A Multicasting Solution for ATM Video Applications

Jer-Min Tsai, Hsin-Hsiung Fang, and Chen-Yi Lee

***Abstract*— This paper presents a multicasting solution for shared-buffer ATM switch. The cell duplicating function is performed by a one-to-many modified demux circuit. A shared multicast server is used to translate input multicast cells to find destination ports and corresponding routing information. By using link-list based ring structure, the multicast server can provide high speed (5 ns to search entry in 256-entry block in 0.8-μm CMOS process) and cascadable multicast translating function. In addition, the channel complexity of multipoint-to-multipoint multicast applications can be reduced to N channels per N users. The shared-buffer controller is modified by adding controlling and scheduling functions for multicasting queues to process the multicast feature. Finally with the aid of the quality of service (QoS) management, this proposed solution provides multi-QoS for multicast cells to meet the needs of various video applications.**

***Index Terms*— ATM switch, communication system, multicast cells, quality of service, VLSI architecture.**

I. Introduction

BROAD-BAND integrated services digital networks (B-ISDN) are expected to provide efficient and reliable transport for various services. The asynchronous transfer mode (ATM) technique [1] has been standardized and widely accepted as a basis for transporting and switching information in B-ISDN. However, the applications that move the needs toward B-ISDN are not the traditional point-to-point telecommunication or data-communication applications, but the applications which need a lot of bandwidth and have the inherent variable-bit-rate characteristic such as video conference and video-on-demand. In order to provide a high-efficiency ATM switch system for those applications, some characteristics of those applications must be taken into account. For example, the maximum packet delay of video cells must be guaranteed, and hence the powerful embedded multicasting feature is needed to reduce the loading on video source.

There are three major video application types, namely point-to-point (P2P), point to multipoint (P2M), and multipoint-to-multipoint (M2M). Video applications such as video phone belong to the P2P type. Applications like video-on-demand belong to the P2M type. And applications like video conference belong to the M2M type. The challenges of P2M and M2M are not only how to provide high-quality video signals through lower bandwidth communication channel, but also how to reduce the loading of video server and network. In addition, the network traffic and the latency for translating the multicasting cells will certainly reduce the quality of video signals of P2M and M2M applications.

In P2M applications, users usually need a small set of video sources in a certain period, so a good request scheduling algorithm with an efficient multicasting function provided by an ATM system can reduce the number of video streams that the video server must provide. In original ATM environment, an N user's video conference needs $N \times N$ communication channels to broadcast video signals to every user. The network traffic and video source loading can be reduced with the aid of efficient multicasting function. The multicasting circuit must translate M2M video cells fast enough to prevent the serious decay of video quality due to video cell delay.

Several multicasting solutions can be found in the literature [2]–[9]. The major architectures can be classified into the following types, namely copy network [6], [7], [9], recycle output cell [4], [5], cell copy [8], and address copy [2], [3]. The major topics which are discussed in those papers are cell duplication, cell routing, and performance analysis. Several excellent solutions on these topics can also be found in the above-mentioned papers. But the detailed descriptions on the mechanisms of multicast cell translation and M2M multicast function cannot easily be found. It is expected that the M2M applications like video conferencing will play an important role in the coming B-ISDN age. Then the N^2 channel complexity of M2M multicast applications will result in heavy loading on the switch and client system. In order to solve this problem, we propose a novel multicast solution based on shared-buffer ATM switch to reduce the channel complexity of M2M multicast applications to N connections for N users. In addition, the proposed solution can provide an efficient, flexible, and cascadable M2M multicasting solution for video applications.

This paper is outlined as follows: in Section II, an overview of the shared-buffer ATM switch system will be described. The complete system design complexity will also be presented for the target 32×32 I/O ATM switch. Section III will present our algorithm for the support of the multicasting function. In addition, a cost-effective very large scale integration (VLSI) architecture is proposed to realize the algorithm. Discussion and comparison will be given in Section IV.

II. System Concept and Overview of Shared-Buffer ATM Switch System

Among the ATM systems found in the literature, the shared-buffer ATM system [10] is attractive for its high memory utilization. From the viewpoint of "cost-effectiveness" of an ATM system, it is for that reason that the shared-buffer

Manuscript received September 30, 1996; revised January 31, 1997. This paper was recommended by Guest Editors B. Sheu, C.-Y. Wu, H.-D. Lin, and M. Ghanbari.

The authors are with the Department of Electronics Engineering, National Chiao Tung University, Hsinchu 300, Taiwan, R.O.C.

Publisher Item Identifier S 1051-8215(97)05887-4.

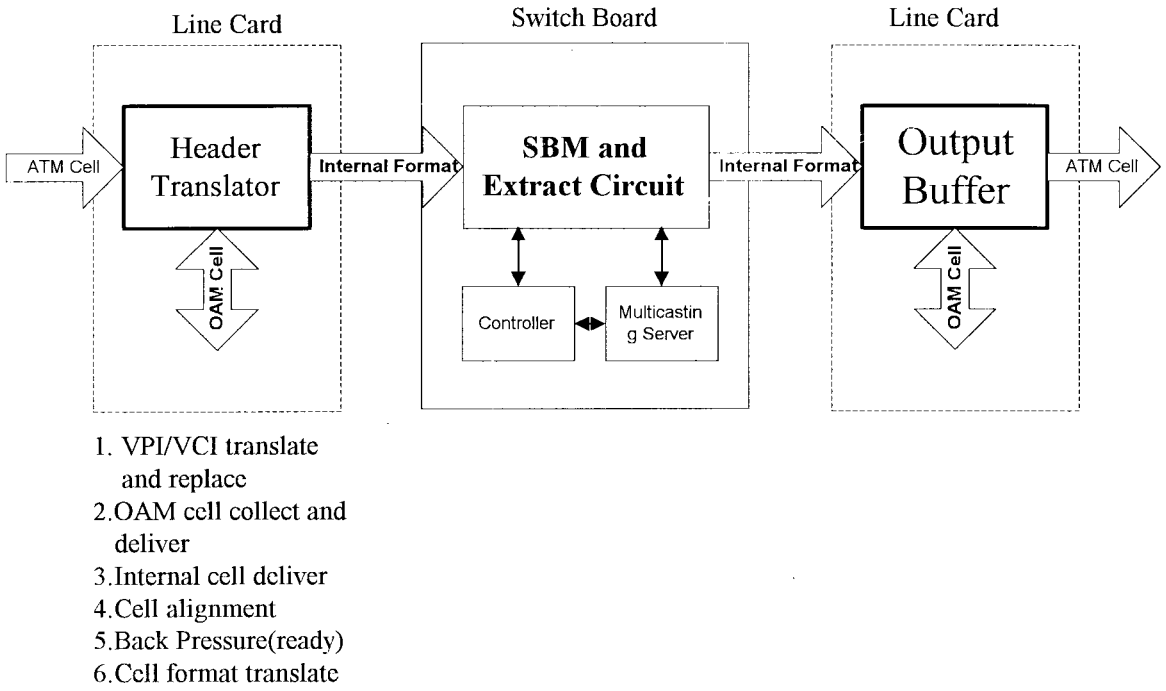

Fig. 1. Data flow diagram of the proposed multicasting shared-buffer ATM switch system.

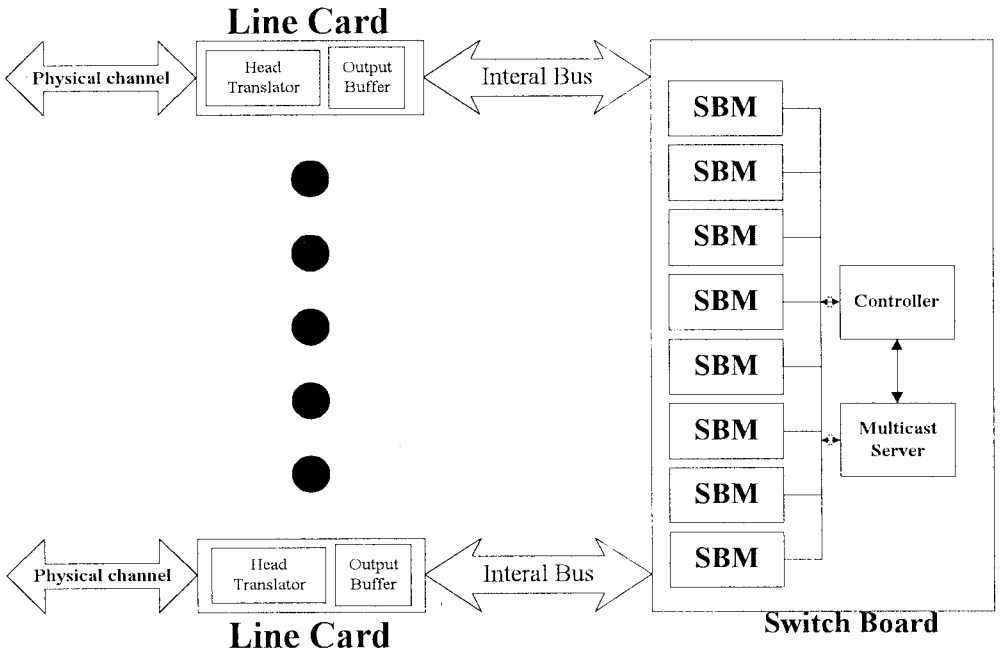

Fig. 2. System architecture of the proposed shared-buffer ATM switch.

architecture is selected as the target architecture to realize our multicasting solution.

Our shared-buffer ATM switch system for video applications mainly consists of three major parts, namely header translator (HT), shared buffer memory (SBM), and multicasting server (MCS) as shown in Fig. 1. HT is used to decode the virtual path identifier/virtual channel identifier (VPI/VCI) of each input ATM cell and find corresponding new VPI/VCI or multicast group ID and other routing information for each input ATM cell. From the routing information found by HT, SBM and its control circuit route input cells to an individual output queue, where these cells are stored in the shared buffer which is waiting to be read out and sent to the destination output port. The multicast cells are translated by a multicasting server. According to the group ID of each multicast cell, the parties of the input cell can be found out by MCS to deliver this multicast cell. Data flow of the shared-buffer ATM switch system can also be found in Fig. 1.

This architecture can achieve a 32×32 shared-buffer ATM switch system, where the speed of each channel is 155.52 Mb/s. In order to reduce the cost and enhance the stability, we assign four input pins and four output pins per channel

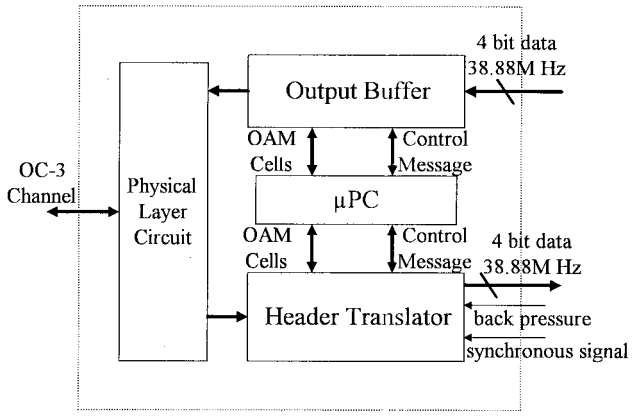

Fig. 3. Block diagram of the line card.

to translate ATM cells between line card and switch board. In order to provide flexibility of VPI/VCI assignment and to reduce the complexity of control programming, a system architecture is proposed in Fig. 2. In this architecture, the HT is placed on the line card and controlled by the microprocessor. The bit-slice shared-buffer and shared-buffer controller are placed on the switch board. The MCS which is shared by all ports on the switch board is used to translate multicast cells. In this way, both computing power and control complexity of the microprocessor on the switch board can be reduced.

A. Line Card and Header Translator

A block diagram of the line card is shown in Fig. 3. The physical layer circuit is used to translate the signals between the communication channel and hardware modules of the switch. The selected protocol between the physical layer circuit and digital circuits is named UTOPIA that was defined by ITU. The microprocessor is used to set up the routing table inside HT and to control the behavior of the whole line card. The output buffer is used to translate the internal cell format to standard ATM cell format.

Fig. 4 illustrates the functionality and data cell format of HT which is the kernel of the line card. The input is a standard ATM cell formatted with five-octet header and 48-octet payload data. If the VCI/VPI part of an input cell header is A, it will be replaced by B after translation, and one additional byte, r, will be attached to the cell. Hence the output of HT will have a six-octet header and a 48-octet payload data. This additional byte and new VCI/VPI are named as routing information data.

To make the HT more flexible and universal, 5-byte routing information is presented here. Fig. 5 shows the format of the routing information table. To expand the word length of the routing information table, the lower 11 bits named as extAddr can be used as the address of an external memory. Since the word length of external memory is flexible, the routing information can be expanded to more than 5 bytes if needed.

B. Switch Board and Shared-Buffer Management

In order to construct the 32×32 shared-buffer ATM switch system, it needs 4 k or more buffer size to meet the constraints of allowable cell loss rate. Since more than 212 kbyte storage space and control circuits must be implemented, we use bit-slice technique [10] to solve this problem. We intend to use eight chips (or nine chips for parity check) to implement shared buffer memory, and one ASIC control chip to implement control circuits.

A block diagram of the shared buffer is shown in Fig. 6. The buffer sharing operation in this switch constructs the address chain output queue using the next address which indicates the address of the succeeding cell to be delivered to the same output port. A pair of write and read address registers is assigned to each output port. For example, a pair of write address register 0 (WAR0) and read address register 0 (RAR0) corresponds to the #0 output port and a pair of WAR31 and RAR31 corresponds to the #31 output port. Here, WAR indicates the end of the chain and RAR indicates the beginning of the chain. This mechanism permits each address of the shared buffer memory to be temporarily allotted to any output port as the occasion demands, not permanently to one particular output port [10].

In Fig. 6, cells from all input ports are multiplexed in the multiplexer (MUX) and written one-by-one into the shared buffer memory pool. The write address that is used is obtained from the WAR corresponding to the destination output port which is indicated by an added routing information item of each cell. A small cache memory (2×12 bits in our design) is used to enhance the process of idle address. With the aid of this small cache, the large first-in, first-out (FIFO) used to store idle address can be eliminated, and the required memory bandwidth is the same as that in the architecture with idle FIFO [10].

When a cell outputs from the switch, the output cell is read out from the SBM with the address obtained from the RAR corresponding to the number of cell output ports. Then, it is sent out through the demultiplexer (DMX).

In addition, an efficient quality of service (QoS) processor [19] is integrated into the SBM controller. It is used to provide a powerful, high speed, and cost-effective QoS processing function. The architecture of this QoS processor is shown in Fig. 7. By using the high-speed hardware sorter [18], this processor can solve an output request in a single cycle. The subtraction cost function provides the starvation-free QoS policy. The service quality and bandwidth assignment can be adjusted by setting the decrementor of the subtractor.

III. Multicasting Hardware Architecture

A. Brief Description of the Multicasting Algorithm

Two procedures are needed to distribute multicast ATM cells. First, multicast cells must be translated to find their corresponding destination port and routing information. The second procedure is to copy multicast cells to corresponding output ports.

The translation circuits of multicast can be separated into three styles: namely, self-routing [4], distributed [2], and centralized [3] styles, respectively. In order to keep the high hardware utilization feature of the shared-buffer ATM switch

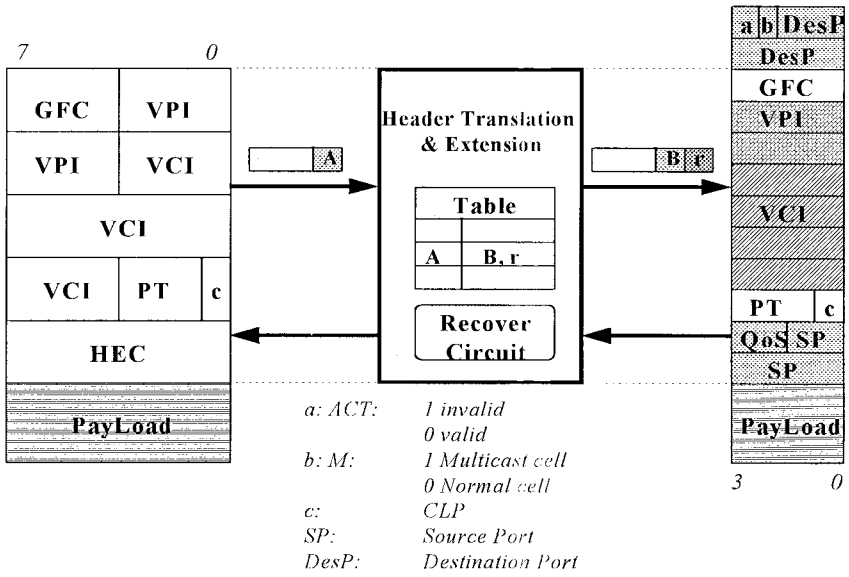

Fig. 4. I/O data cell format and functionality of the header translator.

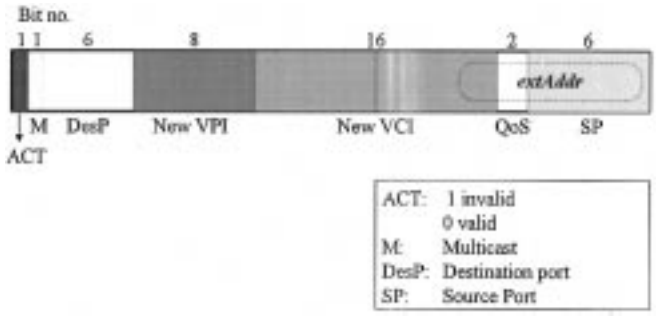

Fig. 5. Data format of the routing information table.

and to reduce the complexity of table maintenance, we use the centralized translation method to construct our multicasting solution.

In order to provide a flexible and efficient M2M cell translation function, we use **ring** structure to represent each multicast group. By linking all members in the same multicast group as a ring, no matter starting from any member, all members can be found by tracing the links one by one. The **ring** structure used to represent multicast groups is shown in Fig. 8. For the given example, a multicast cell is input from port A and then B can be traced, followed by C and D through the links. When the trace is back to A through the pointer from D, the member lookup procedure is finished. In this way, no matter from which source port the input cells come, the rest of the members of the same group can be found. Because of this ring structure feature, the N users M2M applications only need N connections instead of N^2 connections. This feature greatly reduces the cost and traffic load on both switches and end users. Thus, it can be seen that the ring structure is very suitable to implement the M2M multicast translation function.

In real implementation, we can use the link-list data structure to construct the ring structure. In the meantime, we can map the ring structure to a table with an index field and the traced function of the multicast ring to a table search function as well. In order to improve the hardware efficiency, we rearrange the cell copy process after the cell buffering process. It then avoids the duplication of multicast cells in the share-buffer. The flow chart of this multicasting algorithm is given in Fig. 9.

B. System Architecture

The support of multicast requires some type of copy network. Variable types of architectures have been proposed to implement the required copy network to support multicast capability [3], [4], [6], [8]. Based on the description in Section III-A, we propose a new architecture coupled with SBM switch core to support multicast service in an ATM network. This architecture is simple and efficient in implementation, operation, and maintenance. We use an external MCS to translate multicast cells and integrate the copy network to the DEMUX circuit of SBM. The multicast controller is integrated into the SBM controller to control both translation and routing of multicast cells.

The relation among SBM, SBM controller, and MCS is shown in Fig. 10. When the SBM controller receives the multicast cell, the multicast cell will be enqueued to a certain multicast queue. If MCS is free, the controller will send the headers of cells in multicast queues to MCS. The MCS extracts the *Group ID's* and *Source Port* data of input headers to translate multicast cells and find corresponding routing information and destination ports. In addition, the proposed MCS architecture can provide a group-based translation. In other words, no matter from which source port the input cell comes, the MCS will find out the rest of the members of the same group to form the bitmap of destination ports. This feature makes the proposed architecture very suitable to solve the M2M multicast request.

After the translation of MCS, the corresponding messages such as VPI/VCI's and output port bitmap, which decides how to route multicast cells, are then stored in MCS. The handshake signals between MCS and multicast controller will set those

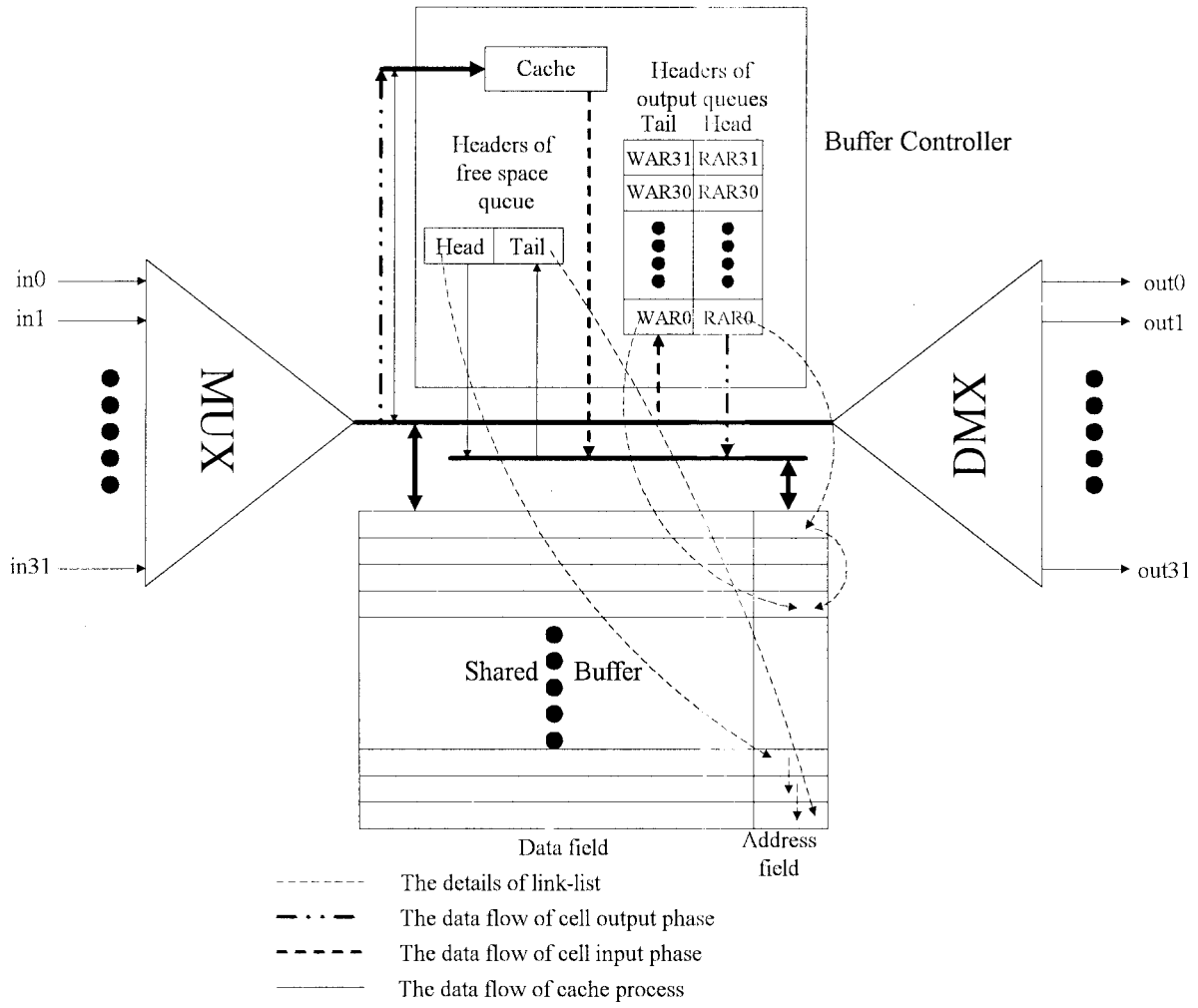

Fig. 6. Block diagram of the shared-buffer memory.

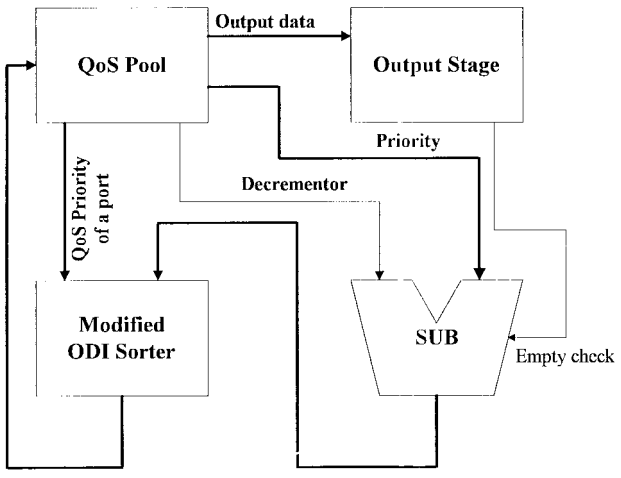

Fig. 7. Architecture of the QoS processor for multiple QoS management.

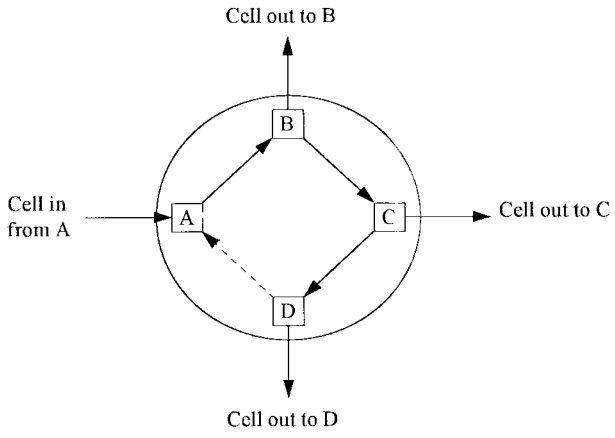

Fig. 8. Illustration of the proposed ring structure for the multicast table (using four outputs as an example).

translated multicast cells to output ready mode and then enable them to be sent out. These output ready multicast cells will be output with the aid of the multicast DMX (one-to-many demux circuit) and the header buffer in MCS. For normal cells, the multicast cell body is read out from SBM through the content of the corresponding queue in the SBM controller. The bitmap of destination ports are provided by MCS to control the routing of multicast DMX. For each output port of multicast DMX, there is an extra header replace circuit which is used to replace new VPI/VCI through the aid of the MCS. Because we use skew policy to reduce the arbiter cost and to enhance the hardware efficiency [10], the width of the header bus can be

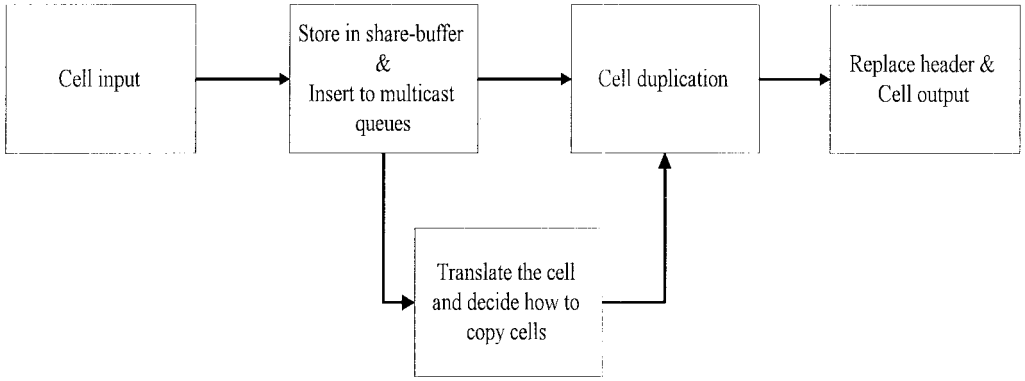

Fig. 9. Flow chart of the proposed multicasting algorithm, where only header info is to be handled.

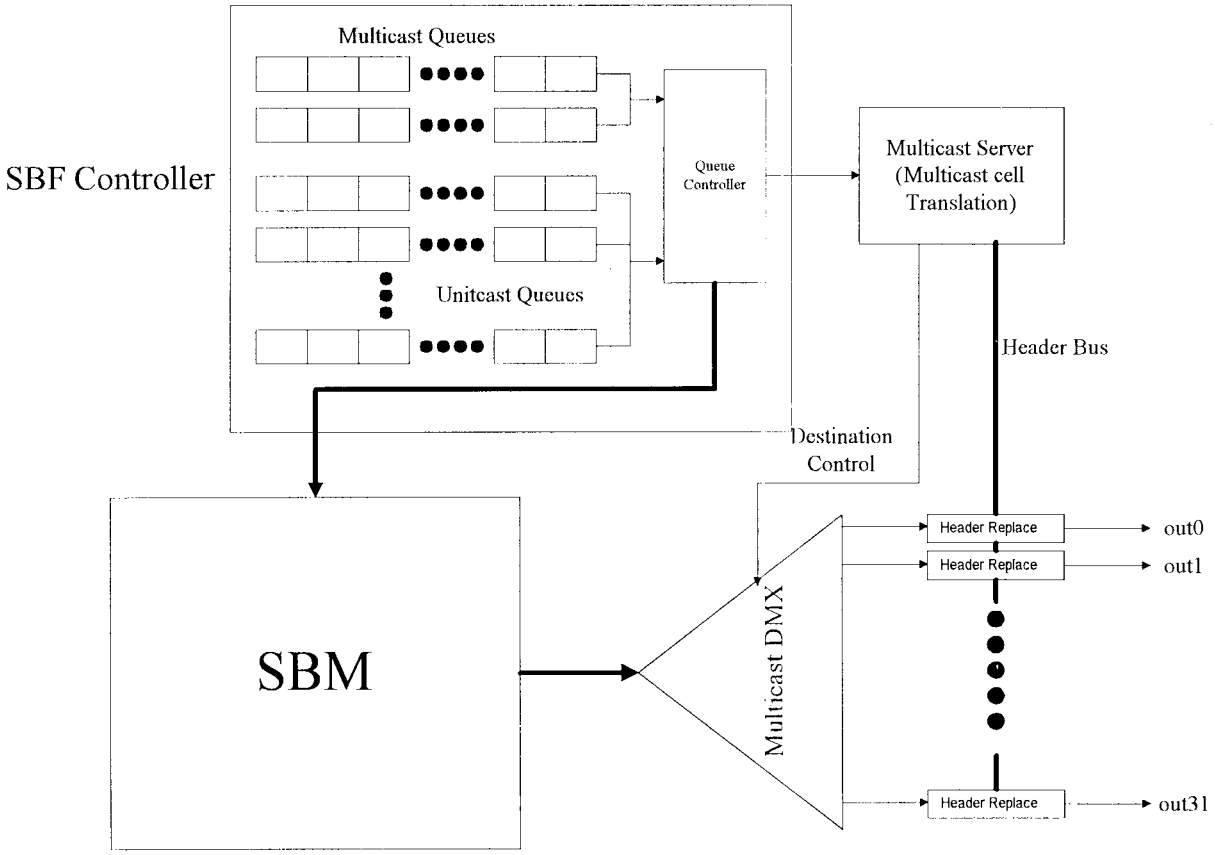

Fig. 10. System architecture of the proposed multicast shared-buffer ATM switch, where multiple queues are added at the DMX.

reduced to about 24 or 28 bits (the same as the word length of ATM VPI/VCI).

By using this approach, the storage space of a multicast cell is the same as a unicast cell, and the cell loss rate of whole system can be reduced. The structure of SBM and MUX are the same as the original shared buffer ATM switch. A transmission gate or pass-transistor based DMX can easily be modified to one-to-many DMX. Thus, with this powerful MCS solution, the whole system will be well-suited to provide a multicast feature for the shared-buffer ATM switch.

C. Multicast Queues and Cell Copying Circuits

Because of the support of the powerful QoS processor, our architecture can provide several multicast queues for different multicast applications to meet the requirements of various multicast applications. The operation of multicast queue can be divided into two phases; one is cell translation, and the other is cell output. When the translated buffer of MCS is not full and the MCS is free, if any multicast cell in multicast queues has not been translated, then the multicast controller will issue a nontranslated cell to be translated by MCS. After the translation of MCS, the multicast cells will be marked as translated cells and be ready for output phase.

There are two methods to separate the translated and non-translated cells, one is to add a flag in each queue entry, and the other is use two queues to store translated and nontranslated multicast cells. No matter which method is chosen, we must make sure that all multicast cells are translated before being

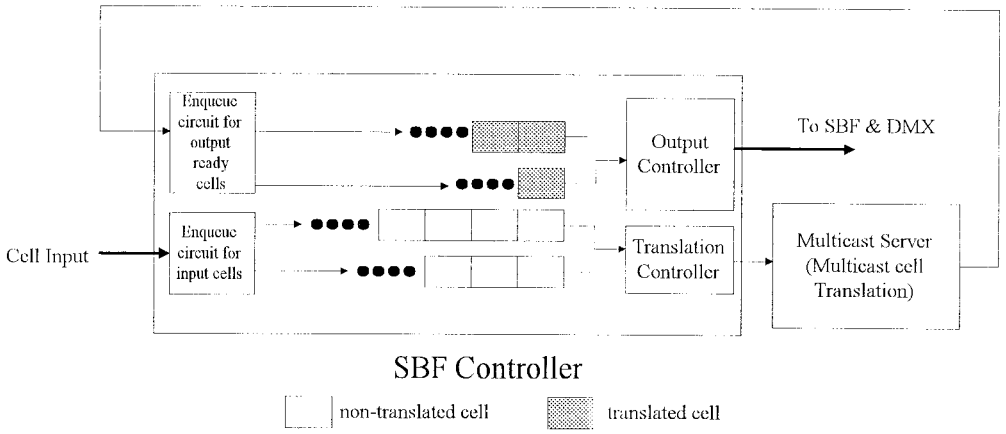

Fig. 11. Block diagram of the multicast queues.

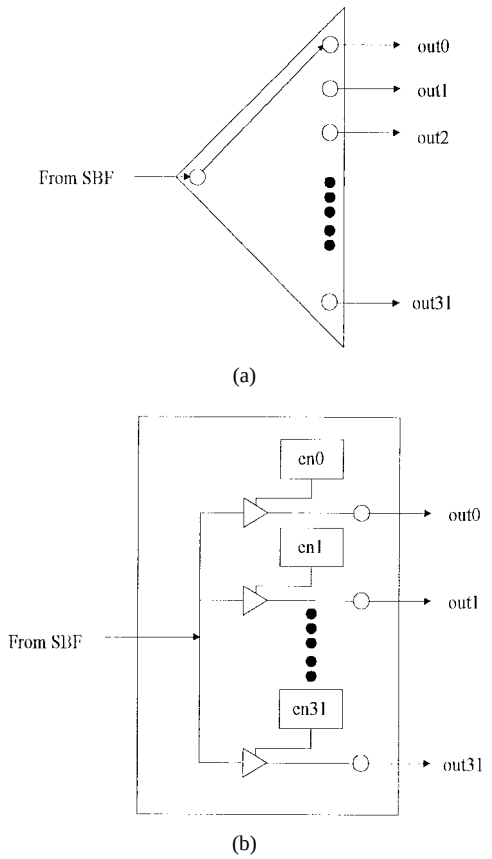

Fig. 12. The block diagram of (a) original and (b) multicast DMX.

outputted to DMX. In addition, the delay caused by translation scheduling[1] must be reduced to enhance the system performance. If the translation time is short enough, the internal buffer size and complexity of scheduling algorithm can be reduced. A block diagram of these multicast queues is shown in Fig. 11.

The original DMX of the shared-buffer ATM switch is used to distribute the output cells of SBM to each output port. In order to provide a cost-effective cell duplicated circuit, we modify the DMX architecture to a multicast DMX. Fig. 12 shows the block diagram of those DMX's. Cell duplication can be achieved by setting the control registers (en0-en31) using the translation result from MCS. The unicast ATM cells can be routed by setting only the corresponding control register to one and the rest to zero.

[1]The multicast controller issues the nontranslated multicast cells which are not currently needed by output circuit, resulting in current output requests which are blocked due to no translated cell in MCS.

TABLE I
MULTICAST TABLE FOR A GROUP OF MEMBERS

	Key field		Link field	Routing Information
Member	Group ID (GID)	Source port (PID)	Next port	Information
A	N	3	1	New VPI/VCI
B	N	1	4	
C	N	4	2	
D	N	2	3	

D. Multicast Server

The multicast server is used to translate input multicast cells to find the destination port bitmap and corresponding VPI/VCI's. As the description in Section III-A, we use ring structure to construct an efficient M2M cell translation mechanism. With the aid of this MCS, the translation delay and the maintenance effort can be reduced.

A block diagram of the proposed multicast server is shown in Fig. 13. The multicast table is used to store and search the routing information of multicast cells and the translated buffer is used to store the translated data (bitmap of destination port and corresponding VPI/VCI's). The actual data format of the multicast table is given in Table I. The data fields of the multicast table include: group ID, source port ID, next port ID, and new VPI/VCI. Group ID (GID) represents the multicast connection ID to specify a certain multicast virtual connection to which the input multicast cell belongs. Source port ID (PID) represents which input port this multicast cell arrives at. Next port ID is used to link current entry to next entry to be searched in the same group.

As depicted in Fig. 14, there are four members in the group: A, B, C, and D. They have a common group ID "N" in the Group ID field. Each member has its own source port ID, next port ID, and new VPI/VCI. We may trace how to search all the members through this structured data.

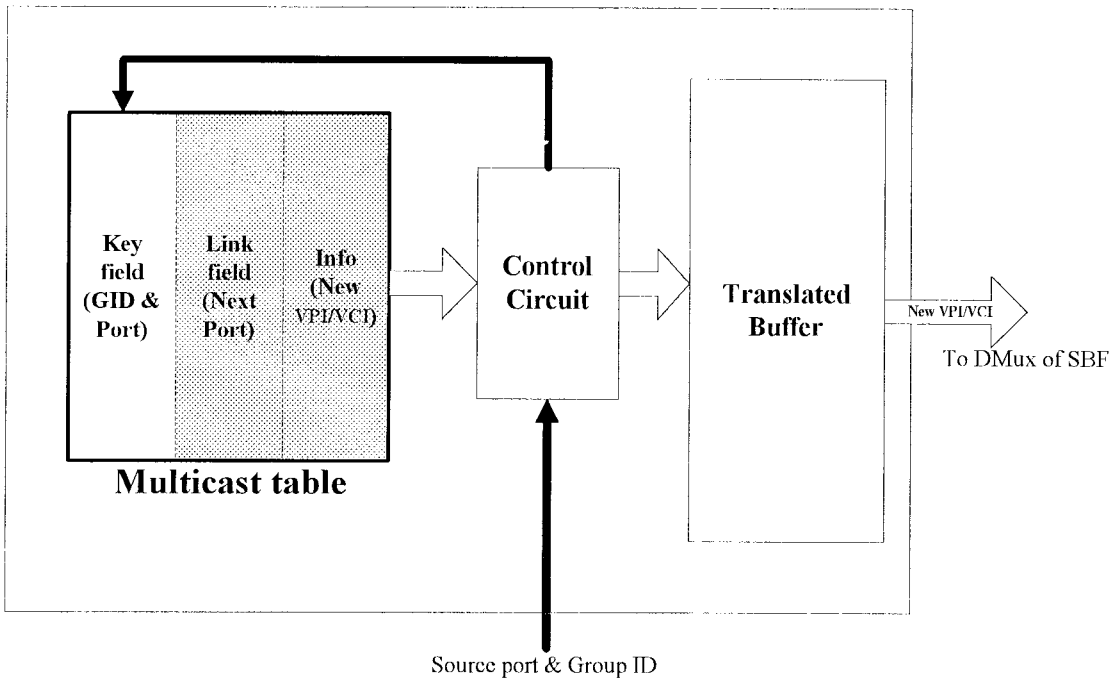

Fig. 13. Block diagram of the multicast server dedicated to handle multicast transmission.

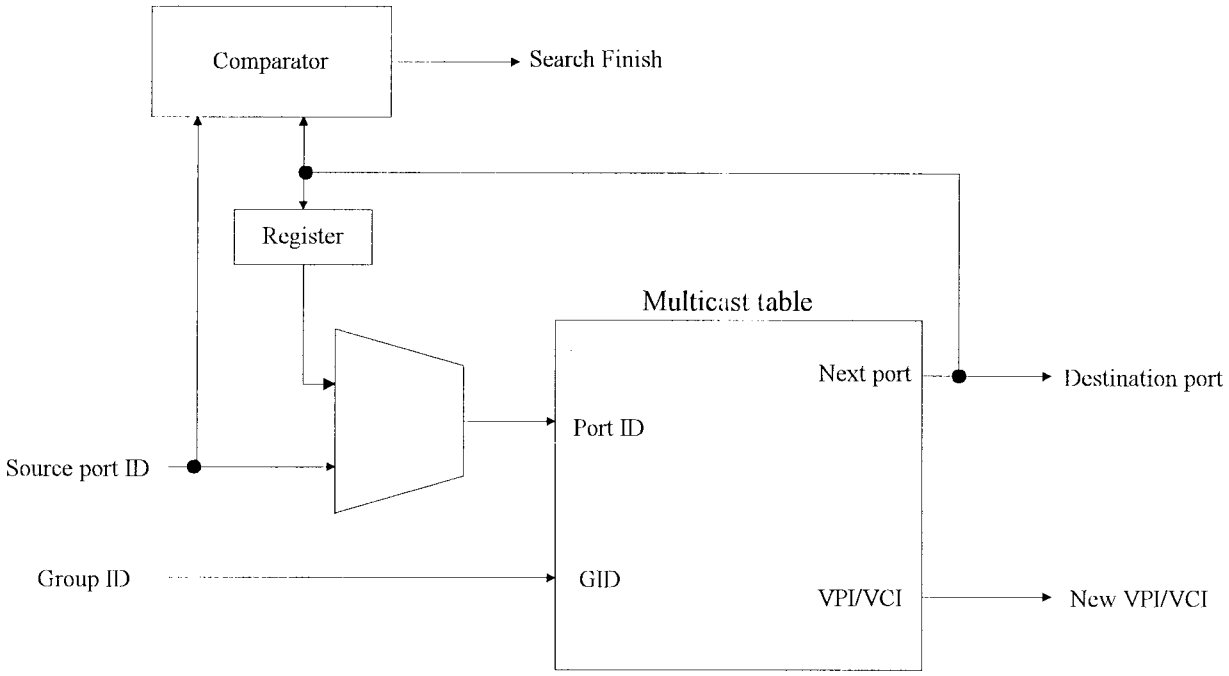

Fig. 14. Block diagram of the searching mechanism.

Both Group and Source port ID's are used as search keys to locate the data entry of the member specified. As the entry being searched is located, the data in the Next port field will replace current Source port ID which is used as a part of searching key for next table search. The Next port and new VPI/VCI in the information field are the output of table lookup operation. The data in Next port field points to the output port to which the multicast cell will be routed. And the data in new VPI/VCI represents the corresponding new VPI/VCI for that output port. Fig. 14 shows the block diagram of this searching mechanism.

As an example, we may trace what would happen when a multicast cell arrives member "A" At the first time, the Group ID "N" and Source port ID "3" drive the GID port and PID port of the multicast table, then member "A" will be located and the information of "A" can be read out. The data in Next port field of "A" is "1," and the value is buffered in the register in the loop path of the Next port data output. At

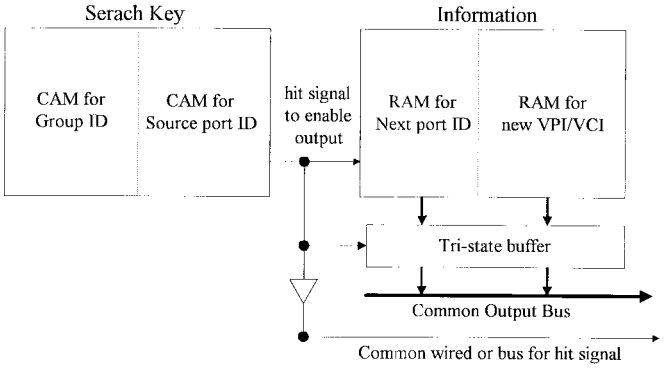

Fig. 15. Block diagram of the table entry which is based on content addressable memory (CAM).

the next search cycle, the PID port is driven by this register, the data driving the PID port becomes "1" instead of "3," and the "B" member is located this time. By repeating this searching process, we can look up the output VPI/VCI and the next member with the same group ID. This search process continues until the data of Next port field is equal to Source port ID of the input multicast cell. In other words, when the next member is the member which starts the table look up, the table look up process is completed.

In order to minimize the translation delay and to enhance flexibility, the whole multicast table is divided into several cascadable table blocks. Each table block is designed by using content addressable memory (CAM). The table block is composed of many table entries as shown in Fig. 15. The table search is performed by all blocks of the whole table simultaneously. The table search of these blocks is synchronized by cascading control signals. The table search and table setup inside a table block are controlled by a local block controller. A block diagram of this table block is shown in Fig. 16.

The usage of table block is monitored by an entry counter. This counter has two functions: one is to provide the data pattern for CAM table initialization before any table operation; the other is to monitor the usage condition of table block, where the counter value indicates the number of entries used. When this counter reaches the table size, the full flag of this block will be enabled to start the operation of next table block. Because this multicast table is designed by using CAM to match the input ID pattern, all the table entries must be initialized before it is activated. During the initialization procedure, the Group ID field of table entries are written with counter values, from 0 to $N-1$, where N denotes the number of entries in a single table block. The entry located at the first physical location is written with zero, and the last one is written with $N-1$. Note, the counter points to the location that could be used to add new entry. When a new entry is added to the table, all blocks are searched with the value pattern of its own counter and the active bit is equal to zero. Then, the new entry is inserted into the matched location of the block which is enabled by the cascading function. To delete an entry, all blocks are searched with the entry pattern whose active bit equals one. The content of the matched location is replaced by (counter -1) and then content of the counter is decremented by one.

TABLE II
COMPARISON AMONG CELL DUPLICATING SCHEMES

	Copy network	Recycle cell	Cell copy	Address copy	Our approach
Extra hardware cost	A copy network (High)	several registers (Low)	no	extra address memory (High)	A one-to-many switch (Low)
Cell delay	Low	High	Low	Low	Low
Increase Cell loss rate	No	No	Yes	No	No
Flexibility and expandability	Depend on network type	Low	High	Depend on the controller	High

During the operation process, many repeated add and delete entry commands are requested. Then the counter patterns in these unused locations are generally distributed. These table entries can be managed efficiently with the aid of this counter. From a procedural viewpoint, the process of adding or deleting an entry looks like the push or pop operation of a stack. The counter plays the role of stack pointer, but the value in this pointer represents the entry pattern instead of physical address. An example demonstrating the table operations with eight entries is given in the Appendix. Moreover with the aid of this counter, cascading multiple table blocks can easily be achieved. This is illustrated Fig. 17. The insertion function of a certain table block is enabled when the **full_in** signal has been set. When this block is full and the **full_in** signal is set, the controller will set the **full_out** signal to enable the insertion function of the next table block. The insertion function of a certain table block can only be activated when its previous table blocks are in full status.

IV. DISCUSSION AND COMPARISON

From above discussion, we can see the proposed multicast scheme provides a cost-effective and cascadable multicast solution with M2M ability for shared-buffer ATM switch. The channel complexity of M2M is now reduced to N connections per N users through the aid of MCS.

However, multicast schemes are strongly dependent on architectures of ATM switches. It is difficult to say which architecture is the best solution for all ATM switch types. But there are still some major parameters that can be used to evaluate available multicast solutions:

1) hardware cost (e.g., extra memory size, circuit complexity);
2) translation delay and delay variance;
3) downgrade of original switch (which increases the cell loss rate and cell delay);
4) functionality (M2M multicast support, multi-QoS's support);
5) flexibility and expandability.

By coupling the cell duplicating circuit into the DMX of the shared-buffer ATM switch, the hardware cost can be greatly reduced and the performance of the whole system remains unchanged. Table II shows the comparison among different cell duplicating schemes. From this comparison we can see our approach provides a more cost-effective cell duplicating scheme compared to available solutions.

Cell translation is the most important part to provide a powerful multicast function such as M2M multicast. But only a few papers [2], [3] discuss that. By using link-list based ring

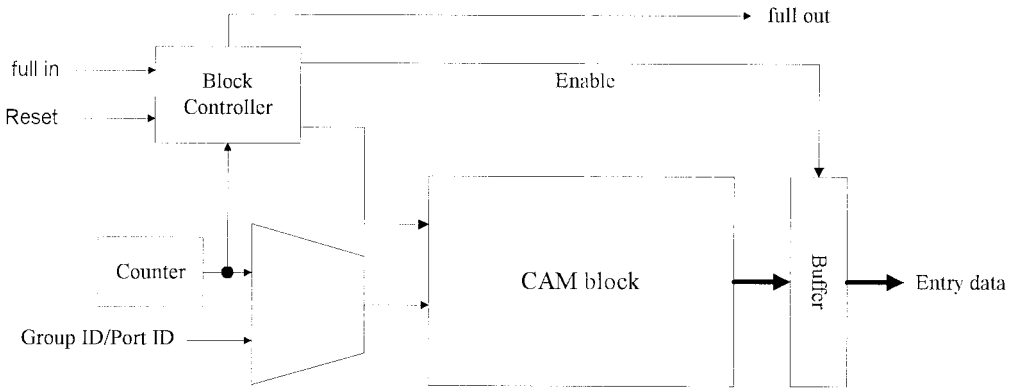

Fig. 16. Block diagram of table block.

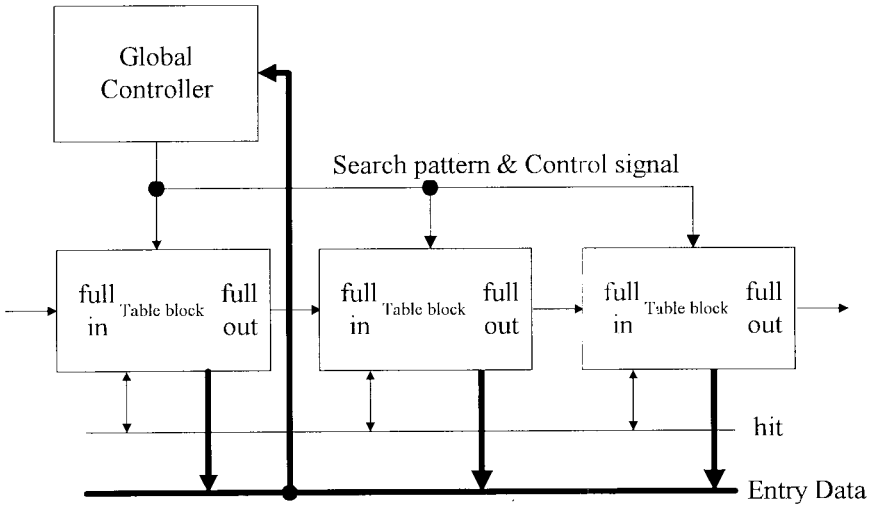

Fig. 17. Block diagram of the cascading MCS table blocks.

structure, our approach provides a high-speed, cascadable, and M2M multicast cell translation function. The translation delay can be reduced by exploiting parallel searching of CAM-based multicast table entries. Based on HSPICE simulation, we find the searching delay is about 5 ns when table block size is equal to 256 in 0.8-μm CMOS SPDM process. Indeed, it is expected that the searching speed can be enhanced by using a smaller table block size. Because the searching speed of MCS is fast enough, the complexity of a multicast cell scheduling circuit in an SBM controller can be reduced, and hence the translation delay variance will not become a problem.

In addition, by using the high-speed QoS processor, our solution can provide two or more QoS's for multicast cells, too. This feature will enhance the flexibility of an ATM switch to process different kinds of multicast applications. For example, LAN emulation and video-on-demand can be assigned to different QoS's to improve network bandwidth utilization.

V. Conclusion

A multicasting solution based on a shared-buffer ATM switch for video applications has been proposed in this paper. The MCS, DMX embedded cell duplicating circuit, and modification on SBM controller are taken into account simultaneously to add the M2M multicast feature for the target shared-buffer ATM switch. The capability of this high-speed and powerful MCS not only reduces the complexity of the SBM controller but also provides an efficient M2M cell translation function. In addition, the channel complexity of our solution can be reduced to N connections, instead of N^2 connections, per N users. Finally, the multicast DMX provides an efficient cell duplicating solution and the multi-QoS's multicast SBM controller provides more suitable multicast features for various video applications.

From the above discussion, we can see the proposed multicast solution can greatly reduce the connection complexity for M2M. It will not only reduce the hardware cost of the switch itself, but also improve the capacity requirement of the whole switch system and clients. A few chips proposed in this paper are now under design and fabrication and will be integrated into a demo multicasting system in the near future.

Appendix
An Example for the Operation of Table Block

Group ID field represents the group id of an active entry. Source port represents the port ID of an active entry. Active

TABLE III
AT INITIAL STATE

Counter=0			
Group ID	Active bit	Source port	Next port and new VPI/VCI
0	0	0	?
1	0	0	?
2	0	0	?
3	0	0	?
4	0	0	?
5	0	0	?
6	0	0	?
7	0	0	?

TABLE IV
INSERT AN ENTRY (N, 1, A)

Counter=1			
Group ID	Active bit	Source port	Next port and new VPI/VCI
N	1	A	a
1	0	0	?
2	0	0	?
3	0	0	?
4	0	0	?
5	0	0	?
6	0	0	?
7	0	0	?

TABLE V
AFTER A CERTAIN ADD/DELETE OPERATION

Counter=3			
Group ID	Active bit	Source port	Next port and new VPI/VCI
N	1	A	a
6	0	0	?
7	0	0	?
N	1	C	c
4	0	0	?
3	0	0	?
5	0	0	?
N	1	B	b

TABLE VI
INSERT AN ENTRY (N, 1, D)

Counter=4			
Group ID	Active bit	Source port	Next port and new VPI/VCI
N	1	A	a
6	0	0	?
7	0	0	?
N	1	C	c
4	0	0	?
N	1	D	d
5	0	0	?
N	1	B	b

bit field is used to decide whether the corresponding entry is valid or not. For example, if Active bit is equal to one, the entry is a valid entry. The Next port and new VPI/VCI field represent the routing information for that entry. The Counter field represents the counter value of the table block. Tables III-VII in this example use an eight-entry table block to show the operations of the table block.

TABLE VII
DELETE AN ENTRY (N, 1, C)

Counter=3			
Group ID	Active bit	Source port	Next port and new VPI/VCI
N	1	A	a
6	0	0	?
7	0	0	?
3	0	0	?
4	0	0	?
N	1	D	d
5	0	0	?
N	1	B	b

ACKNOWLEDGMENT

The authors would like to thank their colleagues within the SI2 group of NCTU for many fruitful discussions, especially K. W. Liu, C. F. Lee, and Y. S. Lee. The MPC support from NSC/CIC is also acknowledged.

REFERENCES

[1] "B-ISDN ATM functional characteristics," Geneva, May 1990, CCITT SG XV III Draft Recommendation I.150.
[2] T.-H. Lee and S.-J. Liu, "Multicasting in a shared buffer memory switch," in *IEEE TENCON'93/Beijing*, vol. 1, pp. 209–212.
[3] H. Saito, H. Yamanaka, H. Yamada, M. Tuzuki, H. Kondoh Y. Matsuda, and K. Oshima, "Multicasting function and its implementation in a shared multibuffer ATM switch," in *IEEE INFOCOM'94*, vol. 1, p. 315.
[4] X. Chen and V. Kumar, "Multicast routing in self-routing multistage networks," in *IEEE INFOCOM'94*, vol. 1, pp. 306–314.
[5] R. Cusani and F. Sestini, "A recursive multistage structure for multicast ATM switching," in *IEEE INFOCOM*, vol. 3, Apr. 1991, pp. 1289–1295.
[6] J. Chao and B.-S. Choe, "Design and analysis of a large-scale multicast output buffered ATM switch," *IEEE/ACM Trans. Networking*, vol. 3, pp. 126–138, Apr. 1995.
[7] C.-T. Lea, "A multicast broadband packet switch," *IEEE Trans. Commun.*, vol. 41, pp. 621–630, Apr. 1993.
[8] X. Chen, J. F. Hayes, and M. K. Mehmet-Ali, "Performance comparison of two input access methods for a multicast switch," *IEEE Trans. Commun.*, vol. 42, pp. 2174–2178, May 1994.
[9] T. T. Lee, R. Boorstyn, and E. Arthurs, "The architecture of a multicast broadband packet switch," in *IEEE INFOCOM*, Mar. 1988, pp. 1–8.
[10] T. Kozaki, "32 × 32 shared buffer type ATM switch VLSI's for B-ISDN's," *IEEE J. Select. Areas Commun.*, vol. 9, pp. 1239–1247, Oct. 1991.
[11] S. Hino, M. Togashi, and K. Yamasaki, "Asynchronous transfer mode switching LSI chips with 10-Gb/s serial I/O ports," *IEEE J. Select. Areas Commun.*, vol. 30, pp. 348–352, Apr. 1995.
[12] H. Kondoh, H. Notani, H. Yamanaka, K. Higashitani, H. Saito, I. Hayashi, S. Kohama, Y. Matsuda, K. Oshima, and M. Nakaya, "A 622-Mb/s 8 × 8 ATM switch chip set with shared multibuffer architecture," *IEEE J. Select. Areas Commun.*, vol. 28, pp. 808–815, July 1993.
[13] W. Fischer, O. Fundneider, E. H. Goeldner, and K. A. Lutz, "A scalable ATM switching system architecture," *IEEE J. Select. Areas Commun.*, vol. 9, pp. 1299–1307. Oct. 1991.
[14] A. Itoh, W. Takahashi, H. Nagano, M. Kurisaka, and S. Iwasaki, "Practical implementation and packaging technologies for a large-scale ATM switching system," *IEEE J. Select. Areas Commun.*, vol. 9, pp. 1280–1288, Oct. 1991.
[15] T. R. Banniza, G. J. Eilenberger, B. Pauwels, and Y. Therasse, "Design and technology aspects of VLSI's for ATM switches," *IEEE J. Select. Areas Commun.*, vol. 9, pp. 1255–1264, Oct. 1991.
[16] M. Katevenis, S. Sidiropoulos, and C. Courcoubetis, "Weighted round-robin cell multiplexing in a general-purpose ATM switch chip," *IEEE J. Select. Areas Commun.*, vol. 9, pp. 1265–1279, Oct. 1991.
[17] Y. Shobatake, M. Motoyama, E. Shobatake, T. Kamitake, S. Shimizu, M. Noda, and K. Sakaue, "A one-chip scalable 8 ∗ 8 ATM switch LSI employing shared buffer architecture," *IEEE J. Select. Areas Commun.*, vol. 9, pp. 1248–1254, Oct. 1991.
[18] C. Y. Lee, P. W. Hsieh, and J. M. Tsai, "High-speed median filter designs using shiftable content-addressable memory," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, pp. 544–549, Dec. 1994.
[19] J. M. Tsai and C. Y. Lee "A novel architecture for ATM QoS management," *IEE Proc.*, submitted.

Jer-Min Tsai received the B.S. degree from National Cheng Kung University, Tainan, Taiwan, in 1989 and the M.S. degree from National Chiao Tung University (NCTU), Hsinchu, Taiwan, in 1993, both in electrical engineering. He is currently working toward the Ph.D. degree at the NCTU.

His research interests include VLSI architectures, high-speed networking, and related VLSI designs.

Hsin-Hsiung Fang received the B.S. degree from National Central University, Chung-Li, Taiwan, in 1988 and the M.S. degree from National Chiao Tung University (NCTU), Hsinchu, Taiwan, in 1996, both in electrical engineering.

His research interests include VLSI architectures, high-speed networking, local area network, and related VLSI designs.

Chen-Yi Lee received the B.S. degree from National Chiao Tung University, Taiwan in 1982 and the M.S. and Ph.D. degrees from Katholieke University Leuven (KUL), Belgium in 1986 and 1990, respectively, all in electrical engineering.

From 1986 to 1990 he was with IMEC/VSDM, working in the area of architecture synthesis for DSP. In February 1991, he joined the faculty of the Electronics Engineering Department, National Chiao Tung University, Hsinchu, Taiwan, where he is now a Professor. His research interests mainly include VLSI algorithms and architectures for high-throughput DSP applications. He is also active in various aspects of very low bit coding, multimedia signal processing, high-speed networking, and system-on-chip design technology.