초고속 포인터 스위칭 패브릭의 설계☆

Design of High-speed Pointer Switching Fabric

류 경 숙* 최 병 석**
Kyoung-Sook Ryu Byeong-Seog Choe

요 약

본 논문은 데이터 메모리 평면과 스위칭 평면을 분리하여 패킷 데이터의 저장과 메모리 주소 포인터의 스위칭이 병렬적으로 처리 가능하며 IP 패킷의 가변 길이 스위칭이 가능한 새로운 스위치 구조를 제안 한다. 제안한 구조는 기존 VOQ방식의 복잡한 중재 알고리즘이 필요 없으며 출력 큐 방식의 스위치에서만 적용되고 있는 QoS를 입력 큐에서 고려한다. 성능 분석 결과 제안한 구조는 기존의 공유 메모리 기반의 구조들에 비해 상대적으로 낮은 평균 지연 시간을 가지며 스위치의 크기가 증가하더라도 일정한 지연 시간을 보장함을 확인하였다.

Abstract

The proposed switch which has separated data plane and switching plane can make parallel processing for packet data storing, memory address pointer switching and simultaneously can be capable of switching the variable length for IP packets. The proposed architecture does not require the complicated arbitration algorithms in VOQ, also is designed for QoS of generic output queue switch as well as input queue. At the result of simulations, the proposed architecture has less average packet delay than the one of the memory-sharing based architecture and guarantees keeping a certain average packet delay in increasing switch size.

☞ keywords: 초고속 스위치(high-speed switch), 스위칭 패브릭(switching fabric), 입력 큐잉(Input queueing), 출력 큐잉(Output queueing)

1. 서 론

인터넷 망에서 증가하는 트래픽 부하와 함께 복잡하고 다양해지는 사용자 트래픽 요구 수준을 수용하기 위한 다양한 하드웨어 및 소프트웨어 기술들이 제시되고 있다. 이 중에서 고속 스위치에서 하드웨어 구현이 간단한 입력 버퍼 형태의 VOQ(Virtual Output Queued) 방식들도 제시되고 있다[3][4][13]. 특히 기가 비트(Gigabit) 이더넷의 발전으로 전송 부분에 있어서 대역폭의 향상이 두드러지고 있으며 교환 장비로써 고성능의 스위치가 출시되고 있다.

최근 인터넷 환경에서의 초고속 스위치의 발전 방향은 기가 비트 스위칭이 가능하도록 하고자 스위치 내부의 입/출력 인터페이스간 패킷 전송에 있어서 ATM 스위치와 같은 고속의 하드웨어 스위칭 기법을 도입하고 있다[1][2]. Abacus와 같은 ATM 스위치는 53 옥텟(Octet)의 고정 셀 스위칭을 하는데 반하여 인터넷 망에서는 가변 길이 패킷을 서비스 해야 하는 차이점이 있다. 일반적으로 이더넷 프레임은 64 바이트에서 1518 바이트의 길이를 가진다. 스위칭 패브릭(switching fabric)을 사용하는 스위치에서 이러한 가변 길이 패킷을 효율적으로 스위칭 하기 위해서는 새로운 형태의 스위치 구조가 필요하다.

현재까지 스위치에 스위칭 패브릭을 적용하여

* 정 회 원 : 동국대학교 정보통신공학과 박사과정
ksryu@dgu.edu
** 정 회 원 : 동국대학교 정보통신공학과 교수
bchoe@dgu.edu
[2007/08/24 투고 - 2007/08/27 심사 - 2007/10/02 심사완료]
☆ 이 논문은 2005년도 동국대학교 연구년 지원에 의하여 이루어졌음.

가변 길이 IP 패킷을 서비스 하기 위한 두 가지 접근 방법이 있다. 하나는 패킷들이 입력 단에서 스위칭 패브릭에 진입하기 전에 패킷들을 일정한 셀 단위로 분할하여 전송하고 출력 단에서 재조립하는 방법이며 다른 하나는 가변 길이 패킷을 별도의 가공 없이 서비스 할 수 있도록 스위칭 패브릭 내에 가변 길이 패킷을 저장하기 위한 저장 공간(crosspoint buffer)을 확보하는 방법이다 [5][7][10][14]. 전자는 패킷을 분할하고 재조립 하기 위한 시간들이 시스템에 부하로 작용할 수 있고 후자는 단편화(fragmentation) 문제는 해결할 수 있지만 스위칭 패브릭 내의 모든 크로스포인트에 버퍼를 두어 가변길이 패킷을 저장하게 되므로 스위칭 패브릭의 속도 면에서 역시 문제가 된다.

최근 고속 스위치에서 하드웨어적 구현이 용이한 입력 버퍼 형태의 다양한 VOQ 방식들이 제시되고 있으나 공유 메모리 기반의 스위치 구조로 인해 복잡한 중재 알고리즘들과 스케쥴링 기법들을 필요로 하는 단점이 있다. 본 논문에서는 가변 길이 패킷 스위칭을 효율적으로 하기 위한 새로운 스위치 구조를 제안 한다. 제안한 구조는 독립적 메모리를 사용하도록 하여 기존의 복잡한 중재 알고리즘(arbitration algorithm)을 필요로 하지 않으며 현재 출력 큐 스위치에서만 적용되고 있는 QoS를 입력 큐 방식에 제공하는 부분을 고려한다.

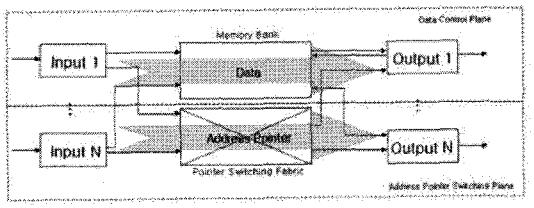
본 논문의 구성은 다음과 같다. 2 장에서는 제안한 스위치의 전체 구조와 내부 패킷 형식 및 패킷 저장 방식 등에 대해서 설명하고 3 장에서는 출력 포트에서의 동작을 자세히 설명한다. 4 장에서는 제안한 구조를 위한 시뮬레이션 환경과 시뮬레이션 결과를 제시하고 5장에서 결론을 맺는다.

2. 제안한 구조

제안한 스위치는 데이터 메모리 평면(data memory plane)과 스위칭 평면(switching plane)을 분리하여 패킷 데이터의 저장 영역인 메모리 뱅크(memory bank) 부분에서의 작업과 패킷의 메모리 뱅크 내 메모리 주소 포인터(memory address pointer)의 스위칭 작업들을 병렬적으로 처리할 수 있도록 하는 새로운 구조를 가진다. 메모리 뱅크를 입/출력 포트 각각에서 독립적으로 메모리들을 관리하도록 구성하여 IP 패킷에 대한 가변 길이 스위칭과 고속의 포워딩이 모두 가능하다. 또한 입력 포트 부분에서 패킷의 메모리 뱅크 내 메모리 주소 포인터와 포워딩을 위한 기본적인 정보들로 구성된 미니 패킷(mini packet)들만 통과하도록 허용한다.

2.1 주소 포인터 스위치 구조

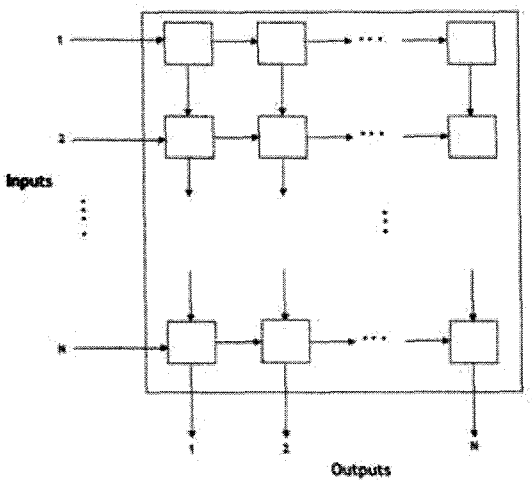
제안한 스위치는 크게 두 개의 평면(plane)으로 구성된다. 하나는 데이터 제어 평면(data control plane) 부분이고 다른 하나는 주소 포인터 스위칭 평면(address pointer switching plane)이다.



(그림 1) 제안한 스위치의 구성도

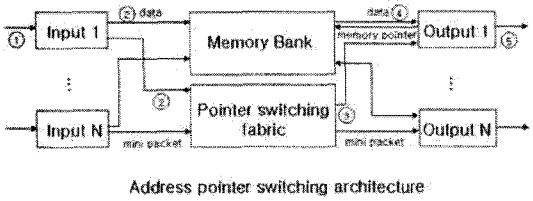
(그림 1)과 같이 입력 인터페이스에 입력된 패킷을 헤더 정보와 분리하여 데이터 부분은 메모리에 저장하며 동시에 저장된 데이터의 주소 포인터 부분만을 스위칭 패브릭을 사용해서 스위칭 하도록 한다. 스위칭 패브릭은 (그림 2)와 같이 구현이 간단하면서도 확장성이 우수한 ATM 스위치의 스위칭 패브릭을 사용한다. 데이터는 입력 포트와 출력 포트 번호로 참조되는 독립적인 메모리들의 집합인 메모리 뱅크에 저장한다. 각 라인 카드에 라우팅 테이블 과 포워딩 엔진을 두어 라우팅 기능을 각 라인 카드에 분산하여 병렬적

인 처리가 가능하도록 하고 있다. 성능 향상을 위해 출력 큐는 ASIC 형태의 시퀀서(sequencer)로 구현하여 스케줄링 알고리즘의 복잡성을 회피하고자 한다[12].



(그림 2) N x N 크로스바 스위치의 스위칭 패브릭

(그림 2)는 제안한 구조에서 사용할 N개의 입력 포트와 N개의 출력 포트에 대해 임의의 입력이임의의 출력으로 연결되는 스위칭 패브릭의 구조이다. 기존의 스위치 구조들에서는 패킷 데이터 전체가 스위칭 패브릭을 통과하도록 하였으나 제안한 구조에서는 패킷의 메모리뱅크 내 주소 포인터 값을 가지는 미니 패킷들만이 스위칭 패브릭을 통과하도록 설계하였다.



(그림 3) 제안한 스위치의 동작

(그림 3)의 제안한 스위치의 동작을 간략하게 살펴보면 다음과 같다.

① 스위치의 입력인터페이스에서 패킷이 들어오면 라우팅 헤더 부분을 분리하고 라우팅 테이블을 룩업(lookup)한다.

② 헤더와 룩업 정보를 바탕으로 제안한 스위치 구조에서 QoS와 스위칭의 목적으로 사용하게 될 미니 패킷을 작성하여 주소 스위칭 패브릭으로 보냄과 동시에 메모리에 데이터를 저장한다.

③ 출력 인터페이스에 도착한 미니 패킷은 출력 시퀀서에 의해서 입력 포트에서 지정한 우선순위에 따라 시퀀서의 오른쪽부터 전송 순서대로 정렬(sorting) 된다.

④ 해당 미니 패킷이 전송될 순서(HOL: Head Of Line)가 되면 메모리 포인터 주소를참조해서 메모리 뱅크에서 데이터 부분을 읽어 오고 참조한 메모리 인덱스 부분은 해제(free) 된다.

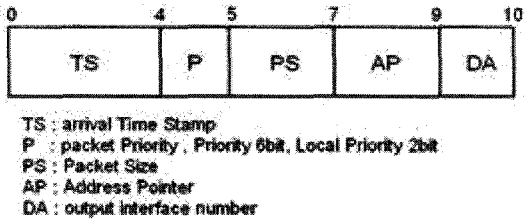
⑤ 메모리 뱅크에서 읽어온 데이터와 라우팅 헤더를 결합하여 다음 홉(hop)으로 전송한 다.

스위치에 진입한 모든 패킷은 입력 단에서 라우팅 테이블 룩업을 위해서 헤더 정보를 읽히게 되는데 이 과정에서 필요한 계산들을 미리 하게 되면 효과적이기 때문에 제안한 스위치구조에서는 QoS 처리를 위한 스케줄링과 출력 큐에서의 정렬을 위한 정보는 입력 포트에서 처리하도록 한다. 출력 포트에서 조절할 경우 보다 정확하고 안정적일 수 있지만 헤더 정보를 한 번 더 읽어야 하고 고속 스위치에서 QoS를 복잡하게 계산하는 것은 지연을 초래할 수 있다. 입력 포트에서 계산된 QoS 정보는 미니 패킷에 저장되어 출력 큐로 전달되며 출력 큐인 시퀀서는하드웨어로 구현되어 있기 때문에 전달 받은 QoS 정보에 따라 자동적으로 정렬이 가능하다.

여기서 미니 패킷의 구성은입력 인터페이스에서 라우팅 테이블 룩업과 함께 처리가 되며 기존 ATM 스위치에서도 스위칭 패브릭 진입 전에 일종의 미니헤더를 구성하고 있으므로 추가적인 지연시간을 요하지 않는다.

2.2 미니 패킷의 구조

제안한 구조의 내부에서 사용하는 미니 패킷은 (그림 4)와 같이 구성된다.

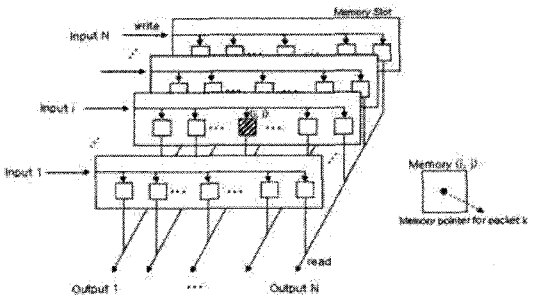


(그림 4) 미니 패킷의 구조

입력 포트로 유입된 패킷은 IPC(Input Port Controller)에서 내부 전용 패킷인 미니 패킷을 생성하여 제안한 포인터 스위칭패브릭(PSF, Pointer Switching Fabric)을 통해서 해당 전송 큐로 전송된다. 미니 패킷은 도착 시각을 나타내는 타임 스탬프(TS: arrival Time Stamp) 4 바이트, 우선 순위(Priority) 6 비트와 로컬 우선 순위(Local Priority) 2 비트를 포함하는 패킷 우선 순위(P: Packet Priority) 1 바이트, 패킷의 크기(PS: Packet Size) 2 바이트, 메모리 뱅크 내의 주소 포인터(AP: Address Pointer) 2 바이트, 출력 포트 주소(DA: output interface number) 1 바이트로 구성된다. 여기서 우선 순위(P) 6 비트는 노드에서의 클래스별 가중치를 적용한다. 또한 출력 포트에서 피드백되는 정보에 의해서 우선 순위가 재조정 되는 경우 우선 순위 재조정을 위해서 로컬 우선 순위(LP) 2 비트가 사용되며 drop된 횟수를 고려하여 우선 순위를 높여 주는 역할을 한다. 로컬 우선 순위는 00, 01, 10, 11의 값을 가지며 값이 높을수록우선 순위가 낮으며 기본 값은 11 이다.

2.3 메모리 뱅크의 구조

패킷의 데이터 부분을저장하기 위해 제안한 메모리 뱅크(Memory Bank)의 구조는 (그림 5)와 같다. 각 입력 포트에 연결된 독립적인 메모리 슬롯들은 출력 포트의 개수만큼의 메모리들이 순서대로 버스로 연결되어 있으며 동일한출력 포트에 속하는 메모리들은 다시 해당 출력 포트로 향하는 버스로 연결되어 있다.

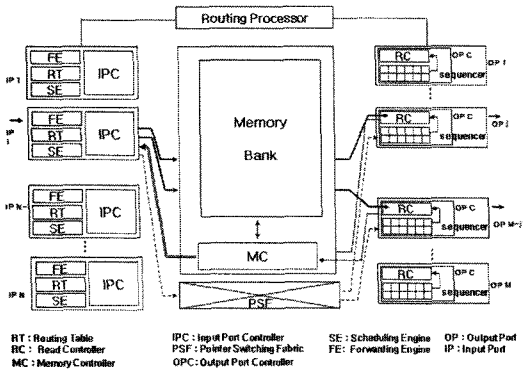


(그림 5) 메모리 뱅크 구조

메모리 뱅크의 접근에 있어서 메모리에 쓸 경우에는 입력 포트 번호(제안한 구조에서는 i로 참조)를 기준으로 저장되며 메모리에서 읽어낼 경우에는 출력 포트 주소(제안한 구조에서는 j로 참조)를 기준으로 읽어낸다. 예를 들어서 입력 포트 i로부터 들어온 패킷 k는 헤더 정보를 읽힌 후 데이터 부분이 추출되어 메모리 뱅크에저장 된다. 입력 i로부터 들어 오는 패킷의 데이터는 입력 인터페이스 i에 연결된 메모리 슬롯 내에서 패킷의 출력 포트 j에 해당하는 위치의 메모리에 저장 된다. 즉, 비어 있는 메모리의 위치에 해당 패킷의 데이터를 저장하고 각 메모리에 할당된 메모리 주소(index) i, j에 대한 포인터를 미니 패킷에 넘겨준다. 출력 포트에서 전송될 패킷은 출력 포트 j에 연결된 버스에서 자신이 들어 왔던 입력 포트 번호에 해당하는 메모리에서 메모리 포인터를 참조하여 패킷을 읽어 내면 된다. 이 과정에서 입력 i로 들어오는 패킷에 할당되는 메모리 셀 내의 주소는 Memory Controller 내의 Idle Address Controller에 의해서 수행된다.

전체 메모리 셀을 논리적으로 하나의 메모리뱅크(Memory Bank)로 보고 할당된 메모리 주소를관리하며 입력된 패킷에 대한 새로운 메모리에 할당과 전송된 패킷에 의해서 반납된 메모리 공간

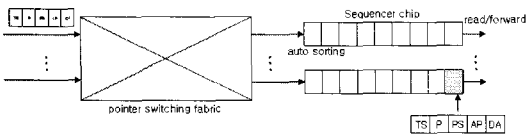
의 제어를 Memory Controller가 수행한다. IPC에 의해서 메모리에 write 시그널이 들어오면 Idle Address Controller에 유지되는 비어 있는 메모리 주소 리스트(Idle Address List)에서 메모리 주소를 할당하고 만약 OPC에 의해서 read 시그널이 도착하면 Idle Address Controller에 유지되는 비어 있는 메모리 주소 리스트에 메모리 주소를 반납한다. (그림 6)은 입/출력 포트의 전체적인 동작을 보여 준다.



(그림 6) 입력 포트와 출력 포트의 동작

3. 출력 포트에서의 동작

입력 포트에서 보낸 미니 패킷이 제안한 PSF를 통과하여 출력 포트에서 정렬된 후 원래의 패킷 데이터와 결합하여 전송되는 과정은 (그림 7)과 같다.



(그림 7) PSF를 통과한 미니 패킷

IPC에서 생성된 미니 패킷은 제안한 PSF를 통해서 해당 출력 큐로 향하게 된다. 이 과정에서 미니 패킷은 정렬된상태로 저장되고 매 전송 시간에 HOL(Head-of-Line) 패킷이 선택된다. 패킷의 전송 순서를 결정하는 정보인 미니 패킷은 입력 포트에서 필요한 값을 할당 받아 전송된다.

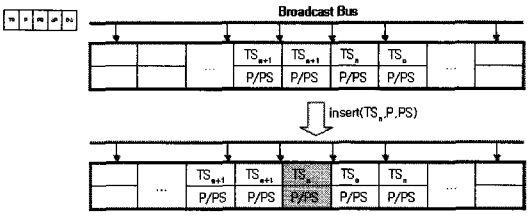
3.1 시퀀서

PSF에서 스위칭 되는 정보는 타임 스탬프(TS), 우선 순위(P), 패킷 크기(PS), 메모리 포인터(AP), 출력 포트 주소(DA)로 구성되어 있다. 타임 스탬프는 패킷이 스위치 입력 단에 도착한 시간을 계산한 값으로 서로 다른 포트에서 도달한 주소 포인터가같은 우선 순위를 가질 경우 FIFO로 동작한다. 경쟁 해결에 사용 되는 우선 순위 필드는 입력 포트 제어기(IPC)에서 트래픽 클래스에 따라 할당되는 서비스 우선 순위 값으로 결정된다. 패킷의 크기는 공평 큐잉(Fair Queuing)을 간편하게 구현 하기 위한 인자이며 패킷의 크기가 작은 패킷을 먼저 서비스 한다. 메모리 포인터는 출력 큐에서 전송 순서가 된 패킷을 메모리 뱅크에서 읽어올 때 사용한다.

3.1.1 시퀀서의 동작

PSF를 통해서 출력 포트로 스위칭 된 미니 패킷은 시퀀서에 정렬 필드로 작용하는 타임 스탬프와 우선 순위, 패킷 크기를 기준으로 정렬 되어 오른쪽에서 왼쪽으로 오름차순 저장이 되며 전송 차례가 되면 시퀀서의가장 오른쪽 쌍의 주소(HOL)부터 RC(Read Controller)에 넘겨준다. 이때 시퀀서에서는 가장 오른쪽 쌍을 밀어내고 모든 쌍을 한 자리씩 오른쪽으로 이동시킨다. 즉, 타임 스탬프와 우선 순위 필드, 패킷 크기로 정렬되는 미니 패킷은 시퀀서에 저장이 되는데 높은 우선 순위의 미니 패킷들이 항상 낮은 우선 순위의 미니 패킷들보다 오른쪽에 저장되며 RC에 의해서 가장 먼저 접근된다.

(그림 8)과 같이 시퀀서의 구현 개념은 매우 단순하다. 입력 포트 i에서 출력 포트로 가는 미니 패킷의 타임 스탬프, 우선 순위, 패킷 크기가

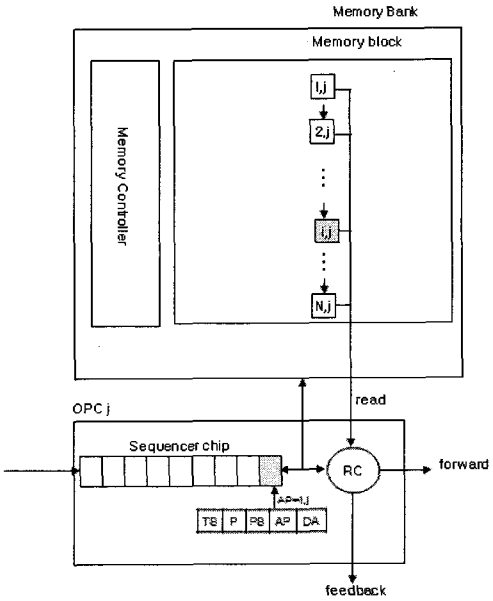


(그림 8) 출력 포트에 도달한 미니 패킷을 시퀀서에 삽입 하기

(TSn, P, PS) 일 때 출력 포트 j의 시퀀서에 삽입되는 과정은 다음과 같다. 정렬 필드인 타임 스탬프, 우선 순위, 패킷 크기는 값이 작을 수록 높은 우선 순위를 가지게 된다. 따라서 위와 같은 우선 순위를 가진 미니 패킷이 도착 했을 때 같은 우선 순위 (TSn, P, PS)를 포함하여 보다 오른쪽에 있는 모든 쌍들은 그대로 두고 우선 순위가 작은 (TSn+1, P, PS)를 포함하여 나머지들은 왼쪽으로 한 자리씩 이동(shift)하고 이동으로 생겨난 빈 위치에 새로운미니 패킷을 삽입한다. 즉, 우선 순위와 패킷 크기가 같고 타임 스탬프 값이 틀린 경우의 예이다. 이 밖에 우선 순위와 패킷 크기 필드가 다른 미니 패킷이 시퀀서에서 정렬되는 자세한 설명은 다음 절에서 다룰 것이다. 시퀀서가 모두 채워지면 시퀀서의 가장 왼쪽에 있는 미니 패킷의 우선 순위 값과 새로 도착한 미니 패킷의 우선 순위 값을 비교 한다. 만약 새로 도착한 미니 패킷의 우선 순위 값이 가장 왼쪽에 있는 우선 순위 값보다 작으면 왼쪽에 있던 미니 패킷은 시퀀서로부터 밀려나고 새로운 미니 패킷이 시퀀서로 삽입된다. 가장 왼쪽에 있던 미니 패킷이 가지고 있던 메모리 포인터의 패킷은 메모리 뱅크에서 새로운 패킷으로 갱신 된다. 만약 새로 도착한 미니 패킷의 우선 순위 값이 가장 왼쪽에 있는 미니 패킷의 우선 순위 값과 같거나 크면 새로 도착한 미니 패킷은 폐기된다.

3.2 읽어내기와 전송하기(Read and Forward)

출력 포트 제어기(OPC, Output Port Controller)에서의 동작은 (그림 9)와 같다. 스위칭 패브릭을 통하여 출력 포트로 온 미니 패킷들은 입력 포트에서 결정된 우선 순위에 의해 시퀀서에서 자동 정렬되고 이를 RC에 의해서 HOL부터 꺼내서 메모리 뱅크의데이터를 참조하여 전송한다. 자세한 동작 과정은 다음과 같다.



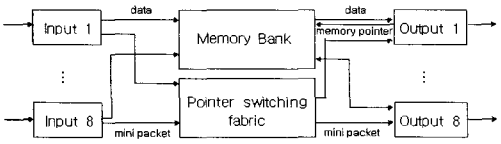
(그림 9) 출력 포트에서의 동작

출력 포트 j에서 RC에 의해서 선택된 HOL 미니 패킷의 주소 포인터를 통해서 OPC는 read 시그널을Memory Controller에 보내고 해당하는 메모리 (i,j)에서 메모리 포인터(index)를 통해 데이터를 읽어온다. 메모리에서 읽어 온 데이터를 새로운 헤더 정보와 결합하여 출력 인터페이스로 전송한다. 전송이 정상적으로 끝나면 메모리 뱅크의 해당 메모리 포인터를Memory Controller 내의 Idle address controller에 반환한다. 폐기된 패킷이 발생하면 local priority 값을 설정해서 IPC로 피드백 시키며 우선 순위를 재조정한다. 출력 포트의 시퀀서에서 미니 패킷이 우선 순위 (TSn+1, P, PS)로 정렬되는 동작은 결과적으로 WFQ를 사용하는 것과 같은 효과를 가진다.

4. 성능 평가

4.1 시뮬레이션 환경

시뮬레이션에 사용될 스위치 구조는 (그림 10)과 같다.



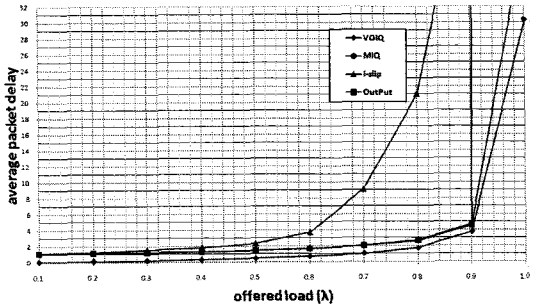
(그림 10) 시뮬레이션 스위치 구조 (8 x 8)

시뮬레이션에 사용된 조건은 다음과 같다. 모든 입력 포트와 출력 포트의 링크 용량은 동일하고 모든 버퍼의 속도는 외부 회선 속도와 동일하다. 출력 포트에 사용되는 시퀀서의 크기는 256이다. 각 입력 포트로 들어오는 패킷들은 베르누이(Bernoulli) 확률 분포를 따르며 각 트래픽 원은 모든 출력 포트에 대해 동등하게 패킷을 발생시키는 uniform traffic 을 가정한다. 트래픽의 클래스는 8-Class를 적용하고 (표 1)과 같이 패킷을 발생 시키고 우선 순위에 따라 처리한다. 각 입력 포트에서 패킷이 들어올 확률은 λ이고 μ 확률로 전송한다고 가정한다. 제안한 구조에서는 HOL 블로킹이 없으므로 μ는1.0 이다. 이 과정에서 uniform 트래픽에 대한 평균 지연 시간 (E(T))은 $E(T) = (1-\lambda)/(\mu-\lambda)$ 이다. 제안한 구조에서 메모리 뱅크의 동작은 VOIQ(Virtual Output/Input Queue) 형태로 볼 수 있기 때문에 VOIQ로 명명한다.

〈표 1〉 트래픽 클래스별 구성 비율

Class	Traffic type	Portion (%)
1	Voice	18
2	Video	15
3	Call-Signaling	5
4	Network Control	5
5	Critical Data	27
6	Bulk Data	4
7	Best-Effort	25
8	Scavenger	1

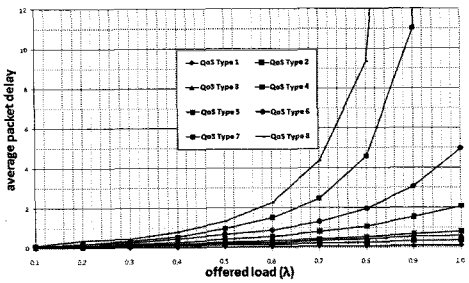
4.2 시뮬레이션 결과



(그림 11) 평균 지연 시간

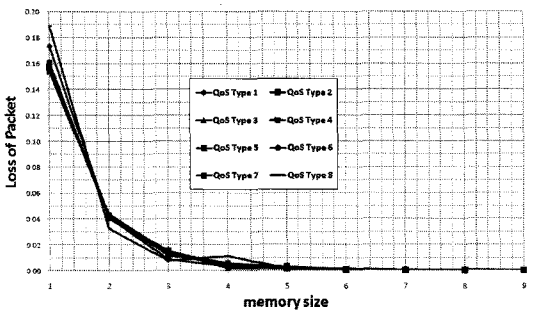
(그림 11)은 주어진 시뮬레이션 환경에서 제안한 구조(VOIQ)와 기존 공유 메모리 기반의 MIQ(Multiple Input Queue), Output, i-slip 방식들을 평균 지연 시간의 관점에서 비교한 실험 결과이다.

제안한 구조에서는 기존의 구조들에서 사용된 복잡한 중재 알고리즘이나 스케줄링 알고리즘을 별도로 구현하지 않았음에도 불구하고 모든 입력 부하(offered load, λ)의 조건에서 기존 공유 메모리 기반의 구조들에 비해 상대적으로 낮은 평균 지연 시간을 보여 준다. 제안한 구조에서는 입력 포트에서 패킷 헤더를 읽어서 필요한 작업을 하는 동안 우선 순위 값을 할당하여 미니 패킷에 삽입 하는 작업을 수행한다. 출력 포트에서는 할당된 우선 순위 값에 따라 시퀀서에서 자동 정렬되기 때문에 별도의 중재 알고리즘이나 스케줄링 없이도 우수한 성능을 얻을 수 있다. (그림 12)는 각 패킷 클래스별 평균 지연 시간을 보여 준다.



(그림 12) 클래스별 평균 지연 시간

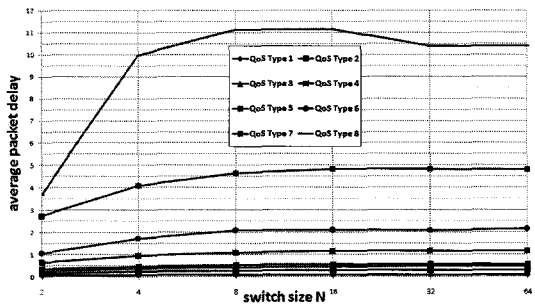
실험 결과 모든 트래픽 클래스에서 만족할 만한 평균 지연 시간을 보여 준다. 특히 우선 순위가 높은 트래픽에대한 부하의 최대치에서도 탁월한 성능을 보여 준다.



(그림 13) 메모리 크기에 따른 패킷 손실률

포화 상태에서 입력 부하 λ가 0.9일 때 메모리 뱅크의 임의의입력 i, 출력 j에 해당하는 독립적인 메모리의 크기에 따른 패킷 손실률은 (그림 13)과 같다.

메모리의 크기는 한번에저장 가능한 가변 길이 IP 패킷의 개수이다. 독립 메모리의 크기가 5 이상일 경우 패킷 손실률이 급격히 감소하며 QoS가 높은 클래스의 손실률은 0이 된다. 메모리 크기가 9 이상일 경우 모든 트래픽에 대한 패킷 손실률은 0이 된다.



(그림 14) 스위치 크기별 평균 지연 시간

(그림 14)는 입력 부하가 0.8로 일정하고 스위치 크기 N이 2, 4, 8, 16, 32, 64로 증가함에 따라 변화하는 평균지연 시간을 보여 준다. 결과적으로 제안한 구조는 스위치 크기가 증가 하더라도 일정한 지연 시간을 보장하는 우수한 확장성(Scalability)을 가지고 있음을 확인할수 있다.

5. 결론

본 논문에서는 인터넷 망에서의 고속스위칭을 고려한 스위치 구조로써 데이터 메모리 평면과 스위치 평면을 분리하여 패킷 데이터의 저장 영역인 메모리뱅크에서의 작업과 패킷의 메모리 뱅크 내 메모리 주소 포인터의 스위칭을 병렬적으로 처리할 수 있도록 하는 새로운 스위치 구조를 제안하였다. 입/출력 포트 각각에 독립적인 메모리 집합을 관리하는 메모리 뱅크를 사용하여 IP 패킷을 위한 가변 길이 스위칭이 가능하고 독립적인 메모리를 사용함으로써 공유 메모리 방식에서의 VOQ와 같은 복잡한 중재 알고리즘을 사용할 필요가 없다. 또한 출력 포트 제어기 부분에 시퀀서를 사용하여 하드웨어 속도의 공평 스케줄링(Fair Scheduling)이 가능하다. 성능 평가 결과 기존 공유 메모리 기반의 스위치에서 MIQ, Output, i-slip 방식을 사용 했을 때와 평균 지연 시간을 비교하여 우수한 성능을 확인 하였고 사용자 트래픽 요구 사항에 대한 QoS 평가를 위해서 클래스별 평균 지연 시간을 비교한 결과 전반적인 입력 부하에 대한 우수한 성능을 확인 하였다. 따라서 제안한 구조는 초고속 인터넷 망의 고속 패킷 스위칭과 사용자 요구를 동시에 만족할 수 있는 구조로 적합하다. 향후 연구 과제로는 제안된 스위치의 멀티 캐스트 기능을 보완하는 것과 입력 부하가 9.0 이상일 때도 우선 순위가 낮은 사용자 트래픽에 대한 QoS를 보장하는 부분이다.

참 고 문 헌

[1] H. Chao and B. Choe, "Design and Implementation of Abacus Switch: A Scalable

Multicast ATM Switch," *IEEE JSAC. vol. 15, no. 5*, Jun., 1997.

[2] H. Chao and B. Choe, "Design and Analysis of a Large-Scale Multicast Output Buffered ATM Switch," *IEEE/ACM Trans. Networking, vol. 3, no. 2*, Apr., 1995.

[3] J. Garcia, L. Cerda, J. Corbal and M. Valero, "A conflict-free memory banking architecture for fast VOQ packet buffers," *in Proc. IEEE GLOBECOM '03*, pp. 4158-4162, 2003.

[4] J. Wang and K. Nahrstedt, "Parallel IP packet forwarding for tomorrow's IP routers," *IEEE Workshop on High Performance Switching and Routing*, pp. 353–357, 2001.

[5] E. Oki and N. Yamanaka, "Scalable Crosspoint Buffering ATM Switch Architecture Using Distributed Arbitration Scheme," *in Proc. IEEE ATM '97 Workshop*, pp. 28-35, 1997.

[6] M. Hashemi and A. Leon-Garcia, "A RAM-Based Generic Packet Switch with Scheduling Capability," *in Proc. IEEE BSS '97*, pp. 28-35, 1997.

[7] C. Minkenberg and T. Engbersen, "A Combined Input and Output Queued Packet-Switched System Based on PRIZMA Switch-on-a-Chip Technology," *IEEE Commun. Mag.*, pp. 70-77, 2000.

[8] J. Bennett and H. Zhang, "Hierarchical Packet Fair Queueing Algorithms," *IEEE/ACM Trans. Networking, vol. 5, no. 5*, Oct., 1997.

[9] S. Moon, J. Rexford and K. Shin, "Scalable Hardware Priority Queue Architectures for High-Speed Packet Switches," *IEEE Trans. Computers, vol. 49, no. 11*, Nov., 2000.

[10] K. Yoshigoe and K. Christensen, "An Evolution to Crossbar Switches with Virtual Output Queuing and Buffered Cross Points," *IEEE Network*, pp. 48-56, Sep./Oct., 2003.

[11] M. Hashemi and A. Leon-Garcia, "The Single-Queue Switch: A Building Block for Switched with Programmable Scheduling," *IEEE JSAC. vol. 15, no. 5*, Jun., 1997.

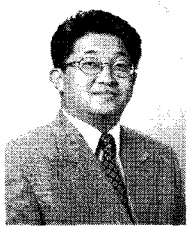
[12] H. Jonathan Chao, "A VLSI Sequencer Chip for ATM Traffic Shaper and Queue Manager," *IEEE JSAC, vol. 27, no.11*, Nov, 1992.

[13] C.Kolias and L. Kleinrock, "The Odd-Even input queueing ATM switch: performance evaluation," *in Proc. IEEE ICC '96, vol.3*, pp.1674-1679, 1996.

[14] "Cisco 12000 Series – Gigabit Switch Routers," *http://www.cisco.com/univercd/cc/td/doc/pcat/12000.pdf*

◐ 저 자 소 개 ◑

류 경 숙(Kyoung-Sook Ryu)
1996년 동국대학교 전자계산학과 졸업(학사)
2001년 동국대학교 교육대학원 졸업(석사)
2001~현재 동국대학교 정보통신공학과 박사과정
관심분야 : 초고속 통신망의 트래픽 제어, 초고속 스위치, 라우터
E-mail : ksryu@dgu.edu



최 병 석(Byeong-Seog Choe)
1985년 서울대학교 전자공학과 졸업(학사)
1987년 미국 Fairleigh Dickinson University EE 졸업(석사)
1993년 미국 Polytechinc University EE 졸업(석사)
1994년 미국 Polytechinc University EE 졸업(박사)
1997~현재 동국대학교 정보통신공학과 교수
관심분야 : 초고속 위성망, 초고속 통신망의 트래픽 제어, 광대역 접속 방식, 라우터
E-mail : bchoe@dgu.edu