells arriving at an OPP are passed along the recycling path to the IPP, which includes a multicast routing table that specifies the output port and VPI/VCI to be used for a particular copy. This multicast routing table can be combined with the virtual path/circuit translation table that the IPP uses for cells arriving on its external link.

The most cost effective system architectures for large-scale multicasting extend this two pass processing to multiple passes, with a bounded number of copies being made on each pass. In the case of *binary copying*, two copies of an original multicast cell are produced in one pass through the network. When cells belonging to a multicast virtual circuit arrive at an IPP from the external link, the IPP table lookup produces two output port numbers and two VPI/VCI fields. These are inserted in the cell header, as the cell is sent to the interconnection network. The interconnection network (which can use either static or dynamic routing), delivers a copy to each of the specified outputs and then each copy is either forwarded on its external link or passed along the recycling path to the IPP, where it passes through another lookup table, acquiring two new output port numbers and VPI/VCI fields. If both copies made in the first pass are recycled to their corresponding IPPs, then they can each produce two copies in the second pass, leading to four copies of the original cell, after two passes through the network. If all four of these copies are recycled, eight copies are produced on the third pass and in general F copies can be produced in $\log_2 F$ passes. The combination of binary copying, cell recycling and a dynamic routing Benes network yields a system with optimal scaling characteristics. That is, the complexity of the interconnection network grows in proportion to $n\log n$ and the number of routing table entries required for multicast is no larger than is required for purely point-to-point virtual circuits (although the entries are somewhat larger). This technique for multicasting is used in the system described in references [CHA97, TUR94].

4. Non-Buffered Multistage Switches

The cell buffers used to resolve contention in buffered multistage networks contribute significantly to their cost, motivating different approaches that attempt to resolve contention in other ways. For ATM, these approaches have not proved to be competitive with the buffered multistage networks, when both are implemented with a high density circuit technology like CMOS. This is because while systems in this class generally require less circuit area for logic devices, they also require either substantially higher bandwidths or increased interconnection wiring, both within and between chips. However they could become competitive in the future, in the context of either higher speed, lower density circuit technologies or larger cell sizes that increase the relative cost of buffered multistage networks.

4.1. Contend and Repeat

The simplest class of non-buffered multistage switches uses a simple “contend-and-repeat” strategy. As an example, consider a system with a $2k$-1 stage Benes network topology, in which the individual switch elements include a crossbar and crossbar controller. In the first k-1 stages, the crossbar controllers simply configure the crossbar randomly so that arriving cells are distributed evenly across the whole network. In the last k stages, cells are routed to the output specified in their cell headers. If two cells arriving at a particular switch element contend for the same output, one of them is forwarded to the requested output, while the other is simply discarded. Cells are typically delayed by just a few clock ticks at each stage, allowing the first part of a cell to emerge from the interconnection network while the last part is still being sent by the IPP. When a cell arrives at its destination OPP, the OPP returns an *acknowledgement* signal which flows back through the network along the path taken by the cell, but in the reverse direction, using a circuit path set up as a side effect of the cell routing process. If an IPP fails to receive an acknowledgement signal when expected, it concludes that the cell was discarded and tries again on the next operational cycle. In order to handle fully loaded external links, such systems must be engineered with a substantial speed advantage relative to the external links. For typical configurations, the required speed advantage is in the range of three to five. While this can represent a significant cost factor, it can be acceptable in the context of higher speed circuit technologies. Newman describes a system of this type in [NEW88].

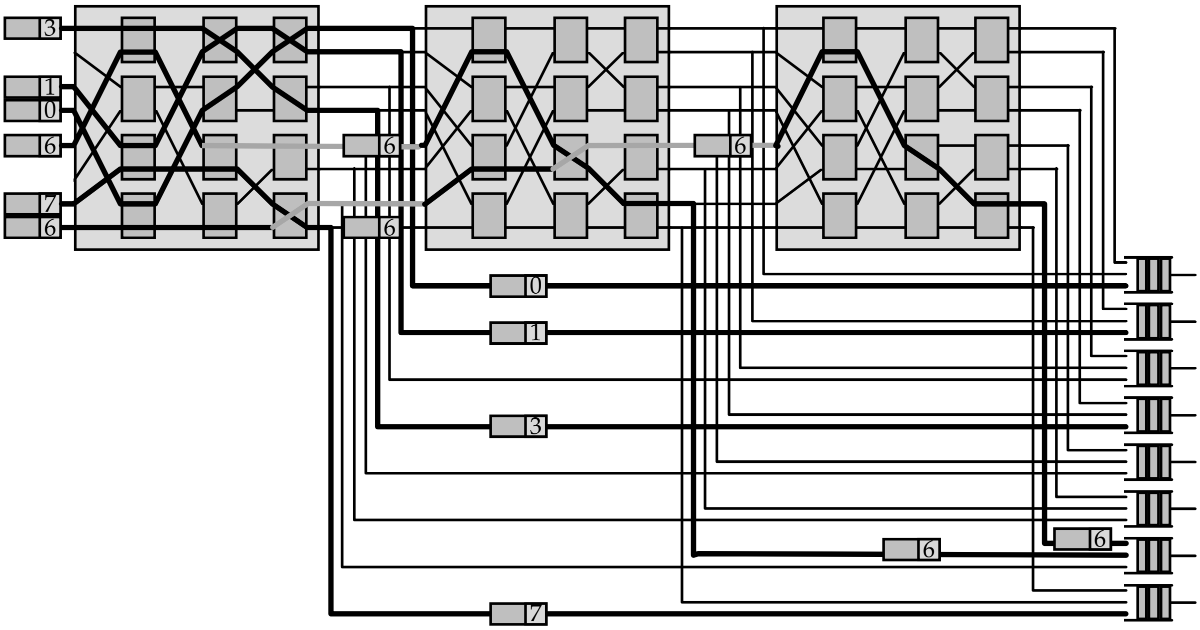

Figure 8. Tandem Banyan Network

There are many variations on the basic approach described above. All require some mechanism for increasing the overall system bandwidth. In some cases, this is done by linking switch elements with multiple parallel links, rather than a single higher speed link (networks of this sort are called *dilated* networks). In other cases, parallel switch planes are used, with input traffic being distributed over the different parallel switch planes. While these alternatives may offer advantages in particular situations, none is clearly superior to the simpler approach described above.

4.2. Deflection Routing

In systems using the contend-and-repeat paradigm, only one of multiple competing cells is propagated beyond a contention point. Deflection routing networks allow both “winning” and “losing” cells to propagate past a contention point, making it possible for the losers to reach the desired output by an alternative path. A simple example of a network of this type is the *tandem banyan* network, illustrated in Figure 8. In this system, several banyan networks are cascaded in series and cells can exit to the desired OPP at the output of any one of the component networks. When a cell loses a contention in any of the banyan networks, a *marker bit* in its header is set and subsequent switch elements pass it on, using any available output. A cell arriving at the output of a banyan network with its marker bit cleared is passed to the OPP at that point. Cells with the marker bit set are propagated to the next banyan network, with the marker bit is cleared. As cells propagate through the series of banyan networks, the traffic drops, increasing each cell’s chance of successfully reaching the required OPP. Of course, even if a large number of banyan networks are used, there remains some non-zero probability that a cell will lose a contention in the last banyan network. If this probability is low enough, it may be acceptable to drop the cell at that point. Alternatively, it can be recirculated back to the input of the last network, and retried at this point.

The tandem banyan network is described in [TOB91]. Another deflection routing architecture, called Shuffleout is described in [DEC91a, DEC91b].

4.3. Sorting Networks

Sorting networks can be used to implement ATM switches that are contention-free, so long as the arriving cells are all addressed to distinct outputs. The most popular sorting network is *Batcher's bitonic sorter* [BAT69], an example of which is illustrated in Figure 9. In this figure, cells are indicated by the small boxes with numbers in them, and the number represents the output address for the given cell. The circles in the figure represent *sorting elements*, which propagate arriving cells to the output, based on the relative values of their output addresses; some sorting elements send cells with larger values to the upper output, others to the lower output, as indicated by the shading in the figure. The sorting network is constructed using a double recursive construction. Some of the components of this construction are highlighted in the figure.

If one happens to have a set of input cells with exactly one cell addressed to each output, the sorting network will deliver each cell to the right output. To handle arbitrary sets of input cells in which some outputs may have zero or more than one cell addressed to them, additional network components are needed. The simplest scalable approach is follow the sorting network by a *filter* block and a *routing network*. The filter block simply compares the output port number of a cell arriving on input i to that of a cell arriving on input i-1 and propagates the cell on input i to output i, only if the compared port numbers are unequal. When this filter is applied to a sequence of cells that have been sorted by output port number, the result is to propagate at most one cell for each output port. The filters can acknowledge propagated cells by sending a signal back along the reverse path taken through the sorting network. IPPs retry cells that are not acknowledged, during subsequent operational cycles. Cells that pass the filter then go through to the routing network. The most commonly used routing network is built around a banyan network. Banyan networks which have the property that a sorted sequence of cells on consecutive inputs can be routed to the outputs without any internal data path contention. A routing network can be constructed by preceding a banyan network with a *concentrator* that places cells on consecutive outputs without changing their relative order.

A system comprising a sorting network, filters and routing network provides a general and scalable switching system. However, the queueing performance can be inadequate to handle fully loaded external links. This limitation can be eliminated by increasing the output bandwidth of the system. One way to do this is to operate the system at a higher internal bandwidth than the external links; for most realistic traffic configurations a two-to-one speed advantage is sufficient. Another approach is to have more than one link from the routing network to each OPP, so that OPPs can receive multiple cells in parallel. For a system that allows r cells to be forwarded to any output in a single cycle, the filter block is modified to propagate up to r cells per output and following the concentrator with r "interleaved" banayan networks.

The major drawback of this class of systems is that they require substantially more interconnection wiring between individual integrated circuits and circuit modules than is needed for buffered multistage networks. This is a consequence of the fact that practical sorting networks require $(1/2)(\log_2 n)(1+\log_2 n)$ stages of sorting elements. Due to pin constraints on integrated circuit packages a larger number of integrated circuits are needed to implement a system with a given number of ports than with alternative architectures. As one example, a 4096 port sorting network implemented with components having 64 inputs and 64 outputs, requires 13 ranks of components, with 64 components per rank. Seven additional ranks of components are required for the filter and routing network. For comparison, a 4096 port Benes network with 64 port switch elements using internal buffering to resolve contention requires just 3 columns of components.

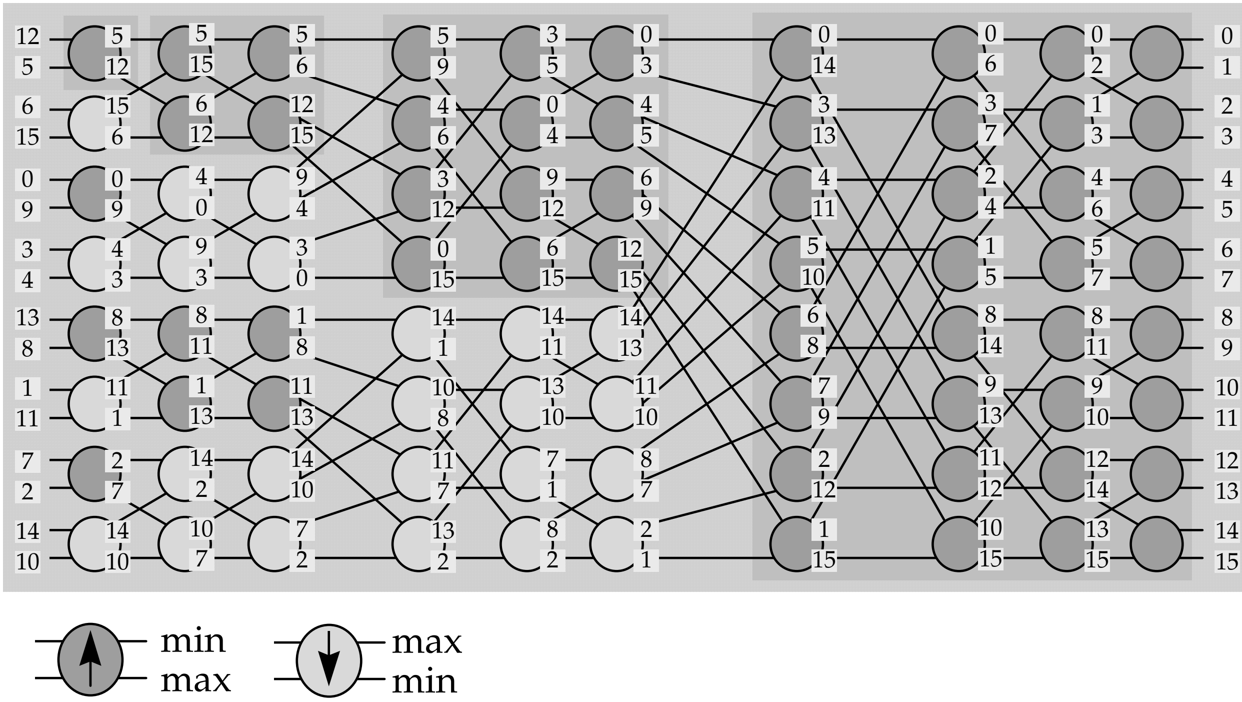

Figure 9. Bitonic Sorting Network

The first ATM switch design based on the use of sorting networks was the Starlite switch [HUA84]. These ideas were developed further in the Sunshine switch, described in [DAY87, GIA91, HAY87a, HAY87b, HUI87]. A system based on a similar approach is described in [MIN95].

Eng et. al. [ENG91, ENG92a, ENG92b, ENG95] have developed a hybrid architecture which combines an unbuffered multistage network with buffered switch elements in the last stage. These buffered switch elements have k inputs and $d<k$ outputs (typically k=32, d=16) and a large shared buffer. The unbuffered switching network preceding this rank of buffered switch elements performs an arbitration function, discarding excess cells if more than k arriving cells are destined for any group of d outputs. This approach is feasible when the memory bandwidth of the shared buffer switch elements is high enough to support d simultaneous output ports, but under these conditions, port multiplexing is also an option and can be a more cost-effective approach. The authors suggest a variety of choices for the unbuffered network, including the use of a sorting network and a three stage Clos network (in which the last stage is replaced by the buffered switch elements). Both of these choices have suboptimal scaling characteristics, weakening the authors' assertion of "growability."

4.4. Multicasting in Non-Buffered Multistage Switches

Multicasting can be implemented in non-buffered multistage networks, by preceding a point-to-point network with a copy network, capable of producing the requesting number of copies of any given input cell. One example of such a system is shown in Figure 10. A *fanout* field is inserted into cell headers by the IPPs before sending cells to the network. The first two components of the network implement a concentration function; the first *adder* block computes the *rank* of each arriving cell (essentially just numbering cells consecutively) and the *concentrator* block uses the rank to route cells to consecutive outputs. The second adder block sums the fanout fields of cells; the computed *fanout sum* of a departing cell on output i equals the sum of all fanouts of cells on outputs 0 through i-1. Using the fanout sum and

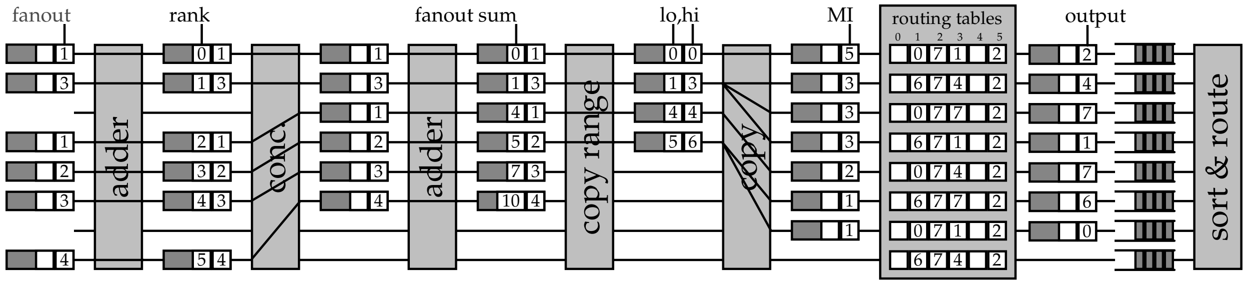

Figure 10. Copy Network

fanout fields, the *copy range* component computes the range of outputs that each cell should be copied to in the succeeding *copy* component. The copy component does the actual copying of cells based on the supplied ranges. The routing tables that follow the copy network specify an output port and outgoing VPI/VCI for each copy of an arriving cell (the VPI/VCI fields are not shown in the example. Each table is indexed by the multicast identifier of the arriving cell. The routing tables are followed by a set of cell buffers, which are in turn followed by a point-to-point network. The copy range components in this system filter out cells if the computed copy range extends beyond the number of outputs of the copy component. Acknowledgments are returned to the IPPs along the reverse of the paths followed by the arriving cells and unacknowledged cells are retried on the next operational cycle. The performance of this design can be improved by expanding the number of outputs of the copy network and modifying the copy range block to distribute the load evenly across the outputs of the copy block. The principal disadvantage of this design is that a multicast virtual circuit consumes an entry in every routing table even when it has a small fanout. This leads to excessive memory requirements in the common case of many virtual circuits with small fanout.

This approach to implementing multicast is described in [LEE88] and was later refined in [TUR93a]. Similar approaches are developed in [BIA92, MIN95b].

5. Summary

Cost is a key factor in evaluating architectural alternatives for ATM switching systems. Throughout the paper, we have touched on issues that contribute to overall system costs. We close with a discussion of a simple cost model for ATM switching systems and a comparison of some of the different design choices, based upon this model. The cost of a single stage switch can be modeled by an expression of the form

$$(C_1 + C_2 m + C_3 B)n + C_4 n^2$$

where the C_i are implementation-dependent constants, n is the number of ports on the switch, m is the number of virtual circuits per port and B is the number of cell buffers per port. The last term in the expression is the part that is determined by the interconnection network. For multistage interconnection networks, this term is replaced by another. For a dynamic routing Benes network, the last term would be $C_4 n \log n$, while for a three stage Clos network, it would be $C_4 n^{3/2}$. In each of these cases the constant may differ. The plot in Figure 11 illustrate the influence of the different components on system cost. The constants were chosen to approximate the cost of systems with 2.4 Gb/s ports. The values chosen are somewhat low at the time or writing, but should be a reasonable reflection of actual costs over the next two to three years. However, they should be viewed as illutrative, not authoritative. A value of $500 was

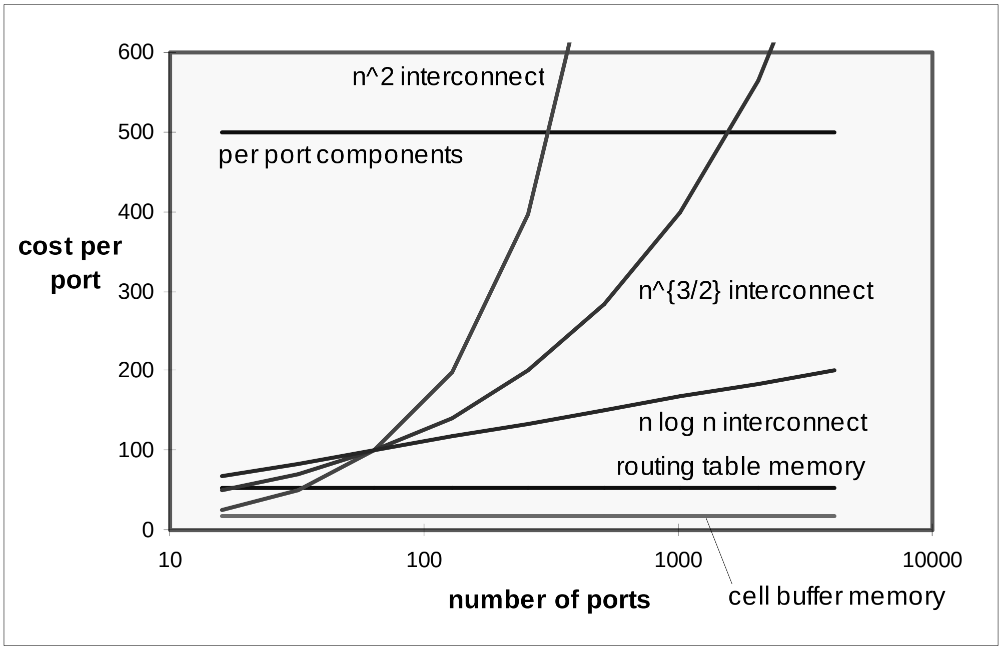

Figure 11. Switching System Cost Components

chosen for the per port component (C_1) of the cost. This captures the cost of optical transceivers, transmission interface circuitry and per port ATM circuits, exclusive of the memory needed for routing tables and cell buffers. The costs for the cell buffers and routing table memory are based on an assumed 4096 routing table entries per port with 20 bytes per entry, 4096 cell buffers per port with 64 bytes per entry and $200 per megabyte of memory (static RAM with 10 ns cycle time). The constants for the interconnect components of the cost were chosen so that all three resulted in a cost of $100 per port in a 64 port system. In most cases, the per port cost component is dominant, but the interconnect cost of the single stage systems becomes important as the systems grow beyond a few hundred ports (note that at 2.4 Gb/s per port, 512 ports are needed for a 1 Tb/s capacity). For the routing table and buffer sizes chosen here, memory cost is not a major component, but if one were to increase the amount of buffering to 32,768 cells (1.6 Mbytes in cell payloads), the memory cost would increase to over $400 per port, making it comparable in cost to the other per port components.

The design and analysis of modern switching systems has a long tradition, extending back to early work by Shannon and Clos. Changes in circuit technologies have opened up new opportunities, enabling the introduction of new services, expanded functionality and vastly greater bandwidths. The pace of change in technology continues unabated and new innovations in switching system design will be needed to keep pace with these changes, in order to realize the promise that the new technologies have to offer. In this paper, we have tried to put recent developments in a common context and provide an intellectual framework in which past and future developments can be related and compared. We hope that it can help future designers learn from the past, while working to create the switching systems that will take us into the future.

References

AHM89 Hamid Ahmadi, Wolfgang E. Denzel, "A Survey of Modern High Performance Switching," *IEEE Journal on Selected. Areas Communications*, vol. 7, no. 7, pp. 1091-1103, Sept. 1989.

BAN91 Thomas R. Banniza, et al., “Design and Technology Aspects of VLSI’s for ATM Switches,” *IEEE Journal on Selected. Areas Communications*, vol. 9, pp. 1255-1264, Oct. 1991.

BAT68 K. Batcher. “Sorting Networks and their Applications,” *Proceedings of the Spring Joint Computer Conferenc*e, pp. 307-314, 1968.

BIA92 Ronald Bianchini, Jr., and Hyong Kim. “Design of a Nonblocking Shared Memory Copy Network for ATM,” *Proceedings of Infocom*, pp. 876-885, 1992.

BIA93 Giuseppe Bianchi, J. S. Turner, “Improved Queueing Analysis of Shared Buffer Switching Networks,” *IEEE/ACM Trans. on Networking*, vol. 1, no. 4, Aug. 1993.

BOS96 B. Bostica, L. Licciardi. “Merging Electronics and Photonics towards Tera bit/s ATM Switching,” *Proceedings of Globecom*,. pp. 943-947, 1996.

BYU94 Byun, Jae W. and Tony T. Lee. “The Design and Analysis of an ATM Multicast Switch with Adaptive Traffic Controller,” *IEEE/ACM Transactions on Networking*, 6/94.

CHA91 H. J. Chao, “A Recursive Modular Terabit/Second ATM Switch,” IEEE J. Sel. Areas in Commun., vol. 9, no. 8, pp. 1161-1172, Oct. 1991.

CHA95a H. J. Chao, Byeong-Seog Choe, “Design and Analysis of a Large-Scale Multicast Output Buffered ATM Switch,” *IEEE/ACM Trans. on Networking*, vol. 3, no. 2, pp. 126-138, April 1995.

CHA95b H. J. Chao, “An ATM Queue Manager Handling Multiple Delay and Loss Priorities,” *IEEE/ACM Trans. on Networking*, vol. 3, no. 6. Dec. 1995.

CHA96 Amit K. Chatterjee, Vijay K. Konangi, “Asymmetric ATM Switch Modules with Imbalanced Traffic,” Infocom’96, pp. 802-809, 1996.

CHA97 Tom Chaney, J. Andrew Fingerhut, Margaret Flucke, J. S. Turner, “Design of a Gigabit ATM Switch”, Infocom 1997.

CHE95 X. Chen, L. E. Moser and P. M. Melliar-Smith, “Flow Control Techniques for Multicasting in Gigabit Networks”

CHO96a Abhijit K. Choudhury, Ellen L. Hahne, “Dynamic Queue Length Thresholds in a Shared Memory ATM Switch,” *Proceedings of Infocom*, 3/96.

CHO96b Gagan L. Choudhury, et al., “Squeezing the Most Out of ATM,” *IEEE Trans. on Communications*, vol. 44, no. 2, Feb. 1996.

COL94 M. Collivignarello et al. A complete set of VLSI circuits for ATM Switching , *Proceedings of Globecom*, 1994, pp. 134-138

COL96 Blair R. Collier, Hyong S. Kim, “Efficient Analysis of Shared Buffer Management Strategies in ATM Networks under Non-Uniform Bursty Traffic,” *Proceedings of Infocom*, 1996.

CLA94 S. Claretto et al., “A Low Power ATM Switching Element for Broadband Applications,” *Proceedings of the International Teletraffic Congress*, 1994, pp. 291-295

COP93 Paolo Coppo, et al., “Optimal Cost/Performance Design of ATM Switches,” *IEEE/ACM Trans. on Networking*, vol. 1, no. 5, pp. 566-575, Oct. 1993.

COR93 G. Corazza, C. Raffaelli, “Performance Evaluation of Input-Buffered Replicated Banyan Networks,” *IEEE Trans. on Commun.*, vol. 41, no. 6, pp. 841-845, June, 1993.

DAD89 G. E. Daddis, H. C. Torng, “A Taxonomy of Broadband Integrated Switching Architectures,” *IEEE Commun. Magazine*, May 1989.

DAY87 C. Day, J. Giacopelli and J. Hickey. “Applications of Self-routing Switches to Data Fiber Optic Networks,” *Proceedings of the International Switching Symposium,* pp. 519-423, 1987.

DEC91a M. Decina, P. Giacomazzi, and A. Pattavina, “Shuffle Interconnection Networks with Deflection Routing for ATM Switching: The Closed-Loop Shuffleout,” Proc. Infocom `91, pp. 1254-1263, April, 1991.

DEC91b M. Decina, P. Giacomazzi, and A. Pattavina, “Shuffle Interconnection Networks with Deflection Routing for ATM Switching: The Open-Loop Shuffleout,” June 1991.

DEN81 W. E. Denzel, A. P. J. Engbersen,I Iliadis. “A Highly Modular Packet Switch for Gb/s Rates,” *Proceedings of the International Switching Symposium*, Oct. 1992.

DEP87a M. De Prycker, M. De Somer, “Performance of a Service Independent Switching Network with Distributed Control,” *IEEE J. on Selected Areas in Commun.*, vol. 5, no. 8, pp. 1293-1301, Oct. 1987.

DEP87b M. De Prycker, J. Bauwens, “A Switching Exchange for an Asynchronous Time Division Based Network,” *Proceedings of the International Communications Conference (ICC)*, June 1987.

DEP90 M. De Prycker, et al., “An ATM Switching Architecture with Intrinsic Multicast Capabilities for the Belgian Broadband Experiment”, *Proceedings of the International Switching Symposium,* Stockholm, May, 1990.

DEV88 Michel Devault, Jean-yves Cochennec, and Michel Servel, “The ‘Prelude’ ATD Experiment: Assessments and Future Prospects,” *IEEE J. Sel. Areas in Commun.*, vol. 6, no. 9, pp. 1528-1537, Dec. 1988.

DOI92 Doi, Y. et al., “A Very High-Speed ATM Switch with Input and Output Buffers,” *Proceedings of the International Switching Symposium*, 1992.

ENG91 K. Eng and M. Kard, The Growable Switch Architecture: A Self-Routing Implementation For Large ATM Applications , *Proceedings of the International Communications Conference*, 1991, pp. 1014-1020.

ENG92a Kai Y. Eng, et al., “A Growable Packet (ATM) Switch Architecture: Design Principles and Applications,” *IEEE Trans. on Commun.*, vol. 40, no. 2, pp. 423-429, Feb. 1992.

ENG92b K. Eng et al., A Prototype Growable 2.5 Gb/s ATM Switch For Broadband Applications, *Journal of High Speed Networks,* Vol. 1, No. 3, 1992, pp. 237-253.

ENG95 K. Eng et al., “Design and Prototype of a Terabit ATM Switch Using a Concentrator-Based Growable Switch Architecture,” *Proceedings of the International Switching Symposium,* 3/95.

FOR97 Fore Systems. “ForeRunner ATM Switches,” http://www.fore.com/products/swtch/index.html, 1997.

FUH93 Steve W. Fuhrmann, “Performance of a Packet Switch with Crossbar Architecture,” *IEEE Trans. on Commun.*, vol. 41, no. 3, March 1993.

GIA91 J. Giacopelli, J. Hickey, W. Marcus, W. Sincoskie and M. Littlewood. “Sunshine: A High-Performance Self-Routing Broadband Packet Switch architecture,” IEEE J. Sel. Areas in Commun., vol. 9, no. 8, pp. 1289-1298, Oct. 1991.

GIA94 Stefano Gianatti, Achille Pattavina, “Performance Analysis of ATM Banyan Networks with Shared Queueing—Part I: Random Offered Traffic,” IEEE/ACM Trans. on Networking, vol. 2, no. 4, pp. 398-410, Aug. 1994.

GEN94 Genda, K., et al., “A 160 Gb/s ATM Switching System using an Internal Speed-up Crossbar Switch,” *Proceedings of Globecom*, 1994, Conf., pp. 123-133.

HAB96 K. Habara, Y. Yamada, A. Misawa, K. Sasayama, M. Tsukada, T. Matsunaga, and K-I Yukimatsu, Demonstration of Frequency-routing Type Photonic ATM Switch (FRONTIERNET) Prototype , 22^{nd} ECOC ‘96, ThC.3.4, 1996.

HAJ88 K. Hajikano, et al., “Asynchronous Transfer Mode Switching Architecture for Broadband ISDN—Multistage Self-Routing Switching (MSSR),” *Proceedings of the International Communications Conference*, 1988.

HAY87a Gary Hayward, et al., “A Broadband ISDN Local Access System Using Emerging Technology Components,” *Proceedings of the International Switching Symposium*, 1987.

HAY87b Gary Hayward, et al., “CMOS VLSI Applications in Broadband Circuit Switching,” *IEEE J. Sel. Areas in Commun.*, pp. 1231-1241, 1987.

HEN90 M. A. Henrion, K. J. Schrodi, et al., “Switching Network Architecture for ATM Based Broadband Communications,” *Proceedings of the International Switching Symposium*, 6/90.

HEN92 M. A. Henrion, G. J. Eilenberger, et al., “Technology, Distributed Control and Performance of a Multipath Self-Routing Switch,” *Proceedings of the International Switching Symposium*, 10/92.

HUI87 Joseph Y. Hui, Edward Arthers, “A Broadband Packet Switch for Integrated Transport,” *IEEE J. Sel. Areas in Commun.*, vol 5, no. 8, pp. 1264-1273, Oct. 1987.

HUA84 A. Huang and S Knauer. “Starlite: A Wideband Digital Switch,” *Proceedings of Globecome*, pp. 121-125, 1984.

ITO91 A. Itoh, et al., “Practical Implementation and Packaging Technologies for a Large-Scale ATM Switching System,” *IEEE J. Select. Areas Commun.*, vol 9, pp. 1280-1288, Oct, 1991.

JEN83 Yih-Chyun Jenq, “Performance Analysis of a Packet Switch Based on Single-Buffered Banyan Network,” IEEE J. Sel. Areas in Commun., vol. 1, no. 6, pp. 1014-1021, Dec. 1983.

KAR92 Mark J. Karol, Chih-Lin, “Performance Analysis of a Growable Architecture for Broadband Packet (ATM) Switching,” *IEEE Trans. on Commun.*, vol. 40, no. 2, pp 431-439, Feb. 1992.

KAR92 M. Karol and K. Eng, “Performance of Hierarchical Multiplexing in ATM Switch Designs,” *Proceedings of the International Communications Conference*, 1992, pp. 269-275.

KAT95 Manolis Katevenis, et al., “Pipelined Memory Shared Buffer for VLSI Switches,” *Proceedings of ACM SIGCOMM*, pp. 39-48, 1995.

KAT96a Manolis Katevenis, et al., "ATLAS I: A General-Purpose, Single-Chip ATM Switch with Credit-Based Flow Control," *Proceedings of IEEE Hot Interconnects*, 8/96.

KAT96b Manolis Katevenis, et al., "VC-level Flow Control and Shared Buffering in the Telegraphos Switch," *Proceedings of IEEE Hot Interconnects*, 8/96.

KIM94 Hyong S. Kim, "Design and Performance of Multinet Switch: A Multistage ATM Switch Architecture with Partially Shared Buffers," *IEEE/ACM Trans. on Networking*, vol. 2, no. 6, pp. 571-580, 12/94.

KIM93 Young Man Kim and Kyungsook Y. Lee, "PR-Banyan: A Packet Switch with a Pseudorandomizer for Nonuniform Traffic," *IEEE Trans. on Commun.* vol. 41, no. 7, pp. 1039-1042, 7/93.

KOZ91 Kozaki, T., et al., "32 x 32 Shared Buffer Type ATM switch VLSI's for B-ISDN's," *IEEE Journal on Selected Areas of Communications*, vol. 9, No. 8, Oct. 1991.

KUW89 H. Kuwahara, N. Endo, M. Ogin, and T. Kozaki, "A Shared Buffer Memory Switch for an ATM Exchange," *Proc. Int. Conf. on Commun.*, 6/89.

LAW96 K. L. Eddie Law, A. Leon-Garcia, "ATM Multiplexers and Output Port Controllers with Distributed Control and Flexible Queueing Disciplines," *Proceedings of Infocom*, pp. 663-670, 1996.

LEA93 Chin-Tau Lea, "A Multicast Broadband Packet Switch", *IEEE Trans. on Commun.*, vol. 41, no. 4, pp. 621-630, 4/93.

LEE88 Tony T. Lee, "Nonblocking Copy Networks for Multicast Packet Switching," *IEEE J. Sel. Areas in Commun.*, vol. 6, no. 9, 12/88.

LEE90 Tony T. Lee, "A Modular Architecture for Very Large Packet Switches," IEEE Trans. Commun., vol. 38, no. 7, pp. 1097-1106, 7/90.

LIC96 L. Liccrardi et al., A Multichip Module Solution for High Performance ATM Switching , MCMC'96, Conf.

LIE90 Soung C. Liew, Kevin W. Lu, "A 3-Stage Interconnection Structure for Very Large Packet Switches," *Proceedings of the International Communications Conference*, 1990.

LIN96a Jeen-Fong Lin and Sheng-De Wang, "*High-Performance Low-Cost Non-Blocking Switch for ATM*," *Proceedings of Infocom*, pp. 818-821, 1996.

LIN96b Yu-Sheng Lin, C. Bernard Shung, "Queue Management for Shared Buffer and Shared Multi-Buffer ATM Switches," *Proceedings of Infocom*, pp. 688-695, 1996.

LOC95 Lockwood, John, Haoran Duan, James Morikuni, Sung-Mo Kang, Sarada Akkineni, Roy Campbell. "Scalable Optoelectronic ATM Networks: the iPOINT Fully Functional Testbed," *Journal of Lightwave Technology*, 6/95.

LUC94 James V. Luciani, C. Y. Roger Chen, "An Analytical Model for Partially Blocking Finite-Buffered Switching Networks," IEEE Trans. on Commun., vol. 2, no. 5, pp 533-540, Oct. 1994.

MCK96a Nick NcKeown, et al., "The Tiny Tera: A Packet Switch Core", *Proceedings of Hot Interconnects*, 1996.

MCK96b Nick Mckeown, “Achieving 100% Throughput in an Input-Queued Switch,” *Proceedings of Infocom*, 1996.

MEL89a Riccardo Melen, J. S. Turner, “Nonblocking Networks for Fast Packet Switching,” Proceedings of Infocom, 3/89.

MEL89b Riccardo Melen and Jonathan Turner. “Nonblocking Multirate Networks,” *SIAM Journal on Computing*, 3/89.

MEL93 Riccardo Melen, J. S. Turner, “Nonblocking Multirate Distribution Networks,” *IEEE Trans. on Commun.*, vol. 41, no. 2, pp. 362-369, 2/93.

MIN95a P. S. Min, H. Saidi and M. V. Hegde, “A Nonblocking Architecture for Broadband Multichannel Switching”, *IEEE/ACM Trans. on Networking*, vol. 3, no. 2, pp. 181-198, 3/95.

MIN95b P. S. Min, Manjunath V. Hegde, et al., “Nonblocking Copy Networks in Multi-Channel Switching”, *IEEE/ACM Trans. on Networking*, vol. 3, no. 6, pp. 857-871, 12/95.

MIS96 A. Misawa, M. Tsukada, Y. Yamada, K. Sasayama, K. Habara, T. Matsunaga, and K-I Yukimatsu, 40-Gbit/s broadcast-and-select photonic ATM Switch Prototype with FDM output buffers , 22nd ECOC ‘96, ThD.1.2, 1996.

MMC96 MMC Networks. “ATMS2000: 5 Gb/s Switch Engine Chip Set,” http://www.mmcnet.com/atms2000.html, 1996.

MUN95 E. Munter et al., A High Capacity ATM Switch Based on Advanced Electronic and Optical Technologies , *Proceedings of the International Switching Symposium,* 1995.

NEW88 Peter Newman, “A Fast Packet Switch for the Integrated Services Backbone Network,” *IEEE J. Sel. Areas in Commun.*, vol. 6, no. 9, pp. 1468-1479, 12/88.

NOJ87 Satoshi Nojima, E. Tsutsui, H. Fukuda and M. Hashimoto, “Integrated Services Packet Network Using Bus Matrix Switch,” *IEEE Journal on Selected Areas in Communications*, vol. 5, no. 8, pp. 1284-1292, Oct. 1987.

OKI97 Oki, Eiji and Naoaki Yamanaka. “Scalable Crosspoint Buffering ATM Switch Architecture Using Distributed Arbitration Scheme,” *Proceedings of the IEEE ATM Workshop*, 5/97.

PAT88 Achille Pattabvina, “Multichannel Bandwidth allocation in a Broadband Packet Switch,” *IEEE J. Select. Areas Commun.*, vol. 6, no. 9, pp. 1489-1499, Dec. 1988.

PAT90 Achille Pattavina, “A Broadband Packet Swtich with Input and Output Queueing,” *Proceedings of the International Switching Symposium*, pp. 11-16, 1990.

PAT94 Achille Pattavina, Stefano Gianatti, “Performance Analysis of ATM Banyan Networks with Shared Queueing—Part II: Correlated/Unbalanced Offered Traffic,” *IEEE/ACM Trans. on Networking*, vol. 2, no. 4, pp. 411-424, 8/94.

PRA96 Balaji Prabhakar, Nick Mckeown, and Ritesh Ahuja, “Multicast Scheduling for Input-Queued Switches,” *Proc. of the Princeton Conf.*, 1996.

PAD95 Krishnan Padmanabhan, “An Efficient Architecture for Fault-Tolerant ATM Switches”, *IEEE/ACM Trans. on Networking*, vol. 3, no. 5, pp. 527-537, 10/95.

RAT88 E. Rathgeb, T. Theimer, M. Huber, “Buffering Concepts for ATM Switching Networks,” *Proceedings of Globecom*, 11/88.

QIA96 Chunming Qiao, “Analysis of Space-Time Tradeoffs in Photonic Switching Networks, Proc. of Infocom’96, pp. 822-829, 1996.

SAS97 K. Sasayama, Y. Yamada, K. Habara and K-I Yukimatsu, FRONTIERNET: Frequency-Routing-Type Time-Division Interconnection Network , *Journal of Lightwave Technology*, vol. 15, No. 3, pp. 1-13, 1997.

SCH91 M. D. Schroeder, et al., “Autonet: A High-Speed, Self-Reconfiguring Local Area Network Using Point-to-Point Links,” *IEEE JSAC*, vol. 9, no. 8, Oct. 1991, pp. 1318.

SUZ89 H. Suzuki, et al., “Output-Buffer Switch architecture for Asynchronous Transfer Mode,” in *Proceedings of the International Communications Conference (ICC)*, pp 4.1.1-4.1.5., 6/89.

SZY89 T. Szymanski and S. Shaikh. “Markov Chain Analysis of Packet-Switched Banyans with Arbitrary Switch Sizes, Queue Sizes, Link Multiplicities and Speedups,” *Proceedings of Infocom*,” 3/89.

TAK87 T. Takeuchi, T. Yamaguchi, H. Niwa, H. Suzuki and S. Hayano. “Synchronous Composite Packet Switching - A Switching Architecture for Broadband ISDN,” *IEEE Journal on Selected Areas in Communications*, pp. 1365-1376, 10/87.

TAK93 Takase, A., et. al. “ATM Transport Node for Flexible and Robust Access Networks,” *Proceedings of Globecom*, 1993, pp. 1481-1487.

TAK96 Takase, A., et. al. “Data-Path Architecture and Technology for Large Scale ATM Switching System,” *Proceedings of Globecom*, 1996, pp. 1395-1399.

TAR91 C. L. Tarng, J. S. Meditch, “A High Performance Copy Network for B-ISDN,” Proc. of Infocom’91, 1991.

THI89 Krishna Thilakam, Ashok Jhunjhunwala, “Proposed High Speed Packet Switch for Broadband Integrated Networks,” *Computer Commun.*, vol. 12, no. 6, Dec. 1989.

TOB90 Fouad A. Tobagi, “Fast Packet Switch Architectures for Broadband Integrated Services Digital Networks,” *Proc. of IEEE*, vol. 78, no. 1, pp. 133-167, 1/90.

TOB91 F. Tobagi, T. Kwok, and F. Chiussi, “Architecture, Performance, and Implementation of the Tandem Banyan Fast Packet Switch,” *IEEE J. Sel. Areas in Commun.*, vol. 9, no. 8, pp. 1173-1193, 10/91.

TOM92 Tomonaga, H., et al., “High-Speed Switching Module for Large Capacity ATM Switching System,” *Proceedings of Globecom*, 1992, pp. 123-127, 1992.

TOM93 Tomonaga, H., N. Matsuoka, Y. Kato, A. Hakata and K. Murakami, “A Structure of Ultra High-Speed ATM Switch,” *Trans. of IEICE*, B-I, vol. J76-B-I, No. 11, pp. 793-800, 1993, in Japanese.

TOM95 Tomonaga, H., N. Matsuoka, T. Katoh, M. Kawai, H. Yamashita, and A. Hakata, “Small Size and Large Capacity ATM Switching System,” *Proc. of 1995 International Symposium on Communication*, pp. 374-381, 1995.

TRA96 TranSwitch. “CUBIT: ATM Cell Bus Switch Device,” http://www.txc.com/cubit.html, 1996.

TU90 S. C. Tu, W. H. Leung, “Multicast Connection-Oriented Packet Switching Networks,” *Proceedings of Infocom*, 1990.

TUR88 J. S. Turner, “Design of a Broadcast Packet Switching Network,” *IEEE Trans. on Commun.*, vol. 36, no. 6, pp. 734-743, 6/88.

TUR93a J. S. Turner, “A Practical Version of Lee’s Multicast Switch Architecture,” *IEEE Trans. on Commun.*, vol. 41, no. 8, pp. 1166-1169, 8/93.

TUR93b J. S. Turner, “Queuing Analysis of Buffered Switching Networks,” *IEEE Trans. on Commun.*, vol. 41, no. 2, pp. 412-420, 2/93.

TUR94 Turner, Jonathan S.“An Optimal Nonblocking Multicast Virtual Circuit Switch,”*Proceedings of Infocom*, June 1994, pp. 298--305.

WEN93 Wen De Zhong, et al., “A Copy Network with Shared Buffers for Large-Scale Multicast ATM Switching,”, IEEE/ACM Trans. on Networking, vol. 1, no. 2, pp. 157-165, April 1993.

WON95 P. C. Wong, and M. S. Yeung, “Design and Analysis of a Novel Fast Packet Switch — Pipleline Banyan”, *IEEE/ACM Trans. on Networking*, vol. 3, no. 1, pp. 63-69, 2/95.

YAM94 Yamanaka, N., et. al., “320 Gb/s High-Speed ATM Switching System Hardware Technologies Based on Copper-Polyimide MCM,” *44th ECTE*, pp. 776-785, 1994

YEH87 Y. S. Yeh, M. G. Hluchyj, and A. S. Acampora, “The Knockout Switch: A Simple, Modular Architecture for High-Performance Packet Switching,” *IEEE J. Sel. Areas in Commun.*, vol. 5, no. 8, pp 1274-1283, 10/87.

YOS96 Yoshisumi et al., 160 Gb/s ATM Switching System for Public Networks, *Proceedings of Globecom*, pp. 1380-1387, 1996.

ZEG93 Ellen Witte Zegura, “Architectures for ATM Switching Systems,” IEEE Commun. Magazine, pp. 28-37, Feb, 1993.

ZHO93 Zhong, Wen De, Yoshikuni Onozato and Jaidev Kaniyil. “A Copy Network with Shared Buffers for Large-Scale Multicast ATM Switching,” *IEEE/ACM Transactions on Networking*,3//93.