DESIGN OF A GIGABIT ATM SWITCH

Tom Chaney, J. Andrew Fingerhut,
Margaret Flucke, Jonathan S. Turner
Washington University, St. Louis

Abstract

This paper describes the design and implementation of a gigabit ATM switching system supporting link rates from 150 Mb/s to 2.4 Gb/s, with a uniquely efficient multicast switch architecture that enables the construction of systems with essentially constant per port costs for configurations ranging from 8 to 4096 ports and system capacities approaching 10 Tb/s. The system design supports many-to-one and many-to-many forms of multicast, in addition to the usual one-to-many. It also provides multicast virtual paths, constant time configuration of multicast connections and an efficient packet-level discard method, that can achieve 100% link efficiencies, without large buffers.

1 Introduction

Asynchronous Transfer Mode (ATM) network technology is widely acknowledged as a key component of the emerging global information infrastructure. ATM technology facilitates the integration of a wide variety of different applications on a common switching and transmission infrastructure, including classical data applications, image-retrieval and real-time audio and video. However, to be useful in large-scale systems with ubiquitous deployment to homes and businesses, ATM networks will require highly efficient switching systems that allow more cost-effective networks than are possible with most current designs. This paper describes the design and implementation of an ATM switching system that is capable of supporting large networks with a wide range of requirements, in a particularly cost-effective way. The system, called the *Washington University Gigabit Switch* (WUGS), has the following distinguishing characteristics.

- *High performance components enabling port speeds up to 2.4 Gb/s.* This enables demanding applications that are beyond the reach of lower speed systems. More importantly perhaps, it enables highly efficient statistical multiplexing of more typical applications with peak data rates from 1–100 Mb/s, greatly reducing the cost of transmission in both local and wide area environments.

- *Optimally efficient multicast switching.* The multicast switching technique used in the WUGS switch has optimal complexity in terms of both the switching network and multicast routing memory, giving it essentially constant per port costs for configurations ranging from 8 to 4096 ports and throughputs approaching 10 Tb/s. This allows large networks to be implemented at a fraction of the cost required when using smaller switches.

- *Efficient handling of many-to-many and many-to-one multicast.* Unlike standard ATM, the WUGS switch supports many-to-many and many-to-one applications directly, rather than requiring one-to-many overlays. This avoids the quadratic scaling limitations inherent in the overlay method. Many-to-many applications requiring explicit inband sender identification can be supported via end-to-end multicast virtual paths with the virtual circuit identifier used for sender identification.

- *Fast reconfiguration of multicast connections.* In previous ATM switch architectures, the time required to add or remove an endpoint from a multicast connection grows with the size of the switch and/or the connection. The WUGS architecture

[0]This work was supported by the Advanced Research Projects Agency, National Science Foundation, Ascom Timeplex, Bay Networks, Bell Communications Research, Bell Northern Research, Goldstar Information and Communications, Italtel SIT, NEC, NTT, Samsung Electronics, Southwestern Bell and Tektronix.

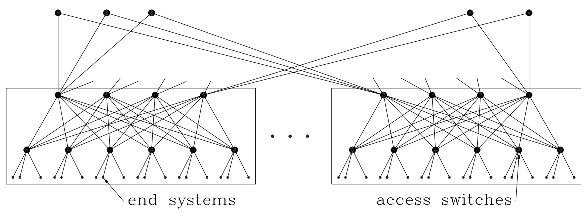

Figure 1: Hierarchical Campus Network

allows constant time addition and removal of endpoints, allowing highly dynamic multicast applications in which endpoints come and go thousands of times per second.

- *Effective packet-level discarding for gigabit links.* The WUGS switch employs an efficient variation of the early packet discard technique that ensures that during overload, the full link capacity is used for complete AAL-5 frames (not fragments). Our technique adds hysteresis to the early packet discard control algorithm, allowing it to achieve ideal throughput with much smaller buffer requirements (the equivalent of two maximum length transport packets). This is an important consideration in high speed applications where memory is a significant cost component.

As stated above, one of the distinguishing features of the WUGS switch architecture is its cost-effectiveness in a wide range of system sizes; but why are large switching systems of value in the first place? One can obviously construct networks of arbitrary size using switches with modest numbers of ports, so why should one bother with large switches? The answer to this is that large switches have an inherent cost and performance advantage when it comes to building large networks. This can be seen best by way of an example. Consider a typical campus network of the future with 8,000 workstations and servers connected to an ATM network with access links of 150 Mb/s. If we are limited to switches with say 16 ports, we need to define an appropriate interconnection topology to link these users together. Figure 1 shows an efficient general interconnection topology that can be used for this purpose. If the access switches are configured so that the number of ports connecting to other switches equals the number connecting to end-systems (a 1:1 concentration ratio), then the total number of ports on all the switches is seven times the number of users. For efficient switch architectures, the total system cost is dominated by the per port cost, making the cost of this hierarchical network roughly seven times that for a single large switch. Going to 64 ports reduces the penalty from $7\times$ to $5\times$. Introducing a three-to-one concentration ratio as well reduces the penalty to $2.3\times$, still a significant cost penalty. The expression below gives the total number of switch ports needed to provide U user ports, when we're limited to switches with n ports each and each of the access switches has u user ports and $n-u$ ports connecting to switches at the next level up in the hierarchy.

$$\left(1+2((n/u)-1)\left(1+\max\left\{0,\left\lceil\log_{n/2}(U/un)\right\rceil\right\}\right)\right)U$$

Note that in addition to having a higher cost, the hierarchical network design yields greatly inferior performance with respect to virtual circuit blocking, queueing delay and cell loss (even with a 1:1 concentration ratio). Its blocking performance is particularly poor for multicast connections.

It's tempting to think that the hierarchical network with concentration can redeem much of the extra switching costs through savings in fiber. However, in campus networks the distances are too short for fiber costs to play a significant role (it takes several miles of fiber to equal the cost of the ATM switch ports at either end of a 150 Mb/s link). This trade-off changes as networks become more geographically distributed, since for distances of ten miles or more, fiber costs become a major consideration, leading to networks with more complex topologies. Within a campus however, a single central switch, or a central switch connected by gigabit links to smaller 'satellites' is by far the most economical choice.

The importance of gigabit links is often attributed solely to their role in supporting high speed applications. Equally (or more) important is their role in lowering system costs through improved statistical multiplexing and more efficient use of fiber. Consider, for example, a collection of on-off bursty sources with a peak rate of 50 Mb/s and an average rate of 2 Mb/s. On a 150 Mb/s link, we can multiplex at most 12 such virtual circuits before the probability of congestion (that is the probability that any given virtual circuit encounters congestion when it sends a burst of data) exceeds 1%. This gives a link efficiency of 16%. A 2.4 Gb/s link can handle 841 such virtual circuits before the congestion probability exceeds 1%, giving an overall link efficiency of 70%. For typical component costs today, this gives the 2.4 Gb/s link a 5-to-1 cost/performance advantage over the equivalent collection of 150 Mb/s links in short distance applications. In long distance applications, where the fiber savings play a dominant role, the advantage is much larger (60-to-1 for a 50 mile link).

In the remaining sections, we describe the design

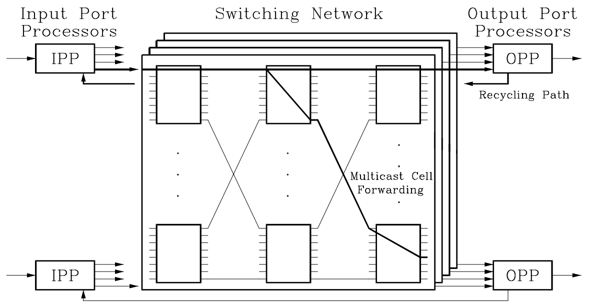

Figure 2: Recycling Switch Architecture (64 port configuration shown)

and implementation of the WUGS switch, which at this writing is approaching completion. An earlier paper [9] describes our basic multicast switching technique and includes an analysis of its scaling efficiency and performance characteristics. Another earlier paper [10] compares our multicast switching approach to competing alternatives and demonstrates order-of-magnitude improvements in cost/performance in large configurations. Here we focus on the specific design implementing this architecture, detail its features and the hardware components we have developed to achieve a highly scalable, cost-effective gigabit switching system.

2 System Overview

Figure 2 shows the overall organization of the WUGS switching system. It consists of three main components, which are each implemented as a single custom integrated circuit. The *Input Port Processors* (IPP) at left, receive cells from the incoming links, buffer them while awaiting transmission through the central switching network and perform the virtual path/circuit translation required to route cells to their proper output (or outputs). The *Output Port Processors* resequence cells received from the switching network and queue them while they await transmission on the outgoing link. Each OPP is connected to its corresponding IPP, providing the ability to *recycle* cells belonging to multicast connections. This is discussed in more detail below. The central switching network is made up of *Switching Elements* (SE) with eight inputs and outputs and a common buffer to resolve local contention. The SEs switch cells to the proper output (or outputs) using information contained in the cell header or can distribute cells dynamically to provide load balancing. The load balancing option is used in the first $k-1$ stages of a $2k-1$ stage

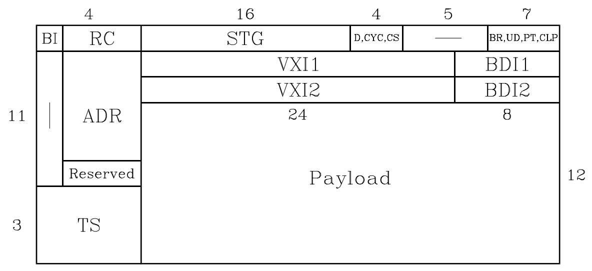

Figure 3: Internal Cell Format

network. In particular, for the configuration shown in Figure 2 ($k = 2$), load distribution is performed in the first stage. Adjacent switch elements employ a simple hardware flow control mechanism to regulate the flow of cells between successive stages, eliminating the possibility of cell loss within the switching network. With this approach, relatively small buffers suffice within the network. Larger buffers are provided at the OPPs.

To provide sufficient bandwidth for 2.4 Gb/s gigabit link rates on the external links, the switch carries ATM cells in a 36 bit wide format, as shown in Figure 3. Four of the 36 bits contain addressing information and the remainder contains the cell payload, together with auxiliary fields that are added by the IPPs and removed by the OPPs (the various fields will be discussed in later sections). The internal cell cycle is 16 clock ticks long and the clock frequency used in our initial prototype is 120 MHz. This yields an internal cell processing rate that is about 1.3 times the cell processing rate for external links operating at 2.4 Gb/s. The switching network is implemented in four parallel planes, with each plane receiving the same four bits of address information, plus eight bits of data. The core of the system operates fully synchronously, and the switch elements' operation is completely deterministic, meaning that the cells proceed through the four planes in parallel, without any explicit coordination, and are reconstructed at the OPP.

2.1 Switching Network

The switching network uses a Beneš network topology. The Beneš network extends to arbitrarily large configurations by way of a recursive expansion. Figure 2 shows a 64 port Beneš network. We can construct a 512 port network by taking eight copies of the 64 port network and adding a first and fifth stage on either side, with 64 switch elements apiece. Output j of the i-th switch element in the first stage is then connected to input i of the j-th 64 port network. Similarly, output j of the i-th 64 port network is connected to input i of the j-th switch element in the fifth stage. Repeating in this way, we can obtain networks with 4,096 ports or 32,768 ports. For $n = 8^k$, the Beneš network constructed using eight port switch elements has $2k - 1$ stages. Since $k = \log_8 n$, the number of switch elements scales in proportion to $n \log n$, which is the best possible scaling characteristic. In [9], it is shown that when dynamic load distribution is performed in the first $k-1$ stages of a Beneš network, that the load on the internal data paths of the switching network cannot exceed the load on the external ports; that is the network achieves ideal load balancing. This is true for both point-to-point and multipoint traffic.

The queueing performance of multistage switching networks with shared buffer switch elements and interstage flow control is studied in [1]. This paper shows that when the internal link speeds have a cell processing rate of more than 1.25 that of the external links, a system with eight port shared buffer switches and 32 cell buffers can handle fully loaded external links under uniform random traffic. Bursty traffic requires some reduction in the average link loading to achieve acceptable congestion probabilities on the outgoing links. For on-off virtual circuits with peak rates of 50 Mb/s and a peak-to-average ratio of 10:1, we find that the probability of output link congestion is about .005 when the average link occupancy is 70%. At this load, the probability of congesting the connection between the switching network output and the OPP of the WUGS switch is .0001. At an offered load of 50%, the corresponding probabilities of congestion are 10^{-6} and 5×10^{-9}.

While the system's per port cost does increase (slowly) with n, the cost remains dominated by transmission component costs in even the largest configurations. In particular, if the cost of transmission components is \$3,000 per 2.4 Gb/s port and the CMOS integrated circuits implementing the ATM switching functions are \$200 each, then the cost of components ranges from \$3,500 per port in an 8 port system to \$4,100 per port in a 4,096 port switch, an increase of just 17%.

The Beneš topology allows for a simple routing algorithm in the last k stages. In particular, if the output port number is expressed in base 8, switch elements in successive stages use successive base 8 digits of the output port number to select the switch element output to route the cell to. This scheme has been extended to provide a *copy-by-two* function for multicast connections. In the multicast case, a cell contains a pair of output port numbers and at each stage, successive pairs of base-8 digits are considered. So long as these digits match, we can route the single copy exactly as before. However, when we reach a stage where the base-8 digits differ, then the cell must be copied to the two switch element outputs specified by the pair of digits. After the copies are made, each is routed as in the point-to-point case, using the remaining base-8 digits of the appropriate output port number. The switch element chip also supports a *copy-range* option that delivers copies of the cell to all outputs in the range bounded by the pair of output port numbers in the cell. This is useful in special applications.

One drawback of the Beneš network topology based on 8 port switch elements is that it grows in factors of 8. We have extended the basic Beneš topology and the routing/distribution algorithms to enable the construction of networks in which the number of ports can be any power of 2.

2.2 Using Cell Recycling for Multicast

The WUGS switch implements multicasting using a technique called *cell recycling*. When cells arrive on an input link, the virtual path and virtual circuit identifiers are used to select an entry from a routing table in the IPP (called the *Virtual Path/Circuit Translation Table* (VXT)). The entry includes a pair of output port numbers, a pair of new virtual path and circuit identifiers and a pair of control bits. The switching network routes a copy of the cell to each of the designated outputs, where they can be forwarded to the outgoing link or optionally recycled back to the input ports (this choice is determined by the control bits), where the virtual path and circuit identifiers are used to perform new VXT lookups, yielding new pairs of destinations. Through this process, a connection with f destinations can be handled in $\log_2 f$ passes through the network. In [9], it is shown that this technique, coupled with a buffered multistage switching network using dynamic routing, yields optimal cost/performance scaling, with respect to both the switching network complexity and the amount of routing memory required. While the recycling technique does add delay, the absolute magnitude is small enough (under 10 μs per pass in the WUGS switch) to make it insignificant in most applications.

Because the cell recycling technique consumes a part of the system's bandwidth to handle multicast connections, engineering rules are needed to specify how much of the system's capacity to set aside for multicast. It's easy to show that in a system with n ports, we can avoid blocking new connection requests

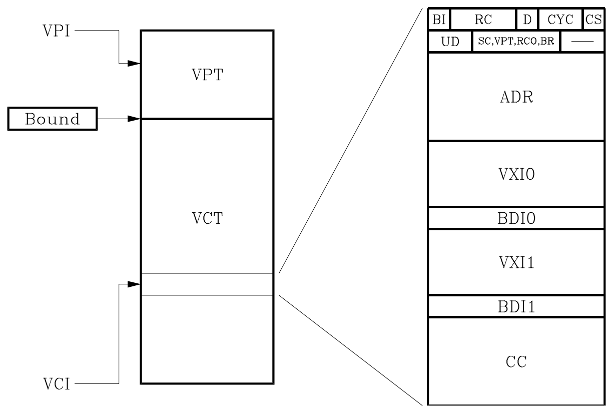

Figure 4: Virtual Path/Circuit Translation Table

if we dedicate h ports for recycling where

$$h \geq \frac{\delta n}{\delta + (1/\beta) - (B/\beta)}$$

and β is the ratio of the speed of the external links to the speed of the switch's internal data path ($\beta \leq 1$), B is the ratio of the maximum virtual circuit bandwidth to the internal data path speed and δ is the fraction of the total output traffic associated with multipoint connections. For the WUGS switch, if we constrain the maximum virtual circuit rate to 150 Mb/s and configure the system for 20% of the total traffic being multicast ($\beta = 1$, $B = 1/16$, $\delta = 0.2$) we get $h \geq .16n$. It's important to note that the fraction of the system's capacity that must be dedicated to multicast is independent of n. In particular, in the worst-case, when potentially all of the traffic is multicast, it's sufficient to have half the ports dedicated to recycling, so long as $\beta + B \leq 1$. We also note that a single port can carry both recycling traffic and traffic from the external links, although in most situations, dedicated recycling ports yield the more cost-effective operation.

2.3 Virtual Paths and Virtual Circuits

As indicated above, when cells arrive at an input port processor (either from the external link or from the recycling path) a VXT lookup is performed. The VXT supports both virtual path and virtual circuit connections, but places certain constraints on *terminating* virtual path connections, in order to conserve memory. In particular, if two different virtual paths terminate at the input port of a switch (that is, if the individual virtual circuits within the virtual path must be switched separately at that point, rather than passed through as a single bundle), the VCIs used for virtual circuits belonging to these virtual paths must be distinct. This constraint is enforced by the software responsible for connection management. There is no constraint on the VCIs associated with non-terminating virtual path connections. In particular, when an end-to-end virtual path is implemented by the switch, the virtual circuit identifiers associated with that virtual path are not restricted by the switch in any way. The VXT allows for a flexible allocation of virtual path and virtual circuit connections, as illustrated in Figure 4. A bounds register separates the *Virtual Path Table* section from the *Virtual Circuit Table* section. When a cell is processed, the VPI is used to select the appropriate entry from the virtual path table. The selected entry contains a *Virtual Path Termination Bit* (VPT). If this bit is cleared, then the other information in the entry specifies how to handle the cell, If the bit is set, then the VCI of the cell is used to select an entry from the virtual circuit table and this second entry is used to process the cell. A typical configuration would set the bounds register at 256, allowing the first 256 table entries to be used for virtual path connections. VPI 0 is reserved for virtual circuit connections (we can view this as a terminating virtual path connecting the switch to its immediate neighbor) while VPIs 1–255 would be available for either user-level virtual paths or network virtual paths. The remaining entries (out of a total of 1024 in the current design) would be used for virtual circuit connections.

Most of the fields of each VXT entry correspond to fields in the switch's internal cell format. The remaining fields are defined here. The *Busy/Idle Bit* (BI) is set to 1 to indicate an entry corresponding to a valid connection. If, when processing a cell, this bit is found to be 0, it means that an error has occurred (either the control software failed to set the bit, or cells are being incorrectly sent on a non-existent connection). The *Set CLP Bit* (SC) can be set to 1 to force the Cell Loss Priority bit of all received cells to be set to 1. The VPT bit was discussed above. The *Recycling Cells Only* bit (RCO) is set to 1 to allow a given entry to be used only for cells received on the recycling port. Cells arriving on the external link that attempt to access that entry are considered erroneous. The *Cell Count* (CC) is a 32 bit value that is incremented whenever a cell is received for a given virtual path or virtual circuit. Its value can be retrieved at any time using a control cell. Control cells are time-stamped, allowing the number of cells received during a given time interval to be determined by comparing the counter values and time stamps from two successive accesses.

2.4 Many-to-Many Connections

Unlike in standard ATM, virtual paths can have all the characteristics of virtual circuits. They can be used on an end-to-end basis, can have arbitrary and time-varying data rates, different QoS characteristics and can be multicast. In particular, end-to-end, many-to-many virtual paths are useful in applications where participants must be able to send information over the channel concurrently while allowing the receivers to separate the information from different senders. With virtual paths, the virtual circuit identifier can be used as an end-to-end source identifier that is either statically or dynamically assigned. Hardware support for virtual paths in the end systems' network interface cards facilitates the use of end-to-end virtual paths but is not strictly necessary. The same effect can be achieved using virtual paths that terminate at the access switches. The use of virtual paths eliminates the quadratic scaling that plagues the standard ATM approach to this class of applications. In standard ATM, it's necessary to overlay one-to-many virtual circuits, meaning that when we add, say the tenth endpoint to a multicast application, we must add a new one-to-nine connection and add the new user as a receiver on nine other connections. This imposes an intolerable setup penalty for large, dynamic multicast applications and leads to wasted bandwidth as well, since there is no way for the participants to allocate bandwidth as a group, rather than as individuals. The use of direct many-to-many connections eliminates these limitations.

The switch includes a special *upstream discard* mechanism to enable most efficient handling of many-to-many virtual paths and circuits. When a many-to-many connection has traffic entering a switch on multiple input links, the entering traffic streams are first brought together by routing them through a common recycling port. From this point the cells are forwarded to the required set of output links using a common multicast connection (either virtual path or circuit). If a given link is used as both an input and an output in the multicast connection, this can result in cells being forwarded back to the sender, which is often not desirable. The upstream discard mechanism can be used to prevent this. To implement upstream discard, the IPPs label cells arriving from the external link with a *Source Trunk Group number* (STG), which is normally equal to the number of the switch port the IPP is connected to. This is carried in the cell until it is sent on an external link. In addition, the VXT entries include an *Upstream Discard bit* (UD) that is included in cells and then interpreted by the output port processors where cells are to be sent on the outgoing link. If the UD bit of an outgoing cell is set and the cell's STG field matches the source trunk group number of the outgoing link, the cell is discarded. This mechanism allows for efficient configuration of many-to-many connections without quadratic scaling limits within a single switch.

3 Switch Element Design

The organization of the switch element is shown in Figure 5. There are two major design alternatives for a shared buffer switch element. In the first, cells from different inputs are multiplexed onto a parallel, very high speed bus, and placed into a common memory with a large word-size to provide sufficient memory bandwidth. On output, this process is reversed, with cells being demultiplexed to individual output ports. The control section for such a switch element is typically implemented using per-output queues, which can be implemented either with independent hardware fifos or lists sharing a common list memory. In a switch element supporting multicast, reference counts are required for the stored cells, so that the storage can be recovered when the last copy has been transmitted. One of the first examples of this type of design can be found in [3].

The second approach, which is adopted here, is to provide a set of independent storage elements, each with its own control circuitry and use crossbars at the input and output to provide access to these storage elements. This approach does not lend itself as readily to the implementation of switch elements with large cell stores. However, the use of crossbars to provide access to the central memory is significantly more area and power efficient than multiplexing. It is also easier to provide the parallelism needed to achieve maximum performance in high speed systems using this approach. Referring to Figure 5, arriving cells enter the switch element at the upper left, passing through a *Skew Compensation Circuit* (SKUC) and a *Distribution Circuit* (DSTC) before entering the *Input Crossbar* (IXBAR). The IXBAR forwards cells to free buffer slots in the central *Cell Buffer* (CBUF). Each row of the CBUF stores one cell (actually one fourth of the cell, plus control information) and has an associated *Buffer Control Circuit* (BCC). Outgoing cells pass through the *Output Crossbar* (OXBAR) and *Header Modification Circuit* (HMC) before proceeding to the output. Downstream neighbors pro-

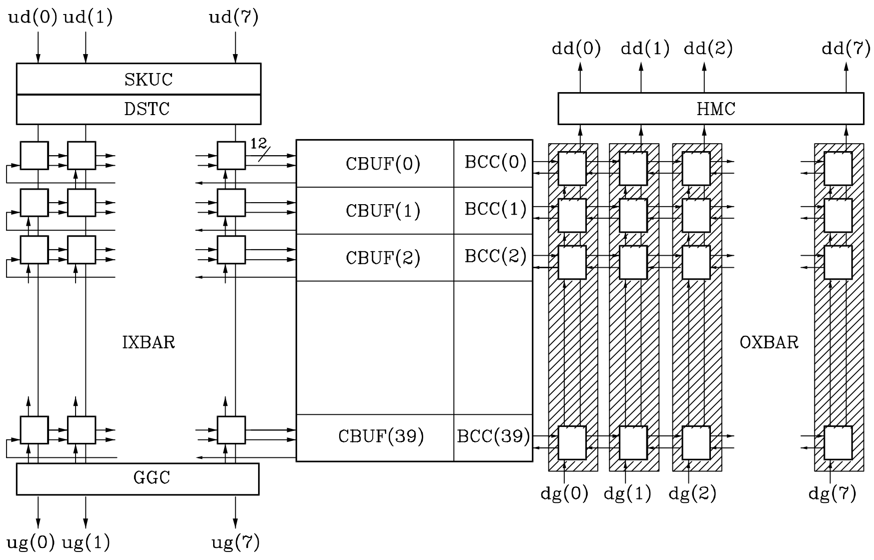

Figure 5: Switch Element Organization

vide a one bit flow control signal each cycle, indicating if they are prepared to receive an incoming cell or not. These *grant* signals `dg(0)` ...`dg(7)` are used within the (OXBAR) to control the flow of cells from the CBUF to the outputs. Upstream grants `ug(0)` ...`ug(7)` are generated by the *Grant Generation Circuit* (GGC) based on the number of available CBUF slots. The CBUF has been dimensioned to hold 40 cells. All the internal data paths are 12 bits wide.

3.1 Skew Compensation

The integrated circuits used to implement the gigabit switch design use CMOS technology, reflecting our belief that the inherent control complexities of ATM and the need for buffers and large tables require a high density circuit technology, and that the limited circuit speed of CMOS can be offset using highly parallel data paths. Having said that, it remains a challenging proposition to design systems that can operate reliably at very high speeds. One of the more difficult parts of implementing systems using CMOS technology with clock rates in excess of 100 MHz is ensuring reliable chip-to-chip communication. In particular, clock skew and variations in inter-chip delay due to differences in wiring path length, can make it difficult to transfer data reliably between chips on a single circuit board or between different circuit boards in a larger system. There are two extreme approaches one can take to this problem. At one extreme, once can construct a classical synchronous system and tightly control the wiring to minimize clock skew and data path lengths through clever packaging strategies. This approach is exemplified by [4]. The other extreme is to adopt a fully asynchronous approach in which clock signals are passed with data from chip to chip. In a switching network, where data can be received from multiple upstream neighbors, this approach still leaves one with the need to synchronize the arriving streams to a local timing reference.

We have taken an intermediate approach (first described in [7]) in which a single clock and cell timing signal are distributed throughout the system so that all chips operate from a common frequency reference. However, we allow some variability in the timing of arriving data to allow for clock skew and variations in wire lengths. Skew compensation circuits at the inputs of every chip monitor the arriving data and insert delay in order to force arriving cells into phase with the local timing reference. Our skew compensation circuit uses a novel design that yields superior timing margins and simple circuitry. It is implemented using only standard cells and yet is compact enough for ubiquitous usage (with a 0.8 μm CMOS process, the area required for one bit of skew compensation is less than the area of a single input pad). The current design can accommodate up to 16 ns of delay variation due to clock skew and differences in wiring length. This can be readily increased to accommodate larger delay variations, with only a small incremental cost. The circuit has been successfully tested at clock frequencies up to 180 MHz. More details can be found in [2].

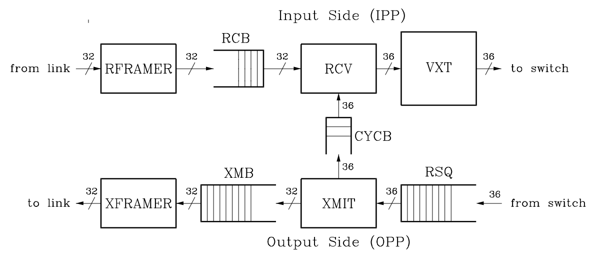

Figure 6: Port Processor Logical Organization

3.2 Output Crossbar

Each column in the output crossbar includes a control circuit that arbitrates requests for access to that output. A particular column has the potential for getting simultaneous requests from any subset of the BCCs. Assuming the output is available (that is, if the downstream neighbor provides a grant signal), the arbitration process selects the requesting BCC whose cell has the highest priority (the priority is initially zero and increases, the longer a cell is forced to wait). This is done using a bit serial maximum-finding circuit to select the maximum priority value. (This technique was borrowed from the original Datakit bus arbitration circuit [6].) If a particular BCC's request to use a particular output is refused, the control circuitry in the output crossbar continues to contend for the output on subsequent cycles until the request is satisfied.

The control circuits along a row of the output crossbar can operate in parallel to allow a given cell to be sent to multiple outputs concurrently. In particular, when a cell first arrives, the BCC for that row issues a request specifying which outputs the cell is to be sent to (using a set of eight output select lines). The control circuit in each selected column then initiates contention for its output, continuing on subsequent cycles until it is successful. After all the initially selected outputs have succeeded, the BCC gets back a signal from the OXBAR indicating that the cell has been sent to all selected outputs. The BCC then makes the CBUF slot available for a new incoming cell.

4 Port Processor

The *Port Processor* (both input and output sections) is shown in Figure 6. Cells coming from the external link enter a *Receive Framer* (RFRAMER) driven by the recovered clock from the link and enter a *Receive Buffer* (RCB) while awaiting transmission to the switch. When the switch is prepared to accept a cell, it is transferred from the RCB, through the *Receive Circuit* to the *Virtual Path/Circuit Translation Table* (VXT) which provides the routing information needed by the switching network. Cells coming from the switch to the link pass first through a *Resequencer* (RSQ) and then through the *Transmit Circuit* (XMIT) to the *Transmit Buffer* (XMB). Cells proceed from there to a *Transmit Framer* (XFRAMER) and the external link. The XMIT circuit can also direct cells to a *Recycling Buffer* (CYCB).

4.1 Resequencing Cells

The resequencer ensures that cells, which may follow different paths through the switching network and be delayed by different amounts of time, are restored to their original order before transmission on the external links. It uses a timing-based resequencing method [8]. Cells being sent from an IPP to the switching network are time-stamped with their time of entry (placed in the cell's *Timestamp field* (TS)). When cells emerge from the switching network, this time stamp field is used to determine the cell's *age* (which is defined as the time that has elapsed since it entered the switching network). Cells are released from the resequencer in 'oldest-first' order, but not until their age reaches a system-dependent *age threshold*, which is set to the maximum delay that cells can be expected to encounter in the switching network (for systems with up to 4,096 ports an age threshold of 64 cell times is ample). In a switch that uses cell recycling for multicast, the resequencing mechanisms must also compensate for the potential misordering that can occur when reconfiguration of a multicast connection causes cells that pass through the connection right after a reconfiguration to experience one fewer pass through the system than cells that go through just before the reconfiguration. This is handled using a *transitional time stamping* mechanism in the IPP. This mechanism causes the time stamps of cells arriving right after a reconfiguration to be artificially inflated (making the cells appear 'younger' than they really are) causing them to be delayed longer at the outgoing resequencer. This allows cells that went through before the reconfiguration a chance to catch up. The added delay is smoothly reduced over a short period of time (10–20 μs in the WUGS switch), causing only a momentary irregularity in the flow of cells through the connection. See [9] for more details.

4.2 Congestion Control

The port processor implements several simple mechanisms to control the effects of congestion. First, the XMB, where most cell queueing occurs in the system, provides separate buffers for reservation-oriented *continuous stream* traffic and bursty *discrete stream* traffic. Cells are identified as belonging to one or the other class through a *Continuous Stream Bit* (CS) inserted in arriving cells by the VXT. The continuous stream traffic includes both constant bit rate and variable bit rate traffic with modest peak-to-average rate variations, while the discrete stream category is intended for bursty data traffic with high variability and greater tolerance for delay. The continuous stream queue is strictly higher priority than the discrete stream queue.

In addition, the XMB implements a simple block discard mechanism on selected virtual circuits, which performs block discarding on the basis of AAL 5 frames, rather than on individual cells. The particular algorithm used is a variation of the well-known *Early Packet Discard* [5] technique in which hysteresis is added to damp the oscillations that can otherwise occur, making it possible to achieve 100% throughput with modest buffer sizes [12]. The block discard is implemented through the use of a *Block Discard Index* (BDI) which is inserted into arriving cells at the VXT. A non-zero BDI indicates a virtual circuit on which block discarding can be performed. The index is used by the *Block Discard Controller* in the XMB to access two bits of state information that control the discarding of cells for that virtual circuit.

While the switching network has sufficient bandwidth to ensure that cells normally proceed to the output port for buffering without congesting the switching network, sustained high rate bursts can lead to congestion in the switching network that will ultimately back up to the input ports, causing the RCBs to fill up and exceed a preset threshold. If this happens at a particular IPP, it begins discarding all received cells with CLP=1 or CS=0 and continues discarding these cells until a timer has expired. The timer is generally set with a long enough period to ensure that the congestion has time to clear completely. This is viewed as a fairly drastic response to an event which should occur very rarely in a network which is properly engineered and controlled at higher levels.

4.3 Control Cells

The chip set has been designed to allow control from a remote processor. In particular, for any link on which the feature is enabled (through a hardware switch), cells received with a VPI field of 0 and VCI field of 32 are interpreted as control cells, with the payload of the cell specifying the desired control action. Options include reading and/or writing entries from the VXT in any port processor and accessing a variety of hardware registers in the IPPs and OPPs. These include hardware counters that record the total number of cells passing over different data paths, the number of cells discarded due to HEC errors, buffer overflows or various error conditions. There are also registers controlling the resequencer age threshold, the low priority cell discard thresholds for various buffers, the VXT bounds register and other variables that affect the switch's operation.

5 System Configurations

The smallest system configuration has a throughput of up to 20 Gb/s and is implemented with a single switch element and eight port processors, together with the associated transmission interfaces. It is packaged in a 10 cm tall rack-mount unit with connections to external fibers through the front panel. Internally, the switch element and port processors are mounted on a large circuit board, which is 35 cm wide by 45 cm deep. Eight line cards are mounted directly above the main circuit board. These contain the transmission line-coding and opto-electronic circuitry, which connects to the front panel by fiber-optic pigtails. A typical line card will be 6 cm wide by 16 cm long. The system packaging also allows for "stacked" line cards to accommodate interfaces that require more real-estate than the standard card provides. Interfaces have been implemented for OC-3C, OC-12C and OC-48C links. Also, interfaces have been implemented using Hewlett Packard G-link series data link chips, operating at 1.2 Gb/s.

A system with a capacity of 160 Gb/s can be implemented in a rack-mounted unit that is 75 cm tall. This configuration is constructed from eight line modules, each containing two switch elements, eight port processors and associated transmission interfaces. Physically, each line module contains a main board that hosts the switch elements and port processors, plus eight line cards which connect to a front panel via fiber-optic pigtails. The line modules are 35 cm tall, by 45 cm deep by 5 cm wide, allowing eight to be housed in a single shelf of a standard equipment rack. The switch elements within the set of eight line modules provide the first and last stage of a three stage network. These switch elements connect to a center stage through a printed circuit backplane. The switch elements for the center stage are hosted by a set of small cards mounted horizontally, above and below the line modules. This is done in a way that allows a fan-shelf to draw air vertically through the unit to provide adequate cooling.

Systems with a capacity of 1.2 Tb/s can be constructed using the same line modules used for the 160 Gb/s system. Obviously these larger systems would require 64 line modules rather than eight and these can be mounted in four equipment racks. The 1.2 Tb/s system also requires a fifth equipment rack to house the center three stages of a five stage network. Interconnection between the four racks containing the line modules and the fifth rack can be accomplished most cost-effectively using parallel transmission through high density ribbon cables with impedance-matched connectors. To provide added fault tolerance, a configuration of this size can be designed using five planes, rather than the four used in the current design. Extending the current chip set to accommodate such a redundancy scheme is fairly straightforward and ensures that systems can operate without total system outages due to hardware failures (failures involving individual ports are still possible however).

6 Closing Remarks

Large scale deployment of ATM network technology will require switching systems that can cost-effectively support thousands or tens of thousands of users in a single location. We have shown that large switches and gigabit transmission offer compelling cost-performance advantages in large configurations and have developed an ATM switch architecture and gigabit technology components that are suitable for such large configurations, offering essentially constant per port costs for systems ranging from 8 to 4096 ports and system capacities approaching 10 Tb/s. The system design supports many-to-one and many-to-many forms of multicast, including multicast virtual paths. It supports constant time configuration of multicast connections and uses an efficient form of Early Packet Discard to guarantee high effective throughputs during overload.

Acknowledgements. This project has involved the efforts of a large number of individuals, in addition to the authors. Brian Gottlieb, Craig Horn, Saied Hosseini, and Randy Richards have contributed to the design of the integrated circuits. Dave Richard designed all the transmission interfaces. Maynard Engebretson and Mike Richards and were responsible for the printed circuit boards and mechanical design. Zubin Dittia helped with the early definition of the system architecture. John DeHart has led efforts to develop general multicast signaling and switch control software for the system. Finally, Jerry Cox and Guru Parulkar have been key supporters of the project from the beginning, handling most of the administrative and management chores needed to bring such an ambitious activity to a successful conclusion.

References

[1] Bianchi, Giusseppe and Jonathan Turner. "Improved Queueing Analysis of Shared Buffer Switching Networks," *IEEE/ACM Transactions on Networking*, August 1993.

[2] Chaney, Tom. "A Digital Phase Adjustment Circuit for ATM and ATM-Like Data Formats," Washington University Applied Research Lab technical report, 11/94.

[3] Coudreuse, J. P. and M. Servel. "Prelude: An Asynchronous Time-Division Switched Network," *Proceedings of the International Communications Conference (ICC)*, 1987.

[4] Day, C. and J. N. Giacopelli and J. Hickey. "A Switching Exchange for an Asynchronous Time Division Based Network," *Proceedings of International Switching Symposium (ISS)*, 1987.

[5] Floyd, Sally and Allyn Romanov. "Dynamics of TCP Traffic over ATM Networks," *Computer Communication Review*, vol. 24, no. 4, 10/94.

[6] Fraser, A. G. "Towards a Universal Data Transport System," *IEEE Journal on Selected Areas in Communications*, 11/83.

[7] Bassett, Paul D., Lance A. Glasser and Randy D. Rettberg. "Dynamic Delay Adjustment: a Technique for High-Speed Asynchronous Communication," *Proceedings of the MIT VLSI Conference*, 4/86.

[8] Turner, Jonathan S. "Resequencing Cells in an ATM Switch," Washington University Computer Science Department, WUCS-91-21.

[9] Turner, Jonathan S. "An Optimal Nonblocking Multicast Virtual Circuit Switch," *Proceedings of Infocom*, 6/94.

[10] Turner, Jonathan S. "Progress Towards Optimal Nonblocking Multipoint Virtual Circuit Switching Networks," *Proceedings of the Allerton Conference*, 9/93, pp. 760–769.

[11] Turner, Jonathan S. "Multicast Virtual Circuit Switch Using Cell Recycling," U.S. Patent #5,402,415, March 28, 1995.

[12] Turner, Jonathan S.. "Maintaining High Throughput During Overload in ATM Switches," *Proceedings of Infocom*, March 1996.