Proceedings of APCC2008 copyright © 2008 IEICE 08 SB 0083

An Efficient Buffered Crossbar Switching Architecture with Multiple Multicast Virtual Output Queues for Mixed Uni- and Multicast Traffic

Fong-Cheng Lee, Wen-Fong Wang, Ming-Da Liu
National Yunlin University of Science and Technology
{ g9213716, wwf, g9517720 }@yuntech.edu.tw

Abstract- **In this study, we purpose an efficient buffered crossbar switching architecture with multiple multicast virtual output queues called *k*MVOQ-CPX switch. The *k*MVOQ-CPX switch consists of N virtual output queues (VOQs), *k* multicast VOQs , an (N+*k*) x N buffered crossbar fabric , N+*k* Input scheduler and N output scheduler. It has the features of simple architecture and great capability to support multicast services. For achieving service quality of multicast applications, we also propose two practical scheduling scheme- MF and MFRR for output schedulers. To evaluate the performance of *k*MVOQ-CPX switch, we assume the *k* equals to one. We have done simulations under different proportions of unicast to multicast traffic. The simulation results show that the 1MVOQ-CPX switch can achieve dramatic service quality of multicast applications and has great capability to supports both unicast and multicast services.**

Keywords— Buffered crossbar fabric , Multicast switch.

I. INTRODUCTION

The explosive growth of traffic on the Internet coupled with the variety of services creates many challenges for the design and implementation of packet switches. To date, the increasing multicast applications such as teleconferencing, distance learning, IPTV etc., switches, which support multicasting play an important role on the Internet. The combined input and buffered crossbar (CICQ) switch is the most popular architecture because of its high throughput, low delay, and distributed scheduler. Although there have been many issues investigated for the CICQ switch, there are few discussions about multicast CICQ switches.

For multicast traffic, traditional unicast switches duplicate a multicast packet to multiple unicast packets before entering the switch system. This causes two problems. First, the size of memory is increasing. Second, the utilization of the bandwidth is decreasing. For solving the above problems, different multicast switch architectures were proposed. Most of switches adopted multicast FIFO queues at the ingress ports [1]. The multicast FIFO queues are used to accommodate arriving multicast cells before entering the switch. However, the HOL blocking problem limits the achievable throughput and degrades the performance of the switch. To solve the HOL problem completely, multicast VOQ was adopted [2]. For an N x N switch using the multicast VOQ architecture, each input must maintain up to 2^N-1 queues to match the number of fanout configurations. It becomes impractically as port number increasing. In [1], a plethora of algorithms had been proposed. However, none of these algorithms has been considered as an efficient solution because they are trouble in performance, fairness, and implementation complexity.

Mhamdi et al. first proposed using buffered crossbar switch with a single FIFO queue at each input port to handle multicast traffic [3]. They also proposed the MXRR algorithm and evaluated its performance. It was shown that this architecture can achieve better performance than the previous schemes. However, under heavy traffic load, it suffered the HOL blocking in the FIFO queues seriously. Shutao et al. enhanced and proposed a buffered crossbar with multiple queues at each input port to handling multicast traffic [4]. In [5] Mhamdi et al proposed a buffered crossbar switching architecture and integrating unicast and multicast scheduling schemes. At each input, there are N VOQs for unicast traffic and *k* ($1 \leqq k << 2^N-1$) FIFO queues for multicast traffic. Although this architecture performed well, the incoming multicast cells and unicast cells share the same crosspoint buffers. This has great effect on the real time multicast services.

To support different multicast services and provide service quality of multicast applications, in this study, we purpose an efficient buffered crossbar switch with multiple multicast virtual output queues called *k*MVOQ-CPX that has the following features: simple architecture, high performance, and great ability to support multicast services.

The rest of this paper is organized as follows. In Section II, we clearly describe the proposed *k*MVOQ-CPX switching architecture. In Section III, we propose and describe the scheduling schemes for input and output schedulers. The simulation results of *k*MVOQ-CPX are shown in Section IV. Finally, conclusions are made in Section V.

II. THE KMVOQ-CPX SWITCHING ARCHITECTURE

For describing the *k*MVOQ-CPX architecture, the following notations are defined.

- C(i,j): a cell comes from input i and destined for output j.
- MC(i, Φ): a multicast cell from input i and destined for fanout set Φ $\{\Phi \mid 2 \leq \Phi \leq N\}$. A fanout set Φ is a vector that records destination ports.

- VOQ(i,j): a buffer for buffering cells in input i, destined for output j.
- MVOQ(*k*,i): a multicast buffer for buffering multicast cells come from input i and assigned to *k*th MVOQ set. A MVOQ set, MVOQs, is composed of N multicast buffer, where N is the number of the switch size.
- CB(i,j): a cross-point buffer for buffering cells from input i and destined for output j.
- IS(i): a input scheduler for arbitrating whether cells can enter the crossbar fabric and be buffered at CB(i,j).
- OS(j): a output scheduler for arbitrating whether cells can be removed from the CB(i,j) and transferred to output port.

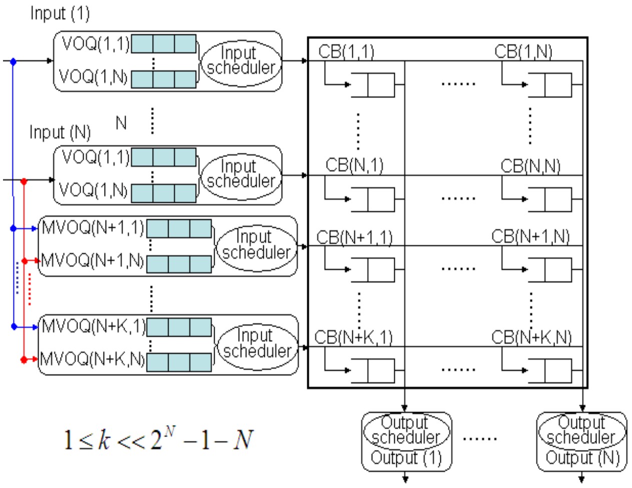

Figure 1. The General Architecture of kMVOQs-CPX

Fig. 1 shows the general architecture of *k*MVOQs-CPX switch with *k* MVOQs. To avoid the HOL blocking problem for unicast cells, each input port adopts a VOQ for buffering unicast cells C(i,j). There are extra queues, MVOQs, adopted for buffering incoming multicast cells. Each multicast cell has a fanout set Φ denoting the output destinations. When a multicast cell MC(i, Φ) arrivals, it will be assigned a number *k* and buffered at the MVOQ(*k*,i) without fanout processing. A (N+*k*) x N buffered crossbar fabric is adopted. For a cross-point buffer CB(i,j),$1 \leqq i \leqq N$,$1 \leqq j \leqq N$, it only buffers the unicast cells and for CB(i,j), $N+1 \leqq i \leqq k$,$1 \leqq j \leqq N$, it buffers the fanout cells which are the duplicate of MC(i, Φ). There are N+*k* IS and N OS arbitrating the cells entering or leaving the buffered crossbar fabric. If a MC(i, Φ) becomes the HOL cell of MVOQ(*k*,i) and selected by the IS(i), it transfers duplicated cells to corresponding CB(i,j) and redefines the fanout set Φ. If Φ becomes empty, the multicast cell is removed from HOL position of MVOQ(*k*,i). In the following, to evaluate the lowest performance of *k*MVOQs-CPX switch, we assume the MVOQ set is one.

The 1MVOQs-CPX switch is the simplest architecture of *k*MVOQs-CPX switch. There is no additional scheme for assigning a MVOQ set buffering the incoming multicast cells. In the next section, we will introduce the proposed scheduling scheme for 1MVOQs-CPX switch.

III. Scheduling Schemes For kMVOQ-CPX Switch

For an NxN 1MVOQ-CPX switch, there are (N+1) Input Scheduler IS(i) and N Output Schedulers OS(j). In this section, we will discuss the details of scheduling schemes for IS and OS.

A. Scheduling Schemes for IS

At each time slot, each IS will select a unicast or multicast cell entering the buffered crossbar fabric which the crosspoint buffer CB(i,j) has free memory space. To maximize the performance, keep fairness property and simplify hardware implementation, two scheduling schemes are investigated:

1. Round-Robin (RR)

   For each IS(i), $1 \leqq i \leqq N$, the scheduler selects and transfers a HOL C(i,j) from VOQ(i,j) to corresponding CB(i,j) according to the round-robin policy. For IS(N+1), the scheduler first selects a HOL MC(i, Φ) from 1st MVOQ set according to the round-robin policy and duplicates fanout cells to corresponding CB(N+1,j). There is a round-robin pointer in each scheduler recording the latest selected position. It is updated after every selection.

2. Shortest Crosspoint Buffer First (SCBF) [6]

   For each IS(i), $1 \leqq i \leqq N$, the scheduler selects a HOL C(i,j) which the occupancy of corresponding CB(i,j) is lowest from VOQ(i,j) and transfers the C(i,j) to CB(i,j). If there are more than two cells are eligible, the RR policy is applied. For IS(N+1), the scheduler first selects the cell which the occupancy of corresponding CB(i,j) is lowest. If there are more than two cells are eligible, the RR policy is applied. When a multicast cell is selected, the IS will duplicates the fanout cells to corresponding CB(i,j), which the CB(i,j) has sufficient space to buffer it.

B. Scheduling Schemes for OS

At each time slot, each OS(j) will select a HOL cell from the CB(i,j), $1 \leqq i \leqq N+1$, $1 \leqq j \leqq N$, and transfer to the output port. To support service quality for multicast applications, two scheduling schemes are investigated:

1. Multicast First Round-Robin (MF):

   For each OS(j), $1 \leqq j \leqq N$, the scheduler first selects the HOL cell from CB(N+1,j). If CB(N+1,j) is empty, the first non-empty CB(i,j), $1 \leqq i \leqq N$, $1 \leqq j \leqq N$, is selected according to the round-robin policy.

2. Multicast and Unicast Round-Robin (MURR)

   To employ the MURR, we divide crosspoint queues into two sets: unicast set and multicast set. The unicast set has CB(i,j), where $1 \leqq i \leqq N$,$1 \leqq j \leqq N$. Multicast set has CB(i,j), where $N+1 \leqq i \leqq N+k$, $1 \leqq j \leqq N$. Since we assume *k* equals to one, the multicast set only owns CB(N+1,j), where $1 \leqq j \leqq N$. Two stage round-robin are included in MURR. At the first stage, the multicast set and the unicast set are selected in the RR manner. If the selected set is empty, the other set is selected. At the second stage, the OS(j) will apply RR manner to select the first non-empty CB(i,j). Finally, the OS(j) will transfer the HOL cell from selected CB(i,j) to output port j.

IV. PERFORMANCE EVALUATION

In this section, we evaluate the performance of 1MVOQ-CPX under the different proportion of unicast to multicast traffic. In the simulations, the 8x8 switch is considered and the set of MVOQ equals to one. In the following figures, each IS act on the RR and SCBF schemes and each OS act on the MF and MURR schemes. For comparison, we also implement the switching architecture and the MURS_mix scheme purposed in [5]. The input arrival rate λ consists of λ_u and λ_m, where λ_u is the unicast arrival rate and λ_m is the multicast arrival rate. Only admissible traffic is considered, that is $\lambda = \lambda_u + \lambda_m \leqq 1$. In addition, the average fanout is 4 and the size of each CB(i,j) is 8. The buffer size of VOQs and MVOQs is infinite, thus there is no cell loss in the simulations. Each simulation runs over 1 million slots. We mainly evaluate the Average Cell Delay, Multicast Cell Delay and Multicast Jitter defined as following:

- Average Cell Delay: the time spent by a cell from the point it enters the switch to when it leaves.
- Multicast Cell Delay: the time spent by the last leaving fanout cell of a multicast cell MC(i, Φ) from the point it enters the switch to when it leaves.
- Multicast Jitter: the interval of cell delay between first leaving fanout cell and last leaving fanout cell duplicated from the same MC(i, Φ).

For multicast applications, the Multicast Cell Delay and Multicast Jitter can be considered as the capability of service quality. The lower Multicast Cell Delay and Multicast Jitter provide the better service quality for multicasting applications.

In Fig.2, we evaluate the throughput of 1MVOQ-CPX under different proportions of unicast arrival rate to multicast arrival rate λ_m. As shown in Fig. 2(a) and Fig. 2(b), when the λ_m is less than 30%, the throughput of 1MVOQ-CPX under all investigated schemes performs dramatically. Fig. 2(c) also shows that if the λ_m increases to 50%, the throughput of 1MVOQ-CPX switches with RR_MURR and SCBF_MUFF scheduling schemes is slightly decreasing. The reason is that, under heavy multicast arrival rate, there are lots of multicast cells arriving and buffered at the 1st MVOQ set. All multicast cells come from different inputs and share one line entering the cross-point buffer. If the output schedulers do not apply MF scheme, the occupancy of CB(N+1,j) will quickly full. The multicast HOL blocking will be happening serially in the 1st MVOQ set. Increasing the size of multicast CB(N+1,j) or adopting more MVOQ set can differently release the multicast HOL blocking in the 1st MVOQ set.

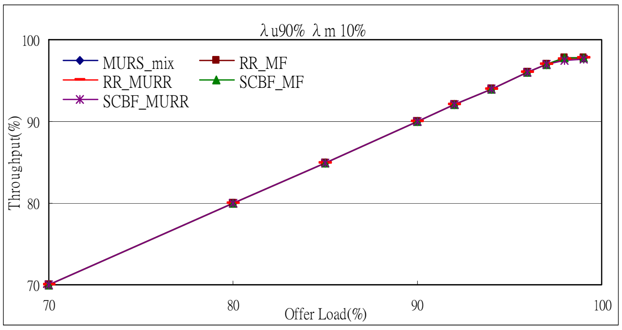

Figure 2(a) Throughput (λu90%, λm10%)

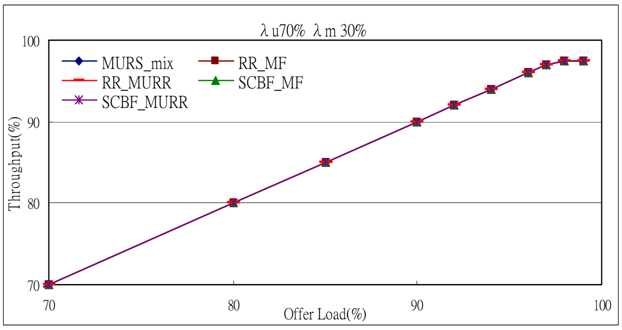

Figure 2(b) Throughput (λu70%, λm30%)

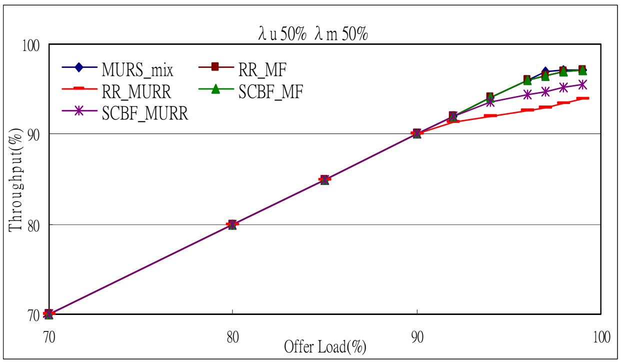

Figure 2(c) Throughput (λu50%, λm50%)

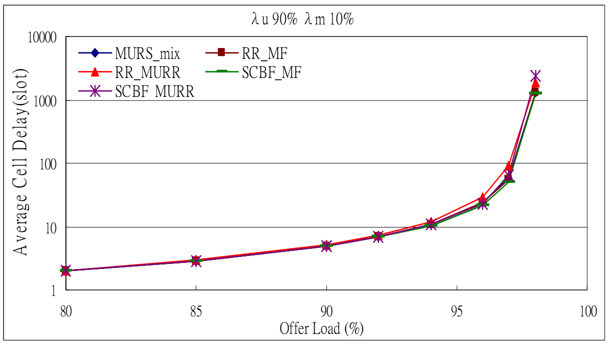

Figure 3(a) Average Cell Delay (λu90%, λm10%)

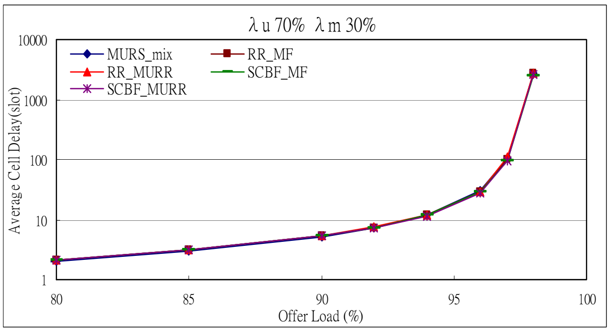

Figure 3(b) Average Cell Delay (λu70%, λm30%)

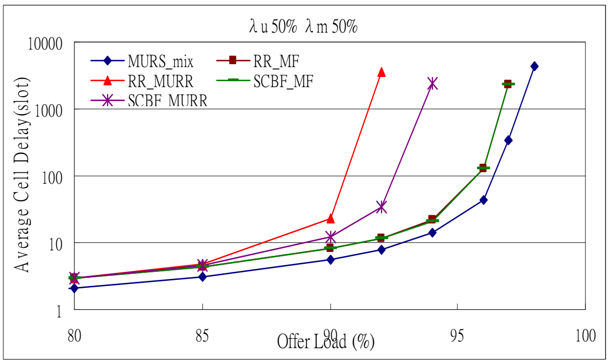

Figure 3(c) Average Cell Delay (λu50%, λm50%)

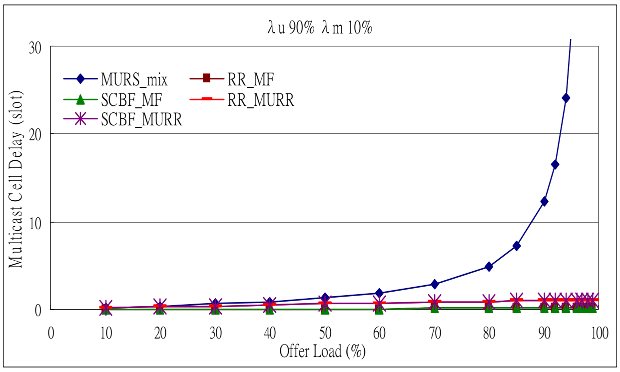

Figure 4(a) Multicast Delay (λu90%, λm10%)

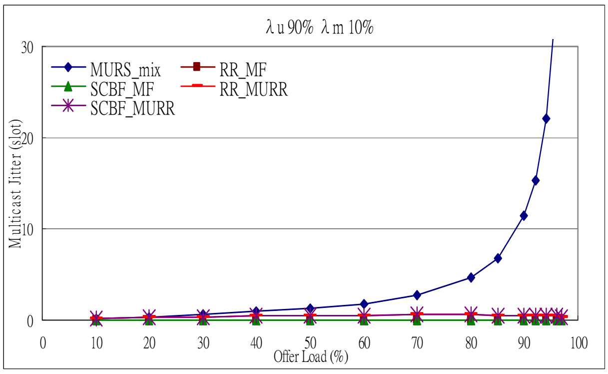

Figure 5(a) Multicast Jitter (λu90%, λm10%)

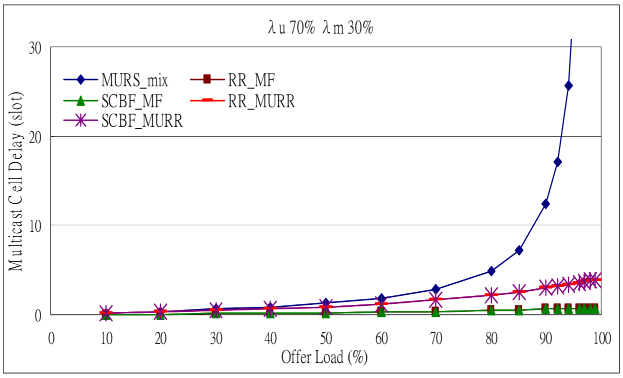

Figure 4(b) Multicast Delay (λu70%, λm30%)

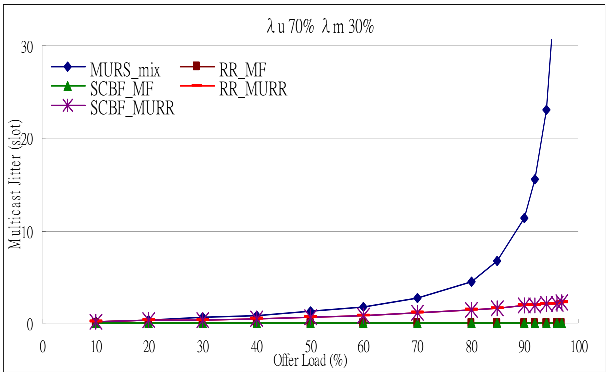

Figure 5(b) Multicast Jitter (λu70%, λm30%)

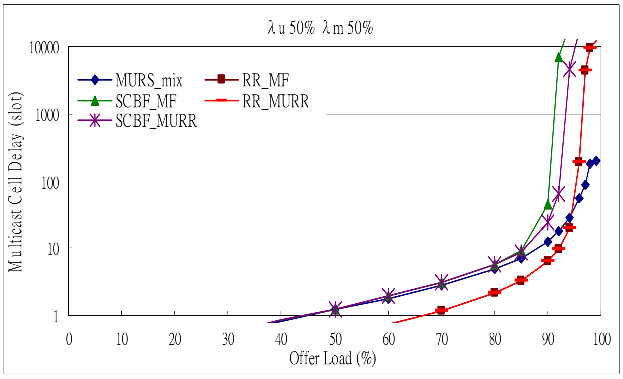

Figure 4(c) Multicast Delay (λu50%, λm50%)

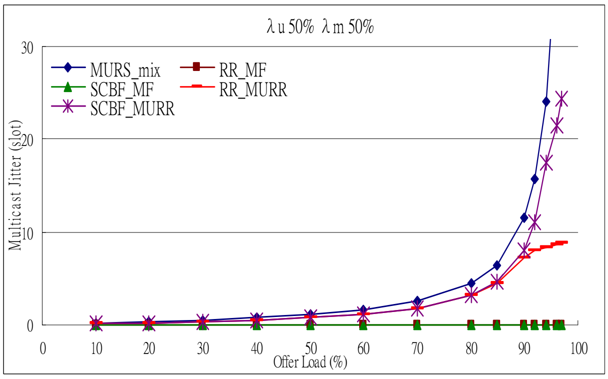

Figure 5(c) Multicast Jitter (λu50%, λm50%)

Fig. 3(a)(b)(c) show the delay performance of 1MVOQ-CPX switch under different proportions of unicast arrival rate to multicast arrival rate. When the λ_m under 30%, all of investigated schemes perform similarly. However, as the λ_m increases to 50%, the 1MVOQ-CPX switch suffers the same problem mentioned above. Increasing the size of crosspoint buffer CB(N+1,j) or adopting more MVOQ set can solve this problem and improve delay performance.

Fig. 4(a)(b)(c) show the Multicast Delay of 1MVOQ-CPX switch under different proportions of unicast arrival rate to multicast arrival rate. Comprising to the switching architecture and MURS_mix scheme proposed in [5], as the λ_m less than 30%, the 1MVOQ-CPX has the feature of low Multicast Delay. It means the 1MVOQ-CPX has excellent capacity of serving multicast applications. As shown in Fig. 4(b), when the λ_m increases to 30%, the MF scheme can provides better multicast delay than MURR. However, when the λ_m increases to 50%, both of the MF scheme and MURR scheme have poorer performance than MURS_mix.

Fig. 5(a)(b)(c) show the Multicast Jitter of 1MVOQ-CPX switch under different proportions of unicast arrival rate to multicast arrival rate. The 1MVOQ-CPX switch can provide very low jitter when the λ_m less than 30%.

We point out interesting observations:

- When the λ_m less than 30%, the performance of throughput, delay, multicast delay, and multicast jitter of the 1MVOQ-CPX switch is dramatically.
- When the λ_m increases to 50%, the performance of 1MVOQ-CPX switch is influenced by the multicast HOL blocking happened in 1st MVOQ set. To solve the multicast HOL blocking problem, two intuitive solutions can be applied. The first solution is increasing the size of cross-pointer buffers. The improved performance depends on how large of the increasing cross-pointer buffer size. The better solution is adopting more MVOQ set. Adopting multiple MVOQ sets can efficient release the multicast blocking problem happened 1MVOQ-CPX switch.
- Under heavy multicast arrival rate, select the MF scheme for output scheduler can release the

influence of multicast HOL blocking. It will provide better Multicast Cell Delay and low Multicast Jitter. Selecting the MURR scheme for output scheduler fairly select unicast and multicast traffic. However, for the 1MVOQ-CPX switch, the MURR scheme does not perform well as the λ_m increases to 50%.

- Under the high multicast arrival rate, the 1MVOQ-CPX switch suffers the multicast HOL blocking in 1st MVOQ set. To provide better performance under high multicast arrival rate, we have to release the multicast HOL blocking problem efficiently. That is to say a *k*MVOQ-CPX switch is needed, where $k > 1$.

V. CONCLUSIONS

In this study, we purpose an efficient buffered crossbar switching architecture with multiple multicast virtual output queues called kMVOQ-CPX switch. The kMVOQ-CPX switch consists of N virtual output queues (VOQs), k multicast VOQs, an (N+k) x N buffered crossbar fabric, N+k Input scheduler and N output scheduler. In this study, we assume the k equals to one.We also propose two practical scheduling scheme- MF and MFRR for output schedulers. To evaluate the performance of 1MVOQ-CPX switch, we have done simulations and compare to the architecture proposed in [5]. The simulation results show that when the multicast arrival rate less than 30%, the performance of throughput, cell delay, multicast delay, and multicast jitter of the 1MVOQ-CPX switch is dramatically. It has better capacity of service quality for both unicast and multicasting applications. When the multicast arrival rate increases to 50%, multicast HOL blocking happened in 1MVOQ-CPX seriously. To provide better performance under high multicast rate, a kMVOQ-CPX switch is needed, where k > 1.

REFERENCES

[1] B. Prabhakar, N. McKeown, and R. Ahuja. “Multicast scheduling for input-queued switches”,IEEE J. Select. Areas Commun., vol. 15, pp.855-866, June, 1997.

[2] M. Ajmone Marsan, A. Bianco, P.Giaccone, E. Lsonardi, and F. Nen. “Multicast traffic in input-queued switches: optimal scheduling and maximum throughput,” ACM Trans. vol. 11, no.3, June 2003, pp. 465-477.

[3] Lotfi Mhamdi and Mounir Hamdi “Scheduling Multicast Traffic in Internally Buffered Crossbar Switches,” IEEE ICC, pp. 1103–1107, June 2004.

[4] Shutao Sun, Simin He, Yanfeng Zheng, Wen Gaol. “Multicast scheduling in buffered crossbar switches with multiple input queues,” IEEE HPSR 2005, pp. 73-77, May 2005.

[5] Mhamdi L., Vassiliadis S... “Integrating uni- and multicast scheduling in buffered crossbar switches,” Workshop on High Performance Switching and Routing, 2006 . pp.6, June 2006.

[6] Xiao Zhang, Laxmi N. Bhuyan,“An Efficient Scheduling Algorithm for Combined Input-Crosspoint-Queued (CICQ) Switches”, in IEEE Globecom 2004, pp1168-1173, Nov. 2004.

[7] Mark J, Karol et al., “Input Versus Output Queueing on a Space-Division Packet Switch”, IEEE Transactions on communications, Vol.35, No.12,pp.1347-1356, December, 1987.

[8] Nick McKeown and Thomas E. Anderson, “A Quantitative Comparison of Scheduling Algorithms for Input-Queued Switches”, Computer Networks and ISDN Systems, Vol.30, No.24, pp.2309-2326, December, 1998.