논문 2004-41SD-5-15

# Diffserv 지원 VOQ-PHB 방식의 MPLS 스위치의 구현에 관한 연구

# (Study on Implementation of an MPLS Switch Supporting Diffserv with VOQ-PHB)

이 태 원*, 김 영 철*

(Tae-Won Lee and Young-Chul Kim)

요 약

인터넷 트래픽의 급격한 증가에 따라, 새로운 멀티미디어 서비스의 요구를 수용하기 위해서 MPLS가 제안되었으며, MPLS는 QoS를 보장하는 Differentiated Service를 제공하는 방향으로 진화되고 있다. 본 논문에서는 Diffserv를 지원하며, 고속의 스위칭이 가능한 MPLS 스위치의 구조를 제안한다. 트래픽 제어기는 분류, 측정, 기록 등의 기능을 수행하도록 구성되었다. 스위치는 입력 큐잉 방식으로 QoS를 보장하도록 VOQ와 PHB별 큐를 확장한 방식이며, 이의 스케쥴링 알고리즘으로는 Priority-iSLIP 알고리즘을 사용하였다. 제안한 구조는 NS-2 시뮬레이터로 모델링하여 검증하였고, VHDL을 이용하여 모델링하여 합성한 후, SYNOPSYS사의 VSS analyzer를 이용하여 그 타당성을 검증하였다. 또한 Apollo tool을 이용하여 layout을 수행하였다.

Abstract

Recently, the growth of Internet and a variety of multimedia services through Internet increasingly demands high-speed packet transmission, the new routing function, and QoS guarantee on conventional routers. Thus, a new switching mechanism, called the MPLS(Multi-Protocol Label Switching), was proposed by IETF(Internet Engineering Task Force) as a solution to meet these demands. In addition, the deployment of MPLS network supporting Differentiated Services is required. In this paper, we propose the architecture of the MPLS switch supporting Differentiated Services in the MPLS-based network. The traffic conditioner consists of a classifier, a meter, and a marker. The VOQ-PHB module, which combines input queue with each PHB queue, is implemented to utilize the resources efficiently. It employs the Priority-iSLIP scheduling algorithm to support high-speed switching. We have designed and verified the new and fast hardware architecture of VOQ-PHB and the traffic conditioner for QoS and high-speed switching using NS-2 simulator. In addition, the proposed architecture is modeled in VHDL, synthesized and verified by the VSS analyzer from SYNOPSYS. Finally, to justify the validity of the hardware architecture, the proposed architecture is placed and routed using Apollo tool.

**Keyword** : MPLS, Diffserv, VOQ-PHB, 스위치

## I. 서 론

최근 인터넷 사용자의 급격한 증가와 멀티미디어 응용 서비스의 증가로 인하여 인터넷 트래픽의 급격한 증가를 초래하고 있다. 이러한 트래픽의 증가와 함께 실시간 서비스의 요구를 수용하기 위해서 현재의 인터넷을 확장한 새로운 인터넷 백본망을 구축할 필요성이 부각되었다. 이에 따라 MPLS망을 이용하여 트래픽 플로우에 대한 QoS를 보장하며 다양한 서비스를 제공하고자 하는 Differentiated Service(Diffserv)의 적용이 본격적으로 진행되고 있다. MPLS망에서 Diffserv를 지원하기 위해서는 Diffserv 트래픽 분류, 미터(Meter), 마킹(Marking) 등의 기능을 수행하는 트래픽 조절기와 각 PHB(Per Hop Behavior)를 지원하는 모듈을 구성해야 한다.

따라서 PHB 처리 모듈의 성능이 Diffserv를 지원하

* 정회원, 전남대학교 전자정보통신공학과
(Department of Electronics and Computer Engineering, Chonnam National University)
접수일자: 2004년1월27일, 수정완료일: 2004년4월21일

는 고속 라우터의 전반적인 QoS와 고속 처리성능에 크게 영향을 미치고 있다.

MPLS 망에서 고속의 Diffserv를 지원하기 위해서는 통상 같은 스케쥴링 알고리즘과 같은 경로 선택 알고리즘을 선택했을 때 무엇보다도 Diffserv 모듈에 의한 지연을 어떻게 줄일 것인가에 중점을 두어야 한다. Diffse-rv 모듈에 의한 지연은 트래픽 조절기에서의 지연과 PHB 처리에 의한 지연을 모두 포함한다. 패킷들이 PHB 처리과정을 거치면서 클래스 별로 경쟁함으로 인해 망 내의 지연도 증가하게 된다. 따라서, PHB 처리 구조의 지연을 줄이고 고속 구현이 가능한 구조가 필요하게 된다.

현재 고속 라우터에서는 Diffserv를 지원하기 위해 기존의 IP 라우터의 전송 엔진에 PHB 처리 모듈을 두어 링크 스케쥴러에 의해 전송되는 구조와 표준화에서 요구하는 PHB 처리를 위해 모든 라우터에 PHB 처리 모듈을 두지 않고, 입구 라우터의 스위칭 전단에 PHB 처리 모듈을 두어 확장성을 개선하고, 망 지연을 개선한 구조가 연구되고 있다. 이 방식들은 망의 지연 요소인 트래픽 조절기에 의한 지연과 버퍼에 의한 지연을 고려하지 않았으며, 고속 스위치의 구조와 PHB 처리 모듈의 구현에 따른 라우터의 성능에 대해서는 다루지 않고 있다. 즉, 라우터에 PHB 처리 모듈을 구성하여 Diffserv를 지원하고 있지만, PHB 처리 모듈 구현에 따른 망의 지연과 PHB 처리 모듈의 하드웨어 구성의 적합성에 대해서는 논의되고 있지 않다. 따라서, 본 논문에서는 Diffserv를 지원하는 구조로서 PHB 처리 모듈의 적합한 구조를 제안한다.

PHB 처리 모듈은 PHB별 저장과 스케쥴러, 그리고 모듈의 위치에 의해 차별화된다. 기존의 고속 라우터에서는 입력 버퍼링 방식으로 VOQ 방식을 사용하여, 라우팅 프로토콜 및 신호 프로토콜에 의해 결정된 출력 목적지별로 가상의 큐에 해당되는 트래픽이 저장되고, 저장된 트래픽은 스케쥴링 알고리즘에 따라 출력 버퍼에 저장해서 다음 노드로 전달된다. Diffserv를 지원하기 위해서는 PHB 처리 모듈이 구성되어 적절한 스케쥴링 알고리즘에 의해 다음 노드로 트래픽을 전송하여야 한다. 결국 기존의 고속 라우터에 PHB 처리 모듈이 추가되는 구조이며, 이는 추가적인 버퍼 지연을 불러오고, 큐의 효율적인 사용이 가능하지 않은 구조이다. 즉, PHB 처리에 대한 지연과 VOQ 버퍼에 의한 지연으로 인해 전체 망의 지연을 증가되므로, PHB 처리와 VOQ 버퍼에 의한 지연 개선이 요구된다.

본 논문에서는 PHB 처리에 따른 망 성능의 저하를 개선하는 구조로써, 가상 목적지별 큐잉과 PHB 처리를 동시에 수행하는 VOQ-PHB 구조의 PHB 처리 모듈을 구현하고 그 성능을 검증한다. 또한 Diffserv 지원을 위해 트래픽 조절기를 구현하여, Diffserv 지원 MPLS 스위치 구조를 제안하고 구현한다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 Diffserv를 지원하는 MPLS 스위치의 구조 및 기능에 대해서 설명하고, Ⅲ장에서는 Diffserv를 지원하는 고속 VOQ-PHB 구조 및 알고리즘에 대해서 설명하며, Ⅳ장에서는 실험결과를 보이고, Ⅴ장에서는 제안한 구조의 하드웨어 설계에 관하여 설명하고, 마지막으로 결론을 맺는다.

## Ⅱ. Diffserv over MPLS Switch

MPLS는 동일한 전송 등급을 갖는 플로우들을 하나로 묶어 하나의 레이블로 할당하여 설정된 경로에 따라 레이블 교환 방식을 이용하여 패킷을 전송한다. 또한, 현재 인터넷의 다양한 멀티미디어 서비스 요구를 만족시키기 위한 방안으로 제시되고 있다. 인터넷의 멀티미디어 서비스 수용을 위해 제시된 Diffserv는 동일한 서비스 클래스별로 동일한 서비스 처리를 수행하는 구조이다. 따라서 MPLS 망에서의 전송 등급과 Diffserv의 서비스 클래스가 유사한 개념으로 제시되고 있고, 이를 이용해 MPLS 망에서 Diffserv를 지원하는 방안들이 제안되고 있다.

### 1. Diffserv over MPLS의 개념

MPLS는 패킷 전달을 고속화하기 위해 짧고 고정된 길이의 레이블을 이용하여 3계층을 거치지 않고 패킷을 포워딩할 수 있게 하는 방식으로 단순한 포워딩 과정을 통해 기존의 IP 포워딩에 비해 망에서의 전달 속도를 증가시킨다는 장점을 가지고 있다.

Diffserv를 지원하는 망에서 트래픽 스트림들은 특별한 홉 단위의 전송방식을 받기 위해 진행하는 경로상의 라우터들로부터 분류되고 마킹되는데, 망의 경계나 호스트들에게서 주로 복잡한 분류와 마킹, 폴리싱(Polic-ing), 쉐이핑(Shaping)등의 트래픽 조절이 수행되고, 내부에서는 마킹된 값에 근거해서 입력된 패킷을 미리 계약된 서비스에 맞추어 처리하기 때문에 플로우의 흐름이 많은 망의 코어 부분에서는 매 플로우별로 상태나 정보를 유지할 필요가 없다는 장점이 있다.

그림 1은 Diffserv 라우터의 기능을 나타내고 있다.

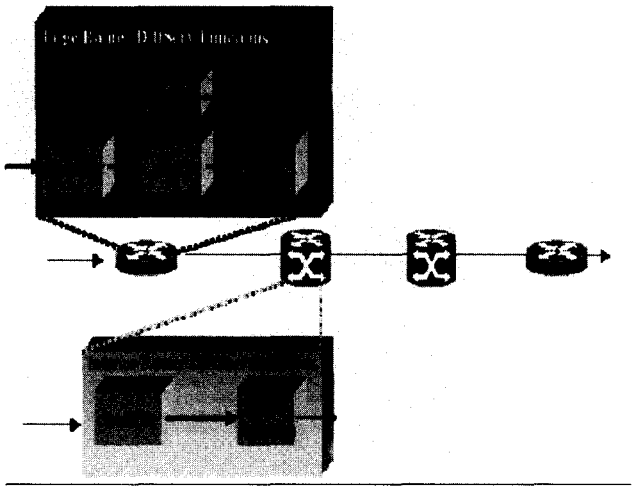


그림 1. Diffserv 지원 라우터의 기능
Fig. 1. The function of the router supporting Diffserv.

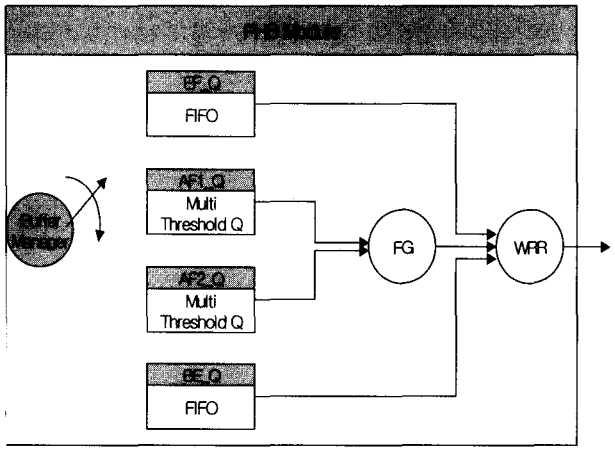



그림 2. PHB 처리 모듈
Fig. 2. The architecture of the general PHB processing module.

경계 노드에서는 패킷을 분류하고, 폴리싱과 사용자 협약에 맞게 적절하게 트래픽을 조절하여 트래픽의 특성에 맞게 PHB를 지정하여 트래픽의 QoS를 보장하는 구조이며, 내부 노드에서는 간단히 트래픽의 분류와 PHB 지정만을 처리하는 구조이다.

MPLS망에서 Diffserv를 지원하기 위해서는 트래픽 조절과 PHB 폴리싱, PHB 매핑, QoS와 레이블 매핑 등의 기능이 지원되어야 한다.

## 2. Diffserv 지원을 위한 PHB 처리 모듈의 구조

그림 2는 PHB 처리 모듈의 일반적인 구조를 나타내고 있다. 트래픽 조절기에서 패킷의 분류와 미터링, 쉐이핑을 거친 패킷들은 정의된 PHB로 마킹된다.

PHB는 EF( Expedited Forwarding), AF(Asured Forwarding)xx, DF (Default Forwarding) 등으로 구분하고 있다.

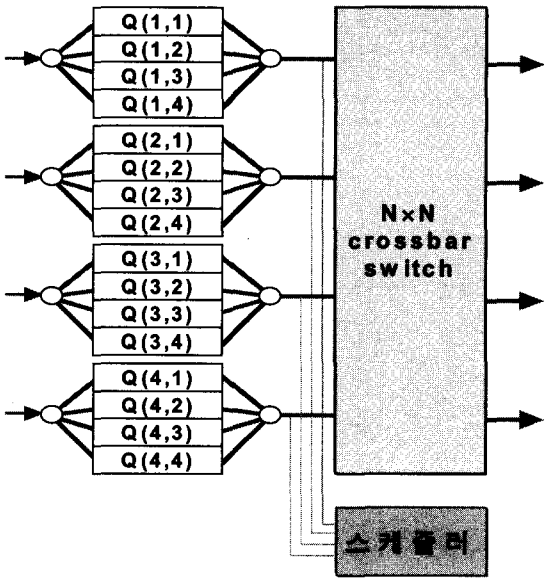



그림 3. 목적지별 저장 방식 크로스바 스위치 구조
Fig. 3. The architecture of the crossbar switch with VOQ.

각 PHB는 QoS의 용이한 보장을 위해 각 PHB별로 구성된다. EF와 DF 트래픽을 위한 버퍼는 단순한 FIFO 구조로 관리되고, AF 트래픽은 CLP비트의 종류에 따라 다른 폐기 확률을 지원하기 위해 멀티 임계값을 갖는 버퍼가 사용된다. 버퍼에서 패킷을 선택하는 스케쥴링 알고리즘으로는 FQ (Fair Queuing), WRR(Weighted Round Robin), PS(Priority Scheduling) 등이 이용된다. 이러한 PHB 처리 모듈은 라우터의 출력 포트에 포함되거나 입력 포트 혹은 스위치 전단에 위치한다.

Diffserv 지원 고속 스위칭 방식으로 기가비트/테라비트 라우터에서는 입력 큐잉 방식과 크로스바 스위치 패브릭, 그리고 출력 큐로 구성된다. 입력 큐잉 방식중 대표적인 방법은 VOQ(Virtual Output Queueing)이다.

목적지별 저장 방식이란 입력단의 버퍼를 출력단별로 관리하는 것으로 물리적으로는 하나의 버퍼가 있지만 논리적으로는 N개의 버퍼가 있는 것처럼 관리하는 것이다. 이 경우 어느 데이터든 먼저 전송 선택되어 나갈 수 있으므로 HOL 차단 현상은 완전히 없어지며 100 % 의 최대 처리율을 얻을 수 있다. 이 구조를 N FIFO 방식, VOQ 등으로도 불린다. 목적지별 저장 방식에 고려되는 스위치 구조는 그림 3과 같다.

목적지별 큐잉방식을 사용하기 위해서는 다음과 같은 가정이 필요하다.

- 입력단(i)에 도착한 셀은 해당 목적지 출력단(j) 큐 Q(i,j)에 저장.

- 전송할 셀이 있는 입력단은 스케줄러로 경로 설정을 요구.
- 스케줄러는 매 슬롯마다 전송될 입력단과 출력단 쌍을 결정하고 해당 크로스 포인트를 개방한다.

목적지별 저장 방식에서 스케줄링을 위한 제약 조건은 한 입력단에서 하나의 셀만 선택해야 하며, 한 출력단으로 하나의 셀만 전송해야 한다. 매 슬롯마다 전송을 위해 선택되는 입·출력단 쌍의 수가 많아야 높은 수율을 얻을 수 있다. 본 논문에서는 Diffserv를 지원하면서 동시에 고속 스위칭을 보장하는 입력 큐잉 방식과 PHB 모듈의 효율적인 구조를 제안한다.

## Ⅲ. VOQ-PHB를 이용한 MPLS 스위치

PHB 처리 모듈을 구성한다는 것은 물리적인 큐를 둔다는 의미이고, 이에 따라 메모리 접근 시간이 추가된다. 따라서, VOQ의 버퍼와 PHB별 버퍼를 효율적으로 사용하면서, QoS보장을 위한 스케줄링 방식이 구현된다면 추가적인 버퍼 사용을 줄이고 지연을 개선하여 전체 망의 성능을 개선시킬 수 있다. 즉, VOQ의 가상의 큐가 입·출력 포트와 PHB로 구분되도록 확장하여, VOQ와 PHB별로 트래픽을 저장하고 QoS를 보장하면서 하드웨어 구현이 용이한 스케줄링 알고리즘을 사용한다면 PHB 처리에 따른 메모리 접근 시간을 VOQ의 메모리 접근 시간으로 대체할 수 있고, 이에 따라 망의 지연을 개선하면서 Diffserv를 지원할 수 있다.

본 논문에서는 PHB 처리에 따른 망 성능의 저하를 개선하는 구조로서 VOQ-PHB 구조의 PHB 처리 모듈을 구성한다. VOQ-PHB 구조는 스위치 내에서 입력 버퍼 방식인 가상 목적지별 큐잉과 PHB 처리를 동시에 수행하는 구조로서, 입·출력 포트에 따라 가상의 큐가 결정이 되며, 각 가상의 큐에서 PHB에 따라 설정된 가상의 큐 내에 새로운 가상의 큐를 선정하여 각 PHB를 저장하게 된다. 또한, 설정된 가상 큐는 입력되는 PHB 트래픽의 종류와 특성에 따라 공유될 수 있는 구조로써, 버퍼 효율을 증가시킬 수 있다. 전송될 트래픽의 선택을 위해 VOQ의 스케줄링 방식인 iSLIP을 이용하여 가상 큐의 입·출력 포트 번호를 결정하며, 그에 따라 선택된 가상의 큐 내에서 우선 순위가 높은 PHB 큐를 결정한다. 선택된 우선 순위 큐에 저장되었던 트래픽이 다음 노드로 전송된다.

그리고 기존의 연구에서 출력 전송 엔진에 PHB 처리 모듈을 두어 PHB별 저장과 스케줄링을 수행했던 것과 달리 본 논문에서는 VOQ-PHB구조에서 PHB처리를 수행하기 때문에 출력 포트는 단순한 출력 버퍼로 구성된다. 그림 4는 제안한 VOQ-PHB 구조이다.

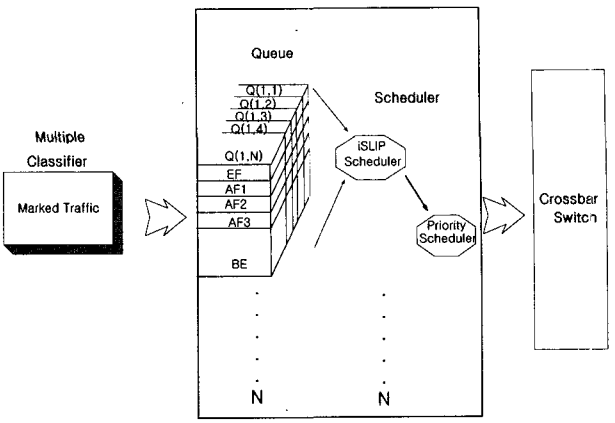



그림 4. 제안한 VOQ-PHB 구조 및 Scheduler
Fig. 4. The proposed architecture of a VOQ-PHB and a scheduler.

입력 큐는 출력 목적지별 가상 큐로 구성되어 있으며, VOQ각각은 각 PHB별로 가상으로 구분된다. 스케줄링 알고리즘으로는 Priority-iSLIP 알고리즘을 사용하였다. iSLIP 알고리즘은 PIM(Parallel Itera- tive Matching)의 랜덤선택과 달리 반복 라운드로빈 방식으로 포인터를 하나씩 증가시켜 최대매칭을 찾아낸다. 이러한 방식을 통하여 대역폭을 동등하고 공평하게 사용할 수 있고, 또한 고속 구현이 가능하다. 그리고 Diffserv QoS보장을 위해 간단한 우선 순위 알고리즘만으로 QoS를 보장할 수 있다.[1][2]

트래픽 조절기로부터 각 PHB로 분류된 트래픽은 신호 프로토콜에 의해 할당된 출력 포트와 PHB에 따라 해당되는 가상 큐에 저장된다. 각 트래픽은 도착한 순서로 스케줄러에 Request를 보내 크로스바 스위치 통과를 요구하게 된다. Request에 따라 iSLIP 스케줄러는 아비터(arbiter)를 통해 Grant와 Accept를 결정하고, Accept된 패킷을 전송하되, PHB 우선 순위에 따라 최우선 순위부터 순차적으로 다음 우선순위 PHB를 보낸다. PHB로는 EF, AF1, AF2, AF3와 DE등으로 구분하였으며, 이는 트래픽 조절기에서 마킹되어 입력된다. 우선 순위는 EF를 제일 높게 두고, DF를 가장 낮게 둔다.

그림 5는 본 논문에서 제안한 VOQ-PHB 구조를 갖는 Diffserv 지원 MPLS 스위치의 구조이다. 그림에서 보듯이 트래픽 조절기, VOQ-PHB 처리 모듈, 크로스바 스위치로 구성되어 있다. 트래픽 조절기에서는 트래픽 분류, 마킹의 기능을 수행하며, 신호 프로토콜에 의해

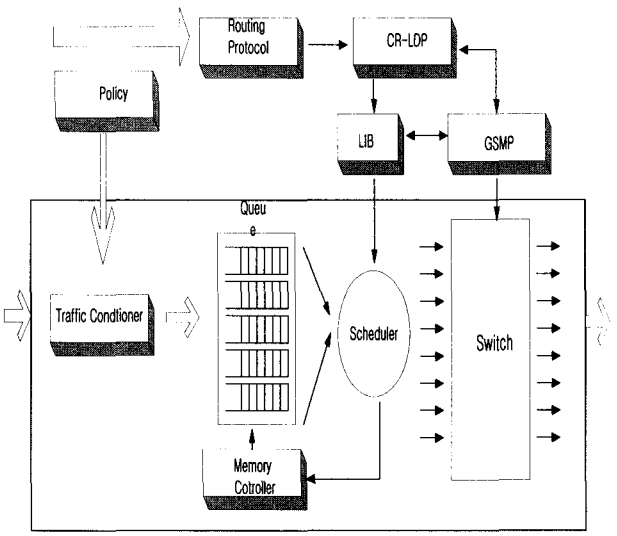



그림 5. 제안한 Diffserv지원 MPLS 스위치의 구조
Fig. 5. The proposed architecture of MPLS switch supporting Diffserv with VOQ-PHB.

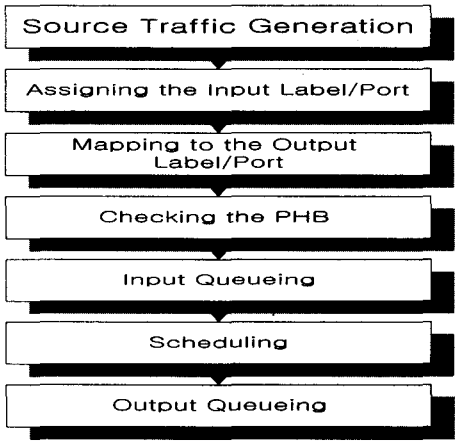



그림 6. 트래픽 처리 절차
Fig. 6. The procedure of traffic processing.

입·출력 인터페이스가 결정되면, 트래픽은 PHB와 출력 포트에 따라 가상 큐에 저장되고 스케줄링되어 전송된다. 큐의 효율적인 사용을 위해 메모리 컨트롤러가 구성되며, 스케줄링 모듈에서는 iSLIP 알고리즘의 아비터와 우선 순위 알고리즘이 구성된다. 스위치는 8X8 크로스바 스위치를 사용하였다.

본 논문에서의 트래픽 처리 절차는 그림 6과 같다. 링크 폴리싱에서 정해진 정책에 따라 입력된 트래픽은 트래픽 조절기에서 분류/미터/쉐이핑 등의 기능을 수행후 DSCP(DiffServ Code Point)를 할당받게 된다. MPLS 신호 프로토콜인 LDP(혹은 CR-LDP)와 FIB(Forward-ing Information Base), 그리고 DSCP를 기초로 하여 LIB(Label Information Base)를 구성하고, 이 LIB를 통해 각 트래픽에 할당되야 할 VOQ의 출력 포트와 PHB

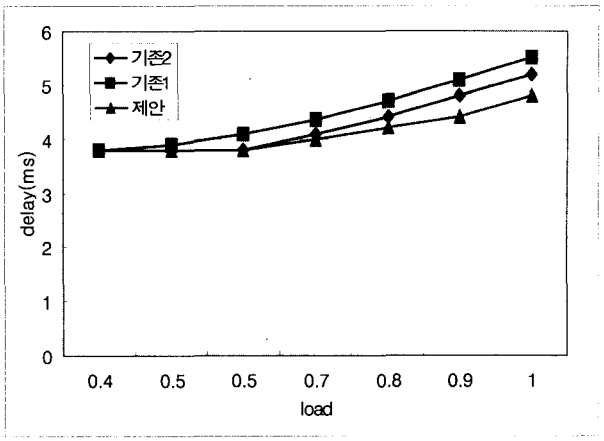



그림 7. EF 트래픽의 패킷 지연 시간
Fig. 7. Packet delay time of EF traffic by load.

큐를 결정하게 된다. 입력 큐에 저장된 각 트래픽은 Priority-iSLIP 스케줄링 알고리즘에 따라 크로스바 스위치를 통과하여 출력 큐에 저장되어 다음 홉으로 전송된다.

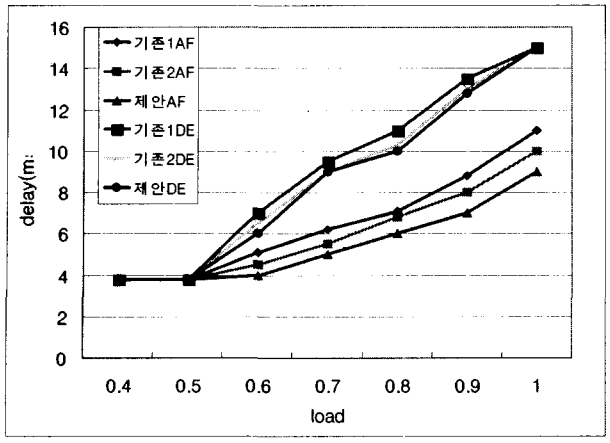
## IV. 실험 결과

제안한 구조의 검증을 위해 NS-2 시뮬레이터를 이용하여 실험하였다. 기존의 입력 큐잉 구조를 가지며 출력 포트에서 PHB별로 저장한 후 스케줄링하는 방식1과 독립적으로 PHB 모듈을 구현한 방식2, 그리고 제안한 VOQ-PHB 방식을 비교하였다. 실험을 위하여 각 PHB별로 트래픽을 발생시켰다. EF 트래픽은 EF의 특성에 맞게 CBR, UDP 패킷으로 발생시켰고, AF, BE는 VBR 트래픽의 On-OFF 소스를 사용하였다. 출력링크의 전송속도는 155 Mbps를 사용하였고, 트래픽은 각 PHB별로 EF 20%, AF1 20 %, AF2 15 %, AF3 15 %, DE 30%의 비율로 발생시켰다.

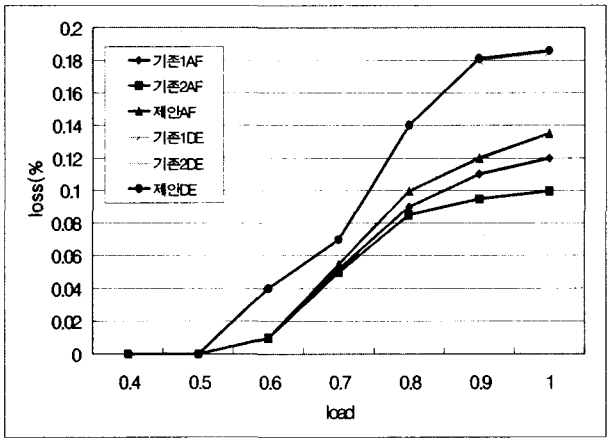
그림 7에서 보듯이 제안한 VOQ-PHB 방식이 기존의 방식들보다 지연 시간이 적었고 이는 큐를 더 효율적으로 사용한 결과이다. 손실률에서는 세 방식에서 거의 손실이 발생하지 않았으며, 방식간에도 거의 차이가 없었다.

그림 8에서는 AF, DF 패킷의 손실률과 지연을 보이고 있다. AF의 경우에서 같은 버퍼 관리 알고리즘이 사용되었다. 기존 방식의 스케줄링 방식에 비해 drop되는 패킷의 수가 많았으나, QoS 보장에는 크게 지장을 초래하지 않았고, 지연은 감소하여 고속 스위칭에 적합함을 알 수 있었다.

그림 9에서는 방식1과 제안한 방식의 버퍼 효율을 나



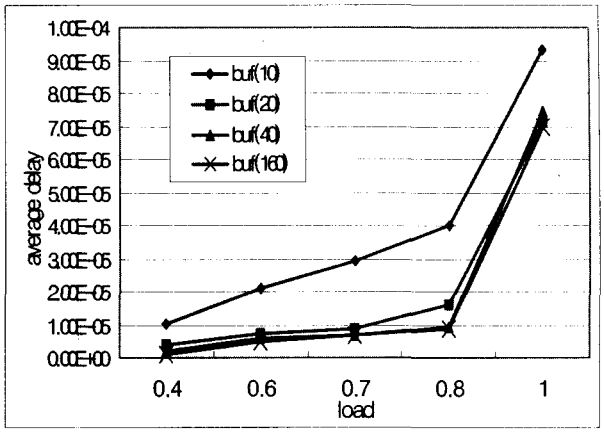

(a)



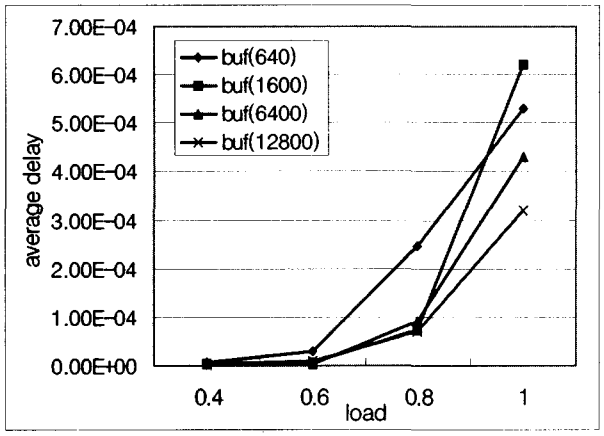

(b)

그림 8. AF, DF 패킷의 손실률과 지연: (a) 트래픽의 지연시간, (b) 트래픽의 평균패킷손실률

Fig. 8. Average packet loss rate and delay of AF, DE traffic by load: (a) Packet delay of AF, DE traffic by load, (b) Average packet loss rate of AF, DE traffic by load.




(a)




(b)

그림 9. 버퍼 효율 시뮬레이션 결과: (a) VOQ-PHB 모듈의 버퍼효율, (b) 출력에서의 PHB 프로세싱모듈의 버퍼효율

Fig. 9. The simulation result of buffer efficiency: (a) the buffer efficiency of VOQ-PHB module, (b) the buffer efficiency of PHB prcessing module in output port.

타내고 있다. 두 방식에서 특정 버퍼 크기 이상에서는 동일한 결과를 나타냄을 알 수 있었고, 제안한 방식에서 단일 포트일 경우 약 2배 이상 적은 양의 버퍼를 사용하여 더 효율적임을 알 수 있다.

그림 10은 AF1과 AF2 트래픽에 대한 전체 성능에 대한 실험 결과이다. 전체 처리량은 노드에서 처리되는 데이터의 량을 나타내며, 망 정책과 SLA를 통해 계약되고, 망에서는 이를 근거로 트래픽을 조절한다.

AF1은 망에서 180의 처리량을 보장받도록 계약한 경우이며, AF2는 80을 보장받도록 계약한 경우이다. 출력 포트 내에서 PHB별 처리 모듈을 구현한 방식과 VOQ-PHB 방식을 비교하였을 때, 거의 동일한 결과를 보이고 있다. 즉 스케줄링 방식에 따라 손실은 증가하지만 전체적인 성능에서는 거의 동일한 결과를 나타내고 있다. 즉 AF1과 AF2 트래픽에 대해 적절한 QoS 보장과 서비스가 이루어지고 있다는 것을 알 수 있다.

## V. Diffserv 지원 VOQ-PHB 방식의 MPLS 스위치 하드웨어 구현

본 논문에서 구현하는 고속 스위치는 8×8 크로스바 스위칭 패브릭을 사용하며, 확장성을 고려하여 입력버퍼 방식을 사용한다. 입력단에는 VOQ-PHB 방식의 지원을 위해 목적지별 FIFO와 각 FIFO안에 PHB 처리를 위해 PHB별 가상 큐를 두고 있다. 이러한 목적지별 큐잉방식의 사용시 발생하는 HOL 차단 현상을 해소하기 위하여 iSLIP알고리즘을 이용하였고, 또한 Diffserv 지

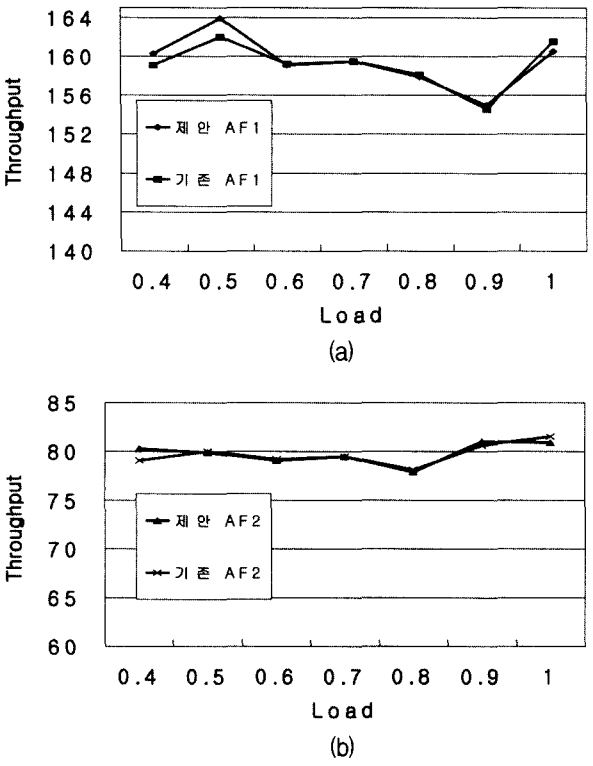



그림 10. AF1과 AF2의 성능: (a) AF1 트래픽의 전체 성능 시뮬레이션 결과, (b) AF2 트래픽의 전체 성능 시뮬레이션 결과

Fig. 10. The simulation result of the throughput: (a) The throughput of AF1 traffic, b) The throughput of AF1 traffic.

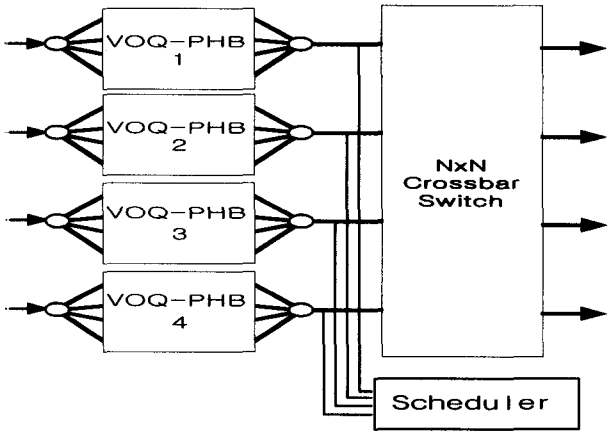



그림 11. 제안한 MPLS 스위치의 구조

Fig. 11. The proposed architecture of MPLS switch.

원을 위해 우선 순위 스케줄러를 사용하여 각 PHB 우선 순위에 따라 입력단의 전송을 제어한다.

## 1. VOQ-PHB 방식의 MPLS 스위치 구현

그림 11은 본 논문에서 제안한 Diffserv 지원을 위한 VOQ-PHB 방식의 MPLS 스위치의 전체 블록도를 나타낸다. 입력단에서는 각 포트별로VOQ-PHB 블록이

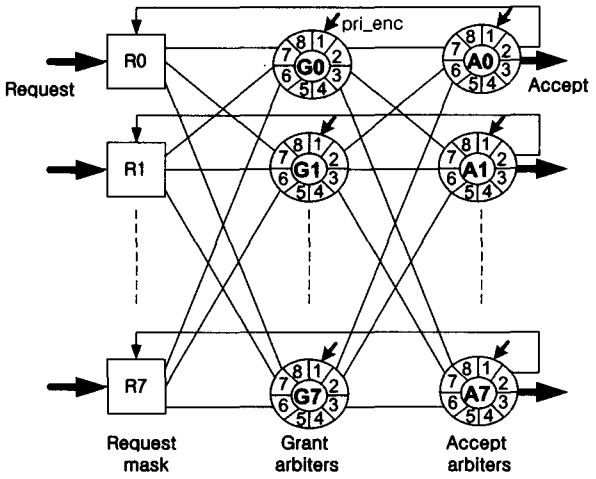



그림 12. iSLIP 스케줄러의 다이어그램

Fig. 12. Diagram of iSLIP scheduler.

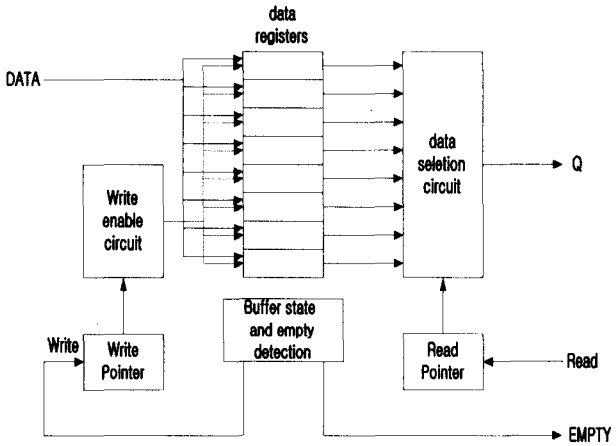



그림 13. FIFO 블록도.

Fig. 13. The block diagram of FIFO.

구성되며, 그림 4에서 설명한 바와 같이 VOQ-PHB 블록에서는 입·출력 포트에 따라 해당 FIFO에 데이터가 저장되며, 각 PHB에 따라 해당 FIFO의 PHB별 버퍼에 저장된다. 버퍼에 저장된 패킷은 스케줄러에 전송 요청 신호를 전송하며 전송 요청 신호를 받은 스케줄러는 iSLIP 알고리즘을 이용하여 전송할 FIFO를 선택하며, iSLIP에 의해 채택된 FIFO에서 우선 순위에 따라 전송된다. 출력 버퍼는 FIFO로 구성되며 패킷을 저장하고, 전송하는 역할을 수행한다.

그림 12는 8×8 크로스바 스위치를 사용할 때 상위레벨 블록도를 나타낸다. iSLIP 알고리즘의3단계(Request -Grant -Accept)가 그림 12의 세 블록들에 대응한다. 세 블록들은 각각의 중재기(Arbiter)를 가지고 있다. 그림에서 accept 중재기에서 request 중재기로 피드백되는 신호는 이미 입력과 출력에 매치된 것으로부터 "Request" 신호를 Mask-off시키기 위하여 사용된다.

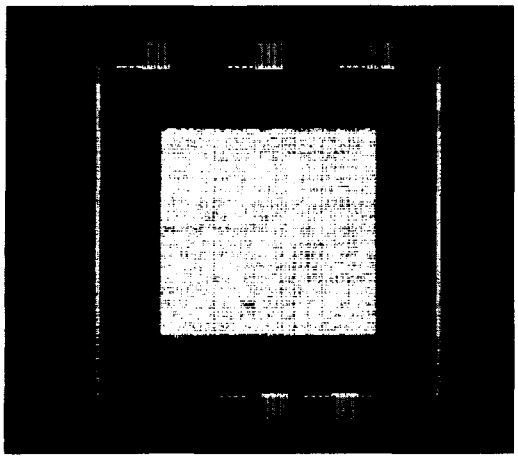


그림 14. VOQ-PHB 방식의 MPLS 스위치의 layout 결과
Fig. 14. The layout result of MPLS switch.

입력 중재기가 출력 중재기로부터 최소 1개의 "Grant" 신호를 받으면 그 입력 포트는 반드시 매치됨을 알 수 있다. 따라서 "Accept" 신호를 받기 전에 출력 중재기들의 "Grant" 신호를 이용해서 "Request" 신호를 Mask-off 할 수 있다. 이 점을 이용하면 한 슬롯의 복수 반복횟수를 파이프라인 형식으로 구현하여 고속으로 동작한다.

중재기는 라운드 로빈 방식으로 구성하여 고속 스위칭이 가능하도록 설계하였다. 중재기는 PPE( Programmable Priority Encoder)와 상태를 저장하기 위한 라운드 로빈 포인터로 구성된다.

본 논문에서 사용된 모든 버퍼는 FIFO 방식을 사용하였다. FIFO는 동기식 FIFO로 구성하였다.

그림 13은 본 논문에서 사용된 FIFO의 구성도이다. 입력 패킷에 대하여 "Write" 신호가 활성화되면 데이터를 레지스터에 저장하며, FIFO는 쓰기 포인터와 읽기 포인터를 가지고 데이터의 읽기, 쓰기를 수행한다. 또한 iSLIP 스케쥴러에 의해 결정된 입력 포트에서 PHB 우선 순위에 따라 데이터를 선택한다.

VOQ-PHB에는 Priority-iSLIP 스케쥴러와 FIFO로 구성된다. 우선 순위는 0이 제일 높고 4가 제일 낮도록 설정하였다. 즉, PHB로 말하자면 EF, AF1, AF2, AF3, DE의 순으로 설정하였다. VOQ 모듈의 구성은 FIFO 구조의 버퍼를 각 목적지별로 8개씩 구성하여 각 포트에 대응시켰다.

패킷은 PHB에 따라 VOQ-PHB 큐에 저장되며, 저장된 패킷은 iSLIP 스케쥴러에서 선택된 목적지 중에서 우선 순위에 따라 패킷을 선택하게 된다. 우선 순위가 낮더라도 상위 우선 순위 패킷이 존재하지 않는다면 낮은 우선 순위의 패킷이 선택되도록 구성하였다. 즉, EF와 AF PHB 패킷이 존재하지 않고 DE 패킷이 존재하면 DE 패킷이 선택되어진다.

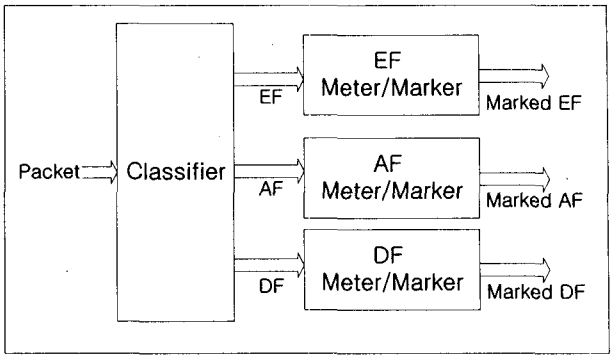



그림 15. Diffserv 모듈의 구조.
Fig. 15. The block diagram of Diffserv module.

그림 14는 VOQ-PHB 처리 모듈과 크로스바 스위치를 포함한 MPLS 스위치를 합성설계하여 back-end 설계를 완료한 layout이다. 현재 8개의 입력 포트와 8개의 출력 포트로 구성되어 있으며, VOQ-PHB 구조를 채택하였고, Prioriyt-islip 스케쥴러가 구성되어 있다.

칩의 크기는 5 mm×5 mm 이며, 삼성 STDA 90 라이브러리를 이용하였고, 약 150,000 게이트가 사용되었다. 예상 처리 용량은 155 Mbps ~ 622 Mbps 이다.

## 2. 고속 Diffserv 모듈 구현

고속 Diffserv 모듈은 Diffserv 영역에서 고속의 패킷 분류, meter, maker의 역할을 수행하는 모듈로서, 전체 구성도는 그림 15와 같다. Diffsev 모듈은 크게 Diffserv 영역에서 DSCP에 따른 패킷의 분류를 맡는 분류기와 각 DSCP 값에 대하여 분류된 패킷의 측정과 기록을 맡는 meter/marker로 구성된다.

분류기는 Diffserv 영역에서 PHB의 분류의 역할을 수행한다. IPv4 헤더의 8 비트의 TOS 필드중 6 비트를 사용하여 각 PHB를 구분하며 본 논문에서는 EF, AF, DE PHB 를 구분 가능하도록 하였고, 각 정의에 따라 확장 가능한 구조이다.

EF Meter/Marker는 토큰 발생률과 큐에 저장된 데이터의 크기로 쉽게 구현될 수 있다. 토큰을 획득한 패킷에는 EF PHB로 표시하여 내보내고, 그렇지 못한 경우에는 폐기 알고리즘에 따라 폐기하거나 수정하여 표시하게 된다. 본 논문에서는 토큰 미터를 이용한 EF Meter/Marker를 구현하였으며, 토큰을 획득하지 못한 패킷은 폐기하도록 하였다. 그림 16은 EF Meter/Mark-er의 구성도이다.

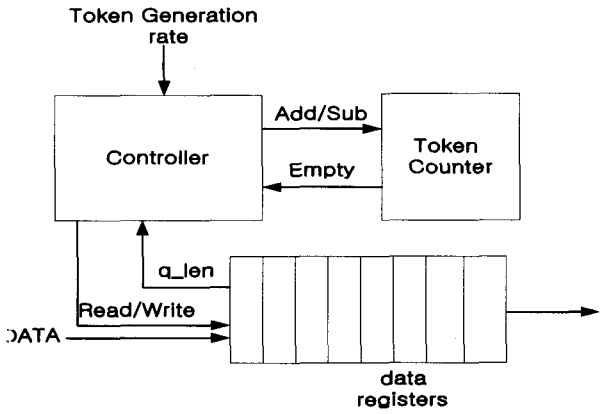



그림 16. EF Meter/Marker의 구성도
Fig. 16. The block diagram of EF Meter/Marker.

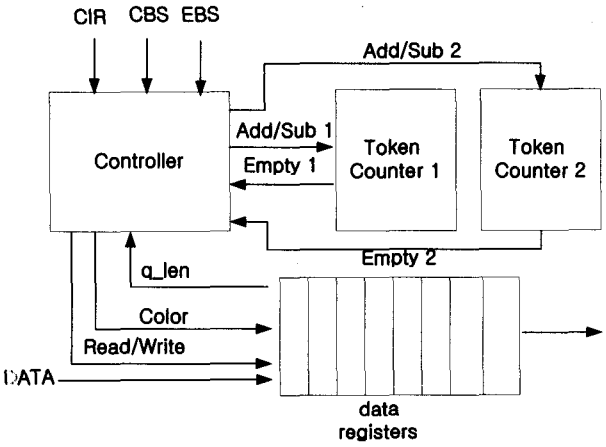



그림 17. AF Meter/Marker의 구성도
Fig. 17. The block diagram of AF Meter/Marker.

제어기에서는 파라미터로 토큰 생성률을 입력받고, FIFO로부터 큐에 저장된 데이들의 수, 그리고 토큰 카운터로부터 토큰의 현재 상황을 입력받는다. 그리고 입력 변수들로부터 토큰을 제어할 수 있는 제어신호를 보내고, FIFO 읽기와 쓰기를 제어한다.

AF Meter/Marker를 구현하기 위해서 토큰 미터와 TCM을 사용하였다. 본 논문에서는 AF1과 AF2 PHB 처리를 위한 Meter/Marker를 구현하였다. 모듈의 입력으로는 CIR, CBS, EBS 등의 파라미터와 패킷 데이터가 사용된다. 두 개의 토큰이 사용되며, 각 파라미터가 토큰의 토큰률을 결정한다.

그림 17은 AF Meter/Marker의 구성도이다. 제어기에서는 각 파라미터와 FIFO의 길이, 각 토큰 카운터의 상태를 입력받아, FIFO를 제어하고, 각 토큰 카운터에 제어 신호를 내보낸다. 또한 각 입력으로부터 폐기 확률을 표시하는 신호를 결정한다.

그림 18은 Diffserv 모듈의 layout 결과이다. 칩 사이즈는 5 mm×5 mm 이며, 삼성 STDA 90 라이브러리를

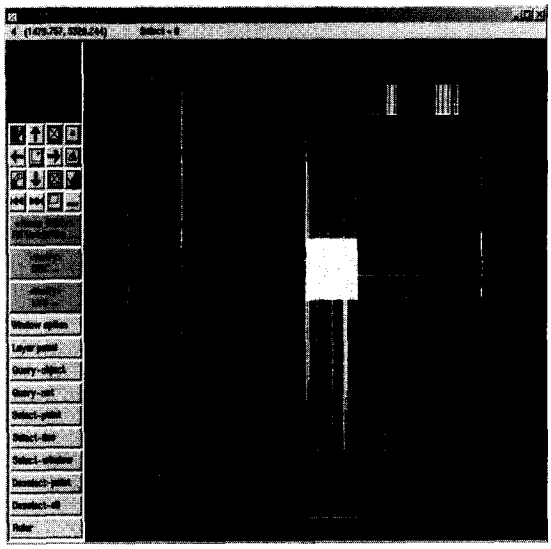


그림 18. Diffserv 모듈의 layout 결과.
Fig. 18. The layout result of Diffserv module.

이용하고, AVANTI 사의 Apollo tool을 사용하였다. 예상 소비 전력은 2.5 mW 이며, 예상 동작 주파수는 100 MHz 이다.

합성설계된 Diffserv 모듈은 분류기, EF Meter/Marker, AF Meter /Marker, DE Meter/ Marker와 멀티플렉서로 구성되어 있다. 분류기에서 분류된 각 패킷이 해당 PHB 처리 모듈에 입력되는 과정에서 멀티플렉서가 사용되고 있음을 알 수 있다. PHB 처리를 위해 EF Meter/Marker, AF Meter /Marker, DE Meter/Marker가 사용되었으며 각 PHB의 특성과 정의에 따라 처리될 수 있도록 구현하였다.

## VI. 결 론

본 논문에서는 MPLS에서 고속의 Diffserv를 지원하기 위해 고속 Diffserv 모듈을 구현하고 PHB 처리에 따른 자원 공유와 하드웨어 구현에 용이한 구조로 VOQ-PHB 구조를 제안하고 구현하였다. VOQ-PHB 구조는 VOQ 각각의 큐를 PHB 종류에 따라 확장한 구조로서, VOQ 각각의 큐에 가상의 PHB별 큐를 두고 있다. 또한 스케줄링 알고리즘으로 Priority-iSLIP 알고리즘을 사용하여, 고속 구현이 용이하면서 적절히 QoS를 보장할 수 있는 구조이다. 고속 라우터에서 멀티미디어 데이터를 처리하기 위해 패킷 처리와 경로 선택 알고리즘을 하드웨어로 구현하고 있으며, 이에 따라, 패킷 처리의 한 부분에 속하는 트래픽 조절기의 하드웨어 구현이 요구되고 있다. Diffserv 모듈은 각 PHB를 처리할

수 있도록 분류기, EF Meter/Marker, AF Meter /Marker와 DE Meter/Marker로 구성되어 있으며, 이들을 하드웨어로 구현하였다.

본 논문에서는 시뮬레이션을 통해 기존의 방식들과 제안하는 방식의 성능을 비교하기 위해서 망의 대역폭을 변화시키면서 평균 패킷 손실률과 지연 시간을 측정하였다. 그 결과, 기존의 방법들에 비해 제안한 방식에서 패킷 손실률이 증가하고 있으나, PHB의 우선 순위와 특성에 따라 패킷들을 전송하고 있음을 보였고, 평균 지연 시간은 개선 효과를 보이고 있어, 자원 공유 측면과 고속 스위칭에서 개선된 결과를 보였다.

본 논문에서는 이러한 시뮬레이션 결과를 토대로 하드웨어를 구현하였으며, 이는 Diffserv 모듈과 VOQ -PHB 구조를 갖는 MPLS 스위치 모듈로 나눌 수 있다. VHDL를 이용하여 각 모듈을 구현하였으며, RTL 기술 방법을 사용하였다. 이의 검증을 위해 논리 합성 후, 시뮬레이션하여 그 결과를 통해 기능을 검증하였고, 각 모듈을 실제 칩 라이브러리를 이용하여 구현하였으며, 이의 성능이 고속 라우터의 패킷 처리와 고속 스위칭에 적용될 수 있음을 보였다.

## 참 고 문 헌

[1] Pankaj Gupta and Nick McKeown, "Design and Implementation of a Fast Crossbar Scheduler", Hot Interconnects VI, Standford University, August 1998

[2] S. Sahu, D. Towsley, J. Kurose, "A Quanti tative Study of Differentiated Services for Internet" , CMPSICI Technical Report 99-09

[3] Francois Le Faucheur, Liwen Wu, "MPLS Support of Differentiated Services", IETF Internet Draft

[4] M. Carlson, W. Weiss, S.Blake, Z. Wang, D. Black, E. Davies, "An Architecture for Differentiated services", IETF RFC 2475, Dec. 1998

[5] Tae-Won Lee, Young-Chul Kim, "Implementation of a MPLS Router Supporting Diffserv for QoS and High-speed Switching", HSNMC2002, P51-55, July, 2002

## 저 자 소 개

**이 태 원**(정회원)
제38권 TC편 제8호 참조
현재 LG전자 선임연구원

**김 영 철**(정회원)
제38권 TC편 제8호 참조
현재 전남대학교 전자컴퓨터정보통신공학부 교수