Prizma switch technology

A. P. J. Engbersen

Advances in packet-switching technology have enabled the rapid growth of the Internet and will be crucial for its continued expansion in the future. For almost ten years, the Prizma switch architecture has been a main component of IBM network technology, and today it is a cornerstone of the IBM Microelectronics Division's engagement with leading network product providers. In this paper we review the evolution of the Prizma architecture over the last decade and discuss the PowerPRS™ range of switch-chip products that are based on the Prizma architecture.

Introduction

The Internet has established itself as a worldwide communications medium for a broad spectrum of communication modes: data, voice, video—both asynchronously and in real time. Its growing popularity for entirely new applications in the field of e-business and entertainment, as well as its emerging use for well-established applications such as telephony, have resulted in a spectacular annual growth factor of at least 4 (and rising [1]) for traffic carried by the net. The Internet is able to grow at this enormous pace because of the exponential growth of optical transmission bandwidth made possible by wavelength division multiplexing (WDM) and commensurate progress in the packet-forwarding capabilities of network nodes. Early routers, the workhorses of Internet traffic forwarding, were based on computer-like architectures in which line cards were connected via buses to main memory, the CPU, and one another. Packet data had to flow at least twice (if not more) over the system bus, creating a severe bottleneck. **Figure 1** shows the anatomy of a modern router with its main functional units: *line interfaces*, which physically attach a multiplicity of transmission technologies to this communication system node and provide framing functionality; *network processors*, which provide the intelligence and processing power to analyze packet headers, look up routing tables, classify packets on the basis of their destinations, source addresses, and other control information and (often complex) rules, as well as providing queueing and policing of packets; *switch fabric*, which provides high-speed (ideally nonblocking) interconnection of the node packet processing units; and the *system processor*, which performs control point functions such as route computation and box and network management. In this paper we concentrate on the development of the switch fabric architecture.

Since the introduction of optical fibers in transport networks, serial time-division multiplex (TDM) transmission speed has grown exponentially at a rate of about 30% per year, and has reached 40 Gb/s today [**Figure 2(a)**]. The vast majority of these fibers deploy the SONET (synchronous optical network) technology in the U.S. and the SDH (synchronous digital hierarchy) technology in Europe and Japan. Both are standardized methods. The speed increase is gated primarily by the electronics of the receivers, which suggests that the data rates will level off in the not-too-distant future: Despite amazing progress in high-speed CMOS, SiGe, and GaAs technologies, it is hard to imagine today that serial transmission rates of commercial transmission systems will rise much above 100 Gb/s because of the intrinsic complexity and the resulting costs of the receiver electronics. Moreover, wavelength division multiplexing is a much lower-cost alternative to drive up fiber transmission utilization. Deployment of WDM transmission technology brought about a radical change: The overall transmission capacity of fiber links began to grow at a rate of about 200% per year, and has reached 1.6 Tb/s (160 times 10 Gb/s or 40 times 40 Gb/s) today. WDM technology is deployed pervasively in the core transport networks and is poised to emerge in metropolitan networks. The WDM capacity trend is expected to continue for some time, although, in the longer term, physical limits will cause saturation—probably of the order of about 50 Tb/s.

The port speeds of routers inevitably had to follow the speed increase of serial transmission over fibers

0018-8646/03/$5.00 © 2003 IBM

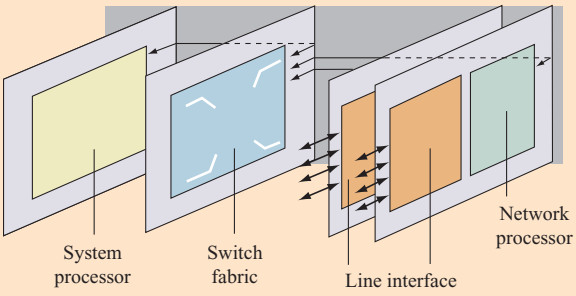

Figure 1

Anatomy of a switch or router. Reprinted with permission from [2]; © 2001 IEEE.

[**Figure 2(b)**]. This was made possible by advances in CMOS technology combined with design optimizations in packet processing and switching hardware. The proliferation of WDM transmission systems gives rise to an interesting question: Are routers going to deal with the multiplication of fiber transmission capacity by a corresponding increase in port speeds? For reasons that will become clear in the discussion below, we are convinced that port speed will continue to grow at the pace of fiber serial transmission, but that the necessary growth in packet forwarding capacity will come from an increase in node size rather than from an increase in node port speed [**Figure 2(c)**].

Building larger systems implies two necessary improvements: distributing the packet processing over more processing units and realizing switches with many more input and output ports than are available in today's designs [**Figure 2(d)**].

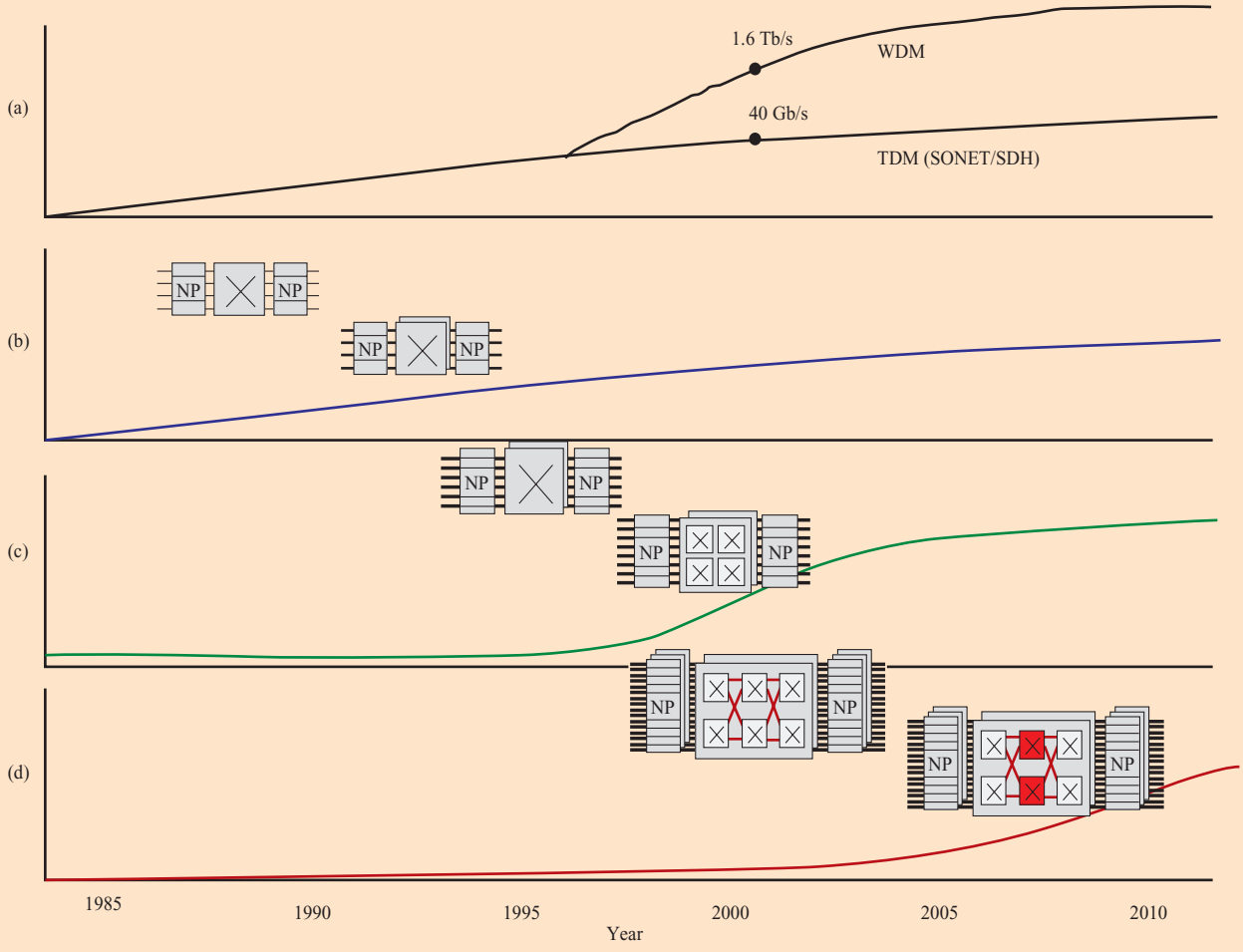

Figure 2

Evolution of transmission and network node technologies: (a) Transmission bandwidth on fiber; (b) port speed; (c) node size; (d) distribution and use of optics. (NP = network processor.) Reprinted with permission from [2]; © 2001 IEEE.

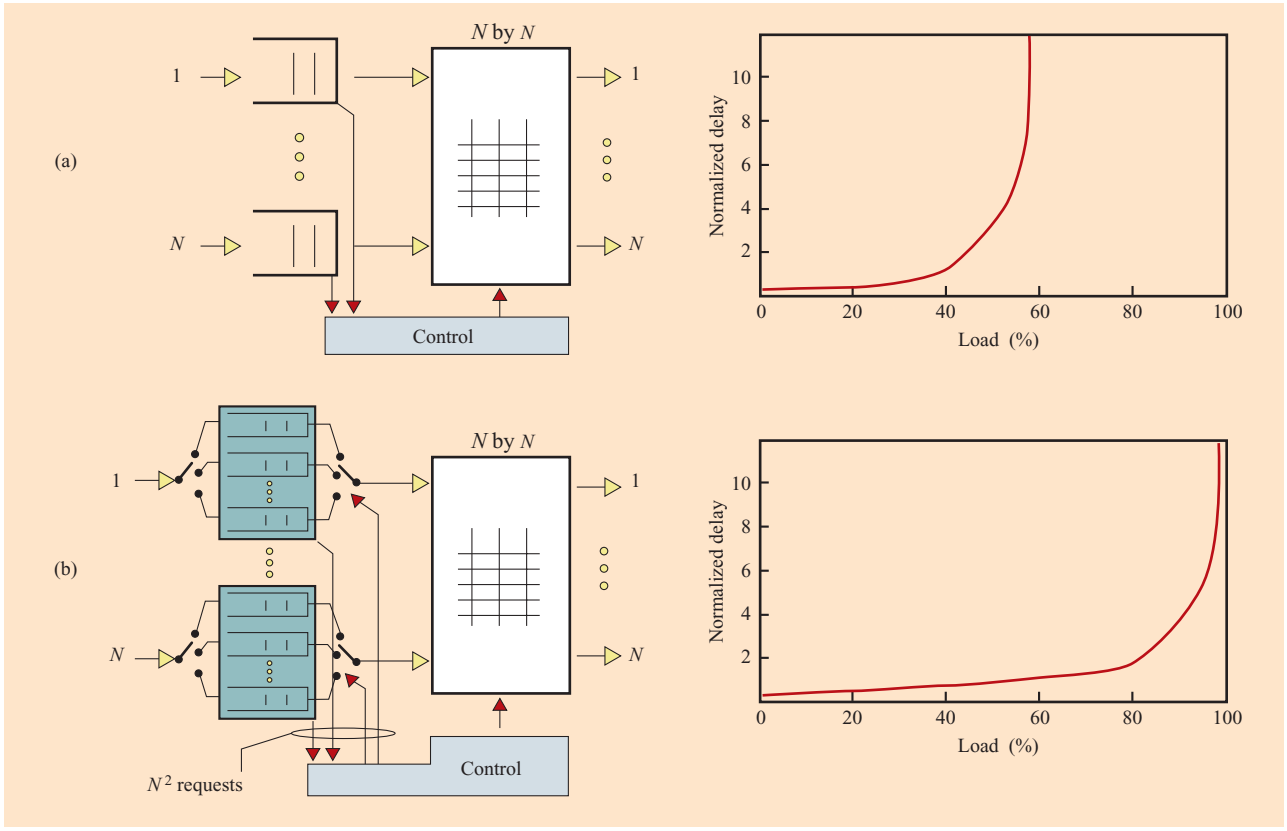

Figure 3

(a) Input queueing. (b) Input queueing with virtual output queueing.

In this paper we review the technological and architectural developments in switch technology that allow us to continue addressing the challenges presented in Figure 2.

Basic switch architectures

The two basic functions of a packet switch fabric are the spatial transfer (switching) of packets from their incoming ports to the destination ports and the resolution of contention, which occurs when two or more packets address the same output at the same time. (We assume here that an architecture that randomly picks one packet out of the set of contenders and drops the others, thus leading to a 37% packet drop under uniformly distributed, random traffic assumptions [3], is not acceptable.) The two classic single-stage packet switch architectures are characterized by the temporal order of queueing and switching functions [4]. Queueing before switching is called *input queueing* (IQ) [**Figures 3(a), 3(b)**], and switching before queueing is *output queueing* (OQ) [**Figures 4(a), 4(b)**]. These two architectures have different performance characteristics, as shown in the graphs in Figures 3(a) and 4(a). For uniform Poisson distributed traffic, OQ achieves 100% throughput with infinite first-in, first-out (FIFO) output buffers, whereas IQ is limited to approximately 58% throughput because of the head-of-the-line (HOL) blocking phenomenon [4]. For nonuniform or "bursty" traffic, the efficiency of IQ can be even worse. In both cases, assuming finite buffers may cause packet losses. The attractiveness of IQ lies in its simplicity and low cost: The queues are only required to support a throughput roughly equal to the line speed, whereas in the OQ case each queue must be able to accept the aggregate rate of all inputs. In the early days of fast packet switching, however, performance was the reason why many designs adopted the OQ concept in spite of the more complex and expensive multiport buffers required (e.g., the Bell Laboratories Knockout [5], the NEC Atom [6], and the Siemens Sigma as well as the IBM high-performance switch fabric [7] and Prizma switch [8].

Both of these basic switch architectures have been improved over time to overcome their respective deficiencies: The HOL blocking was addressed by a scheme called "*virtual output queueing*" for which the early

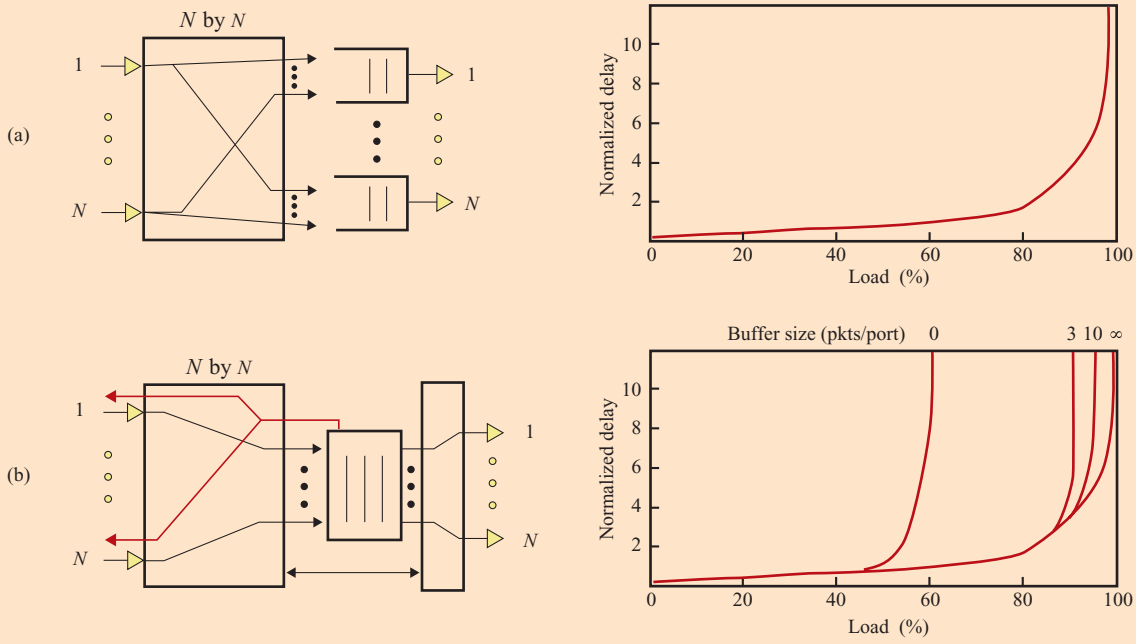

Figure 4

(a) Output queueing. (b) Shared output queueing.

work, dating back to a 1984 patent, was done by McMillen [9] and by Tamir and Frazier [10]. The approach circumvents the HOL problem by sorting the packets in input queues according to output destination and offering N^2 requests to the switch controller [Figure 3(b)]. A practical approach was used in the IBM 3746 Communications Controller and described by LaMaire and Serpanos [11]. Under the assumption that the ideal controller servicing the N^2 requests could be built such that under all circumstances it guarantees maximum use of the crossbar fabric and fairness, this switch architecture would achieve ideal performance [Figure 3(b)]. Despite intense research [12–17], however, the ideal algorithm has not yet been achieved.

A careful inspection of the OQ architecture in Figure 4(a) reveals that, although every single output queue has N inputs, it will never be the case that on each of these N^2 inputs a packet will arrive at the same time (except when all switch input ports are receiving simultaneously broadcast packets). This allows the architecture to be optimized via a single, larger memory and the sharing of all or part of the memory space among all—now logical—output queues. The result, also called a *shared output-buffered* switch, has become the main architecture for output-buffered switches: Hitachi [18, 19], IBM Vulcan [20], and follow-ons, Atlas I [21, 22] and the IBM Prizma [8, 23], are among the long list of shared output-buffered switches. It should also be noted that this architecture is ideally suited to support multicast because every memory location is connected to every input port and every output port. Figure 4(b) shows the architecture and performance characteristics of a shared output-buffered switch. A crucial observation is that ideal throughput is achieved—as always—only when the shared buffer is infinite. Since the output buffer requires a throughput of $2N$ times the individual line rate, in practice only on-chip buffers are suitable to address this task. This obviously limits the size of the buffer memory. Fortunately, with only a moderate amount of buffering (say 8 to 10 packets per output) the performance of such a switch is drastically improved over having zero buffer (i.e., an IQ switch). At the same time, this is a major drawback of the shared output-buffered architecture: If a switch is engineered for a certain packet size, an increase of either packet size or packet length variation, "burstiness," in the traffic (which have the same effect) will reduce the performance of the switch. In particular, one could argue that, depending on the momentary packet size, the switch will operate on a different performance characteristic—certainly not a very desirable behavior. It can be argued that input-buffered switches have the same drawback. However, input queues require a significantly lower aggregate throughput and are

therefore likely to be implemented with normal memory chips on the network processor subsystem. This also means that this memory can relatively easily be extended to accommodate increased burstiness of traffic.

Prizma architecture

Prizma pioneered the separation of control and data paths in its internal architecture. **Figure 5** shows this architecture, which operates as follows. A free address pool contains all of the addresses of free packet buffer locations. Each input router is given an address out of this pool, such that when a packet arrives, it is directed to the free packet buffer location, and a copy of the packet header, together with the address of the packet buffer where the packet is stored, is sent to the control section. In the control section these packet-buffer pointers are stored in the output queue indicated by the packet header, which is then discarded. The key innovation is that the relatively small address pointers can be placed sequentially in the output queues, whereas the longer packet data "shifts" to the packet buffer location.

Once the pointers move through the output queue, they are fed to the output selectors and the packet data shifts out of the packet buffer and into the destined output, while the pointers are fed back to the free address pool. Adding multicast to this scheme is straightforward [24] and uses only one packet buffer location, but multiple pointers in multiple output queues. Inspection of the information that flows between the control section and the data section (interface AA′ in Figure 5), shows that besides headers flowing from the data to the control section, the control section sends only (rather small) pointers to the data section. This allows multiple data sections to operate in parallel, each working on a part of the packet data, thus increasing the system throughput (**Figure 6**). Of course, the more data sections are put in parallel, the higher the total throughput will be. When the packet size does not increase according to the number of data sections put in parallel, the effective packet time, i.e., the time it takes to store a full packet, decreases because each data section receives only a fraction of the packet. The limit is reached when the control section can no longer process all pointers within the reduced packet time. We call this mode of operation "speed expansion." In the early Prizma switch chips, the speed-expansion limit was approximately 4; i.e., four data sections were controlled by one control section. In practice, identical chips were used to achieve speed expansion whereby one chip (the "master") has both control and data section operational, whereas in the other chips only the data section was active (**Figure 7**).

As argued above, the ability for the number of ports to increase is an important capability, especially since WDM deployment started. Fundamentally, two approaches are

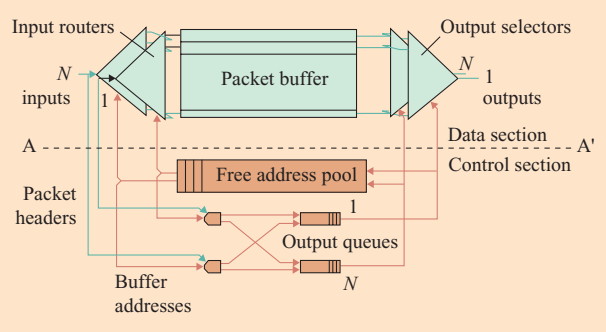

Figure 5

Prizma switch base architecture.

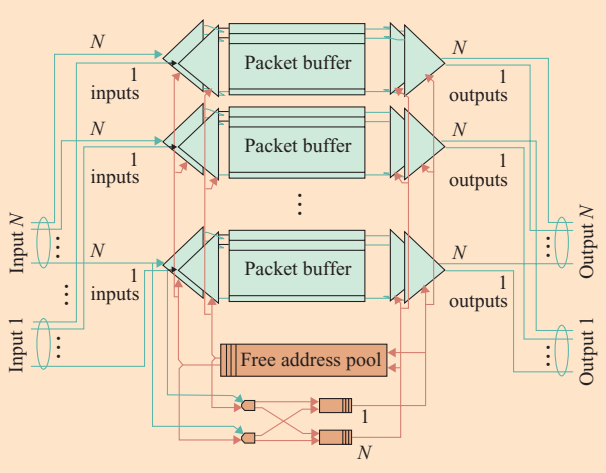

Figure 6

Prizma switch speed expansion.

supported by Prizma, the single-stage port expansion (see **Figure 8**) and multistage expansion. Single-stage port expansion is of special importance because later versions of Prizma-based products use this approach on-chip. In this mode, four chips are used to double the number of switch ports according to the arrangement shown in Figure 8. The packets from the first set of ports are fed to two switch chips, and an address filter at each input port accepts only incoming packets, which address output ports 1 to N in the upper chip pair and outputs $N + 1$ to $2N$ in the lower chip pair. The programming of the address filters is indicated in Figure 8 by the select codes "0" and "1" attached to the corresponding chips. Furthermore, only one switch chip element at a time is allowed to feed an output line. Therefore, each output port is controlled by an external "grant" signal. Only activation of the grant

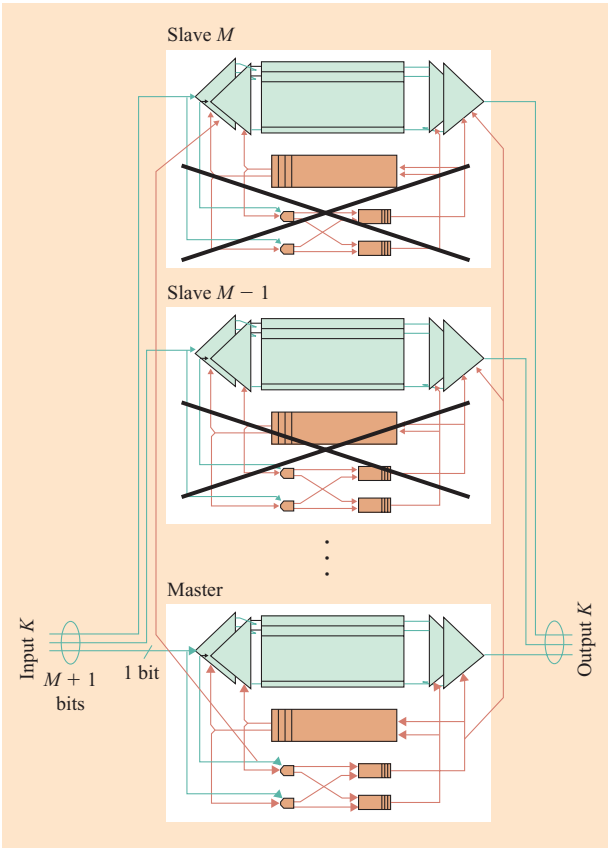

Figure 7

Prizma speed expansion with single port number.

will cause the output port to drive the chip pins. An external arbiter is needed to control the arbitration between two chips. In principle it is also possible to split this arbiter into two parts and implement the arbiter logic on-chip. In this arrangement, the resulting $2N \times 2N$ switch fabric retains its full single-stage characteristic. Because the output queues of two switch chips are actually used in parallel to feed one output port, the resulting performance is even better than with a single chip [25]. A disadvantage is the quadratic growth characteristic: A doubling of the number of ports requires a quadrupling of the number of chips required. For a switch fabric of moderate size, this is feasible. For larger switches, however, a multi-stage configuration must be chosen. The single-stage expansion and the auto-routing mechanisms used for multi-stage expansion have been inherited from an earlier switch project, known as the "tree switch" [7, 26].

In real networks, there exist links of a variety of speeds. It is necessary to aggregate and de-aggregate packets between links of different speeds. The Prizma architecture supports a mode, called link paralleling, in which two or four physical ports can be combined into one logical port to support this aggregation and de-aggregation. Packet order while traversing the switch is guaranteed, and the physical port number of the switch port determines the temporal order of the packets: first packet on the lowest-numbered physical port, second packet on the next-lowest-numbered switch port, etc. (**Figure 9**). Packets can be routed freely between (logical) links of different capacity and single-port links.

Probably the most important feature of the Prizma architecture is that the mechanisms to increase port speed and number of ports as well as to mix ports of different speed are freely combinable, within technology limits, to achieve a required switch-fabric design point. **Figure 10** shows how combinations of these expansion modes can be used to arrive at fabrics that have, for example, twice the number of ports and twice the port speed per port.

Any system with limited memory resources, such as the on-chip output queue buffer, requires flow-control means

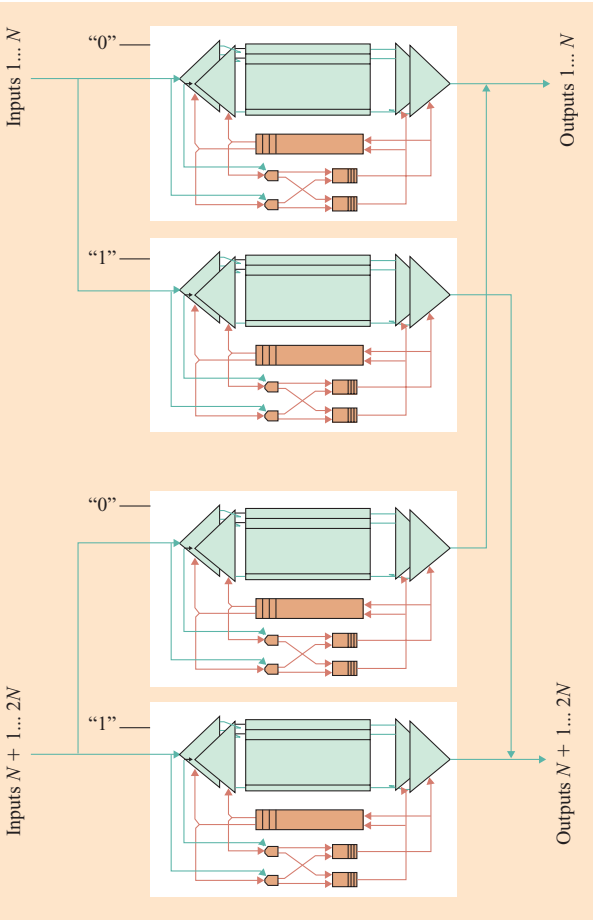

Figure 8

Prizma single-stage port expansion.

to prevent packet loss. At the system design level, Prizma allows lossless operation. In contrast to switch designs that take a loss rate under certain traffic assumptions as their basis for determining the on-chip output buffer size, Prizma originally provided a flow-control signal that becomes active when a packet cannot be stored internally in the buffer. The activation of this *back-pressure* signal indicates that the current packet arriving at this input port is not accepted (i.e., cannot be stored in the buffer memory) and could be retransmitted. With this signal, it is up to the input port adapter to decide whether this packet can be discarded, another packet sent, or this packet retransmitted.

We describe below several implementations of the Prizma architecture. Over time, details of the architecture have changed, often prompted by implementation choices and triggered by new possibilities made possible by advances in CMOS technology. We also discuss the extent to which the expansion modes were used to arrive at particular switch-fabric design points.

Prizma-based products

A range of products have been developed based on the Prizma architecture; we refer to them by their product marketing names.

First-generation products

PRS and PRS-P

The PRS (packet-routing switch) released in 1993 and shown in **Figure 11** was the first implementation of the Prizma architecture. It is a 14-mm × 14-mm CMOS 4S (0.8-μm feature size) chip, with 16-input and 16-output ports of 400 Mb/s each. It supports link paralleling for 2× and 4×, has 128 64-byte packet storage locations, and does not support internal priorities. In order to achieve its 6.4-Gb/s memory throughput, PRS employs on-chip bit-writable register arrays: A control line of the input router enables the bit-write control, and by stepping through the memory address range, the bits of ALL arriving packets are written sequentially into the memory in parallel. With its 128 8-bit-wide packet locations, the on-chip memory has a 50-Gb/s throughput capability at a cycle time of 20 ns. Because there are only 16 input routers, only 1/8 of this can be used. Two products built at the IBM La Gaude development laboratory use this PRS chip. **Figure 12** shows a card with the PRS chip visible in the middle, used in one of these products. The La Gaude development team built a four-way speed-expanded switch system with a 25.6-Gb/s aggregate throughput and released it about a year later, clearly showing the reduction in development time by using one or more of the expansion modes. **Figure 13** shows a single PCB with the four PRS chips and the necessary support chips for data transmission. In **Figure 14**, half of a 50-Gb/s switch system is shown, based on two 25-Gb/s cards. Besides being used in networking products from IBM, these cards formed the basis of the IBM Microelectronics Division's current OEM business in switching technology and verified the Prizma architecture.

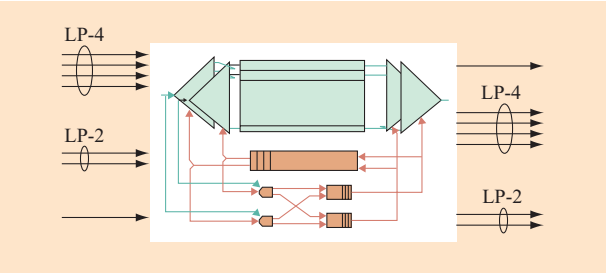

Figure 9

Prizma link paralleling (trunking).

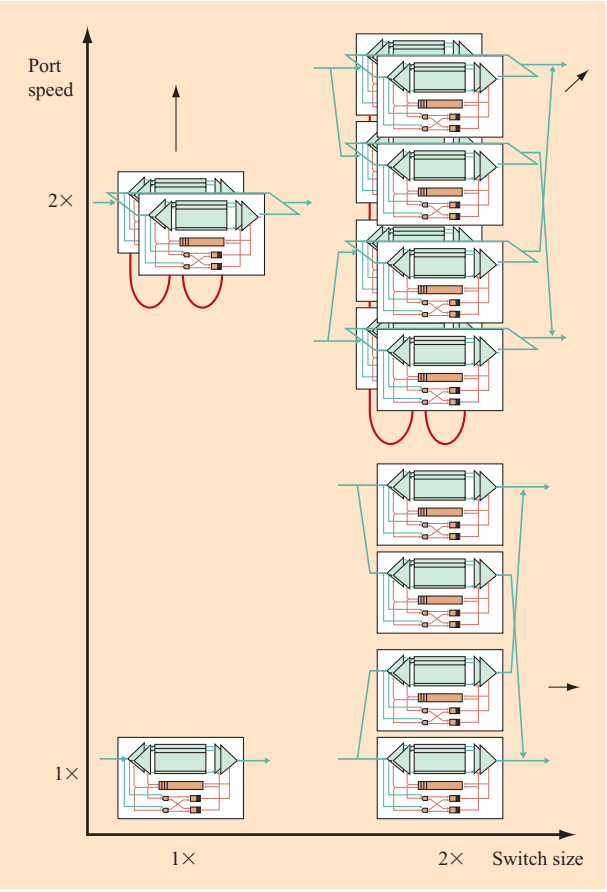

Figure 10

Prizma design space combining expansion modes.

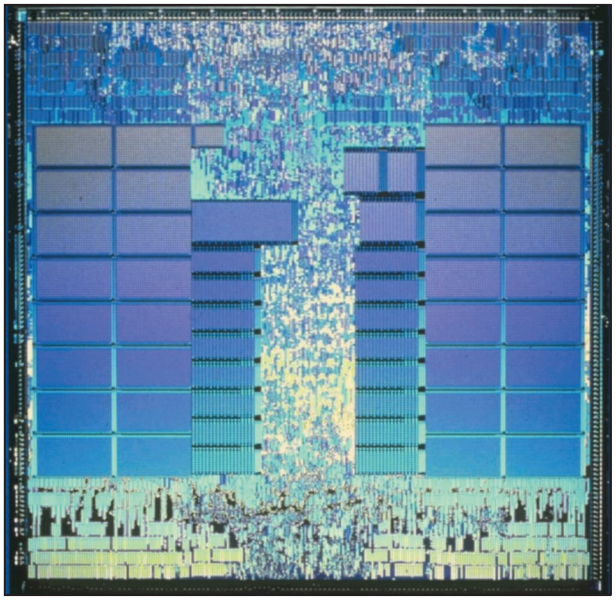

Figure 11

Photograph of packet-routing switch—PRS-P chip.

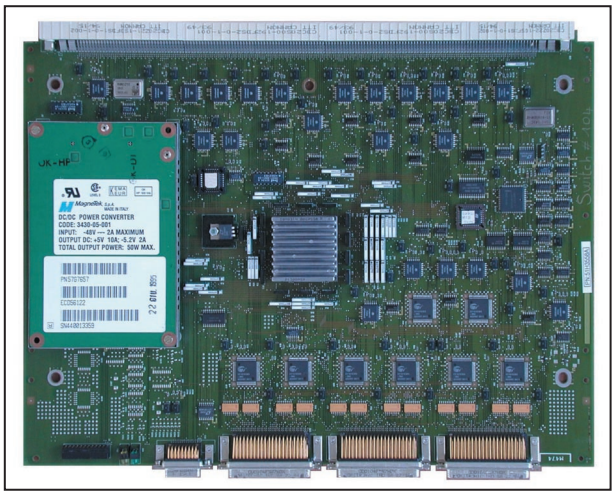

Figure 12

Photograph of single-chip 6.4-Gb/s switch card.

Priorities not supported in this first PRS chip later became a necessity, and a simple yet efficient way was found to limit the amount of lower-priority traffic in the FIFO queues: *nested thresholds*. Each output queue has one counter for the total queue occupancy and associated multiple thresholds, one threshold for each traffic class. When the counter for a certain class exceeds the class threshold, the back-pressure signal is activated upon receipt of a packet for the associated class and all lower-priority classes [27]. PRS-P implemented these nested thresholds. Priorities could now be supported, but it was not necessary to send additional signals out of the chip: Back-pressure remained a per-port signal, which is activated when the currently arriving packet is not accepted because the output queue or the traffic class or the on-chip memory is full. Consequently, new, improved functionality was introduced within the same package and pin layout.

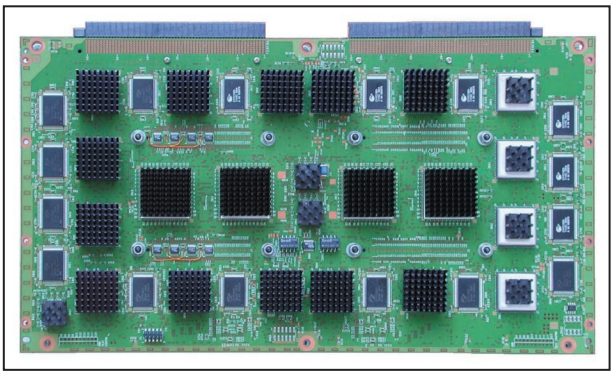

Figure 13

Photograph of quad PRS-P chip card—25 Gb/s using speed expansion.

Second-generation products

PRS-28.4

PRS-28.4 marks the second generation of Prizma switches. Whereas the first generation focused on achieving the highest possible throughput, PRS-28.4 was a deliberate small step back from the leading edge in terms of throughput in an effort to reduce costs. The challenge was to achieve close to 30 Gb/s of aggregate throughput while keeping intact all of the salient expansion modes of the original Prizma architecture. The memory bandwidth is attained by the traditional, yet unscalable, shared-bus approach. The limitations of the available on-chip memory bandwidth were overcome (see **Figure 15**) by using the two-way speed-expansion architecture option of the Prizma architecture on the same chip. The system was completed with an address manager (equivalent to the free-address queue), an output-queue read, and an output-queue access manager to implement the link paralleling and multicast features. Figure 15 shows an annotated chip image of PRS-28.4 and the logical architecture: One can clearly recognize the two memory structures described above. At constant packet size on the communication link, the speed-expansion concept causes each speed-expanded

chip to receive only a fraction of the packet. Special memory-access modes were implemented in the PRS-28.4 to use memory more efficiently in this case. This improved memory efficiency substantially over the first generation, which always allocated memory for a full packet, even if only partial packet data was received in speed-expansion mode. In PRS-28.4 the complete on-chip memory could be used in speed expansion with the commensurate performance improvement. PRS-28.4 allowed only a single stage of external speed expansion because one stage of speed expansion was already used by the on-chip implementation. This next generation of CMOS technology allowed more packet buffer memory to be packed onto the chip, thus improving the performance. It also allowed four preemptive traffic priorities to be implemented.

PRS-28.4 marks the introduction of serial I/O technology integrated directly into the switch chip to transport the packet data between the switch chip and the adapters and (if speed expansion is used) for the speed-expansion bus, which carries the signals from the control section to the multiple data sections. In the first generation, on-card (Figures 13 and 14) serial-to-parallel converters had already been used for the purpose of connecting to the adapters. At 30 Gb/s projected throughput, an I/O capacity of 60 Gb/s is required. Adding the necessary bandwidth for the speed-expansion signals to a slave chip increases this bandwidth by approximately 20%, resulting in a 72-Gb/s I/O bandwidth. Approximately one half of the 600-chip package I/Os in a single-ended driver design are available for data I/O, resulting in a 4-ns I/O pulse width. At these speeds it is not possible to construct a backplane with single-ended connection technology. A backplane is defined as approximately 1 meter of PCB trace and two high-quality connectors. A balanced solution would reduce the pulse width further (because the number of package I/Os is more or less fixed) and increase the power dissipation significantly over the 15-W target. Instead, a true asynchronous serial I/O technology at 440 Mb/s per balanced pair, targeted for 1-meter FR-4 card material and two card-to-card connectors, was used. By using four of these serial I/Os in parallel, a port bit rate of 1.77 Gb/s was achieved. All PRS-28.4 chips receive the same reference clock, implying that the serial I/O circuits have to perform only bit-phase alignment. To optimize chip-area real estate, a single phase-alignment controller was used per switch I/O port. More details can be found in [28]. In Figure 15 the areas showing high-speed drivers/receivers contain these circuits.

Second-generation Prizma architecture updates

The introduction of a serial link technology in PRS-28.4 required a rethinking of the back-pressure concept used in Prizma. Since there was now a clear-cut asynchronous,

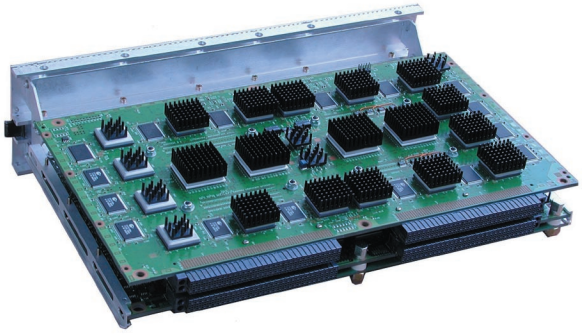

Figure 14

Photograph of half a 16-PRS-P switch—50 Gb/s using speed expansion and single-stage port expansion.

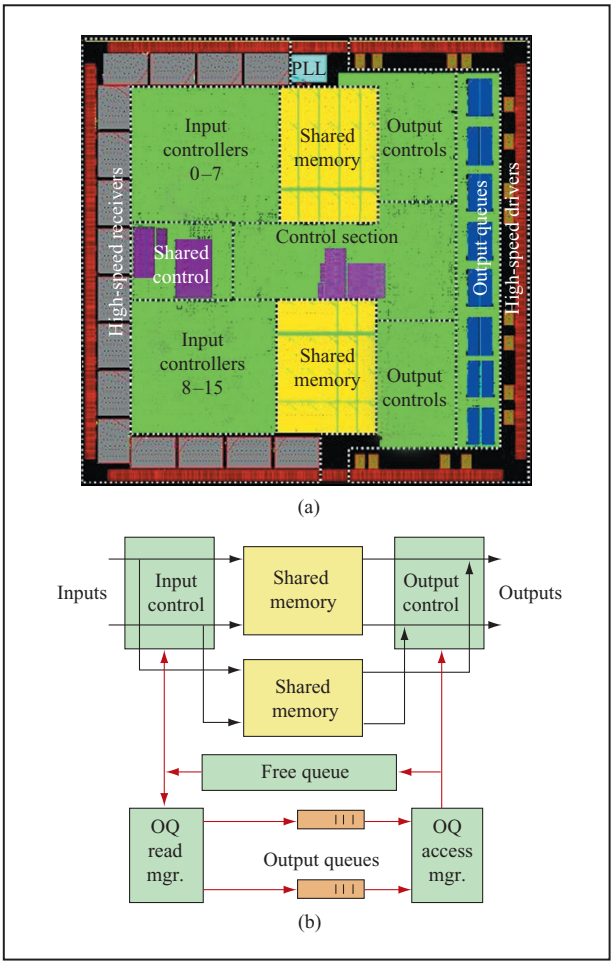

Figure 15

Packet-routing switch—PRS-28.4: (a) Annotated chip layout image; (b) color-coded architecture diagram.

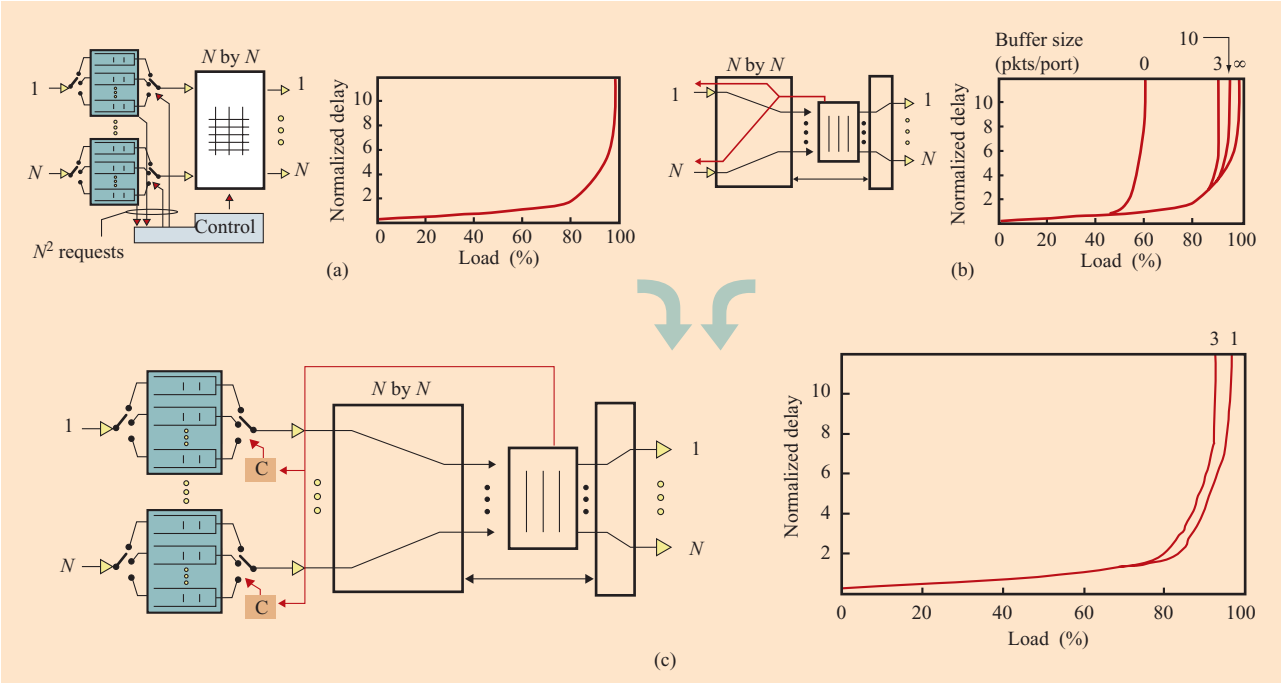

Figure 16

Virtual output queueing (VOQ) and output buffering: (a) VOQ principle; (b) limited shared output-buffering principle; (c) combination of VOQ and output buffering.

high-speed interface between adapter and switch, a level-sensitive, clock-edge-synchronized back-pressure signal would be difficult to integrate into this interface. At the projected 4-Gb/s port speed target (in speed expansion), a 64-byte packet would have a duration of only 16 ns—not enough time to ensure a stable back-pressure level indication at the sending adapter for the *current* packet cycle. In other words, the packet would have left the adapter completely before back-pressure would arrive and trigger a retransmission. It became necessary to change to output-queue grant signals, which are transported as bits in the headers of cells flowing from Prizma output ports to the adapters. Each output-queue grant signal is active as long as the respective output queue can receive packets. By transmitting these grant signals as bits in the outgoing switch packet headers, which flow back to the adapters in single-stage switch systems, all adapters are updated with the full chip output-queue grant status.

In doing so, however, it became possible to operate the system very much like *virtual output queueing* as used in crossbar switches to solve the HOL performance degradation, but without the need for the complicated central controller shown in Figure 3(b)! Further study showed that this approach achieved close to 100% throughput performance, with very little sensitivity to packet length (burstiness) and without the need for the central controller, which has not yet been shown to be implementable at acceptable complexity for high throughputs and port counts. **Figure 16** shows this principle along with the performance. Especially note the burstiness factor as a parameter of the load–delay performance characteristic. On the basis of the virtual-output-queueing architecture and the output-buffered architecture, a new architecture, *combined input/output queueing*, was designed. The output-queue status information is made available to every input port, such that each input port can make a local decision, using only the available packets in its own input queues and the above-mentioned output queue status. If multiple input queues happen to make their individual decisions such that multiple packets are sent to the same switch output port, the on-chip output buffer memory is used to resolve the resulting contention. It is no longer necessary to have a global arbitration, as was needed in the classical virtual-output-queueing crossbar shown in Figure 16(a). Nor is the on-chip output buffer memory used as a traffic queueing point, as was the case with the classical output-buffered architecture shown in Figure 16(b), but rather as a contention-resolving mechanism. Consequently, in this new architecture, the classical central virtual-output-

queueing scheduler of superlinear complexity is replaced by simple arbiters of complexity O(N) located at each input port. As indicated by extensive performance investigations, which are beyond the scope of this paper (the interested reader is referred to [29, 30]), this architecture achieves the highest throughput known in the industry today and has low burst sensitivity by effectively separating the contention resolution and buffering functions, such that the switch performs the former and the input queues cope with the bursts. Because of this fundamental change in the function of the output buffer memory of the switch, an optimum size of this buffer memory can be determined. Assuming that the output-queue grant signal back to the adapters is immediate (i.e., with no latency), each output buffer (associated with one switch output) can receive at maximum one packet from each input in every packet cycle. If we designate the number of input ports by N, every output buffer requires at least N packet storage locations. As soon as all N packet locations are empty, this output will activate its output-queue grant again, and any packets for this output will flow into the output queue again. The only reason to have more than N packet storage locations is to compensate for the latency in the grant-signal propagation time.

Sharing the output memory, as is done in classical output queueing switches to enhance the performance of the queueing point [31], is no longer meaningful when this memory becomes merely a contention-resolution point, as shown in [32]. In fact, [32] shows that a degradation of performance may even occur when sharing is allowed because of the likelihood that the output queue information fed back to the inputs is not correct under high loads. Consider an on-chip output queue with no packets queued. In this new architecture, this will result in a positive queue grant status being fed to the input adapters. Under high load conditions, one can safely assume that packets for this output will arrive within a few packet cycles at most, probably even from more than one input. At the same time, for high loads and with allowed sharing, other outputs may have "borrowed" queue positions from this empty output queue. These two effects increase the probability that this output queue will receive many packets and quickly cause an overflow of the total on-chip packet memory.

PRS-64G

The second chip of the second generation reduced the internal cycle time by a factor of 2 to support twice the number of input and output ports. Improvements in technology allowed the clock of the serial I/O circuits to be increased to 500 MHz. PRS-64G is a 64-Gb/s part, with 32 ports and 2 Gb/s per port. A significant improvement in the handling of the various traffic classes was implemented in PRS-64G: Previous Prizma-architecture-based chips had

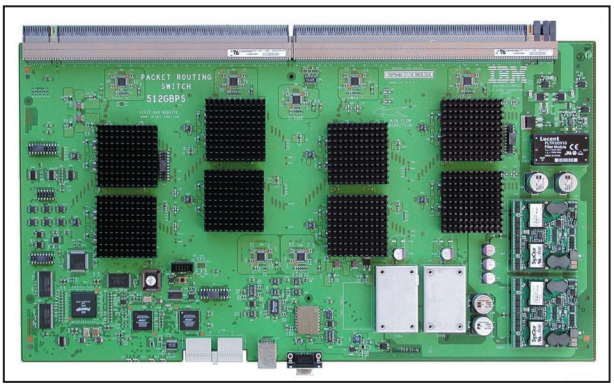

Figure 17

Photograph of a 512-Gb/s card based on eight-way speed expansion of PowerPRS Q-64G.

multiple priorities implemented using the nested threshold principle described above and a strict preemptive output queue priority scheduling at packet boundaries to select the next packet from the output queue. This strict scheduling has the disadvantage that under high loads, lower-priority traffic could suffer from starvation. In PRS-64G the concept of a *credit table* was introduced: Each priority level is associated with a number P. The address pointer to the credit table is increased by 1 every time a packet has been served, and wraps back around to 0. Every credit table entry contains a number P, which determines that this slot is for the associated priority P. If there is a packet of priority P in the output queue, that priority is served; otherwise, a packet of the highest non-empty priority is served. In this system, priorities can be given a guaranteed minimum share of the output bandwidth.

PowerPRS Q-64G*

The PowerPRS Q-64G is the third chip of the second generation. PowerPRS Q-64 uses the same architecture as PRS-64G but has improved the cycle time of the control section on-chip to allow up to eight data sections to be employed (see Figure 6). Having 32 input and 32 output ports of 2 Gb/s each per chip, with eight chips in parallel (speed expansion), results in a 32-port system of 16 Gb/s per port, sufficient to support OC-192c (10-Gb) communication links. Although this is a 512-Gb/s throughput system, high port speeds at lower total system throughputs can also be achieved by using the link paralleling method described above; e.g., with two chips in speed-expansion mode and four-way link paralleling (on every port), an 8 × 8 OC-192 system can be built. It is also possible to have only one OC-192 port and use the remaining 28 ports for OC-48 traffic. **Figure 17** is a

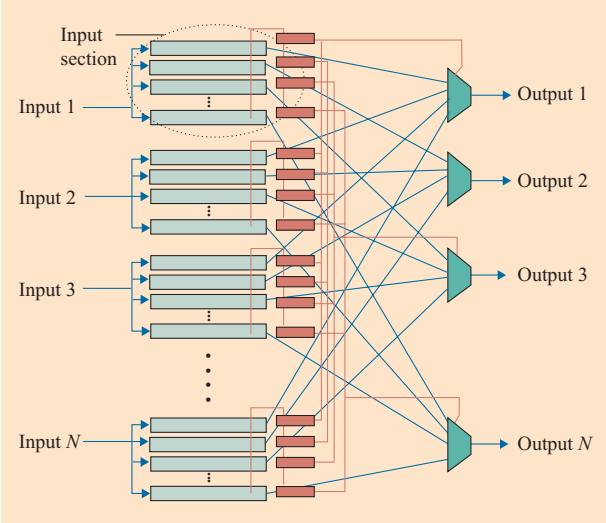

Figure 18

Implementation architecture of third-generation Prizma.

photograph of the 512-Gb/s eight-way speed-expanded switch fabric system.

Third-generation Prizma

Owing to the deployment of WDM, as argued above, there is less pressure to increase the port speed and more pressure to increase the number of ports to achieve a higher total switch capacity. With OC-192 (10-Gb/s) line cards, a 2-terabit/s (Tb/s) system would contain some 200 line cards. If we correct the 2-Tb/s throughput number for internal fabric packet header overhead, etc., such a system would support at least 128 OC-192c line interface and network processor cards (also known as *adapters*). Systems of the first and second generations, as described above, would reach less than one-quarter of this (i.e., 32 adapters at maximum). Thirty-two adapters, each the size of approximately two A4 sheets of paper, fit fairly easily in a two-shelf rack and still leave enough space for a switch fabric card, as shown in Figure 17. The packaging of the first and second generations posed no significant problems concerning round-trip times and flow control. However, 128 to 256 adapters require systems spanning multiple racks and require cables or fibers between the central switch fabric rack and the adapter racks, which easily reach several tens of meters in length. Although one could argue that large switch systems for telephony and even computer interconnect (for example, IBM ESCON* Director) have been built in the past, one should recognize that these are circuit switches, in which the connection between an input and an output is made for a relatively long duration of time (several seconds to hours), whereas for high-performance packet switches, the connection between an input and an output is set for only the duration of ONE relatively small packet. Realistic assumptions of 2×10^8 m/s propagation speed in coaxial (or fiber-based) cables, 64-Gb/s port speeds and 64-byte switch packets result in a "length" of such a packet of approximately 0.5 meter. For the cable lengths assumed, this means that there are twice as many packets "on the cable" as the cable length is measured in meters! Recalling the operation shown in Figure 16(c), this means that there is a significant number of packets underway until the simple arbiter on every adapter receives notification that the switch queue is full. In practice, this means that the output queue controller must negate the grant signal such that twice the number of packets that "fit" on the cable can still be stored after negation. By means of a credit mechanism, the number of required storage locations can be reduced by a factor of 2. Since on-chip packet storage comes at a premium, a credit mechanism with the logic to perform on-chip buffer space accounting has been developed and integrated with the output-queue status mechanism shown in Figure 16(c).

The third-generation Prizma makes full use of the combined input/output queueing architecture: The on-chip buffer memory, at the targeted design parameters of 64 ports of 64 Gb/s each, becomes a significant challenge. Again referring to Figure 16(c), this amounts to an aggregate memory throughput of 8 Tb/s! At a 1-ns memory cycle time, this requires an 8000-bit-wide memory bus (single-port memory assumed). Not only is this an insurmountable challenge, even with the latest dense silicon technology, or with two-port memories; it would also require that up to 8 or 16 switch packets (of 64 bytes each) be aggregated on this parallel bus and transported between the input ports, memory, and output ports. The amount of control and multiplexing logic is likely to prevent operation at the required 1-ns cycle time. A solution was developed using the fact that the new combined input/output queueing architecture no longer requires the shared output buffer design for performance reasons, as argued earlier. When the buffer memory is not fully shared, there is no longer a need for full connectivity between all memory locations and the input and output ports. Three options are then open: partition the memory per output port, partition the memory per input port, or partition the memory per input and per output port. Since the last option yields the most flexible solution and also provides flexibility with respect to possible implementation synergies between parts in adjacent input and output ports, we have chosen this approach, as shown in **Figure 18**. In particular, recalling the color-coding from previous figures, note how the control section has been distributed in a way similar to the data memory. As is the case with the data memory throughput, the control section must also

be distributed in order to achieve acceptable cycle times. In particular, the total memory for one output (i.e., the sum of the respective segments in each input section) is managed as one output queue, and the appropriate output queue status is generated for the virtual input queue arbiters. The total memory for one input (i.e., the entire input partition) is managed as one memory for determining the outstanding flow-control credit. The combined input/output queueing architecture is clearly crucial to establishing this distribution because it allows a maximum decoupling of memory resources and flow control. The resulting structure resembles buffered crossbar architectures [33–35], which have a long history and have recently become more attractive because of the improvements in silicon technology.

Increasing system throughput creates ever-greater challenges in terms of power consumption. Both total power consumption and per-chip power consumption are limited to around 2000 W per rack and a maximum of 35 W per chip. The global power consumption limit encourages the use of as few chips as possible. A single chip with 1000–2000 I/O pins is limited in its aggregate data throughput by the maximum I/O speed per pin. Practical numbers must assume differential I/O technology, especially as long as there is not yet a viable, cost-effective alternative available for copper interconnects. Under these conditions, less than 500 (bidirectional, symmetric) links can be supported from a single chip. Assuming 500 links at 2.5 Gb/s per link, the maximum capacity is 1.25 Tb/s per chip, and at 5 Gb/s per link, 2.5 Tb/s per chip. The third-generation Prizma addresses this requirement by integrating the long cable drivers directly into the switching chips, thus eliminating intermediate redrive chips. The per-chip power limit is addressed by exploiting the speed-expansion concept, resulting in a multichip solution.

The future

Advances in WDM technology have significantly increased the capacity of fibers and the number of channels to be switched for the same number of fibers. Prizma has leveraged advances in architecture, implementation, and silicon technology to evolve from a single 6-Gb/s chip to multi-terabit single-stage switch systems that can support a few hundred ports of OC-192 (**Figure 19**). As outlined above, the serial-link technology will determine the switch chip throughput more than any other technology parameter. Chip packages with more than 1500 pins are very expensive, and in the presence of an alternative such as speed expansion, multiple smaller chips with lower I/O count are more economical and flexible. All of these considerations lead us to believe that in the coming decade, the limit for single-stage solutions will be somewhere in the 4–16-Tb/s aggregate throughput regime.

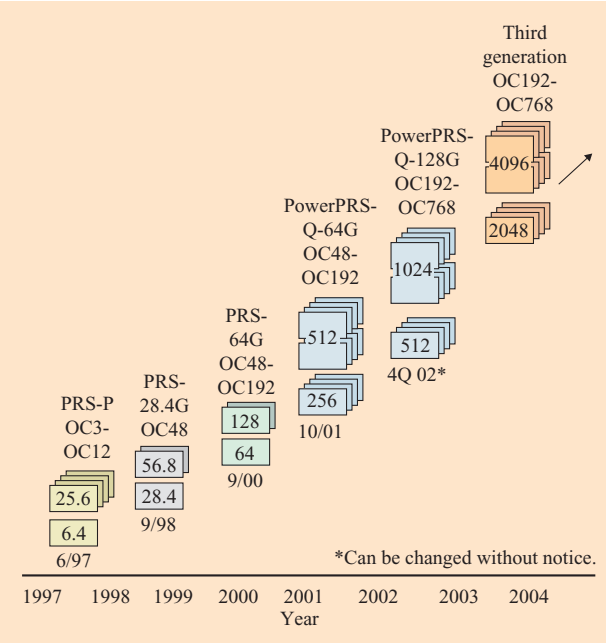

Figure 19

PRS and PowerPRS roadmap.

The high-speed interconnection of chips, boards, and racks, and the associated power and space limitations, will be among the most challenging problems in switch system design. Building larger switches requires multistage architectures with a commensurate increase in interconnect challenges. As long as there is no viable optical alternative for the electrical forwarding and switching on a packet-by-packet basis, electronics will play a paramount role in routers and switches. A hybrid electro-optical combination of fast circuit switching interconnects (likely all-optical switches) between—possibly lower-capacity—electrical switches is a promising possibility. In the absence of optical memory and logic of sufficient capacity, hybrid switches will be with us for a long time to come.

Conclusion

We have described the history of the Prizma switch architecture and its evolution from a pure output-buffered switch with back-pressure to a robust switch for Internet traffic using a balanced combination of input queueing and output buffering in the presence of long cables between switch core and adapters. Implementations of the architecture have been presented where appropriate, and the synergies between implementation and architecture development have been highlighted. It has been argued that optical switching can make some inroads into larger switch systems, but only to the extent that a hybrid

optical-electrical multistage switch is built to achieve aggregate capacities well beyond 10–50 Tb/s.

Acknowledgments

We are indebted to many colleagues in the IBM Research and Microelectronics divisions, especially those at the La Gaude center, who, throughout a decade of joint research and development, helped create the Prizma architecture and implementations, and Prizma-based systems. They all were instrumental in turning this project into a viable switch technology for IBM and its customers.

*Trademark or registered trademark of International Business Machines Corporation.

References

1. "The Network Letter: IP Traffic Growth—Revisited," *The ATM & IP Report*, July/August 2001, Broadband Publishing; see *http://www.broadbandpub.com/atmreport/*.
2. W. Bux, W. E. Denzel, T. Engbersen, A. Herkersdorf, and R. Luijten, "Technologies and Building Blocks for Fast Packet Forwarding," *IEEE Commun. Mag.* **39,** 2–9 (January 2001).
3. J. K. Patel, "Performance of Processor-Memory Interconnections for Multiprocessors," *IEEE Trans. Computing* **30,** No. 10, 771–780 (October 1981).
4. M. J. Karol, M. G. Hluchyj, and S. P. Morgan, "Input vs. Output Queuing on a Space-Division Packet Switch," *IEEE Trans. Commun.* **35,** No. 12, 1347–1356 (1987).
5. Y. Yeh, M. G. Hluchyj, and A. S. Acampora, "The Knockout Switch: A Simple, Modular Architecture for High-Performance Packet Switching," *IEEE J. Sel. Areas Commun.* **5,** No. 8, 1274–1283 (October 1987).
6. H. Suzuki, H. Nagano, and T. Suzuki, "Output Buffered Switch Architecture for Asynchronous Transfer Mode," *Proceedings of the International Communications Conference* (*ICC'89*), Boston, June 1989, pp. 99–103.
7. H. Ahmadi, W. E. Denzel, C. A. Murphy, and E. Port, "A High-Performance Switch Fabric for Integrated Circuit and Packet Switching," *Proceedings of the IEEE Conference on Computer Communications* (*INFOCOM'88*), New Orleans, March 1988, pp. 9–18.
8. W. E. Denzel, A. P. J. Engbersen, I. Iliadis, and G. Karlsson, "Highly Modular Packet Switch for Gb/s Rates," *Proceedings of the International Communications Conference* (*ICC'92*), Yokohama, Japan, October 25–30, 1992, pp. 237–240.
9. R. J. McMillen, "Packet Switched Multiple Queue N × M Switch Node and Processing Method," U.S. Patent 4,623,996, filed October 18, 1984; published November 18, 1986.
10. Y. Tamir and G. Frazier, "High-Performance Multi-Queue Buffers for VLSI Communication Switches," *Proceedings of the 15th International Symposium on Computer Architecture* (*ACM SIGARCH*), Vol. 16, No. 2, May 1988, pp. 343–354.
11. R. O. LaMaire and D. N. Serpanos, "Two-Dimensional Round-Robin Schedulers for Packet Switches with Multiple Input Queues," *IEEE/ACM Trans. Networking* **2,** No. 5, 471–482 (October 1994).
12. C. Lund, S. Phillips, and N. Reingold, "Fair Prioritized Scheduling in an Input-Buffered Switch," *Proceedings of the Conference on Broadband Communications*, Montreal, Canada, 1996, pp. 358–369.
13. N. W. McKeown, "Scheduling Algorithms for Input-Queued Switches," Ph.D. thesis, University of California, Berkeley, 1995.
14. N. W. McKeown, "The iSLIP Scheduling Algorithm for Input-Queued Switches," *IEEE/ACM Trans. Networking* **7,** No. 2, 188–201 (April 1999).
15. N. W. McKeown, "Fast Switched Backplane for a Gigabit Switched Router," Cisco Systems white paper; see *http://www.cisco.com/warp/public/cc/cisco/mkt/core/12000/tech/fasts_wp.pdf*.
16. D. N. Serpanos and P. I. Antoniadis, "Firm: A Class of Distributed Scheduling Algorithms for High-Speed ATM Switches with Multiple Input Queues," *Proceedings of the IEEE Conference on Computer Communications* (*INFOCOM 2000*), Tel Aviv, Israel, March 2000, Vol. 2, pp. 548–555.
17. I. Stoica and H. Zhang, "Exact Emulation of an Output Queuing Switch by a Combined Input Output Queuing Switch," *Proceedings of the Sixth IEEE/IFIP Workshop on Quality of Service* (*IWQoS'98*), Napa Valley, CA, May 1998, pp. 218–224.
18. H. Kuwahara, N. Endo, M. Ogino, and T. Kozaki, "A Shared Buffer Memory Switch for an ATM Exchange," *Proceedings of the International Communications Conference* (*ICC'89*), Boston, June 1989, pp. 118–122.
19. T. Kozaki, N. Endo, Y. Sakurai, O. Matsubara, M. Mizukami, and K. Asano, "32 × 32 Shared Buffer Type ATM Switch VLSI's for B-ISDN's," *IEEE J. Sel. Areas Commun.* **9,** No. 8, 1239–1247 (October 1991).
20. C. B. Stunkel, D. G. Shea, B. Abali, M. G. Atkins, C. A. Bender, D. G. Grice, P. Hochschild, D. J. Joseph, B. J. Nathanson, R. A. Swetz, R. F. Stucke, M. Tsao, and P. R. Varker, "The SP2 High-Performance Switch," *IBM Syst. J.* **34,** No. 2, 185–204 (1995).
21. M. Katevenis, D. Serpanos, and P. Vatsolaki, "ATLAS I: A General-Purpose, Single-Chip ATM Switch with Credit-Based Flow Control," presented at the IEEE Hot Interconnects IV Symposium, Stanford, CA, August 15–17, 1996.
22. G. Kornaros, D. Pnevmatikatos, P. Vatsolaki, G. Kalokerinos, C. Xanthaki, D. Mavroidis, D. Serpanos, and M. Katevenis, "ATLAS I: Implementing a Single-Chip ATM Switch with Backpressure," *IEEE Micro Mag.*, pp. 30–41 (January–February 1999).
23. W. E. Denzel, A. P. J. Engbersen, and I. Iliadis, "A Flexible Shared-Buffer Switch for ATM at Gb/s Rates," *Computer Networks & ISDN Syst.* **27,** No. 4, 611–624 (January 1995).
24. A. P. J. Engbersen, "Multicast/Broadcast Mechanism for a Shared Buffer Packet Switch," *IBM Tech. Disclosure Bull.* **34,** No. 10a, 464–465 (March 1992).
25. H. Ahmadi, W. E. Denzel, C. A. Murphy, and E. Port, "A High-Performance Switch Fabric for Integrated Circuit and Packet Switching," *Int. J. Digital & Analog Cabled Syst.* **2,** No. 4, 277–287 (1989).
26. H. Ahmadi, J. G. Beha, W. E. Denzel, A. P. Engbersen, R. P. Luijten, C. A. Murphy, and E. Port, "High-Speed Modular Switching Apparatus for Circuit and Packet Switched Traffic," U.S. Patent 5,008,878, April 16, 1991.
27. P. Austruy, A. Fichou, C. Galand, and I. Iliadis, "Back Pressure Access Control System for a Shared Buffer with Allocation Threshold for each Traffic Class," U.S. Patent 5,838,922, 1999.
28. M. Colmant and R. P. Luijten, "A Single-Chip Lossless 16 × 16 Switch Fabric with 28 Gb/s Throughput," *Research Report RZ-3087*, IBM Zurich Research Laboratory, Rüschlikon, Switzerland, December 1998.
29. C. Minkenberg and T. Engbersen, "A Combined Input- and Output-Queued Packet-Switch System Based on PRIZMA Switch-on-a-Chip Technology," *IEEE Commun. Mag.* **38,** No. 12, 70–77 (December 2000).
30. C. Minkenberg, "On Packet Switch Design," Ph.D. thesis, Technical University Eindhoven, The Netherlands,

September 2001; also available as IBM Research Report RZ-3387, 2001.
31. I. Iliadis, "Performance of a Packet Switch with Shared Buffer and Input Queueing," *Teletraffic and Datatraffic in a Period of Change* (*ITC-13*), Copenhagen, Denmark, June 1991, A. Jensen and V. B. Iversen, Eds., Elsevier, Amsterdam, 1991, pp. 911–916.
32. R. P. Luijten, T. Engbersen, and C. Minkenberg, "Shared Memory Switching + Virtual Output Queuing: a Robust and Scalable Switch," *Proceedings of the IEEE International Symposium on Circuits and Systems* (*ISCAS*), Sydney, Australia, May 2001 (IEEE, Piscataway, NJ, 2001), pp. IV-274–IV-277.
33. R. Bakka and M. Dieudonne, "Switching Circuit for Digital Packet Switching Network," U.S. Patent 4,314,367, February 2, 1982.
34. S. Nojima, Y. Kato, T. Shimoe, K. Hajikano, and K. Murakami, "Experimental Broadband ATM Switching System," *Proceedings of the IEEE Global Telecommunications Conference* (*GLOBECOM'88*), Hollywood, FL, November 1988, pp. 1288–1292.
35. K. Yoshigoe and K. J. Christensen, "A Parallel-Polled Virtual Output Queued Switch with a Buffered Crossbar," *Proceedings of the Workshop on High Performance Switching and Routing*, Dallas, TX, May 2001, pp. 271–275.

Received April 30, 2002; accepted for publication September 20, 2002

Antonius P. J. (Ton) Engbersen *IBM Research Division, Zurich Research Laboratory, Säumerstrasse 4, CH-8803 Rüschlikon, Switzerland (apj@zurich.ibm.com).* Dr. Engbersen received his master's (EE) degrees from Eindhoven Technical University, The Netherlands, in 1978 and his Ph.D. from the Zurich Federal Institute of Technology in 1983, when he joined the IBM Zurich Research Laboratory. He was instrumental in bringing VLSI design skills to the laboratory in the mid-1980s, and in the early 1990s developed the PRIZMA switch architecture. PRIZMA has become a family of communication switch offerings IBM is marketing through its Microelectronics Division. He spent 1996 and 1997 at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, where he led the initial development of multi-protocol label switching (MPLS). Since 1997 Dr. Engbersen has managed the Network Technology Research Group at the Zurich Research Laboratory. His current research interests are in networking technology, scalable network processors, hardware and software, scalable switching technology, and SDH/SONET.