IEICE TRANS. COMMUN., VOL.E83–B, NO.2 FEBRUARY 2000

PAPER IEICE/IEEE Joint Special Issue on Recent Progress in ATM Technologies

Performance of Scalable-Distributed-Arbitration ATM Switch Supporting Multiple QoS Classes

Eiji OKI[†], **Naoaki YAMANAKA**[†], *and* **Masayoshi NABESHIMA**[†], *Members*

SUMMARY A multi-QoS scalable-distributed-arbitration (MSDA) ATM switch is described that supports both high- and low-priority traffic under the head-of-line-priority discipline. It uses crosspoint and transit buffers, each consisting of a high- and low-priority buffer. The buffers arbitrate in a distributed manner the selection of which cellsto transmit. The MSDA switch supports multiple QoS classes while still providing the scalability of a previously described single-QoS scalable-distributed-arbitration (SSDA) switch. A problem occurs when the delay-time-based cell-selection mechanism used in the SSDA switch is applied to the low-priority traffic: it cannot achieve fairness in terms of throughput. This problem is overcome by introducing a distributed-ring-arbiter-based cell-selection mechanism at each crosspoint for the low-priority traffic. The low-priority transit buffer at each crosspoint has virtual queues, one for each upper input port. Cells for the low-priority traffic are selected by distributed-ring arbitration among the low-priority crosspoint buffer and these virtual queues. For the high-priority traffic, the same delay-time-based cell-selection mechanism is used as in the SSDA switch. Simulations show that the MSDA switch ensures fairness interms of delay time for the high-priority traffic and ensures fairness in terms of throughput for the low-priority traffic.
key words: *ATM, switch, arbitration, QoS, fairness*

1. Introduction

Asynchronous transfer mode (ATM) is being used to provide multimedia communication networks. The demand for multimedia services, such as high-speed data communications and high-definition television broadcasting, will increase. Therefore, ATM switches that have large throughput (over 1 Tbit/s) must be developed [1]–[3].

For the speeds required, the crosspoint-buffer-type ATM switch, which has cell buffers at each crosspoint, is superior to the shared-memory-type or input-output-buffer-type ATM switches because it does not require high-speed memory access or high-speed internal cell transmission [4]–[7]. This makes it easy for a crosspoint-buffer-type switch to act as ahigh-speed ATM switch [8].

However, a crosspoint-buffer-type switch has trouble when the output lines are fast or there are many input ports. The faster the output lines, the shorter the cell-transmission time. The more input ports there are, the bigger the arbitration ring. Therefore, in a switch with many input ports and/or fast output lines, ring arbitration cannot be completed within one cell-time. Therefore, in conventional switches based on ring arbitration, the arbitration time and the number of input ports limits the output-line speed; ring arbitration must be completed within the cell-time [9].

We previously described a crosspoint-buffer-type switch that solves the problem of conventional switches based on ring arbitration [10], [11]. It is based on scalable distributed arbitration (SDA) and provides a single quality-of-service (QoS) class. This single-QoS SDA (SSDA) switch has a crosspoint buffer and a transit buffer at every crosspoint. When a cell is sent to an output port, it is transferred from a crosspoint buffer to the output port by way of several transit buffers. Arbitration is performed between the crosspoint buffer and a transit buffer at each crosspoint. A cell is selection for transmission based on the delay time by using a synchronous counter. If two cells have the same delay times, one of them is selected based on a certain probability for each crosspoint buffer. The longest control-signal transmission length for arbitration within one cell-time is only between two adjacent crosspoints. Therefore, because the arbitration time does not depend on the number of input ports, an SSDA switch can handle a large numberof input ports while supporting high output-line speeds.

To support multiple QoS classes, priority queueing control at each crosspoint buffer is needed. One priority queueing approach is the head-of-line priority (HOLP) technique [13]. Consider two priority buffers. Under the HOLP system, cells waiting in the low-priority buffer (delay-tolerant) are served only if there are no cells a waiting transmission in the high-priority (delay-sensitive) buffer. Therefore, in the HOLP discipline, the low-priority traffic uses only the residual bandwidth.

The low-priority class tolerates delay time while the high-priority class requires that the delay time be short. In addition, the low-priority class is a best-effort service class, such as unspecified bit rate (UBR). It requires fairness in terms of throughput rather than in terms of delay time, which the high-priority class requires, even when total input traffic is overloaded.

However, a problem occurs when we use the SSDA mechanism in a HOLP system supporting multiple QoS classes. The delay-time-basedcell-selection mechanism used in the SSDA switch does not ensure fairness in

Manuscript received May 31, 1999.
Manuscript revised August 25, 1999.
[†]The authors are with NTT Network Service Systems Laboratories, Musashino-shi, 180-8585 Japan.

terms of throughput for the low-priority class. Consider that the input traffic intended for a certain output port is excessive. In the delay-time-based cell-selection mechanism, if the number of cells destined to the output port from one input port is larger than that from another input port, the throughput of the former tends to be higher that that of the latter. This is because the delay-time-based cell-selection mechanism corresponds to a first-in-first-out (FIFO) buffer to some extent. Therefore, we need to develop a cell-selection mechanism that maintains fairness in terms of delay time for the high-priority class and fairness in terms of throughput for the low-priority class, while maintaining the scalability of SSDA switches.

There is another reason the delay-time-based cell-selection mechanism for the low-priority class is not used in the HOLP system. The delay time of cells in the low-priority buffer will be very large and the upper bound of the low-priority delay time cannot be designed because of the limited number of bits in the cell header available for measuring the delay.

In this paper we describe a multi-QoS SDA (MSDA) switch that supports both the high- and low-priority classes [12]. To provide a cell-selection mechanism for the low-priority class, we introduce a distributed-ring-arbiter-based selection mechanism at each crosspoint for the low-priority class. The low-priority transit buffer at each crosspoint has virtual queues, one for each upper input port. Cells to be transmitted from the low-priority class are selected by distributed-ring arbitration from among the low-priority crosspoint buffer and these virtual queues. For the high-priority class, we use the same delay-time-based cell-selection mechanism as in the SSDA switch. As a result, the proposed MSDA switch ensures fairness interms of delay time for the high-priority class and ensures fairness in terms of throughput for the low-priority class.

The remainder of this paper is organized as follows. Section 2 explains the problems in using switches based on conventional ring arbitration. Section 3 describes the SSDA switch for a single QoS class. Section 4 presents the MSDA switch for multiple QoS classes. Section 5 discusses the performance of the MSDA switch. Finally, Sect. 6 summarizes the key points.

2. Conventional Switch

A crosspoint-buffer-type switch using conventional ring arbitration is shown in Fig. 1. To simplify the discussion, we assume that the switch supports a single QoS class. In this switch, the ring arbiter searches, from some starting point, for a crosspoint buffer that has made a request to transfer a cell to the output line. It starts at the crosspoint buffer immediately below the one from which a cell was sent at the previous cell-time. If it finds such a buffer, the cell at the head of the buffer is transferred. At the next cell-time, the starting point is the next crosspoint buffer. Thus, in the worst case, the control signal for ring arbitration must pass through all the crosspoint buffers with the same output line within one cell time. The maximum output-line speed is thus limited by the number of input ports (i.e., the switch size) and by the transmission delay of the control signals in each crosspoint.

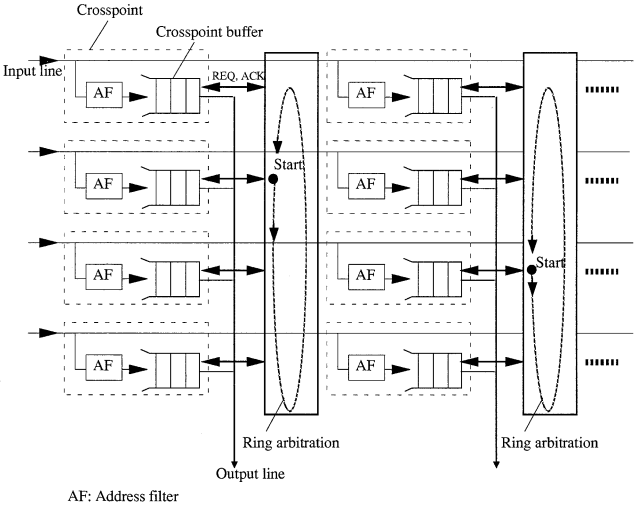

Fig. 1 Conventional crosspoint-buffer-type switch structure based on ring arbitration.

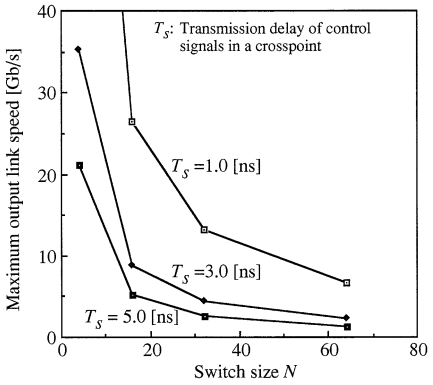

Fig. 2 Maximum output-link speed in conventional switch.

Maximum output-line speed C_{max} [bit/s] is given by

$$C_{max} = \frac{L}{NT_s}, \tag{1}$$

where N is the number of input ports, or the switch size, T_s [s] is the transmission delay of the control signals in a crosspoint, and L [bits] is the length of a cell. The delay depends on the device performance and the distance between crosspoints. We assume that there is no guard time in each cell-time.

Figure 2 shows the relationship between the number of input ports and the maximum output link

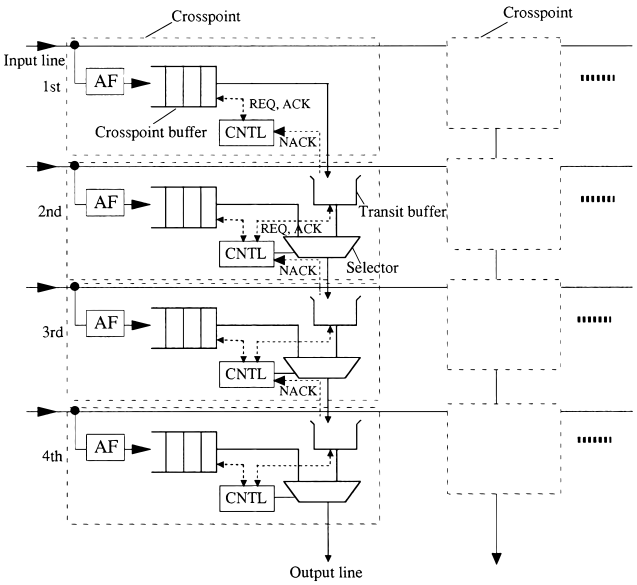

Fig. 3 Single-QoS scalable-distributed-arbitration (SSDA) switch structure.

speed for three different delay times in the conventional switch. The cell length was set to 53×8 [bits]. As N increased, C_{max} decreased. For example, at $T_s = 3.0$ [ns] and $N = 16$, C_{max} was 8.8 [Gb/s]. The decrease in C_{max} depends on T_s. As a result, unless the delayis made small by using ultra-high-speed devices, the conventional switch cannot achieve large throughput.

3. Single-QoS SDA (SSDA) Switch

The crosspoint-buffer-type switch based on a distributed arbitration scheme we described [10] is illustrated in Fig. 3. This single-QoS scalable-distributed-arbitration (SSDA) switch has an address filter (AF), a crosspoint buffer, a transit buffer, an arbitration-control part (CNTL), and a selector at every crosspoint (except that the first crosspoint does not need a transit buffer or selector).

Each cell-time, the crosspoint buffer sends a request (REQ) to CNTL if it contains at least one cell. The transit buffer stores several cellssent from either the upper crosspoint buffer or the upper transit buffer. Each cell-time, it sends a REQ to CNTL, like the crosspoint buffer does, if it contains at least one cell. If it is close to becoming full, it sends a not-acknowledgment (NACK) to the upper CNTL.

If there are one or more REQs and CNTL does not receive a NACK from the next lower transit buffer, it selects, within one cell-time, a cell to send. It determines which cell to send according to the following cell-selection rule. The selected cell is sent through the selector to the next lower transit buffer or to the output line.

A cell is selected within a crosspoint as follows. If either the crosspoint buffer or the transit buffer requests a cell release, the cell in the requesting buffer is selected. If both buffers request a cell release, then the cell with the larger delay time is selected. The delay time is defined as the time since the cell entered the crosspoint buffer.

One way of comparing the delay time of competing cells is to use a synchronous counter, which needs S bits, which are included in the overhead bits in each cell. The synchronous counter is incremented by one at each 2^G cell-time. The granularity of the delay time is a parameter, G. The delay time is measured most accurately when $G = 0$. When a cell enters a crosspoint buffer, the value of the synchronous counter is written in the cell's overhead. When both the crosspoint and transit buffers in a crosspoint issue requests for cell release, the values in the counters are compared. If the difference is less than 2^{S-1}, the cell with the smaller value is selected. Conversely, if the difference is equal to or more than 2^{S-1}, the cell with the larger value is selected. This delay-time comparison is effective if the maximum delay time is less than $2^{G+S-1} (= 2^G 2^{S-1})$. We thus set the value of G so that this relation is satisfied.

A simple example may help to clarify the above explanation. If the counter value reaches the maximum value (256 $(=2^8)$ when $S = 8$), then the next value will become 0. Thus, the counter is reset each 2^{G+S} $(=2^G 2^S)$ cell-time. We assume that the maximum delay time is less than 128 $(=2^7)$ and $G = 0$. If one value is 110 and the other value is 13, the cell having the value of 110 is older than the other cell if we consider the cyclic increment of the counter. Note that if the condition that the maximum delay time is less than 128 $(=2^7)$ is not satisfied, this comparison does notwork. If $G = 1$, the value of S can be reduced by one.

When the delay time of the head-of-line cell in the crosspoint buffer equals that in the transit buffer, CNTL determines which cell to send by using a second cell-selection rule. Let us consider the kth crosspoint and transit buffers starting at the top. The second rule is that the kth crosspoint buffer is selected with probability $1/k$, while the kth transit buffer is selected with probability $(k-1)/k$. For example, the third crosspoint and transit buffers are selected with probabilities of 1/3 and 2/3, respectively. According to this rule, the total probability that a cell from any crosspoint buffer is selected for delivery to an output line is a constant value $1/N$.

The SSDA switch thus achieves distributed arbitration at each crosspoint. The longest control-signal transmission distance for arbitration within one cell-time is obviously the distance between two adjacent crosspoints. (In the conventional switch, the control signal for ring arbitration must pass through all crosspoint buffers with the same output line.) Therefore, the arbitration time does not depend on the number of

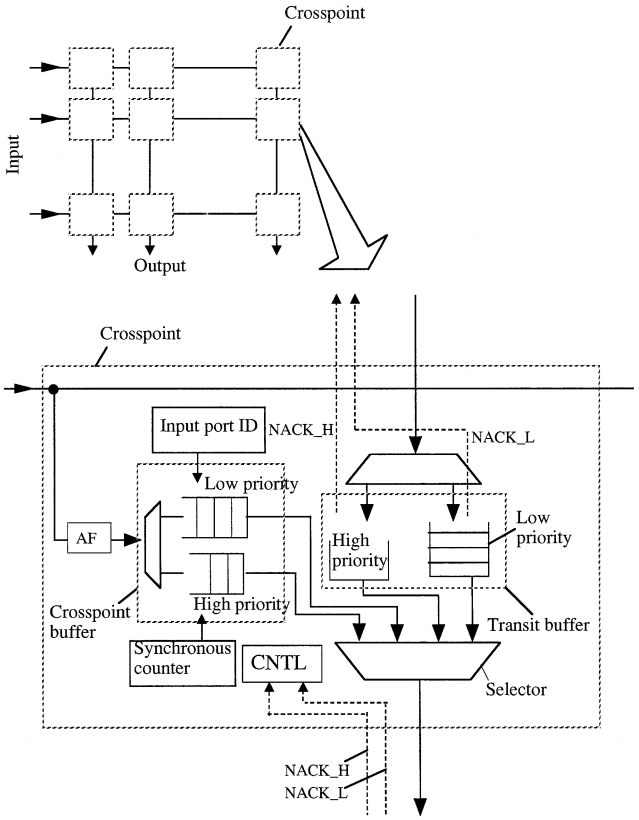

Fig. 4 Multi-QoS SDA (MSDA) switch structure.

input ports.

4. Multi-QoS SDA (MSDA) Switch

This section describes our proposed multi-QoS SDA switch (MSDA), which supports multiple QoS classes under the HOLP discipline. Although we consider only two priority classes in this paper for simplicity, more than two classes can easily be handled.

To avoid using the delay-time-based cell-selection mechanism for the low-priority class, a distributed-ring-arbiter-based cell-selection mechanism is used at each crosspoint for the low-priority class.

The MSDA switch has a matrix of crosspoints. As shown in Fig. 4, each crosspoint has a crosspoint buffer and a transit buffer, each consisting of a high-priority and low-prioritybuffer, an arbitration-control part (CNTL), and a selector.

Each incoming cell passes through an address filter (AF) and is placed in either the high- or low-priority crosspoint buffer according to its priority class. At that time, at the high-priority buffer, the value of a synchronous counter is written into the cell header, like in the SSDA switch. At the low-priority crosspoint buffer, an input port ID (identifier) is written into the header instead. Note that the values of the synchronous counter and port ID are written into the same bits. Therefore, no additional cell-overhead bits

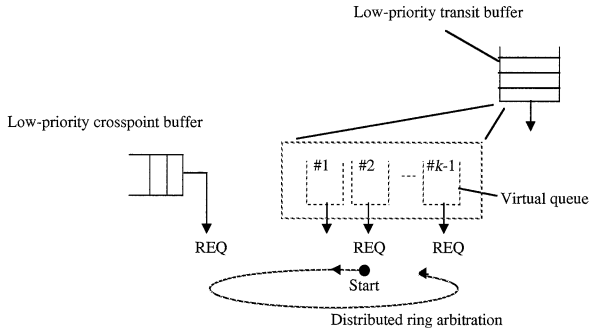

Fig. 5 Low-priority selection rule.

are required to move from SSDA to MSDA. For example, at the kth crosspoint, the value of the input-port ID is k. This is used to identify from which input port a low-priority cell came†. The high- and low-priority crosspoint buffers send a REQ to CNTL if there is at least one cell stored in each.

A cell transmitted from an upper crosspoint enters into either the high-priority transit buffer or the low-priority transit buffer of the next lower crosspoint according to the priority class. The low-priority transit buffer has a FIFO (first-in-first-out) buffer function to transmit low-priority cells. In addition, it has $k-1$ virtual queues, which are $\#1$, $\#2$, $\cdots$, $\#k-1$, as shown in Fig. 5. Using these virtual queues makes the implementation simple [14], [15]. Each virtual queue has a counter. When a low-priority cell arrives from the upper crosspoint (the $k-1$th crosspoint), it is simply stored at the end of the line in the low-priority transit buffer. At the same time, the counter value of the virtual queue $\#i$ corresponding to the input-port ID i $(1 \leq i \leq k-1)$ of the cell is incremented by one. When a cell leaves the queue, the counter for its virtual queueis decremented by one.

Each cell-time, the high-priority transit buffer sends a REQ to CNTL if it has at least one cell, and each low-priority virtual queue sends a REQ to CNTL if its counter value is not zero. If the high- orlow-priority transit buffers are about to become full, they send not-acknowledgments NACK_H or NACK_L, respectively, to CNTL in the upper crosspoint.

The cell-selection algorithm in the MSDA switch works as follows. If CNTL receives a NACK_H from the high-priority transit buffer in the lower crosspoint, neither a high-priority cell nor a low-priority cellis transmitted. This is because the lower high-priority transit bufferis about to become full, so there is no chance for

†This statement is true until alow-priority cell which enters the kth crosspoint buffer is transmitted from the $k+1$th transit buffer. However, as will be explained later, due to an input-ID-replacement operation, after the low-priority cell is transmitted from the $k+1$th buffer, the input-port ID is not always the same as the input port from which the cell came.

a low-priority cell in a lower transit buffer to be transmitted. Low-priority cells can not be transmitted when there is at least one high-priority REQ from the crosspoint or transit buffer. When both the high-priority crosspoint buffer and the high-priority transit buffer send REQs to CNTL, the high-priority cell-selection rule used is the cell-selection rule used in the SSDA switch.

Low-priority cells can be transmitted only when there are no REQs from either the high-priority crosspoint buffer or the high-priority transit buffer. If this condition is satisfied and CNTL does not receive either a NACK_H or NACK_L from the lower transit buffer, then the low-priority selection rule is used. The low-priority crosspoint buffer and virtual queues in the low-priority transit buffer send REQs to CNTL as shown in Fig. 5. Ring arbitration is executed at each crosspoint in a distributed manner. If the ring arbiter accepts the REQ of the low-priority crosspoint buffer, the head-of-line cell in the low-priority crosspoint buffer is transmitted to the lower crosspoint. If the ring arbiter accepts a REQ that is sent from one of the virtual queues, the head-of-line cell in the low-priority transit buffer is transmitted to the lower crosspoint, and the counter value of the accepted virtual queue is decremented by one.

When the head-of-line cell in the low-priority transit buffer is transmitted to the lower crosspoint, a newly introduced input-ID-replacement operation replaces the input ID of the cell by the ID number of the selected virtual queue. Thanks to this operation, the series of IDs for the cells are transmitted from the transit buffer in round-robin order; therefore, these cells enter the lower transit virtual queues in the same round-robin order. As a result, the threshold at which a NACK_L in the low-priority transit buffer is sent to the upper CNTL has only to be a few cells to achieve fairness in terms of low-priority throughput.

The MSDA switch thus achieves distributed arbitration at each crosspoint. It uses the delay-time-based cell-selection rule for the high-priority class and the distributed-ring-arbiter-based cell-selection rule for the low-priority class.

5. Performance of MSDA Switch

We evaluated the performance of the MSDA switch by computer simulation. We assumed that in an $N \times N$ crosspoint-buffer-typeswitch, the input traffic for both the high- and low-priority classes is random for simplicity. We set $G = 0$.

The high-priority class is not affected by, but does affect, the low-priority class. We first present the performance of the MSDA switch for the high-priority class described in Sect. 5.1. Because the performance for the high-priority class has nothing to do with the state of the low-priority class, it is the same as that of the SSDA switch, which was previously described [10]. Therefore, in Sect. 5.1, we mainly discuss fairness in terms of delay time for both balanced and unbalanced input traffic conditions. Then we present the performance for the low-priority class in Sect. 5.2.

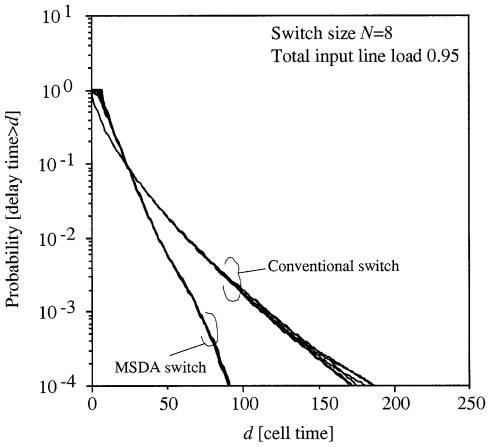

Fig. 6 Probability of delay time in balanced traffic for high priority class.

5.1 High-Priority Class

The MSDA switch ensures fairness in terms of delay time for the high-priority class. Figure 6 shows the probability of the delay time being larger than a certain time d at$N = 8$. The high-priority input load was 0.95. We assumed that the input traffic loads from the different input ports were the same, that is, the traffic load was balanced. The probability is shown for each high-priority crosspoint buffer entered by cells. We set the threshold at which a NACK_H is sent to the upper CNTL, TH_h, to two cells in this simulation. In other words, when the queue length reached TH_h, a NACK_H was sent to the upper CNTL. In this case, each high-priority transit buffer needed four cells ($2 * TH_h$) to avoid cell loss in their high-priority buffer. The minimum value of TH_h was determined by considering the transmission delay of a NACK_H signal. In an overload situation, the minium value of TH_h should be set so that at least one cell is stored in the high-priority buffer. This will achieve the highest possible throughput without cell loss. The minimum value is determined by the implementation. The performance of the high-priority class does not depend on the value of TH_h if it is equal to or more than the minimum value. The delay time is defined as the time from the cell's entering the high-priority crosspoint buffer until it reaches the output line.

In the MSDA switch, when d is more than about 10 [cell times], all delay times have basically the same probability and delay time fairness for the high-priority class is achieved. (Because it takes at least N=8 [cell times] for a cell in the top high-priority crosspoint

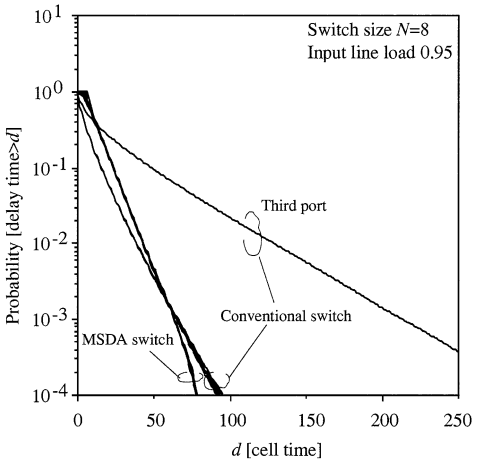

Fig. 7 Probability of delay time in unbalanced traffic for high priority class.

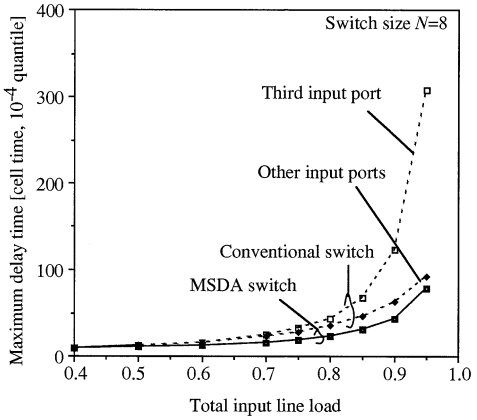

Fig. 8 Maximum delay time in unbalanced traffic for high-priority class.

buffer to enter the output line, fairness is not maintainedat shorter values.)

In addition, when d is larger than a certain time, the probability of the MSDA switch delay time being larger than d is smaller than that of the conventional switch based on ring arbitration, as shown in Fig. 6. This is because, in the MSDA switch, the high-priority cell with the largest delay time is selected. Here, the conventional switch uses ring arbitration, as shown in Fig. 1. This delay-reduction effect enables us to reduce the required high-priority crosspoint buffer size while still guaranteeing a certain cell-loss ratio [10].

Note that because this subsection focuses on only the high-priority class, we can consider the conventional switch shown in Fig. 1. Even if the conventional switch based on ring arbitration for a single QoS class is extended to that for multiple QoS classes, the performance of the high-priority class is the same as that for a single QoS class.

Even when the input traffic is not balanced, the MSDA switch keeps fairness in terms of delay time for the high-priority class, as shown in Fig. 7. The input traffic load at the third input port from the top is twice that at the other input ports. The total input load is 0.95, which is the same as that in Fig. 6. In the MSDA switch, all delay times forthe high-priority class have basically the same probability, while in the conventional switch, the delay time of the third input port is much larger than that of others. Thus, the MSDA switch ensures delay-time fairness for the high-priority class even under unbalanced input traffic condition.

Figure 8 shows the maximum delay time (10^{-4} quantile) of an MSDA switch in unbalanced traffic-load conditions. The traffic conditions are the same as those in Fig. 7 except for the total input line load. Because the maximum delay times of the MSDA switch for all input ports are the same, only one solid line is depicted, while in the conventional switch, a dotted line for the third input port and a dotted line for other input ports are depicted. For various total input line loads, we can see that the difference between the maximum delay time for the third input port and those for the other ports becomes larger in the conventional switch as the total input line load increases, while in the MSDA switch, fairness with respect to delay time is maintained for all the total input line loads.

5.2 Low-Priority Class

Tables 1, 2, 3, 4 and 5 show that the MSDA switch maintains fairness in terms of throughput for the low-priority class. We present the results for five sets of input traffic conditions, case 1, case 2, case 3, case 4, and case 5. The switchsize was set to $N = 8$. We set the threshold at which a NACK_L is sent to the upper CNTL, TH_l, to two cells. Therefore, each low-priority transit buffer needs at least four cells ($2 * TH_l$) to avoid cell loss in the low-priority transit buffer. The minimum value of TH_l is determined by considering the transmission delay of a NACK_L signal, as is the case with the high-priority class. In an overload situation, the minium value of TH_l should be set so that at least one cell is stored in the low-priority buffer. This will achieve the highest possible throughput without cell loss. Here, we assume that the minimum value is $TH_l = 2$. As will be discussed below, the desired fairness is achieved if the value of TH_l is equal to or more than the minimum value.

In this simulation, the size of the low-priority transit buffer was set to four ($=TH_l * 2$) cells. When examining TH_l dependency, the size of the low-priority transit buffer was set to $TH_l * 2$. The size of the low-priority crosspoint buffer was set to 256 cells. Note that the value of the low-priority crosspoint-buffer size does not affect throughput fairness if it is set enough large, even when traffic overloading exists.

In case 1, the high-priority load of the fourth input port is 0.18 and that of the other input ports is 0.06.

Table 1 Throughput in MSDA switch (case 1).

Input port	Input load (High)	Input load (Low)	Throughput (High)	Throughput (Low)
1	0.060	0.050	0.060	0.050
2	0.060	0.050	0.060	0.050
3	0.060	0.150	0.060	0.050
4	0.180	0.050	0.180	0.050
5	0.060	0.050	0.060	0.050
6	0.060	0.050	0.060	0.050
7	0.060	0.050	0.060	0.050
8	0.060	0.050	0.060	0.050
Total	0.600	0.500	0.600	0.400

Table 2 Throughput in MSDA switch (case 2).

Input port	Input load (High)	Input load (Low)	Throughput (High)	Throughput (Low)
1	0.060	0.050	0.060	0.050
2	0.060	0.050	0.060	0.050
3	0.060	0.050	0.060	0.050
4	0.180	0.050	0.180	0.050
5	0.060	0.050	0.060	0.050
6	0.060	0.050	0.060	0.050
7	0.060	0.050	0.060	0.050
8	0.060	0.150	0.060	0.050
Total	0.600	0.500	0.600	0.400

Table 3 Throughput in MSDA switch (case 3).

Input port	Input load (High)	Input load (Low)	Throughput (High)	Throughput (Low)
1	0.060	0.030	0.060	0.030
2	0.060	0.030	0.060	0.030
3	0.060	0.390	0.060	0.190
4	0.180	0.030	0.180	0.030
5	0.060	0.030	0.060	0.030
6	0.060	0.030	0.060	0.030
7	0.060	0.030	0.060	0.030
8	0.060	0.030	0.060	0.030
Total	0.600	0.600	0.600	0.400

Table 4 Throughput in MSDA switch (case 4).

Input port	Input load (High)	Input load (Low)	Throughput (High)	Throughput (Low)
1	0.060	0.030	0.060	0.030
2	0.060	0.030	0.060	0.030
3	0.180	0.030	0.180	0.030
4	0.060	0.030	0.060	0.030
5	0.060	0.030	0.060	0.030
6	0.060	0.390	0.060	0.190
7	0.060	0.030	0.060	0.030
8	0.060	0.030	0.060	0.030
Total	0.600	0.600	0.600	0.400

The low-priority load of the third input port is 0.15 and that of the other input ports is 0.05, as shown in Table 1. The total input load is 1.1 (0.6 +0.5), which is an overload. The output load, which we call throughput,for the high-priority class is the same as the high-priority inputload for each input port. The residual bandwidth of 0.4 $(1.0 - 0.6)$ is divided by the number of ports (8), resulting in a bandwidth of 0.05 for each input port, as along as all the low-priority input loads

Table 5 Throughput in MSDA switch (case 5).

Input port	Input load (High)	Input load (Low)	Throughput (High)	Throughput (Low)
1	0.060	0.070	0.060	0.050
2	0.060	0.070	0.060	0.050
3	0.060	0.070	0.060	0.050
4	0.180	0.070	0.180	0.050
5	0.060	0.070	0.060	0.050
6	0.060	0.070	0.060	0.050
7	0.060	0.070	0.060	0.050
8	0.060	0.070	0.060	0.050
Total	0.600	0.560	0.600	0.400

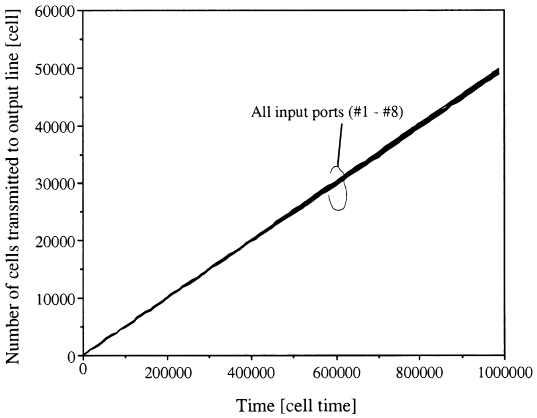

Fig. 9 Number of low-priority cells transmitted to output line (case 1).

are equal to or greater than 0.05. In this way, the residual bandwidth is shared fairly among the low-priority input traffic loads, although their requests for bandwidth are different. We verified that the results shown in Table 1 are satisfied independently of time, as shown in Fig. 9. We plotted the number of low-priority cells transmitted from each input port to the output line over time. These plots overlap in a straight line.

In case 2, to determine whether the location of the low-priority heavy-load input affects throughput, we moved the low-priority heavy-load traffic from the third input port (case 1) to the eighth one. The other input traffic conditions were the same as in case 1. Table 2 shows the input load conditions and results. Comparing the results in Table 2 with those for case 1, we found no change in the throughput.

In case 3, the low-priority load of the third input port was 0.39 and that of other the input ports was 0.03, as shown in Table 3. The high-priority input load was the same as in case 1. The total input load was 1.2 (0.6 + 0.6), which is again an overload condition. In case 3, the low-priority throughput for the input ports, except for the third input port was 0.03, which was the same as the input load, and the low-priority throughput for the third input port was 0.19, which is larger than 0.03. As shown in Table 3 for the low-priority class, the MSDA switch achieved a max-min fair share [16]. This is explained as follows. A bandwidth of 0.03 was first

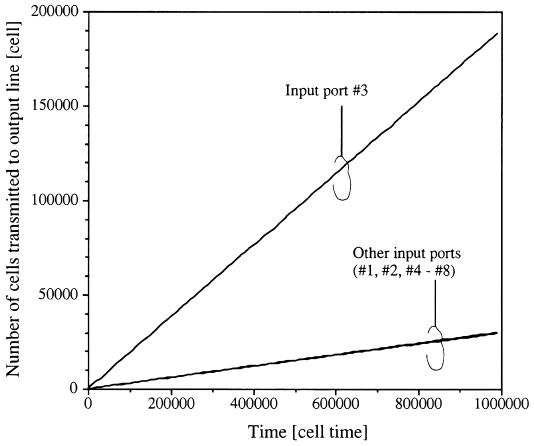

Fig. 10 Number of low-priority cells transmitted to output line (case 3).

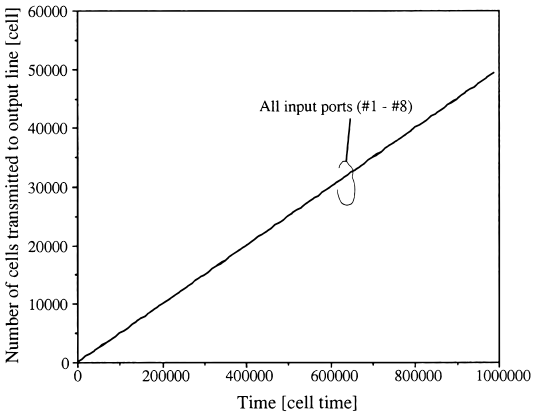

Fig. 11 Number of low-priority cells transmitted to output line (case 5).

assigned to each input port for the low-priority traffic. This satisfied the requirements of all the input ports, except for the third one. The residual bandwidth was 0.16 $(1 - 0.6 - 0.03 \times 8)$. It was given to the third input port, so the low-priority throughput of the third input port was 0.19. The MSDA switch thus achieved a max-min fair share for the low-priority class. We also verified the low-priority time-dependent performance, as shown in Fig. 10. The max-min fair share presented in Table 3 was maintained independently of time.

In case 4, to determine whether the locations of the high-priority and low-priority heavy-load inputs affect throughput, we moved the high-priority heavy-load traffic load from the fourth input port (case3) to the third one and the low-priority heavy-load traffic load from the third port (case 3) to the sixth one. The other input traffic conditions were the same as in case 3. Table 4 shows the input load conditions and results. We again did not observe anyinput-port dependency, comparing the results in Table 4 with those for case 3.

In case 5, each low-priority load was 0.07, as shown in Table 5. The high-priority input load was the same as incase 1. The total input load was 1.16 (0.6 + 0.56), which is again an overload condition. The low-priority throughput for all input ports was 0.05. Again, the low-priority time-dependent performance was verified, as shown in Fig. 11. These results indicate as well that the MSDA switch achieves a max-min fair share for the low-priority class even under heavily overloaded traffic conditions.

To show that the value of TH_l does not impact the performance of the low-priority throughput, we examined the TH_l dependency of the low-priority throughput quantitatively. To do this, we used the fairness index described by Jain [17], [18]. In general, if a scheme gives an allocation that is different from the optimal one, its unfairness is quantified as follows. Supposea scheme allocates $\{\tilde{x_1}, \tilde{x_2}, \ldots, \tilde{x_n}\}$ instead of the optimal allocation $\{\hat{x_1}, \hat{x_2}, \ldots, \hat{x_n}\}$. We calculate the normalized allocation $x_i = \tilde{x_i}/\hat{x_i}$ for each source and compute the fairness indexas follows.

$$Fairness = \frac{(\sum_i x_i)^2}{n \sum_i {x_i}^2}. \tag{2}$$

When the value of the index is 1.0, optimal fairness is achieved. In this case, the optimal allocation means the max-min allocation.

Using the above index, the results for all the cases are shown in Table 6. All the fairness indices reached 1.0 when TH_l was equal to or larger than the minimum value, which we assume to be 2. The reason is as follows. Due to the input-ID-replacement operation mentioned in Sect. 4, the low-priority cells were transmitted from the transit buffer in round-robin order in terms of their input ID and entered the lower transit virtual queues in the same round-robin order. Because at the lower crosspoint, alow-priority cell is again selected using ring arbitration, it is sufficient that the number of stored cells in the low-priority transit buffer is one or two. This consideration is satisfied in all the transit and crosspoint buffers. Thus, the value of TH_l does not impact the low-priority fairness for the above traffic conditions.

The examples described in this paper are such that all optimal allocations, $\hat{x_i}(i = \{1, \ldots, n\})$, are equal. When each optimal allocation $\hat{x_i}$ is different, the proposed MSDA schemecan meet this situation by employing a weighted-round-robin-based mechanism instead of the ring-arbiter-based mechanism in each crosspoint. The performance of the control scheme for different optimal allocations should be examined in further studies.

Table 6 TH_l independency of low-priority throughput.

TH_l	case 1	case 2	case 3	case 4	case 5
2	1.000	1.000	1.000	1.000	1.000
3	1.000	1.000	1.000	1.000	1.000
4	1.000	1.000	1.000	1.000	1.000
6	1.000	1.000	1.000	1.000	1.000
8	1.000	1.000	1.000	1.000	1.000
10	1.000	1.000	1.000	1.000	1.000

6. Conclusion

A multi-QoS scalable-distributed-arbitration (MSDA) ATM switch that supports both the high- and low-priority classes under the HOLP discipline has been described. It uses crosspoint and transit buffers, each consisting of a high-priority and low-priority buffer. Arbitration is executed between these two buffers in a distributed manner. The MSDA switch extends the advantage of our previously proposed single-QoS SDA (SSDA) switch. It is expandable while permitting high output-line speeds due to the distributed arbitration.

To solve the problem of the delay-based cell-selection mechanism used in the SSDA switch not ensuring fairness in terms of throughput for the low-priority class, we introduced a distributed-ring-arbiter-based cell-selection mechanism at each crosspoint. The low-priority transit buffer at each crosspoint has virtual queues, one for each upper inputport. This virtual-queue technique is combined with a newly introduced input-ID-replacement operation. Simulations confirmed that the MSDA switch ensures fairness in terms of delay time for the high-priority class and ensures fairness in terms of throughput for the low-priority class. Thus, the MSDA switch supports both service requirements while maintaining the scalability of the SSDA switch.

References

[1] K. Genda and N. Yamanaka, "TORUS-switch: Scalable Tb/s ATM switch architecture based on the internal speed-up ATM switch," Proc. IEEE GLOBECOM'95, pp.1738–1745, 1995.
[2] E. Munter, J. Parker, and P. Kirkby, "A high-capacity ATM switch based on advanced electronic and optical techniques," IEEE Commun. Mag., pp.64–71, 1995.
[3] A. Takase, T. Kozaki, M. Takatori, and H. Abe, "Datapath architecture and technology for large scale ATM switching systems," Proc. IEEE GLOBECOM'96, pp.1395–1399, 1996.
[4] H. Kuwahara, N. Endo, M. Ogino, and T. Kozaki, "A shared buffer memory switch for an ATM exchange," Proc. IEEE ICC'89, pp.118–122, 1989.
[5] H. Yamada, S. Yamada, H. Kai, and T. Takahashi, "A multi-purpose memory switch LSI for ATM-based systems," Proc. IEEE GLOBECOM'90, pp.1602–1607, 1990.
[6] N. Yamanaka, K. Endo, K. Genda, H. Fukuda, T. Kishimoto, and S. Sasaki, "320 Gb/s high-speed ATM switching system hardware technologies based on copper-polyimide MCM," IEEE Trans. CPMT'95, vol.18, pp.83–91, 1995.
[7] T. Kawamura, H. Ichino, M. Suzuki, K. Genda, and Y. Doi, "Over 20 Gbit/s throughput ATM crosspoint switch large scale integrated circuit using Si bipolar technology," Electron. Lett., vol.30, pp.854–855, 1994.
[8] H. Tomonaga, N. Matsuoka, Y. Kato, and Y. Watanabe, "High-speed switching module for a large capacity ATM switching system," Proc. IEEE GLOBECOM'92, pp.123–127, 1992.
[9] K. Genda, Y. Doi, K. Endo, T. Kawamura, and S. Sasaki, "A 160-Gb/s ATM switching system using an internal speed-up crossbar switch," Proc. IEEE GLOBECOM'94, pp.123–133, 1994.
[10] E. Oki and N. Yamanaka, "A high-speed ATM switch based on scalable distributed arbitration," IEICE Trans. Commun., vol. E80-B, no.9, pp.1372–1376, Sept. 1997.
[11] E. Oki, N. Yamanaka, and Y. Ohtomo, "A 10 Gb/s (1.25 Gb/s ×8) 4 × 2 CMOS/SIMOX ATM switch," Proc. IEEE ISSCC'99, pp.172–173, Feb. 1999.
[12] E. Oki, N. Yamanaka, and M. Nabeshima, "Scalable-distributed-arbitration ATM switch supporting multiple QoS classes," Proc. IEEE ATM Workshop'99, May 1999.
[13] A.Y. Lin and J.A. Silvester, "Priority queueing strategies and buffer allocation protocols for traffic control at an ATM integrated broadband switching system," IEEE J. Sel. Areas Commun., vol.9, no.9, pp.1524–1536, 1991.
[14] K.-Y. Siu, Y. Wu, and W. Ren, "Virtual queueing techniques for UBR+ service in ATM with fair access and minimum bandwidth guarantee," Proc. IEEE GLOBECOM'97, pp.1081–1085, 1997.
[15] H.-Y. Tzeng and K.-Y. Siu, "Performance of TCP over UBR in ATM EPD and virtual queuing techniques," Proc. Workshop on Transport Layer Protocols over High-Speed Networks, IEEE GLOBECOM'96, 1996.
[16] S. Keshav, An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network, Addison Wesley, 1998.
[17] R. Jain, The Art of Computer Systems Performance Analysis, Wiley, New York, 1991.
[18] R. Jain, "Congestion control and traffic management in ATM networks: Recent advances and a survey, " Computer Networks and ISDN Systems, vol.28, pp.1723–1738, 1996.

Eiji Oki received B.E. and M.E. degrees in instrumentation engineering and a Ph.D. degree in electrical engineering from Keio University, Yokohama, Japan, in 1991, 1993, and 1999, respectively. In 1993, he joined Nippon Telegraph and Telephone Corporation's (NTT's) Communication Switching Laboratories, TokyoJapan. He has been researching multimedia-communication network architectures based on ATM techniques and traffic-control methods for ATM networks. He is currently developing high-speed ATM switching systems in NTT Network Service Systems Laboratories as a Research Engineer. Dr. Oki received the Switching System Research Award and the Excellent Paper Award from the IEICE in 1998 and 1999, respectively. He is a member of the IEEE.

Naoaki Yamanaka was born in Sendai City, Miyagi Prefecture, Japan in 1958. He graduated from Keio University, Tokyo, Japan, where he received B.E., M.E. and Ph.D. degrees in engineeringin 1981, 1983 and 1991, respectively. In 1983 he joined Nippon Telegraph and Telephone Corporation's (NTT's) Communication Switching Laboratories, Tokyo, Japan, where he researched and developed high-speed switching systems and high-speed switching technologies, such as ultra-high-speed switching LSI chips/devices, packaging techniques, and interconnection techniques, for broadband ISDN services. Since 1989 he has been developing broadband ISDN things based on ATM techniques. He is now researching ATM-based broadband ISDN architectures and is engaged in traffic management and performance analysis of ATM networks. He is currently a senior research engineer, supervisor, research group leader in the Broadband Network System Laboratory at NTT. Dr. Yamanaka received the Best of Conference Award at the 40th, 44th and 48th IEEE Electronic Componentsand Technology Conferences, the TELECOM System Technology Prize from the Telecommunications Advancement Foundation, the IEEE CPMT Transactions Part B: Best Transactions Paper Award, and the Excellent Paper Award from the IEICE in 1990, 1994 , 1999, 1994, 1996, and 1999, respectively. Dr. Yamanaka is the Broadband Network Area Editor of the IEEE Communication Surveys, Editor of the IEICE Transactions, and the IEICE Communication Society International Affairs Director, as well asthe Secretary of the Asia Pacific Board of the IEEE Communications Society. Dr. Yamanaka is a senior member of the IEEE.

Masayoshi Nabeshima received B.E. and M.E. degrees from Waseda University, Tokyo, Japan, in 1992 and 1994, respectively. He joined Nippon Telegraph and Telephone Corporation (NTT), Tokyo, Japan,in 1994. He is currently researching high-speed ATM switching systemsand traffic management and scheduling mechanisms for ATM networks inthe NTT Network Service Systems Laboratories. He is a member of the IEEE Communication Society.