The Wayback Machine - https://web.archive.org/web/20040112004448/http://www.ic.uff.br:80/~michael/kr19…

4.6 What's inside a router?

Our study of the network layer so far has focussed on network layer service models, the routing algorithms that control the routes taken by packets through the network, and the protocols that embody these routing algorithms. These topics, however, are only part (albeit *important* ones) of what goes on in the network layer. We have yet to consider the **switching function** of a router - the actual transfer of datagrams from a router's incoming links to the appropriate outgoing links. Studying just the control and service aspects of the network layer is like studying a company and considering only its management (which controls the company but typically performs very little of the actual "grunt" work that makes a company run!) and its public relations ("Our product will provide this wonderful service to you!"). To fully appreciate what really goes on within a company, one needs to consider the workers. In the network layer, the real work (that is, the reason the network layer exists in the first place) is the forwarding of datagrams. A key component in this forwarding process is the transfer of a datagram from a router's incoming link to an outgoing link. In this section we study how this is accomplished. Our coverage here is necessarily brief, as an entire course would be needed to cover router design in depth. Consequently, we'll make a special effort in this section to provide pointers to material that covers this topic in more depth.

A high level view of a generic router architecture is shown in Figure 4.6-1. Four components of a router can be identified:

- **Input ports.** The input port performs several functions. It performs the physical layer functionality (shown in light blue in Figure 4.6-1) of terminating an incoming physical link to a router. It performs the data link layer functionality (shown in dark blue) needed to interoperate with the data link layer functionality (see Chapter 5) on the other side of the incoming link. It also performs a lookup and forwarding function (shown in red) so that a datagram forwarded into the switching fabric of the router emerges at the appropriate output port. Control packets (e.g., packets carrying routing protocol information such as RIP, OSPF or IGMP) are forwarded from the input port to the routing processor. In practice, multiple ports are often gathered together on a single *line card* within a router.
- **Switching fabric.** The switching fabric connects the router's input ports to its output ports. This switching fabric is completely contained with the router - a network inside of a network router!
- **Output ports.** An output port stores the datagrams that have been forwarded to it through the switching fabric, and then transmits the datagrams on the outgoing link. The output port thus performs the reverse data link and physical layer functionality as the input port.
- **Routing processor.** The routing processor executes the routing protocols (e.g., the protocols we studied in section 4.4), maintains the routing tables, and performs network management functions (see chapter 8), within the router. Since we cover these topics elsewhere in this book, we defer discussion of these topics to elsewhere.

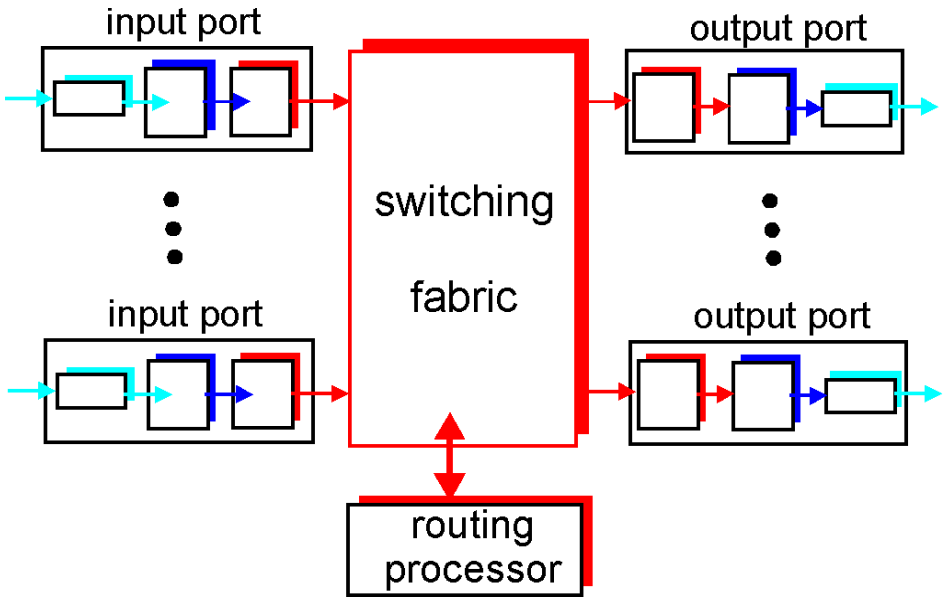


Figure 4.6-1: Router architecture

In the following, we'll take a look at input ports, the switching fabric, and output ports in more detail. [Turner 1988, Giacopelli 1990, McKeown 1997a, Partridge 1998] provide a discussion of some specific router architectures. [McKeown 1997b] provides a particularly readable overview of modern router architectures, using the Cisco 12000 router as an example.

4.6.1 Input ports

A more detailed view of input port functionality is given in Figure 4.6-2. As discussed above, the input port's line termination function and data link processing implement the physical and data link layers associated with an individual input link to the router. The lookup/forwarding function of the input port is central to the switching function of the router. In many routers, it is here that the router determines the output port to which an arriving datagram will be forwarded via the switching fabric. The choice of the output port is made using the information contained in the routing table. Although the routing table is computed by the routing processor, a "shadow copy" of the routing table is typically stored at each input port and updated, as needed, by the routing processor. With local copies of the routing table, the switching decision can be made locally, at each input port, without invoking the centralized routing processor. Such *decentralized* switching avoids creating a forwarding bottleneck at a single point within the router.

In routers with limited processing capabilities at the input port, the input port may simply forward the packet to the centralized routing processor, which will then perform the routing table lookup and forward the packet to the appropriate output port. This is the approach taken when a workstation or server serves as a router (e.g., [Microsoft 1998]); here, the "routing processor" is really just the workstation's CPU and the "input port" is really just a network interface card (e.g., a Ethernet card).

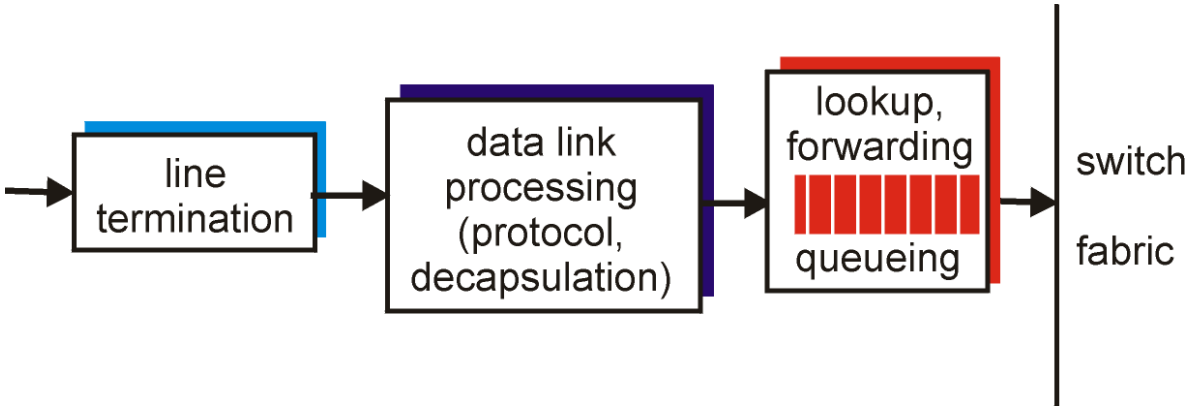


Figure 4.6-2: Input port processing

Given the existence of a routing table, the routing table lookup is conceptually simple -- we just search through the routing table, looking for a destination entry that matches the destination address of the datagram, or a default route if the destination entry is missing. In practice, however, life is not so simple. Perhaps the most important complicating factor is that backbone routers must operate at high speeds, being capable of performing millions of lookups per second. Indeed, it is desirable for the input port processing to be able to proceed at **line speed**, i.e., that a lookup can be done in less than the amount of time needed to receive a packet at the input port. In this case, input processing of a received packet can be completed before the next receive operation is complete. To get an idea of the performance requirements for lookup, consider that a so-called OC48 link runs at 2.5 Gbps. With 256 byte long packets, this implies a lookup speed of approximately a million lookups per second.

Given the need to operate at today's high link speeds, a linear search through a large routing table is impossible. A more reasonable technique is to store the routing table entries in a tree data structure. Each level in the tree can be thought of as corresponding to a bit in the destination address. To lookup an address, one simply starts at the root node of the tree. If the first address bit is a zero, then the left subtree will contain the routing table entry for destination address; otherwise it will be in the right subtree. The appropriate subtree is then traversed using the remaining address bits -- if the next address bit is a zero the left subtree of the initial subtree is chosen; otherwise, the right subtree of the initial subtree is chosen. In this manner, one can lookup the routing table entry in N steps, where N is the number of bits in the address. (The reader will note that this is essentially a binary search through an address space of size 2^N.) Refinements of this approach are discussed in [Doeringer 1996].

But even with N=32 (e.g., a 32-bit IP address) steps, the lookup speed is not fast enough for today's backbone routing requirements. For example, assuming a memory access at each step, less than a million address lookups/sec could be performed with 40 ns memory access times. Several techniques have thus been explored to increase lookup speeds. Content addressable memories (CAMs) allow a 32-bit IP address to be presented to the CAM, which then returns the content of the routing table entry for that address in essentially constant time. The Cisco 8500 series router [Cisco 1998a] has a 64K CAM for each input port. Another technique for speeding lookup is to keep recently accessed routing table entries in a cache [Feldmeier 1998]. Here, the potential concern is the size of the cache. Measurements in [Thompson 1997] suggest that even for an OC-3 speed link, approximately 256,000 source-destination pairs might be seen in one minute in a backbone router. Most recently, even faster data structures, which allow routing table entry to be located in log(N) steps [Waldvogel 1997], or which compress routing tables in novel ways [Degemark 1997], have been proposed. A hardware-based approach to lookup that is optimized for the common case that the address being looked up has 24 or less significant bits is discussed in [Gupta 1998].

Once the output port for a packet has been determined via the lookup, the packet can be forwarded into the switching fabric. However, as we'll see below, a packet may be temporarily **blocked** from entering the switching fabric (due to the fact that packets from other input ports are currently using the fabric). A blocked packet must thus be queued at the input port and then scheduled to cross the switching fabric at a later point in time. We'll

take a closer look at the blocking, queueing and scheduling of packets (at both input ports and output ports) within a router in section 4.6.4 below.

4.6.2 Switching Fabrics

The switching fabric is at the very heart of a router. It is through this switching that the datagrams are actually moved from an input port to an output port. Switching can be accomplished in a number of ways, as indicated in Figure 4.6-3:

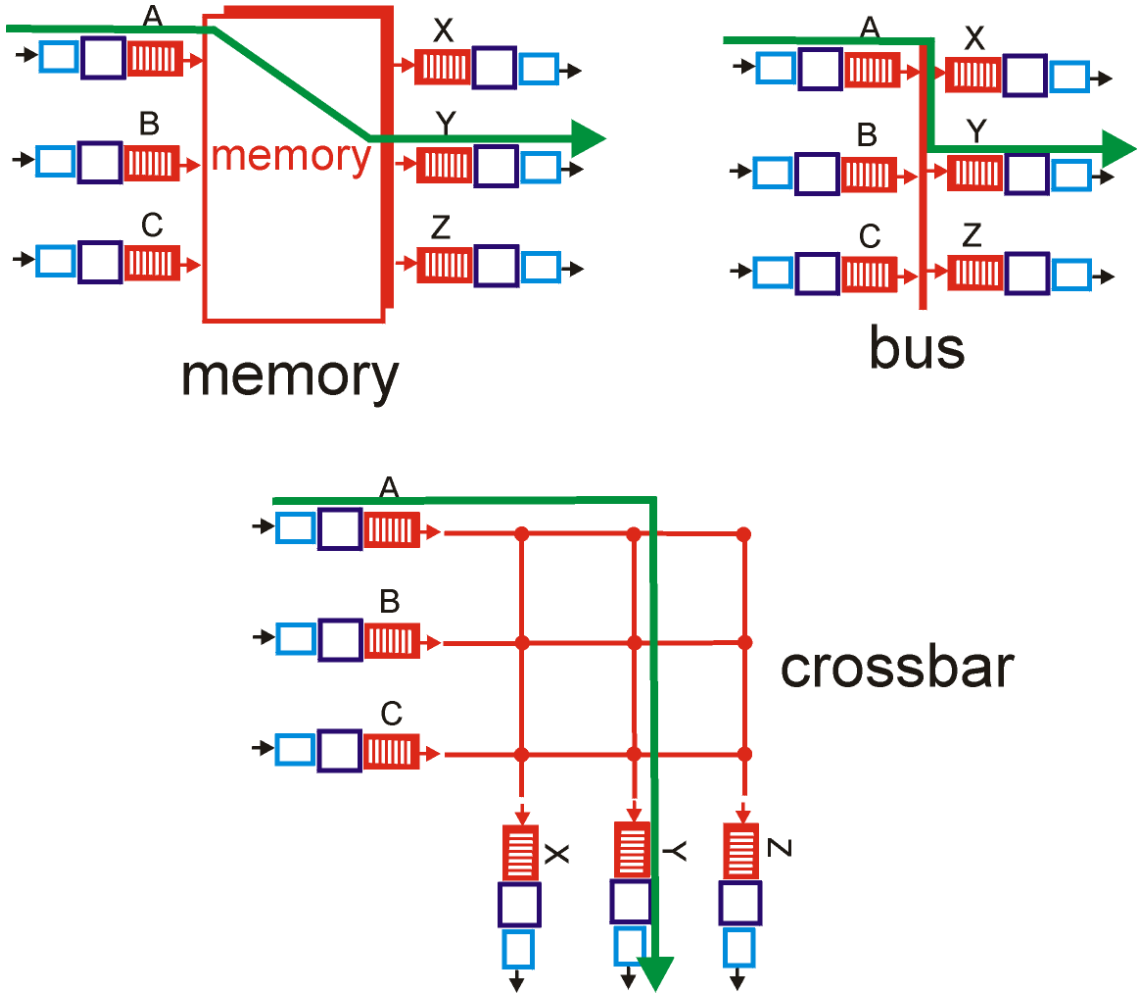


Figure 4.6-3: Three switching techniques

- **Switching via memory.** The simplest, earliest routers were often traditional computers, with switching between input and output port being done under direct control of the CPU (routing processor). Input and output ports functioned as traditional I/O devices in a traditional operating system. An input port with an arriving datagram first signaled the routing processor via an interrupt. The packet was then copied from the input port into processor memory. The routing processor then extracted the destination address from the header, looked up the appropriate output port in the routing table, and copied the packet to the output port's buffers. Note that if the memory bandwidth is such that B packets/sec can be written into, or read from, memory, then the overall switch throughput (the total rate at which packets are transferred from input ports to output ports) must be less than B/2.

IPR2025-01468; Citrix

Many modern routers also switch via memory. A major difference from early routers, however, is that the lookup of the destination address and the storing (switching) of the packet into the appropriate memory location is performed by processors on the input line cards. In some ways, routers that switch via memory look very much like shared memory multiprocessors, with the processors on a line line card storing datagrams into the memory of the appropriate output port. Cisco's Catalyst 8500 series switches [Cisco 1998a] and Bay Networks Accelar 1200 Series routers switch packets via a shared memory.

- **Switching via a bus.** In this approach, the input ports transfer a datagram directly to the output port over a shared bus, without intervention by the routing processor (Note that when switching via memory, the datagram must also cross the system bus going to/from memory). Although the routing processor is not involved in the bus transfer, since the bus is shared, only one packet at a time can be transferred over the bus at a time. A datagram arriving at an input port and finding the bus busy with the transfer of another datagram is blocked from passing through the switching fabric and queued at the input port. Because every packet must cross the single bus, the switching bandwidth of the router is limited to the bus speed.

Given that bus bandwidths of over a gigabit per second are possible in today's technology, switching via a bus is often sufficient for routers that operate in access and enterprise networks (e.g., local area and corporate networks). Bus-based switching has been adopted in a number of current router products, including the Cisco 1900 [Cisco 1997b], which switches packets over a 1Gbps Packet Exchange Bus. 3Com's CoreBuilder 5000 systems [Kapoor 1997] interconnects ports that reside on different switch modules over its PacketChannel data bus, with a bandwidth of 2 Gbps.

- **Switching via an interconnection network.** One way to overcome the bandwidth limitation of a single, shared bus is to use a more sophisticated interconnection network, such as those that have been used in the past to interconnect processors in a multiprocessor computer architectures. A crossbar switch is an interconnection network consisting of 2N busses that connect N input ports to N output ports, as shown in Figure 4.6-3. A packet arriving at an input port travels along the horizontal bus attached to the input port until it intersects with the vertical bus leading to the desired output port. If the vertical bus leading to the output port is free, the packet is transferred to the output port. If the vertical bus is being used to transfer a packet from another input port to this same output port, the arriving packet is blocked and must be queued at the input port.

  Delta and Omega switching fabrics have also been proposed as an interconnection network between input and output ports. See [Tobagi 90] for a survey of switch architectures. Cisco 12000 Family switches [Cisco 1998b] use an interconnection network, providing up to 60 Gbps through the switching fabric. One current trend in interconnection network design [Keshav 1998] is to fragment a variable length IP datagram into fixed length cells, and then tag and switch the fixed length cells through the interconnection network. The cells are then reassembled into the original datagram at the output port. The fixed length cell and internal tag can considerably simplify and speed up the switching of the packet through the interconnection network.

4.6.3 Output Ports

Output port processing, shown in Figure 4.6-4, takes the datagrams that have been stored in the output port's memory and transmits them over the outgoing link. The data link protocol processing and line termination are the send-side link- and physical layer functionality that interact with the input port on the other end of the outgoing link, as discussed above in section 4.6.2. The queueing and buffer management functionality are needed when the switch fabric delivers packets to the output port at a rate that exceeds the output link rate; we'll cover output port queueing below.

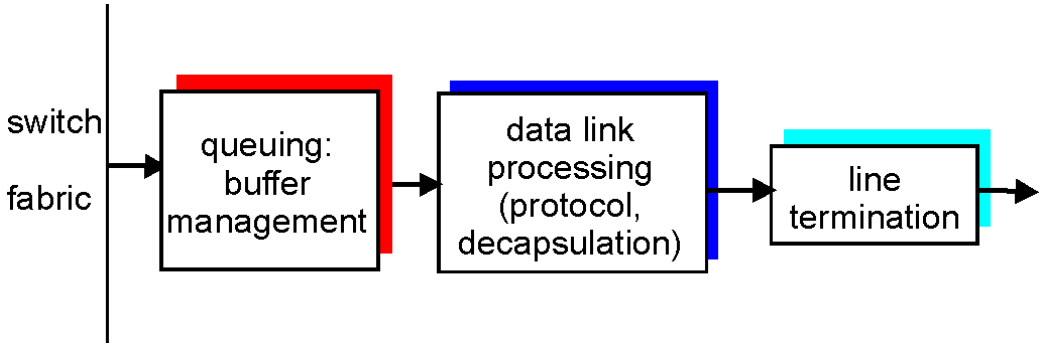


Figure 4.6-4: Output Port processing

4.6.4. Where Does Queueing Occur?

Looking at the input and output port functionality and the configurations shown in Figure 4.6-3, it is evident that packet queues can form at both the input ports *and* the output ports. It is important to consider these queues in a bit more detail, since as these queues grow large, the router's buffer space will eventually be exhausted and **packet loss** will occur. Recall that in our earlier discussions, we said rather vaguely that packets were lost "within the network" or "dropped at a router." It is here, at these queues within a router, where such packets are dropped and lost. The actual location of packet loss (either at the input port queues or the output port queues) will depend on the traffic load, the relative speed of the switching fabric and the line speed, as discussed below.

Suppose that the input line speeds and output line speeds are all identical, and that there are n input ports and n output ports. If the switching fabric speed is at least n times as fast as the input line speed, than no queuing can occur at the input ports. This is because even in the worst case that all n input lines are receiving packets, the switch will be able to transfer n packets from input port to output port in the time it takes each of the n input ports to (simultaneously) receive a *single* packet. But what can happen at the output ports? Let us suppose still that the switching fabric is at least n times as fast as the line speeds. In the worst case, the packets arriving at each of the n input ports will be destined to the *same* output port. In this case, in the time it takes to receive (or send) a single packet, n packets will arrive at this output port. Since the output port can only transmit a single packet in a unit of time (the packet transmission time), the n arriving packets will have to queue (wait) for transmission over the outgoing link. n more packets can then possibly arrive in the time it takes to transmit just one of the n packets that had previously been queued. And so on. Eventually, buffers can grow large enough to exhaust the memory space at the output port, in which case packets are dropped.

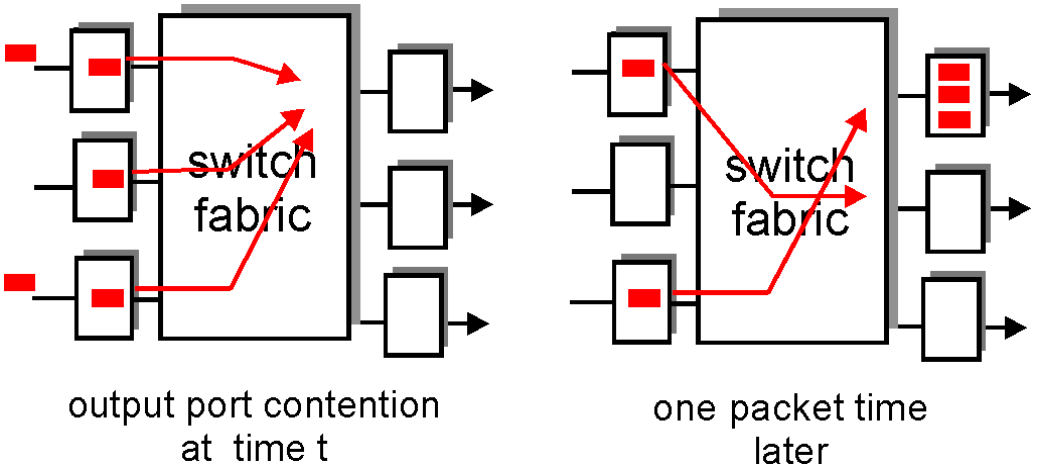


Figure 4.6-5: output port queueing

Output port queueing is illustrated in Figure 4.6-5. At time *t*, a packet has arrived at each of the incoming input ports, each destined for the uppermost outgoing port. Assuming identical line speeds and a switch operating at three times the line speed, one time unit later (i.e., in the time needed to receive or send a packet), all three original packets have been transferred to the outgoing port and are queued awaiting transmission. In the next time unit, one of these three packets will have been transmitted over the outgoing link. In our example, two *new* packets have arrived at the incoming side of the switch; one of these packets is destined for this uppermost output port.

A consequence of output port queueing is that a **packet scheduler** at the output port must choose one packet among those queued for transmission. This selection might be done on a simple basis such as first-come-first-served (FCFS) scheduling, or a more sophisticated scheduling discipline such as weighted fair queueing (WFQ), which shares the outgoing link "fairly" among the different end-to-end connections that have packets queued for transmission. Packet scheduling plays a crucial role in providing **quality of service guarantees.** We will cover this topic extensively in section 6.6. A discussion of output port packet scheduling disciplines used in today's routers is [Cisco 1997a] .

If the switch fabric is not fast enough (relative to the input line speeds) to transfer *all* arriving packets through the fabric without delay, then packet queueing will also occur at the input ports, as packets must join input port queues to wait their turn to be transferred through the switching fabric to the output port. To illustrate an important consequence of this queueing, consider a crossbar switching fabric and suppose that *(i)* all link speeds are identical *(ii)* that one packet can be transferred from any one input port to a given output port in the same amount of time it takes for packet to be received on an input link and *(iii)* packet are moved from a given input queue to their desired output queue in a FCFS manner. Multiple packets can be transferred in parallel, as long as their output ports are different. However, if two packets at the front of two input queues are destined to the same output queue, then one of the packets be blocked and must wait at the input queue - the switching fabric can only transfer one packet to a given output port at a time.

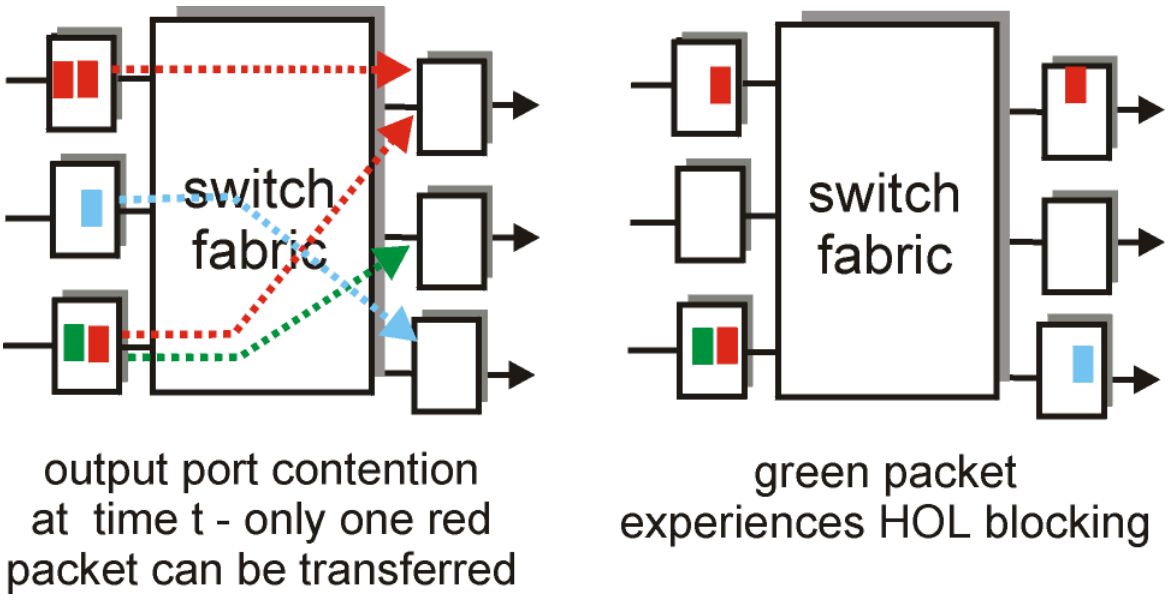


Figure 4.6-6: HOL blocking at an input queued switch

Figure 4.6-6 shows an example where two packets (red) at the front of their input queues are destined for the same upper right output port. Suppose that the switch fabric chooses to transfer the packet from the front of the upper left queue. In this case, the red packet in the lower left queue must wait. But not only must this red packet wait, but so too must the green packet that is queued behind that packet in the lower left queue, even though there is *no* contention for the middle right output port (the destination for the green packet). This phenomenon is known as **head-of-the-line (HOL) blocking** in an input-queued switch - a queued packet in an input queue must wait for transfer through the fabric (even though its output port is free) due to the blocking of another packet at the head-of-the-line. [Karol 1987] shows that due to HOL blocking, the input queue will grow to unbounded length (informally, this is equivalent to saying that significant packet loss will occur) as soon as packet arrival rate on the input links reaches only 58% of their capacity. A number of solutions to HOL blocking are discussed in [McKeown 1997b].

References

[Cisco 1997a] Cisco Systems, "Queue Management," http://www.cisco.com/warp/public/614/quemg_wp.htm, 1997.
[Cisco 1997b] Cisco Systems, Next Generation ClearChannel Architecture for Catalyst 1900/2820 Ethernet Switches, http://www.cisco.com/warp/public/729/c1928/nwgen_wp.htm, 1997.
[Cisco 1998 a] Cisco Systems "Catalyst 8500 Campus Switch Router Architecture," http://www.cisco.com/warp/public/729/c8500/csr/8510_wp.htm, 1998.
[Cisco 1998b] Cisco Systems, "Cisco 12000 Series Gigabit Switch Routers," http://www.cisco.com/warp/public/733/12000/12000_ov.htm, 1998.
[Degemark 1997] M. Degemark et al., "SMall Forwarding Tables for Fast Router Lookup," *Proc. 1997 ACM SIGCOMM Conference,* (Canes, France, Sept. 1997).
[Doeringer 1996] W. Doeringer, G. Karjoth, M. Nassehi, "Routing on Longest Matching Prefixes," *IEEE/ACM Transactions on Networking,* Vol. 4, No. 1 (Feb. 1996), pp. 86-97.
[Giacopelli 1990] J. Giacopelli, M. Littlewood, W.D. Sincoskie “Sunshine: A high performance self-routing broadband packet switch architecture”, 1990 International Switching Symposium.
[Gupta 1998] P. Gupta, S. Lin, N.McKeown. “Routing lookups in hardware at memory access speeds”, Proc. IEEE Infocom 1998, pp 1241-1248.
[Kapoor 1997] H. Kapoor, "CoreBuilder 5000 SwitchModule Architecture," http://www.3com.com/technology/tech_net/white_papers/500645.html, 1997.

[Karol 1987] M. Karol, M. Hluchyj, A. Morgan, "Input Versus Output Queueing on a Space-Division Packet Switch," *IEEE Transactions on Communications,* Vol. COM-35, No. 12, pp. 1347-1356, December 1987.
[Keshav 1998] S. Keshav, R. Sharma, "Issues and Trends in Router Design," IEEE Communications Magazine, Vol 36, No. 5 (May 1998), pp. 144-151.
[Microsoft 1998] Microsoft Corp., "Microsoft Routing and Remote Access Service for Windows NT Server 4.0], http://www.microsoft.com/ntserver/basics/communications/basics/remoteaccess/routing/default.asp
[McKeown 1997a] N. McKeown, M. Izzard, A. Mekkittikul, W. Ellersick, M. Horowitz, "The Tiny Tera: A Packet Switch Core", IEEE Micro Magazine, Jan-Feb 1997.
[McKeown 1997b] N. McKeown, "A Fast Switched Backplane for a Gigabit Switched Router," Business Communications Review, Vol. 27. N0 12.
[Partridge 1998] C. Partridge et al. "A Fifty Gigabit per second IP Router", IEEE/ACM Transactions on Networking, 1998.
[Tobagi 1990] F. Tobagi, "Fast Packet Switch Architectures for Broadband Integrated Networks," *Proc. IEEE,* Vol. 78, No. 1, pp. 133-167.
[Turner 1988] J. S. Turner "Design of a Broadcast packet switching network", IEEE Trans Comm, June 1988, pp. 734-743.
[Feldmeier 1988] D. Feldmeier, "Improving Gateway Performance with a Routing Table Cache," *Proc. 1988 IEEE Conference,* (New Orleans LA, March 1988).
[Thomson 1997] K. Thomson, G. Miller, R. Wilder, "Wide Area Traffic Patterns and Characteristics," *IEEE Network Magazine,* Dec. 1997.
[Waldvogel 1997] M. Waldvogel et al., "Scalable High Speed IP Routing Lookup," *Proc. 1997 ACM SIGCOMM Conference,* (Canes, France, Sept. 1997).