IXP1200 네트워크 프로세서를 이용한 IPv4 라우터의 구현

정영환° 박우진 황광섭 배국동 안순신
고려대학교 전자 공학과
{youngh°, progress, hanriver, gooibil}@dsys.korea.ac.kr

The Implementation of the IPv4 Router on IXP1200 Network Processor

Young-hwan Jung° Woojin Park Kwang-seop Hwang Kuk-dong Bae Sun-shin An
Computer Network Lab. Dept. of Electronics Eng., Korea University

요 약

인터넷의 급격한 성장으로 요구되는 고속의 데이터 처리 능력과 시장의 급격한 변화에 빠르게 대응하기 위하여 기존의 범용 프로세서를 사용한 방법과 주문형 반도체를 이용한 네트워크 라우터/스위치 시스템의 단점을 보완하고, 두 방식의 장점만을 취합한 네트워크 프로세서가 개발 되었다. 네트워크 프로세서는 네트워크 관련 기능에 특화된 구조를 채택하면서 프로그램이 가능하여 고속의 데이터 처리와 동시에 다양한 응용 프로그램의 개발을 가능하게 한다. 본 논문에서는 인텔사의 IXP1200 네트워크 프로세서를 이용하여 IPv4 라우터를 구현하여 네트워크 프로세서가 가지는 특징을 평가해 본다.

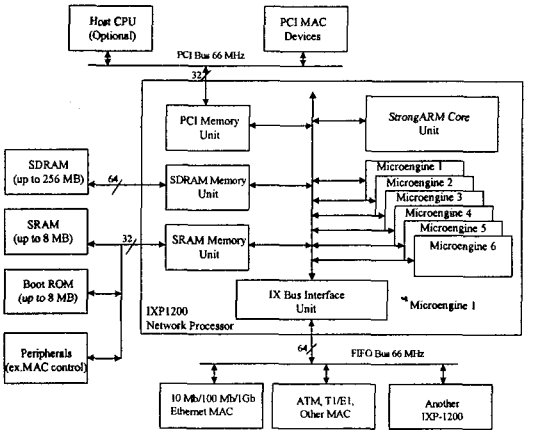
1. 서 론

1990년대까지 대부분의 네트워크 라우터/스위치 시스템은 일반적인 범용 프로세서를 사용한 구조를 가지고 있었다. 이 범용 프로세서 기반 방식은 무어의 법칙에 의해 증가하는 마이크로 프로세서의 처리 속도를 훨씬 뛰어넘는 속도로 급격히 증가하는 전송 속도 요구량과 다양한 서비스를 지원하기에는 한계를 가지고 있었다. 이에 따라, 1990년대 이후에는 주문형 반도체(ASIC)를 이용한 방식이 주류를 이루게 되었다. 주문형 반도체 기반 방식은 사용 용도에 최적화되어 빠른 데이터처리 속도의 장점을 최대한 이용하여, 초고속 라우터/스위치 시스템의 등장을 가능하게 하였다. 그러나, 주문형 반도체는 시장 진입까지의 기간이 길다는 단점을 가지고 있다. 만약, 제품상에 문제점이 발견되는 경우에는, 추가의 기간이 더 소요되게 되며, 이미 출시된 제품에 대해서는 기능, 성능에 변화를 줄 수 없다는 단점 역시 주문형 반도체 기반 방식의 큰 문제로 작용하였다. 네트워크 프로세서는 이전의 두 방식의 장점만을 취합하여, 고속의 데이터 처리 능력과 응용성의 두 가지 목표를 동시에 이루는데 목적을 두고 있다. 네트워크 프로세서는 네트워크 관련 기능을 수행하기 위해 특별히 설계된 구조를 가지고 있으면서 프로그래밍이 가능하기 때문에 범용 프로세서의 응용성과 주문형 반도체의 고속의 데이터 처리능력을 가지는 라우터/스위치 시스템의 개발을 가능하게 한다. 본 논문에서는 인텔사의 IXP1200 네트워크 프로세서를 이용하여 Forwarding Engine, 라우팅 프로토콜, 라우터 관리 모듈을 개발하여 기본적인 IPv4 라우터를 구현해 보고 네트워크 프로세서가 가지는 장, 단점등을 평가해 본다.

2. 관련 연구

2.1 IXP1200 네트워크 프로세서의 구조

인텔사에 의해 개발된 IXP1200 네트워크 프로세서는 6개의 32비트 RISC Programmable MicroEngine과 StrongARM Core Processor를 탑재하고 있다. 6개의 마이크로 엔진은 기본적인 패킷 포워딩 기능을 수행하며, Layer 3에서 1초에 3백만 개의 64바이트 패킷을 포워딩 시킬 수 있다(약 1.5Gbps). StrongARM Core는 포워딩 테이블의 유지 및 보수, 망 관리, OSPF, RIP등의 Protocol 처리 등 더욱 복잡한 작업을 수행하게 된다.



[그림 1] IXP1200 네트워크 프로세서의 블록 다이어그램

2.2 IPv4 라우터의 요구 조건

RFC1812에 정의되어 있는 기본적인 IPv4 라우터가 가지고 있어야할 기능들은 다음 [표 1]과 같다. 본 연구에서는 표에 나타나있는 기능들 중 필수적인 기능들을 구현하였다.

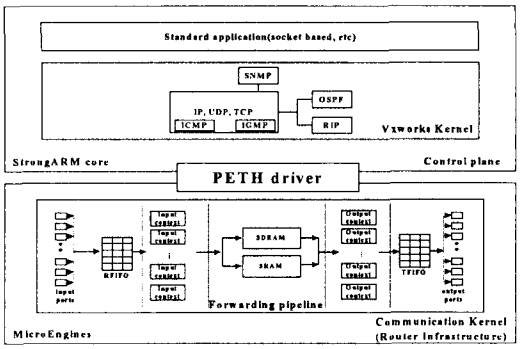
Layer	Function	Implementation
Link Layer	Trailer Encapsulation	Opt
	Header Validation and Modification	M
	MTU Verification	M
	ARP Filtering	M
	Ethernet and 802.3 coexistence	M
	PPP	Opt
	Forwarding of Link Layer Broadcast	M
Internet Layer	IP Header Verification	M
	Local Delivery Decision	M
	Next Hop Address Decision	M
	Fragmentation and Reassembly	M
	ICMP	M
	IGMP	Opt

	Source Address Validation	Opt
	Martian Address Filtering	Opt
	Forwarding of Internet Layer Broadcast	Opt
	IP Option Filtering	M
	TTL Validation	M
	Multicast Routing	Opt
	Congestion Control	Opt
	TOS	Opt
	IP Precedence	Opt
Transport Layer	TCP	M
	UDP	M
Application Layer	OSPF	M
	BGP	Opt
	Static Routing	Opt
	Filtering of Routing Information	M

[표 1] IPv4 라우터의 요구 조건

3. 설계 및 구현

IXP1200을 이용한 IPv4 라우터의 전체적인 구조는 [그림 2]와 같다. Microengine을 이용하는 Communication Kernel은 기본적인 Packet Forwarding 기능(IP헤더 검사, 수정, 라우팅 테이블 lookup등)을 수행하며, Communication Kernel에서 수행할 수 없는 복잡한 연산이 필요한 예외 패킷들(ICMP, IP option, ARP, 라우팅 패킷 등)은 PETH드라이버를 통하여 Control Plane으로 전달되어 VxWork Kernel상에서 StrongArm Core가 수행하게 된다.



[그림 2] IXP1200을 이용한 IPv4 라우터의 구조

3.1 Communication Kernel의 구현

Communication Kernel의 Forwarding Engine은 Receive Thread, Transmit Scheduler, Transmit Thread 이 세 가지의 모듈로 구성되어진다.

- Receive Module

입력 포트의 receive ready flag를 검사함으로써 새로운 패킷이 도착한 포트를 찾고, 어느 Receive Thread가 그 패킷을 처리할 것인지를 결정하여 기본적인 IP 패킷에 대한 처리를 수행하고, 목적지 주소를 보고 SRAM속에 저장된 라우팅 테이블을 참조해서 출력 포트를 찾는다. 이런 메모리를 참조하는 연산은 긴 지연이 생기기 때문에 이런 지연을 보상하기위해 Multithreading기법을 사용한다. 기본적으로 IXP1200 Network Processor는 Context Switching을 통해서 Multithreading을 지원한다.

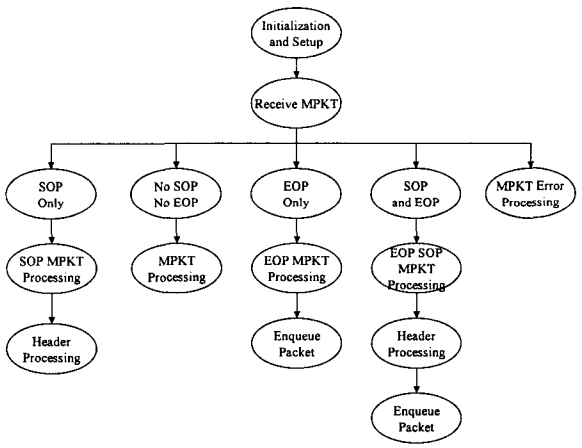
-Transmit Scheduler Module

Transmit Scheduler 모듈은 출력포트로의 패킷 간의 전송 순서를 결정하고, Transmit Module에게 그 순서를 알려준다. QoS를 제공하기 위해서는 효과적인 패킷 전송에 대한 스케줄링 기술이 요구된다.

-Transmit Module

Transmit 모듈은 Transmit Scheduler에게서 받은 메시지를 읽고, 출력 포트로 패킷을 전송한다. 이때, 패킷은 SDRAM에서 TFIFO로 이동하게 되고, 이 과정은 하드웨어에 의해서 자동으로 이루어진다. 여기서, 주의해야 할 것은 TFIFO를 circular queue구조로 되어있고 순차적으로 처리가 되기 때문에 순서를 잘 맞춰주어야 한다는 것이다.

[그림 3]은 수신측 즉, Receive Thread 모듈의 계략적인 동작흐름을 보여준다. 수신용으로 할당된 4개의 마이크로 엔진의 모든 쓰레드는 그림에는 보여지는 모든 기능을 수행한다.



[그림 3]Receive Processing 모듈의 동작흐름

각 MAC 패킷의 형태에 따른 수행과정은 다음과 같다.

- SOP MAC 패킷의 처리 (SOP only)

-RFIFO로부터 SDRAM의 패킷 버퍼로 64바이트중 하위 32바이트를 기록한다.

-출력을 위한 큐잉을 제외하고 2계층과 3계층에 대한 처리를 수행한다.

-수정된 이더넷과 IP헤더(상위 32바이트)를 SDRAM의 패킷 버퍼에 기록한다.

- MAC 패킷의 처리 (No SOP and No EOP)

-RFIFO로부터 SDRAM의 패킷 버퍼로 64바이트를 그대로 옮긴다.

- EOP MAC 패킷의 처리 (EOP only)

-EOP MAC 패킷으로부터 타당한 바이트 수를 얻어낸다.

-MAC 패킷의 타당한 바이트를 SDRAM Transfer 레지스터로부터 SDRAM의 패킷 버퍼로 쓴다.

- SOP이면서 EOP인 MAC 패킷의 처리 (SOP and EOP)

-RFIFO로부터 SDRAM의 패킷 버퍼로 64바이트중 하위 32바이

트를 쓴다.

-2계층과 3계층 처리를 수행한다.

-수정된 이더넷과 IP헤더(상위 32바이트)를 SDRAM의 패킷 버퍼 버퍼에 기록한다.

Transmit Thread가 패킷을 전송하는 단계는 다음의 순서로 이루어진다.

-Transmit Thread는 Task Assignment를 선택하기 위해서 Task Assignment Mailbox들을 읽는다. 만약, Assignment Valid Bit이 Set되어 있다면, Global Message ID를 증가시키고, 다음 Transmit Thread에게 신호를 준다.

-Transmit Thread는 SRAM에서 Packet Descriptor를 읽고, SDRAM으로부터 Transmit FIFO까지 Packet Data를 전송한다.

-Transmit Thread가Packet Queue Linked List를 Update하고, 사용된 Descriptor를 Free했을 때, FBI에게 상태를 기록하고 나서 다시 다음 할당으로 Loop Back한다.

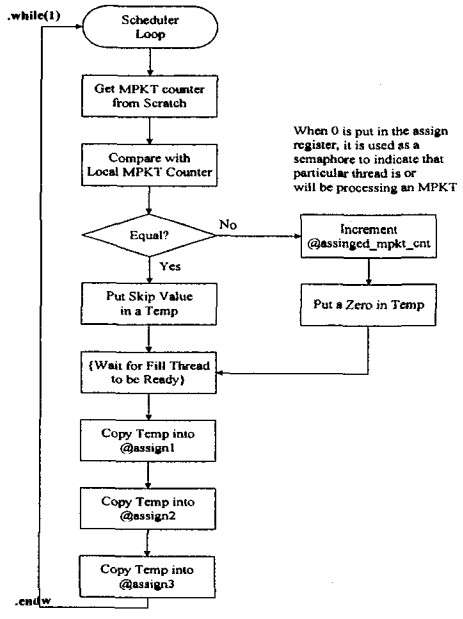
IX Bus로 데이터를 전송하는 단계는 다음과 같다.

-Transmit State Machine은 XMIT_PTR 레지스터에서 나타나는 현재 Element의 Valid Flag를 확인하다.

-만약 그 Valid Flag가 Set되어 있다면, Transmit State Machine는 Control Field를 읽고, Prepend Field에서의 데이터를 먼저 전송하고 나서 Data Field에서의 데이터를 전송한다. 만약, 그 Valid Flag가 Clear되어 있다면, Transmit State Machine은 Valid Flag가 Set될 때까지 그 Element를 기다린다.

-IX Bus를 통한 Transmit FIFO Element의 전송 시작에서 Transmit Pointer는 1만큼 증가된다.

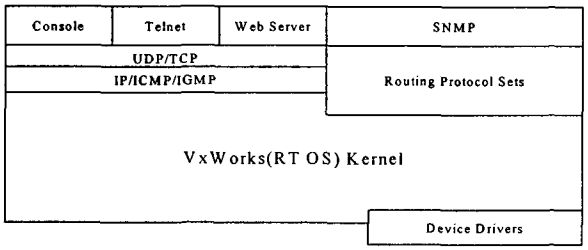
[그림 4]는 Transmit Thread의 개괄적인 Flowchart를 보여준다.



[그림 4]Transmit Thread의 개괄적인 Flowchart

3.2 Control Plane의 구현

[그림 5]는 본 연구에서의 IPv4 라우터 개발을 위한 Control Plane의 소프트웨어의 전체 구조를 나타낸다. Device Drivers 모듈은 OS와 하드웨어를 연결해주는 역할을 한다. IP/ICMP/IGMP 모듈과 UDP/TCP 모듈은 VxWorks kernel에서 기본적으로 제공해주는 모듈로 라우팅 프로토콜 모듈의 개발 환경을 제공해 준다. Console, Telnet, Web Server module, SNMP는 라우터 관리자가 라우터를 관리할 수 있도록 해주는 관리 툴 모듈이다. 본 연구에서는 VxWorks Kernel에서 기본적으로 제공되어지지 않는 OSPF, BGP 라우팅 프로토콜과, SNMP 망관리 프로토콜을 기존 리눅스기반 소스를 VxWorks상으로 포팅하여 IPv4 라우터의 요구사항을 충족 시켰다.



[그림 5] 네트워크 프로토콜 구조

4. 결론 및 향후 연구

본 논문에서는 인텔사의 IXP1200 네트워크 프로세서를 이용하여 Communication Kernel 및 Control Plane의 여러 가지 프로토콜을 구현하여 기본적인 IPv4 라우터를 구현해보았다. 네트워크 프로세서가 가지는 프로그래밍이 가능한 장점을 살리면서 Line Speed로 Forwarding할 수 있는 데이터 처리 능력을 보여 주었다. 성능 측정의 결과를 보면, Forwarding Engine은 2.42Mpps (1.24Gbps)의 속도로 패킷을 처리할 수 있으며, 이는 이론적 최대 전송속도인 2.67Mpps (1.37Gbps)의 90.51%이다. 이는 SDRAM의 스피드가 충분히 빠르지 못해 병목구간이 생기기 때문으로 분석된다. 특히 각종 프로토콜과 함께 구동을 하게 되면 PETH 드라이버 또한 SDRAM에 접근하기 때문에 성능 하락이 조금 더 생기게 되어 SDRAM BUS의 보완이 필요해 보인다.

향후에는 네트워크 프로세서의 programmability를 이용하여 IPv6와 MobileIP, QoS를 지원하기 위한 DiffServ를 구현하여 차세대 네트워크 장비의 핵심 기술인 네트워크 프로세서의 활용 기술을 연구할 계획이다.

5. 참고 문헌

[1] Intel "IXP1200 Network Processor Programmer's Reference", June, 2001

[2] Intel "IXP1200 Network Processor Hardware Reference Manual", June, 2001

[3] Intel "IXP1200 Network Processor Software Reference Manual", June, 2001

[4] F. Baker, Cisco Systems, "Requirements for IP Version 4 Routers", RFC1812, June 1995

[5] Woojin Park, "Enhancement of a Layer 3 IP Forwarding Function on the IXP1200", Korea University, June, 2002