Multicast Communication Implementation in High Speed Packet Switching Systems

Paul Landsberg, Shu-Ping Chang, and Paul Chang

Advanced Networking Laboratory
IBM Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Multicast Communication Implementation in High Speed Packet Switching Systems

Paul Landsberg, Shu-Ping Chang, and Paul Chang

Advanced Networking Laboratory
IBM, Thomas J. Watson Research Center
Route 134
Yorktown Heights, NY 10598

ABSTRACT

Without a doubt, packet switching (especially ATM) has made great strides in being accepted as the transport mechanism for broadband networks and in any discussion of broadband networks, a multicast capability is viewed as a vital component of the underlying switching fabrics. Multicast has always been a desired feature in communication for a single originator to communicate with multiple recipients all at one time with the same information. This capability is essential for electronic conferencing applications where every attendant communicates with all other parties simultaneously with the same video, image, voice, text and etc.

In this paper, issues for implementing multicast communication in the high speed packet switching systems are looked at from two perspectives; one, selecting where to implement the multicast function and two, the means for handling multicast traffic.

Index Terms: ATM, Computer Communication, Packet Switching, Multicast Packet.

1. Introduction

In this paper, two key issues are examined that permit novel solutions for a multicast design for a variety of packet switching systems; the placement of the multicast function and the means to implement it. Fundamentally, multicast can be looked at in terms of where the logic is introduced to handle multicast traffic and what mechanism is used to generate multicast traffic. Of course, there exist relations between the "where" and "how" for implementing a multicast function on a packet switch system, and facets of this interaction will be examined.

Before delving into the details of multicast traffic, note that although ATM [EKL88] has made great strides in being accepted as the transport mechanism for broadband networks, its acceptance has been without the benefit of real-time network experience on broadband networks. In general, using small fixed length packets (cells) makes switch design far simpler in terms of hardware complexity but when video traffic is considered, using relatively small fixed length packet sizes (e.g. 53 bytes in the case of ATM) introduces inordinate segmentation and reassembly costs at the edge of the envisioned integrated broadband networks. Hence, very strong arguments can be made for using variable length packets. Furthermore, traffic studies on existing Ethernet networks [KHA90] have explicitly shown that the distribution of packet length is not Poisson but rather, bimodal, (in exact terms, even multi-modal) where one mode is below 200 bytes, and the other mode is close to 1500 bytes. This would seem to indicate a need for at a minimum of two packet sizes being allowed in a switch architecture. Regardless of arguments about the usefulness of variable length packets, this paper assumes that the high-speed packet switched communication network under study can process variable length packet. Note that the term "packet" will include fixed size ATM cells and variable length packets.

In this paper, Section 2 describes a switching model and discusses the multicast function placement issue. The mechanism for generating multicast packets is deal with in Section 3. Also, in Section 3, one particular multicast communication implementation is described. Section 4 details implementation issues for packet retransmission for the shared memory type of switching system and, finally, Section 5 summarizes this paper.

2. The Multicast Function Placement in a Switching System

In the most basic form, a switch can be represented as shown in Figure 1 where there are input ports, output ports, and some sort of switching structure. In this general form, the input ports can range from simple FIFOs to adapters that can perform segmentation and reassembly, bridging, routing, or protocol conversion. From a reliability standpoint, it might even be desirable to partition a switch in the above manner, where the ports are removable hardware adapters. Hence, if a port adapter fails, the entire system does not have to be replaced. Furthermore, if the interface between the switching structure and the input/output ports has been well defined and architected, it allows an enormous flexibility on the part of switching node upgrades and future adapter development because the functions have been separated.

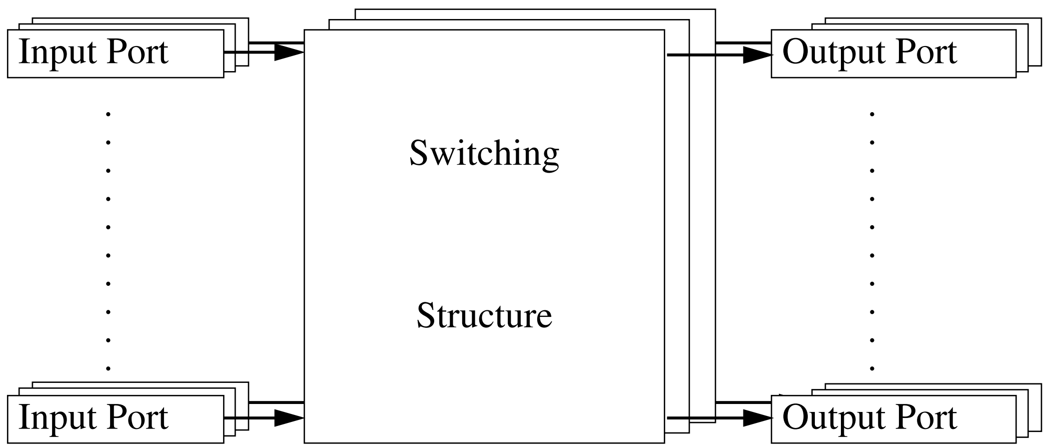

Figure 1 - General Switch Structure

As seen in [TOB90], packet switching systems can be classified into three major types; shared medium, shared memory and space division. Among them, the shared-memory type of switching system requires an important resource management function in the switching structure for its shared memory, a memory manager. The "switching" is done through memory access by the switch output ports. One standard memory management technique is to define a fixed size block of memory, called buffer, as the smallest granularity handled by the memory manager. Packets that are larger than a system buffer of the shared memory (e.g. 64 bytes) would use multiple buffers to form a complete packet. Many of the recently introduced shared-memory switches [END93],[WEI92],[KOZ91] handle only fixed size ATM cells. In this case, the size of the system buffer is normally defined to be larger than or equal to the fixed size cell. However, when variable sized packets must be routed, data structures for linking multiple buffers to form the packet and for linking multiple packets to form a queue must be employed by the system. Effectively, a buffer manager capable of creating and manipulating a two-dimensional linked list would be required. To serve multiple switching ports in the system, such a buffer manager normally be implemented by using a pipelined design technique in order to maintain very high system throughput.

2.1 Routing and Multicast

Since routing information provides recipient(s) with destination(s), it is intuitively true that the multicast implementation must be located at or near the routing function. Given the general system structure shown in Figure 1, routing can take place at a variety of locations for a shared memory switching system. In contrast, the routing function for shared medium and space division switching systems is always implemented at the input port. If the switching system under consideration has output ports which only feed a single link, the routing function can be placed within the central switching structure or at the input ports. Routing done within the switching structure would have to be in effect time-division-multiplexed such that all input traffics from input ports address the shared centralized routing table. On the other hand, if routing is done on each input port, then multiple routing tables are maintained by the system management function. For output port adapter which feeds several slower speed links or must generate multiple VCIs/VPIs for a given packet, some form of routing table/hardware is also required on the output port adapter. Each slower link attached to an output adapter is call a sub-port and can have an associated sub-port routing function. The placement of the routing tables (the decision points) is a design consideration that has to satisfy system requirements and to address the over all system implementation cost.

The implication of having to implement the multicast function whereever there exists routing logic (tables or otherwise), is that a buffering mechanism need to be implemented for the routing and multicast. Strictly from a hardware perspective, if routing were implemented at all three places, at the input port adapter, at the switching structure, and at the output port adapter, by necessity, each location would have to be able to store an entire packet. From our previous discussion, if routing has to be done upon the adapters that are ideally as low cost subsystems, would place significant buffering and control function requirements on these adapters.

2.2 Multicast Packets in Shared Memory Switching Systems

From previous discussion on routing considerations, distributing the routing functions to the input and output ports increases the complexity and cost of each port adapter. Furthermore, sufficient memory must exist on each port adapter to store and forward the packet. In the case of ATM traffic, the memory to store an entire cell is relatively small, only 53 bytes of actual packet data. But in the general case of packet switching, the memory to store an entire packet is determined by the maximum packet size. It is not unrealistic to consider packet sizes of 4K bytes (e.g. FDDI maximum). Furthermore, to maximize the throughput, the switching structure tends to be implemented by pipelined technology. It would be inefficient for the port adapters, input or output, to only store one packet before allowing another packet to enter. Hence, the actual number of packets stored is multiplied by the number (usually small) of packets that need to be stored. However, even if only four packets are stored, that would required 16 Kbytes of storage for the preceding FDDI traffic example.

A shared memory switching system by definition has an extremely large memory space devoted for packet storage within the switching structure in Figure 1. Therefore, it makes engineering sense to re-use that memory capability for reducing the complexity and cost of port adapters. This can be accomplished by using a RETRANSMIT mechanism that works between the output adapter and the shared memory switching structure. A retransmit mechanism is shown in Figure 2, where

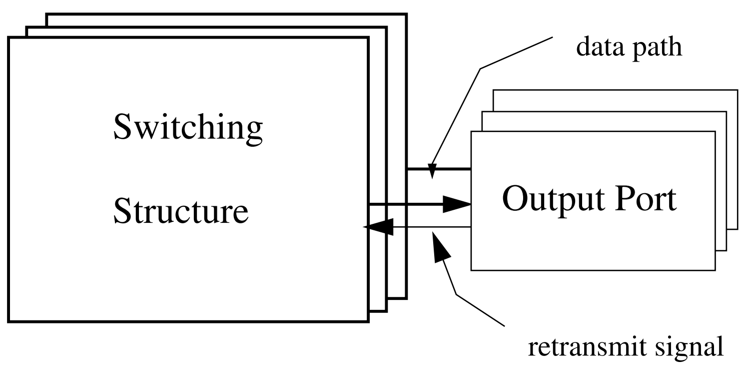

Figure 2 - Switching structure to single output port connection.

a retransmit signal notifies the switching structure that a given packet should be retransmitted. Note that from a product engineering perspective, since the data path between the switching structure and the output port adapter is over signal lines that can be affected by noise, environmental effects, or electrical signal problems, a point-to-point retransmit signal would be a desired, if not required, feature. In addition, if status control signals already exist that indicate the output port can accept packets, the retransmit signal is trivial encode as part of these signals. Most importantly, if the output port adapter is a subsystem that feeds several slower links, the sub-port multicast function can also be implemented by using the switch structure as the central storage room for multicast packet generation.

3. Multicast Methods

There are fundamentally two way to implement a multicast solution for packet switches: 1) multiple packet generation and 2) multiple READs from a single stored packet, termed single-write, multiple read. Although the two methods sound similar, consider case 1) where M multiple packets are generated at the input port and thereby, M packets are then stored in the shared memory. If a packet is stored at an input port, and it is destined for M output ports, a back-pressure mechanism must exist to halt any further inbound traffic, or some significant buffering (maybe even a combination of the two) must exist on the input port adapter. Whether back-pressure, buffering, or some combination is used, the number of packets waiting in input queues, or at the node immediately prior to the inbound link, grows. A form of head-of-the-line blocking has been introduced. On the other hand, if a single multicast packet is stored in the shared memory, along with a multicast count of all desired destinations, not only has memory storage requirement for packet data been reduced from M to 1, but in terms of bandwidth, in the first case, it took M packet times to transmit the M multicast packets and in the second case, it effectively only took one packet time to transmit M multicast packets, assuming the multicast count could be updated within one packet time. Considering that packets are generally much longer in duration than VLSI system clock speeds, the assumption that a multicast count is updated within one packet time is reasonable.

In terms of current multicast solutions, some of the proposed packet switches use the extremely inefficient method where a given cell or packet is copied as many times as need to produce the proper multicast sequence [KIM93][BIA93][TUR93], or, some form of copy network [KWO90] is used. However, this is normal in the space division type of switching systems. On the other hand, shared medium type of switching system has the advantage of the broadcast nature of the shared medium. Therefore, multicast by input port does not introduce bandwidth issue and shared medium switch does not have memory inside the switching strucutre to begin with. But for shared memory type of switching system, from previous discussion, it is more desirable from bandwidth and memory storage perspective to not use a copy network, or to generate multiple copies of the same packet.

3.1 Single-Write, Multiple-Read Multicast Implementation

The Single-Write, Multiple-Read multicast mechanism has been identified as a superior way of supporting multicast packets for the shared memory type of switching system. In this section, one particular implementation of single-write, multiple-read multicast is presented.

Generally speaking, the control structure for a shared memory system can contain a Multicast

Packet Count memory, MPC-RAM. This memory must contain the number of multicast copies for every possible packet in the system. Therefore, the total number of entries in the MPC-RAM is the same as the total number of packets/cells the system can store. In terms of memory accesses, when a packet is written into the shared memory, the multicast information must be written into the MPC-RAM. Later, when the packet is to be transferred to the output port, the stored information, the multicast packet copy count will be decremented by one until the count reaches zero. At that time, the hardware knows that all copies of the packet have been transmitted. Therefore, the system can instruct the buffer manager to release all the memory buffers associated with this packet (multicast or not) back to a free buffer pool. Note that, for a pipeline designed switching structure, in any given clock cycle, there can be up to three memory operations occurred on the MPC-RAM and they are: 1) write the original multicast packet count into the MPC-RAM, 2) read from the MPC-RAM to retrieve the current multicast packet count and 3) write to the MPC-RAM to store the updated multicast packet count. Therefore, the MPC-RAM memory bandwidth must be able three concurrent accesses.

An alternative design can be used to reduce the MPC-RAM bandwidth to 2 operations per system clock cycle. This alternative design is to split the MPC-RAM into two memory modules called the Original Multicast Packet Count Memory (OMPC-RAM) and Current Multicast Packet Count Memory (CMPC-RAM) respectively. Under this design, all the entries in the CMPC-RAM will be set to zero during system initialization. After system initialization, when a packet is stored into the shared memory, the original multicast packet count will be written into the OMPC-RAM. Later, when the packet is to be transferred from the shared memory to the output port, a simultaneous read of the OMPC-RAM and CMPC-RAM will be performed. If the MPC read from the CMPC-RAM does not equal to the MPC read from the OMPC-RAM minus 1, the CMPC will be increment by one and the updated CMPC value is written back to the CMPC-RAM. Otherwise, it indicates the end of multicast packet copy and, in this situation, a zero will be written into the CMPC-RAM. The buffer manager then releases all memory buffers associated with the current packet back to the free buffer pool.

As noted earlier, the memory management associated with the shared memory must be capable of handling two dimensional linked lists. The extra control logic used to handle multicast packets can be evaluated as to how much additional complexity is introduced to such a buffer manager.

4. Implementation Issues for Retransmission

Using a retransmit signal in a shared memory type of switching system is an attractive way of optimizing resource usage. The memory intense switching structure performs the brunt of packet storage, leaving the adapters to perform processing intensive (if needed) tasks. However, there is one implementation issue that should be noted. Shared memory managers that yield high throughput normally require some form of pipelined architecture. The shared memory manager that presents the buffer location of the current packet being transmitted and waits for the adapter to signal "no re-transmission is needed" (in order to free up the buffer spaces), may not maintain the highest possible throughput. Ideally, as a packet is being transmitted from the shared memory (the switching structure) to the output port adapter, the occupied packet buffer should be concurrently released back to a free buffer pool. Hence, any sort of retransmit signal must a) reach, b) be recognized, and c) be processed, by the shared memory manager before the buffer is released.

In a tightly coupled system, where the port adapters are synchronous with the shared memory, a retransmit signal must return within a required time. On the other hand, if the port adapters are physically separated from the shared memory, some latency is introduced into the timing of any retransmit signal. If the latency is fixed and small, the pipelined design of the memory manager can be easily modified (e.g. the pipeline extended). However, if the delay can vary (e.g. the port adapters are housed in a separate rack that can be located anywhere from 1 foot to 50 feet away), this presents a more difficult engineering problem. This problem can be solved by using two-level retransmission mechanism based on the length of the packet, provided that the maximum delay of the retransmission signal has a reasonable bound. If the memory manager waits for the retransmit signal, the latency of the retransmit signal might prevent the transmission of back-to-back short packets (assume size L). Therefore, for packet shorter than or equal to size L, the output port is responsible for the sub-port multicast. Longer packets can still use the previously described retransmit signal and the switching structure as the storage area for sub-port multicast function. In this way, the output port only needs to buffer small packet for multicast function instead of the packet with maximum size.

5. Summary

This paper discussed issues for various issues in multicast communications in different broadband switching architectures. Shared medium and space division switching systems are forced towards a certain type of multicast solution. The shared memory switch architecture, such as the shared memory switches recently proposed, allows the multicast function can be placed inside the switching structure such that the packet remains stored in the shared memory, and multicast is supported by using single-write, multiple-read mechanism to save inbound bandwidth and memory storage. An implementation of the single-write, multiple-read multicast mechanism was presented, and can be easily developed with commercial dual-port memories.

Using the shared memory inside the switch structure to support sub-port multicast communication, one implementation issue was considered where the retransmit signal might arrive asynchronously at the shared memory manager. In the case of an asynchronous retransmit signal, a variation on the proposed scheme was presented. For short packets in systems where the adapters are physically separated from the switching structure, the sub-port multicast function is handled strictly by the output port adapter. This is a supplementary mechanism for the switching structure that stores longer packets.

6. References

[KIM93] H.S. Kim, "Multinet Switch: Multistage ATM Switch Architecture with Partially Shared Buffers," *IEEE INFOCOM*, 1993, pp. 473-487.

[BIA93] R.P. Bianchini Jr. and H.S. Kim, "Design of a Non-Blocking Shared-Memory Copy Network for ATM," *Int. J. of Dig. and Analog Comm. Sys.*, Jan-Mar 1993, pp. 39-48.

[EKL88] B. Eklundh, K. Sallberg, and B. Stavenow, "Asynchronous Transfer Modes -- Options and Characteristics," *Proc. 12th Int. Teletraffic Cong.*, 1988.

[END93] N. Endo, T. Kozaki, et al, "Shared Buffer Memory Switch for an ATM Exchange," *IEEE Trans. on Comm.*, Vol. 41, No. 1, Jan. 1993, pp. 237-245.

[ENG92] K.Y. Eng, M.J. Karol, and Y-S. Yeh, "A Growable Packet (ATM) Switch Architecture: Design Principles and Applications," *IEEE Trans. on Comm.*, Vol. 40, No. 2, Feb. 1992, pp. 423-430.

[KHA90] K.M. Khalil, K.Q. Luc, and D.V. Wilson, "LAN Traffic Analysis and Workload Characerization," 15th Conference on Local Computer Networks, 1990, pp. 112-122.

[KOZ91] T. Kozaki, et al, "32 Multiplied by 32 Shared Buffer Type ATM Switch VLSIs for B-ISDN," *IEEE Int. Conf. on Comm.*, 1991, pp. 711-715.

[KWO90] T. Kwok and F. Tobagi, "The Tandem Banyan Switching Fabric: A Simple High-Performance Fast Packet Switch," *IEEE INFOCOM '91*, pp. 1245-1253.

[TOB90] F.A. Tobagi, "Fast Packet Switch Architectures for Broadband Integrated Services Digital Netwroks," *Proceedings of the IEEE*, vol. 78, no. 1, Jan. 1990, pp. 133-167.

[TUR93] J.S. Turner, "A Practical Version of Lee's Multicast Switch Architecture," *IEEE Trans. on Comm.*, Vol. 41, No. 8, Aug. 1993, pp. 1166-1169.

[WEI92] Sherry X. Wei, and V.P. Kumar, "On the Multiple Shared Memory Module Approach to ATM Switching," *IEEE INFOCOM '92*, pp. 116-123.

[ZHO93] W.D. Zhong, Y. Onozato, and J. Kaniyil, "A Copy Network with Shared Buffers for Large Multicast ATM Switches," *IEEE Int. Conf. on Comm.*, 1993, pp. 722-725.