고성능 입력큐 스위치를 위한 버퍼관리기의 설계

정갑중* · 이범철**

*경주대학교 · **한국전자통신연구원

Design of High Performance Buffer Manager for an Input-Queued Switch

Gab Joong Jeong* · Bhum-Cheol Lee**

*Kyongju University, **Electronics and Telecommunications Research Institute

E-mail : gjjeong@kyongju.ac.kr

요 약

본 논문은 고성능 입력큐 스위치 패브릭을 위한 입력버퍼 관리기의 설계 및 구현에 관한 연구이다. 본 논문에서 설계된 버퍼관리기는 멀티기가비트 크로스바 스위치의 입력 및 출력 포트에 연결되어 하나의 스위치 패브릭으로 구성된다. 본 버퍼관리기는 입력 및 출력포트의 와이어 속도로 셀 및 패킷의 라우팅을 지원하며 중앙중재기와 정보전송에 있어서 중재요청신호 및 출력허가신호의 파이프라인 전송지연을 수용하는 구조로 설계 되었다. FPGA 칩을 이용하여 구현된 버퍼관리기는 포트당 2.5Gbps의 OC-48c 속도를 지원하며 외부 입력 및 출력 형식으로 CSIX 인터페이스를 지원한다.

ABSTRACT

In this paper, we describe the implementation of high performance buffer manager that is used in an advanced input-queued switch fabric. The designed buffer manager provides wire-speed cell/packet routing with low cost and tolerates the transmission pipeline latency of request and grant data. The buffer manager is implemented in a FPGA chip and supports the speed of OC-48c, 2.5Gbps per port.

Keywords

ATM, Buffer Manager, Pipeline, Switch Fabric, Wire-speed Routing

I. Introduction

Input-queued packet switches have been extensively studied in recent years to resolve the output contention problem that occurs when multiple packets destined for the same output arrive simultaneously [1-3]. The loser of an output contention has to wait for the next arbitration in an input buffer. The input buffer needs to communicate request and grant data with an arbiter, which determines the winners for the next packet transmission. The transmission time of request and grant data is more serious when the switch system uses a high-speed serial transmission technology as point-to-point communication line between an input buffer and the central arbiter for the implementation of a large distributed switching system.

In this paper, we introduce the design and implementation of a high performance buffer manager for ingress and egress buffer of an advanced input-queued ATM switch, which has transmission latency of arbitration information and a pipelined approach to improve the throughput [4]. The proposed buffer manager as an input buffer of an input-queued ATM switch provides wire-speed pipelined queue management that has dynamically allocated virtual output queues [5]. The dynamic

allocation of queues minimizes the amount of packet memory in an input buffer. The designed buffer manager adopts a novel method to manage the transmission latency between input buffers and a central arbiter, which is based on request shifting. It has been implemented in FPGAs.

II. Proposed Buffer Manager

The buffer manager in an advanced input-queued ATM switch controls VOQs at each input port of the switch and output queues for traffic management. Figure 1 illustrates the pipelined buffer manager for ingress and egress buffers. We explain mainly the architecture of the buffer manager for the input queue management. The operations are divided into incoming cell writing, outgoing cell reading, policing, and request control for arbitration latency. Each operation has to be connected in a pipelined manner for wire-speed routing with low cost. The input queue manager consists of a VOQ and idle queue (IDQ) modules, write pointer manager (WPM), read pointer manager (RPM), and request first-in-first-out controller (RFC).

The RFC has a request first-in-first-out (FIFO) register for each VOQ and communicates with a central arbiter. It controls request generation and deletion according to the VOQ and request FIFO status. The RFC generates a valid request signal for each VOQ that has one or more queued cells, when the first element of the request FIFO register of the VOQ is not occupied with a valid request. It then stores the request signal in the request FIFO register after shifting existing data in the request FIFO, and decreases the VOQ length by 1. The VOQ length in the RFC is smaller than the VOQ length in the policing module (PM) by the number of valid request signals in the request FIFO. If the VOQ length in the RFC is zero and the first element of the request FIFO of the VOQ is not occupied by a valid request, the RFC generates an invalid request signal and just shifts the existing data in the request FIFO. When a granted output port number arrives at the RFC from the central arbiter, the RFC deletes the oldest request signal in the request FIFO of the granted output port, and it propagates the granted output port number to the outgoing cell reader (OCR).

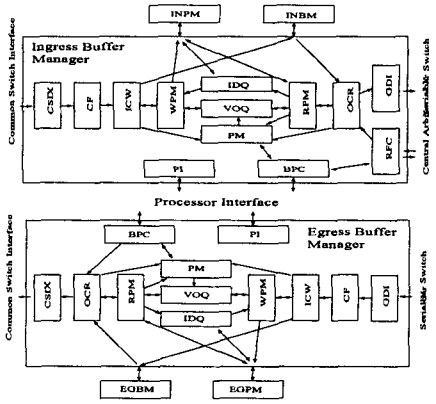

Fig. 1 Pipelined ingress/egress buffer manager.

The OCR propagates the granted output port number to the RPM and PM, and reads an outgoing cell from the ingress buffer memory (INBM) using an output cell address that comes from the RPM. The RPM updates the currently selected port queue of the VOQ module with the next address extracted from the next cell pointer information in the ingress pointer memory (INPM). The RPM stores the current outgoing cell address in the next destination port queue of multicast leaves if the outgoing cell is a multicast cell. A new arriving cell is stored in the INBM by the incoming cell writer (ICW) after receiving a new cell writing address from the WPM. The WPM updates the destination output port queue of the current incoming cell in the VOQ module with the new cell writing address. The RFC aggregates the new cell arriving and multicast cell stitching information from the PM through the backpressure controller (BPC).

The RPM stores the current outgoing cell address in the next destination port queue of multicast leaves if the outgoing cell is a multicast cell. A new arriving cell is stored in the INBM by the incoming cell writer (ICW) after receiving a new cell writing address from the WPM. The WPM updates the destination output port queue of the current incoming cell in the VOQ module with the new cell writing address. The RFC aggregates the new cell arriving and multicast cell stitching information from the PM through the backpressure controller (BPC). Figure 2 shows entire data paths to manage all VOQs and an idle queue that fully share the INPM, a single dual-port synchronous SRAM, including the multicast cell address stitching.

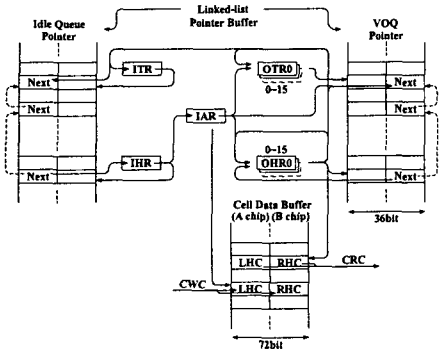

Fig. 2 Entire data path for the pointer movement with output leaf multicasting.

III. Implementation

Every small block of the buffer manager is pipelined by pipeline register and the operation is independent of each adjacent block. We illustrate the micro-operation of the read pointer manager (RPM) to show how to operate the blocks each other. The RPM block eliminates a head cell address in the VOQ selected by the central arbiter for each cell slot and adds the read cell address in the idle queue or it attaches the read cell address in the next destination output queue if the outgoing cell is a multicast cell. The operation steps of the RPM are shown below.

The first cell slot:
Step a) takes a granted output port identification from the OCR.
Step b) gives the ID of step (a) to VOQ block.
Step c) takes a head cell address from the VOQ block.
Step d) gives the cell address of step (c) to the OCR and INPM.
Step e) takes the next cell address and multicast bitmap from the INPM.
Step f) gives the next cell address of step (e) to the VOQ block and the cell address of step (c) to the IDQ if it is not a multicast cell.

The second cell slot:
Step g) gives the next destination output leaf ID to the PM if the read cell in step (f) is a multicast cell.
Step h) gives the next destination output leaf ID and the cell address in step (c) to VOQ block.
Step i) takes the tail address of the queued cell in the selected destination in step (h) from the VOQ block.
Step j) gives the tail address in step (i) and the cell address in step (c) to INPM for queuing the next multicast output cell.
Step k) gives the cell address in step (c) and the updated multicast bitmap of the cell to INPM for eliminating current destination output port in its output leaf.

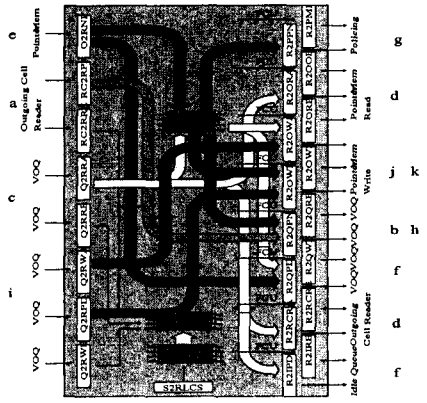

Fig. 3 Pipelined process steps in the RPM block.

Since the stepwise operation of the RPM described above occurs at every cell slot, the reading of a current outgoing cell through the first 6 steps is processed with the writing of a multicast cell in the next output leaf of the previous output cell through the last 5 steps in a single cell slot. Every stepwise operation of all blocks in the buffer manager is scheduled its own pipeline clock to evade any collision of pipelined processes. Figure 3 illustrates the RPM block and the pipelined process steps. Main pipeline stages consist of four stages for processing and one stage for synchronizing, and each pipeline stage has two clock cycles for 72-bit pointer data processing using the data bus width of 36-bit between the buffer manager and the external pointer memory chip. Figure 4 shows the timing diagram of the INPM access and the four pipeline stages for data processing. The arrows in the timing diagram show the concatenation operation.

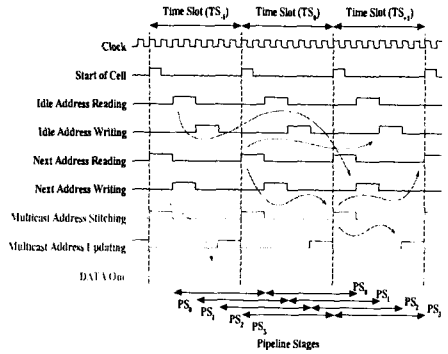

Fig. 4 Pointer memory r/w timing and the movement of data.

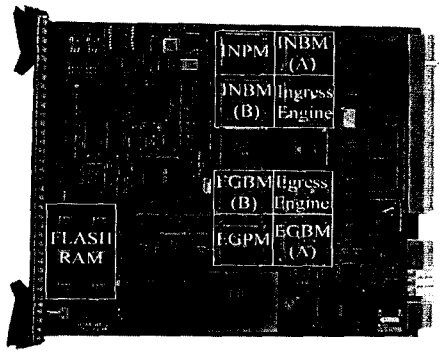

Fig. 5 Photograph of the designed ingress and egress buffer board.

The proposed buffer manager has been designed and implemented in a field programmable gate array (FPGA). The designed buffer manager has been verified in a core ATM switch system that has a multi-gigabit serial crossbar structure, a 62.5MHz operating frequency, and OC-48c speed per port. The designed buffer manager has been used as an egress cell buffer manager as well as an ingress cell buffer manager. It supports common switch interface (CSIX) format to communicate with an ingress and egress port processor. Figure 5 shows the photograph of the ingress and egress buffer board using the proposed buffer manager, which is implemented in a high performance backbone switch of the Internet.

IV. Conclusions

In this paper, we proposed and implemented a new architecture of the VOQ buffer manager for an advanced input-queued ATM switch. The proposed buffer manager provides wire-speed routing at low cost using a pipelined buffer management. In the design of the proposed VOQ manager, we used a novel request shifting method to manage the transmission latency of request and grant data between input buffers and central arbiter. The wire-speed pipelined VOQ management has been implemented in a FPGA and has been tested for all functions in an internet backbone switch system which has 16x16 switch size, 2.5Gbit/s per port speed, and 40Gbit/s aggregated switching capacity.

References

[1] R. O. LaMaire and D. N. Serpanos, "Two-dimensional round-robin schedulers for packet switches with multiple input queues," IEEE/ACM Trans. Networking, vol. 2, no. 5, pp. 471-482, 1994.

[2] N. McKeown, "The iSLIP scheduling algorithm for input-queued switches," IEEE/ACM Trans. Networking, vol. 7, no. 2, pp. 188-201, 1999.

[3] P. Gupta and N. McKeown, "Designing and implementing a fast crossbar scheduler," IEEE Micro, vol. 19, no. 1, pp. 20-28, 1999.

[4] G. J. Jeong, J. H. Lee, and B. C. Lee, An advanced input-queued ATM switch with a pipelined approach to arbitration, in Proc. IEEE GLOBECOM, pp.496-499, Nov. 2000.

[5] G. J. Jeong, J. H. Lee, and B. C. Lee, "Design of pipelined routing engine for input-queued ATM switches," Electron. Lett., vol. 37, no. 2, pp. 137-138, Jan. 2001.