Journal of The Institute of Electronics and Information Engineers Vol.52, NO.8, August 2015 http://dx.doi.org/10.5573/ieie.2015.52.8.074
ISSN 2287-5026(Print) / ISSN 2288-159X(Online)

논문 2015-52-8-9

PCI Express 기반 시스템 인터커넥트의 설계 및 구현

(Design and Implementation of an Alternate System Interconnect based on PCI Express)

김 영 우*, 런 예*, 최 원 혁*

(Young Woo Kim©, Ye Ren, and WonHyuk Choi)

요 약

PCI Express는 프로세서와 시스템의 IO 장치들을 연결하기 위하여 널리 사용되는 업계 표준이다. PCI Express는 이전 PCI 표준에서 유래하며, 전통적으로 하나의 PC 혹은 서버 내에서 사용되어져 왔다. PCI Express의 고속, 저전력, 고효율 특성은 기존 시스템 연결망과는 다른 형태의 대안 연결망으로써 고려되고 있다. 본 논문에서는 이와 같은 PCI Express를 이용한 시스템 연결망(PCIeLINK)을 설계, 구현하고 초기 시험 결과를 제시한다. 본 논문에서는 PCI Express를 이용한 fail-over 시스템에 자주 사용되는 non-transparent bridging(NTB) 기법을 이용하여 PCI Express 기반 시스템 연결망을 설계, 구현 하였다. NTB는 PCI Express 장치를 단순 연결할 경우 발생되는 전기적, 논리적 충돌을 방지하는 기법으로써, PCI Express Gen2 규격에 기반한 20 Gbps급의 x4 연결을 하나의 카드에 복수개 구현하고 이를 시험하였다. 개발된 PCI Express 기반 시스템 인터커넥트 장치는 최대 8.6 Gbps의 단방향 성능을 보였으며, Linux 기반의 TCP/IP 환경에서 최대 5.1 Gbps의 성능을 나타내는 것으로 측정 되었다.

Abstract

PCI Express is a well-known and widely used de-facto system bus standard for connecting among a processor and IO devices. PCI Express is originated from old PCI standard, and its most of applications are limited to be used within a PC or server system. But, because of its fast speed, low power consumption, and good protocol efficiency, it is considered as one of a good candidate for an alternate system interconnect for many years. In this paper, we present design, implementation and early evaluation of an alternate system interconnect by utilizing PCI Express. The developed alternate system interconnect using PCI Express (named PCIeLINK) utilizes non-transparent bridging (NTB) technic which generally used in fail-over system in PCI and PCI Express. By using NTB technic, PCI Express device can be extended to outside of a system without electrical and logical problems arising during system boot and enumeration. To build up an alternate system interconnect, we designed and implemented a network interface card having multiple PCI Express x4 connections (theoretically 20 Gbps) and tested, The early test results revealed that an x4 port in the card showed 8.6 Gbps peak performance for bulk transmission and 5.1 Gbps peak for normal TCP/IP transfer.

Keywords : PCI Express, Interconnect, Non Transparent, PCIeLINK

* 정회원, 한국전자통신연구원 클라우드컴퓨팅연구부
(Cloud Computing Research Department, ETRI)
© Corresponding Author (E-mail: bartmann@etri.re.kr)
※ 이 논문은 2015년도 정부(미래창조과학부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.B0101-15-0104 , 유전체 분석용 슈퍼컴퓨팅 시스템 개발)
Received ; June 1, 2015 Revised ; July 9, 2015
Accepted ; July 27, 2015

Ⅰ. 서 론

무어의 법칙에 기반 한 지난 수십 여 년 간 반도체 공정, 설계 및 제작 기술의 발전에 따라, 컴퓨팅 시스템의 성능은 비약적인 향상과 더불어 점차 고도화, 복잡화되고 있다. 컴퓨팅 시스템 성능의 핵심인 마이크로프

로세서는 무어의 법칙으로 대표되는 반도체 기술의 발전에 따라, 약 2년에 2배의 속도로 지속적인 성능향상을 보이고 있다. 최근 고집적에 따른 발열문제로 인하여 마이크로프로세서는 이전의 고성능 단일 프로세서 구조에서 멀티/매니코어로 발전하고 있는 추세이다[1].

고성능 컴퓨팅 시스템의 필수적인 요소 중 하나로서 시스템 인터커넥트 기술을 꼽을 수 있다. 슈퍼컴퓨터로 대표되는 고성능 시스템의 시스템 인터커넥트는 크게 인피니밴드(InfiniBand) 기술과 이더넷(Ethernet) 기술이 대다수를 차지하고 있으며[2], 이들 인터커넥트 기술은 1990년대 후반 초당 1 기가비트(Gigabit per second, Gbps) 성능을 돌파한 이후 현재 100 Gbps의 시대에 돌입하고 있다. 그러나 이와 같은 시스템 인터커넥트 기술의 발전은 마이크로프로세서의 발전 속도를 따라가지 못하고 있는 상황으로, 대규모 고성능 시스템에서 네트워크의 성능이 전체 시스템의 성능을 결정하는 중요한 변수의 하나로 부각되고 있다.

현재 일반 네트워크로서는 1/10 Gbps 이더넷이 널리 사용되고 있으며, 40/100 Gbps 이더넷에 대한 규격이 완료되었고, 향후 50/200/400 Gbps급의 규격으로 발전할 것으로 예상하고 있다[3]. 슈퍼컴퓨팅 시스템 및 고성능 컴퓨팅 시스템의 고속 연산 네트워크로 널리 사용되는 인피니밴드는 2001년 10 Gbps급(x4) SDR의 개발을 시작으로, 40/54.54Gbps급의 QDR/FDR이 널리 사용되고 있으며, 2014년 100Gbps급의 EDR 기술이 발표 되었다. 인피니밴드는 2017년 200 Gbps급의 기술 개발을 목표로 하고 있으며, 고성능 시스템 네트워크 기술에서 이더넷 기술과 경합 중에 있다[4].

이와 같은 시스템 인터커넥트 기술과 더불어, 서버 시스템 내의 IO를 위한 시스템 버스 기술 또한 지속적으로 발전 되고 있다. 대표적으로 프로세서와 IO 장치 간의 연결을 위한 시스템 버스인 PCI Express 규격은 2002년 단일 레인 기준 2.5 Gbps(Gen1)를 시작으로, 현재 2010년 발표된 8 Gbps급의 성능(Gen3)을 가지는 규격이 널리 사용되고 있다. 3 세대 PCI Express 규격인 Gen3 규격의 4개 레인을 엮을 경우(x4) 32 Gbps, 16개의 레인을 엮을 경우(x16) 128 Gbps의 성능을 가지며, 이는 이더넷 보다 빠르며, 인피니밴드 EDR에 준하는 성능을 나타낸다. PCI Express는 단일 레인에서 16 Gbps 성능을 가지는 4 세대(Gen4)규격의 2015년 발표를 목표로 지속적인 성능향상을 추구하고 있다.

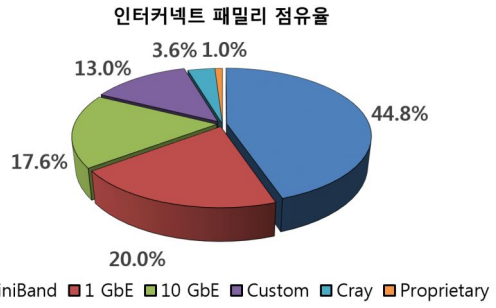


그림 1. 슈퍼컴퓨터 인터커넥트 분포[2]
Fig. 1. Interconnect Family Share[2].

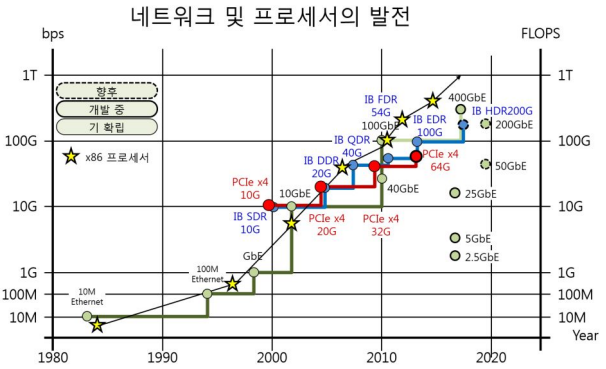


그림 2. 네트워크 기술 및 프로세서 기술 발전
Fig. 2. The evolution of network technology and processor performance.

그림 2는 이와 같은 시스템 인터커넥트와 PCI Express, 마이크로프로세서의 성능을 비교한 도표이다[3~6]. 최근 십 수 년 간 이와 같은 PCI Express의 고속도, 저전력, 고효율 프로토콜 등의 특성을 이용하여, PCI Express를 이용한 기술 개발과 시스템 인터커넥트/네트워크로 사용하고자 하는 다양한 시도가 이루어지고 있다[7~13].

본 논문에서는 PCI Express 기술을 사용하여, 이를 고성능 시스템 연결망으로 확장하기 위한 하드웨어 및 소프트웨어의 설계 및 구현에 대하여 기술한다. 구현된 기술은 유전체 분석을 위한 고성능 컴퓨팅 시스템 개발의 일환으로 개발되고 있는 MAHA 슈퍼컴퓨팅 시스템의 로컬 네트워크로 사용하는 것을 목표로 하고 있다[14~15].

본 논문의 제 Ⅱ장에서는 PCI Express의 일반적인 기술 개요와 특징을 기술하며, 제 Ⅲ장에서 PCI Express 기술의 시스템 네트워크 적용을 위한 네트워크 장치의 설계와 구현을 설명한다. 제 Ⅳ장에서는 개발한 PCI Express 기반의 시스템 네트워크 장치의 실

험 결과를 기술하며, 마지막으로 제 V장에서 결론 및 향후 연구 방향을 제시한다.

Ⅱ. PCI Express 기술

PCI Express(PCIe) 기술은 단일 서버 혹은 PC 시스템에서 프로세서와 IO 장치간의 연결을 위한 IO 버스 기술에서 유래한다. 시스템 IO 버스는 초장기의 ISA(Industry Standard Architecture), VESA(Video Electronics Standards Association), AGP(Accelerated Graphics Port)와 같은 다양한 IO 버스 기술 들이 점차 PCI(Peripheral Component Interconnect)로 통합되었으며(그림 3 참조), 시스템 및 IO 장치의 고속화에 따른 병렬 버스의 한계(용량성 부하 증가로 인한 성능저하, 버스 선폭의 증가에 따른 보드 크기 증가 등)로 인하여 점차 직렬 점 대 점 연결 방식으로 변화 하였다. 다양한 고속 직렬 점 대 점 연결 방식의 패킷 기반 시스템 버스 기술 중 가장 대표적인 기술이 2002년 등장한 PCI Express 기술 이다.

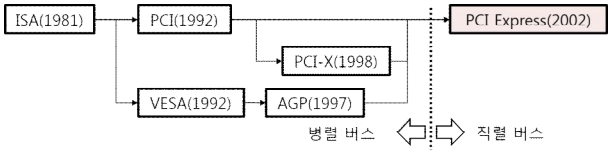
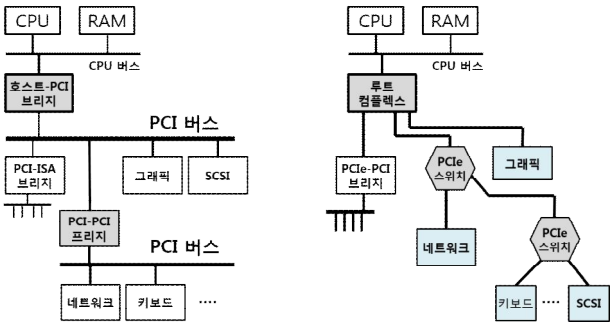


그림 3. 시스템 IO 버스 기술의 발전
Fig. 3. The evolution of system IO bus.

1. PCI Express

PCI Express는 전술한 바와 같이 병렬 버스 기술의 성능 한계를 극복하기 위하여 개발된 기술로써, 병렬-직렬(Serializer/Deserializer, SerDes) 변환을 통한 고속 신호 전송 기법에 기반 한 직렬 버스 기술이다. PCI Express는 이전 세대 기술인 PCI와 소프트웨어적인 호환성을 유지하도록 개발된 기술 표준으로, 이에 따라 시스템 내에서의 하드웨어적인 구조 또한 유사성을 가진다. 다만 패킷 기반의 직렬 버스 기술을 적용함으로써 개별 포트 당 1개의 연결을 가진다는 점이 구조상의 차이점이다. 다음 그림 4는 PCI와 PCI Express의 시스템 상에서의 계층구조를 비교한 그림이다.

PCI Express는 레인(lane)이라는 직렬 쌍(serial pair) 신호에 기반 하여, 각 장치의 연결(link)에 최소 1 레인



(a) PCI 계층구조, (b) PCI Express 계층구조

그림 4. PCI 및 PCI Express 계층구조 비교
Fig. 4. Comparison between PCI and PCI Express hierarchy.

표 1. PCI Express 세대별/레인별 성능[9, 16]
Table 1. Speed of PCI Express technology by generation and lane width[9, 16].

PCI Express 세대	링크 폭에 따른 성능						
	x1	x2	x4	**x8**	x12	**x16**	x32
PCI Express 1.0 (Gbps)	2.5	5	10	**20**	30	**60**	80
PCI Express 2.0 (Gbps)	5	10	20	**40**	60	**80**	160
PCI Express 3.0 (Gbps)	8	16	32	**64**	96	**128**	256

(x1)에서 최대 32 레인(x32)의 연결을 제공할 수 있다.

예로써 이더넷 혹은 인피니밴드와 같은 인터커넥트 장치는 x8 연결을 널리 사용하며, 고성능을 요하는 그래픽 장치의 경우 x16 연결이 널리 사용되고 있다. 2010년 8 Gbps의 PCI Express Gen3 규격의 경우, x16 레인은 단방향으로 128 Gbps의 최대 성능을 나타내며, 이와 같은 고속, 고성능 규격을 기반으로 현재 거의 모든 x86계열의 프로세서는 PCI Express 장치를 내장하여 시스템 내부 버스의 실질적인 표준이라 할 수 있다.

2. PCI Express와 시스템 인터커넥트

이번 절에서는 PCI Express와 여타 시스템 인터커넥트 기술과의 차이점, 시스템 인터커넥트로써 PCI Express 기술의 문제점 등을 간략히 기술한다.

가. PCI Express와 기타 인터커넥트 기술

PCI Express 기술은 다른 시스템 인터커넥트 기술과 비교하여 볼 때, 높은 프로토콜 효율, 동일 수준의 지연 성능, 적은 전력소모 및 낮은 단가 등의 우수한 특징을 가진다. PCI Express의 이와 같은 우수한 특징을 이용하여 이를 시스템 인터커넥트에 적용하고자 하는 연구

가 꾸준히 이루어지고 있다[10~13]. 다음의 표 2는 PCI Express의 기술적 특징을 다른 시스템 인터커넥트 기술과 비교한 표이다.

또한 PCI Express 기술을 시스템 인터커넥트로 활용할 경우, 일반적인 네트워크 장치에서 필요한 프로토콜 처리를 위한 네트워크 엔진이 불필요하게 되므로, 이로 인한 부가적인 네트워크 지연 감소 등과 함께 PCI Express 종단장치(endpoint)만으로 구성 가능한 최소한의 하드웨어로 시스템 연결망을 구현할 수 있는 장점이 있다(그림 5 참조).

나. PCI Express의 시스템 인터커넥트 적용 시 문제점

PCI Express 규격은 단일 서버 혹은 PC 시스템 내에서 프로세서와 IO 장치를 연결하는 것을 전제로 설계된 직렬 버스 시스템으로써, 직접적으로 시스템 인터커넥트에 적용할 경우, 여러 문제점이 발생한다[9, 19].

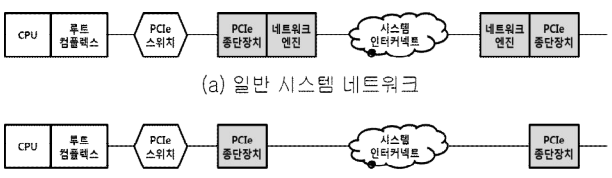
첫 번째 문제점으로 꼽을 수 있는 것이, 호스트 장치 간의 다른 클록 형상으로 인한 전기적인 충돌 문제이다. 일반적으로 PCI Express는 Spread Spectrum

표 2. PCI Express와 기타 연결망 기술 비교[9, 16]
Table 2. Comparison among PCI Express and other system interconnect technologies[9, 16].

	PCI Express	Ethernet	InfiniBand
프로토콜 효율*	93%	86 %	88 %
End-to-End 지연	~ 1μs	~10 μs to 30 μs	~ 1μs to 2 μs
가격/포트	HBA* < $100 Switch < $200	HBA ≈ $486 Switch ≈ $454	HBA ≈ $392 Switch ≈ $421
전력소모**	HBA ≈ 1.9W Switch ≈ 10.2W	HBA ≈ 5.5W Switch ≈ 36W	HBA ≈ 8.8W Switch ≈ 34W

* 256 바이트 페이로드 기준

** 16-SerDes/HBA, 96-SerDes/Switch 의 경우



(a) 일반 시스템 네트워크

(b) PCI Express 기반 시스템 네트워크

그림 5. 일반 시스템 네트워크와 PCI Express 기반 시스템 네트워크 구조
Fig. 5. Diagram for general system interconnect and PCI Express based system interconnect.

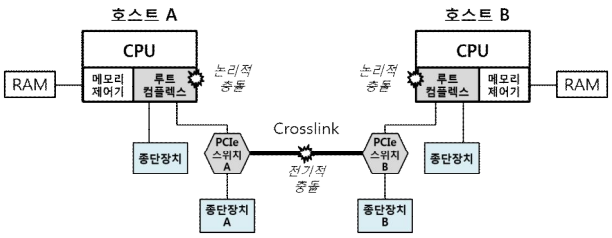


그림 6. PCIe Crosslink로 연결된 두 호스트 시스템[9, 19]
Fig. 6. Two host systems connected with PCI Express crosslink[9, 19].

Clock(SSC)의 사용을 권장하고 있으나, 서로 다른 호스트 간에 SSC를 이용할 경우, 클록 타이밍의 충돌로 인한 링크의 형성이 불가능해지는 문제점이 발생한다.

두 번째로는 시스템의 설정 시 발행하는 PCI Express 시스템의 계층 구조상 어드레스 및 ID 번호체계(Bus, Device, Function, BDF 번호)의 충돌 문제이다. 그림 6은 일반 PCI Express 스위치를 사용하여 두 호스트 시스템을 연결한 다이어그램이다. 시스템 부팅 시 PCI Express는 계층 구조상의 종단장치를 검색하고 이를 등록하게 된다. 이때, configuration cycle이라는 설정 동작을 수행하며, 이 설정 동작은 계층 구조상의 downstream(계층 구조상 위에서 아래)을 따라가며 계층 구조 및 장치 정보를 획득한다. 그림 6에서 호스트 A의 스위치 포트는 호스트 B와 연결되어 있는데, 서로 동일하게 다운스트림(downstream) 포트가 상호 연결됨으로 인하여, 규격 상 지원하지 않는 요청을 전송하게 되는 문제가 발생한다. 또한, 두 호스트 시스템이 동일한 구조를 가질 경우, 호스트 A쪽에 생성된 요청 신호에 대하여 호스트 B의 종단장치 B가 응답을 생성할 경우, 어느 호스트로 응답을 전달하여야 하는지 알 수 없는 문제가 발생한다. 즉, 루트 콤플렉스(Root Complex)의 BDF가 동일한 번호(일반적으로 0, 0, 0)를 가지게 되므로, 목적지가 동일한 2 곳으로 나타나게 되어 시스템의 오류가 발생하는 상황이다.

제 Ⅲ장에서는 이와 같은 문제점을 해결하며, PCI Express 장치를 시스템 인터커넥트에 적용하기 위한 설계 및 구현에 대하여 설명한다.

Ⅲ. PCI Express 시스템 네트워크 설계 및 구현

이번 장에서는 전술한 PCI Express 기술을 이용하여 시스템 인터커넥트를 구현하기 위한 설계 및 구현에 대

하여 논의한다.

1. PCI Express Non-Transparent Bridging

PCI Express 스위치의 downstream 연결을 직접 두 시스템 간에 연결할 경우, 전술한 바와 같은 여러 문제점이 발생한다. 따라서 시스템 인터커넥트 구현 시 전기적, 시스템적으로 두 시스템을 분리 하여야 하며 이를 가능케 하는 것이 Non-Transparent Bridging(NTB)이다[9, 19~20].

NTB는 이전 PCI 기반 시스템에서부터 사용되어 왔으며, 이전까지 시스템의 fail-over를 위하여 사용되었다. PCI Express 계층 구조상의 어드레스 및 번호체계 충돌의 문제는, configuration cycle의 응답 주체가 종단장치여야 하는데 두 시스템을 단순 연결 할 경우 두 시스템이 논리적으로 분리되어 있지 않음으로 인하여 발생한다. NTB는 하나의 포트 내부에 두 개의 PCI Express 종단장치를 내장하고, 이들 두 종단장치를 논리적으로 분리시켜, 두 시스템 간의 PCI Express 계층 구조 번호체계를 분리 시켜주는 구조를 가진다.

그림 7의 NTB는 두 개의 시스템 간의 연결 시 전기적인 상이점(SSC의 사용에 따른 클록킹 문제점)을 해결한다. 또한 동시에 두 시스템에게 각각 독립적인 논리적 종단장치를 제공함으로써 PCI Express 계층 구조상의 번호체계를 분리시키며, 동시에 메모리 어드레스 및 번호체계의 변환을 통하여 두 시스템간의 데이터 전송을 가능케 한다. NTB는 이전까지 fail-over를 위하여 사용되었으며, 이를 이용하여 시스템 연결망으로 활용하는 연구가 최근 이루어지고 있다[9~19].

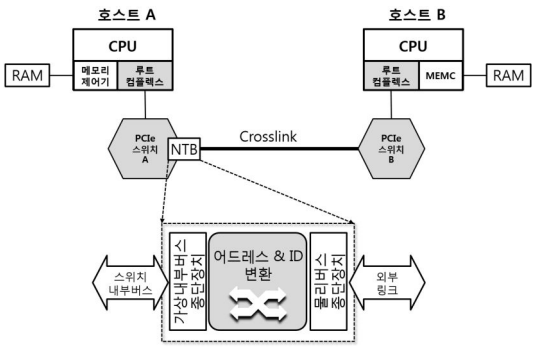


그림 7. PCI Express NTB 구조
Fig. 7. General structure of non-transparent bridging device in PCI Express switch.

2. PCI Express기반 시스템 인터커넥트 장치

이번 절에서는 PCI Express를 기반으로 하여 시스템 인터커넥트를 구현하기 위한 장치 설계 및 구현에 대하여 논의한다.

가. PCI Express 기반 시스템 인터커넥트 연결형상

NTB를 포함하는 PCI Express 스위치 장치는 호스트 CPU 쪽으로 연결되는 upstream 포트와 종단장치로 연결되는 복수개의 downstream 포트를 가지며, 내부적으로 가상의 내부 PCI 버스로 상호 연결되어 있다. NTB는 PCI Express 스위치 장치의 내부에 위치한 종단장치와 downstream 포트의 기능을 동시에 가진다. 내부적으로는 두 개의 PCI Express 종단장치가 상호 연결되어 하나의 포트를 구성하며, 내부의 개별 종단장치에는 PCI Express 어드레스 및 번호체계 변환을 위한 변환 테이블, IPC를 위한 도어벨 및 임시 메모리 등을 가진다(그림 8 참조).

그림에서 NTB는 Virtual side와 Link side로 분리된 연결을 가짐으로써, SSC와 같은 전기적인 상이점을 분리하여 두 시스템의 상호 연결 시 발생하는 문제점을 해결하는 구조이다.

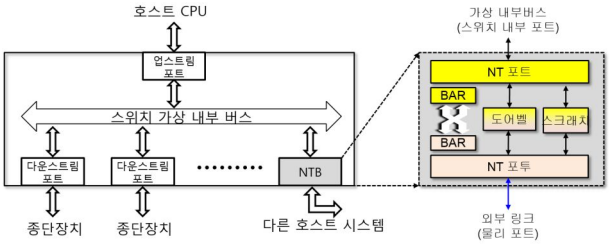


그림 8. PCI Express 스위치 및 NTB 내부 구조
Fig. 8. General structure of non-transparent bridging device in PCI Express switch.

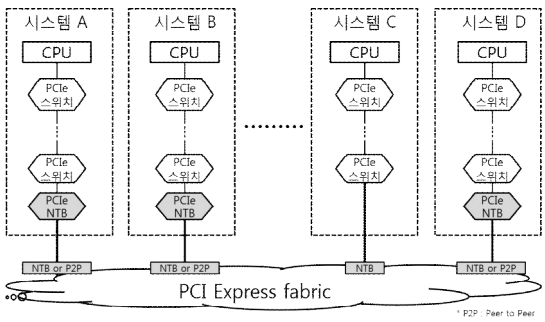


그림 9. PCI Express NTB 기반의 인터커넥트 형상
Fig. 9. Possible interconnect configuration using PCI Express NTB.

NTB기반의 네트워크 장치를 사용하여 시스템 인터커넥트를 구현할 경우, 그림 9와 같이 다양한 형태의 연결 형상의 구성이 가능하다.

나. PCI Express 기반 시스템 인터커넥트 장치

구현된 PCI Express 기반 시스템 인터커넥트 장치(PCI Express Link, PCIeLINK)는 PCI Express Gen2.1 규격에 기반 하여 설계하였다. PCIeLINK 하드웨어는 다음의 표 3과 같은 규격에 기반 하여 설계 및 구현 하였다.

하드웨어의 설계 및 구현에 사용된 PCI Express 스위치 장치는 복수개의 Non-Transparent(NT) 포트를 내장한 장치를 우선적으로 고려하여, 향후 다양한 네트워크 토폴로지에 대한 대응이 가능 하도록 설계하였다. PCIeLINK 하드웨어는 총 4개의 외부 연결을 위한 NT 포트와 호스트 시스템 연결을 위한 1개의 upstream을 지원하는 스위치 장치를 사용한다. 외부 연결은 1 레인 당 최대 10 Gbps의 전송이 가능한 QSFP+ 규격의 모듈을 사용하여 포트 당 x4 레인의 연결을 통한 최대 20 Gbps(단방향)의 전송이 가능하도록 설계하였다[21].

표 3. PCI Express 인터커넥트 장치 규격
Table 3. Hardware specifications of PCIeLINK.

구분	규격
프로토콜	PCI Express Base Specification Rev. 2.1
호스트 연결	PCI Express Gen2, x8 lane
외부 포트	PCI Express Gen2, x4 lane
외부 포트의 수	4
외부 포트 연결	QSFP+
카드 형상 규격	PCI Express CEM Rev. 2

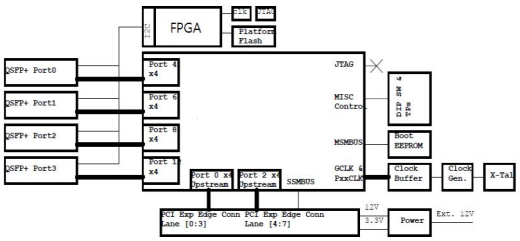


그림 10. PCIeLINK 하드웨어 장치의 구조도
Fig. 10. Block diagrams for PCIeLINK hardware.

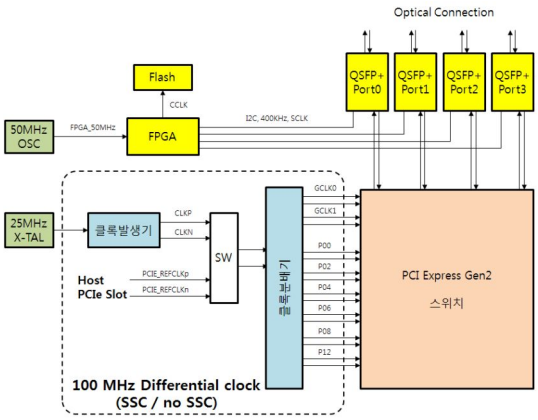


그림 11. PCIeLINK 하드웨어 클록 분배 및 연결
Fig. 11. Connection diagram of PCIeLINK hardware.

다. PCIeLINK 하드웨어 구현

PCIeLINK 하드웨어 장치는 IDT사의 PCI Express Gen2 스위치 장치인 89HPES24NT6AG2 스위치 칩을 사용하여 구현하였다[20, 22]. 89HPES24NT6AG2 스위치는 총 24 레인, 6 포트를 지원하며, 최대 6개의 x4 NT 포트 혹은 3개의 x8 NT 포트의 지원이 가능한 장치이다. PCIeLINK 하드웨어 장치를 위하여, 8 레인의 일반 P2P(Peer to Peer) 브리지 설정(upstream)을 사용하여 호스트 시스템에 연결하며, 4개의 x4 NT 포트를 통하여 외부 PCIeLINK 장치 혹은 PCI Express 스위치에 연결되도록 구현하였다[16 ~ 18].

8 레인의 x8 호스트 연결은 최대 40 Gbps(단방향)의 성능으로 호스트 시스템과의 데이터 전송을 수행할 수 있으며, x4 NT 포트를 통한 외부 연결은 각각 최대 20 Gbps의 데이터 전송이 가능하도록 구현하였다. PCIeLINK하드웨어 장치는 PCI Express full profile 카드 규격을 준수하여 구현하였다(그림 12, 그림 13).

2. PCI Express기반 시스템 인터커넥트 SW

가. PCIeLINK 소프트웨어 구조

PCI Express 기반의 시스템 인터커넥트를 위한 소프트웨어의 전반적인 구조는 다음의 그림 14와 같다.

PCIeLINK 소프트웨어는 크게, PCIeLINK 하드웨어 장치 드라이버 모듈, 네트워크 SW 스택, 네트워크 관리 모듈로 구성된다[17~18].

PCIeLINK 하드웨어 장치 드라이버는 PCIeLINK 하드웨어 장치의 초기화, 하드웨어 장치의 설정 및 변경,

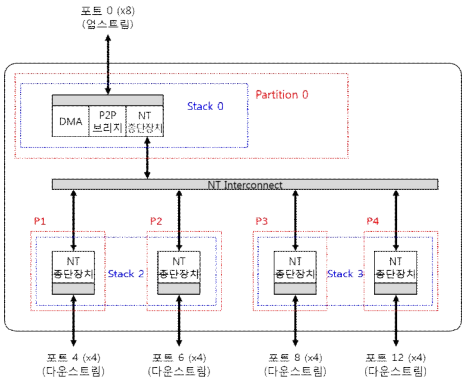


그림 12. PCIeLINK 스위치 칩 내부 설정 구조
Fig. 12. Configuration scheme of PCIeLINK switch.

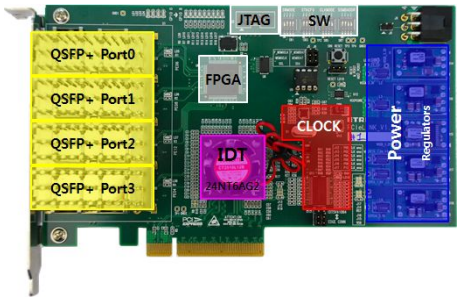


그림 13. PCIeLINK 하드웨어 장치
Fig. 13. PCIeLINK hardware.

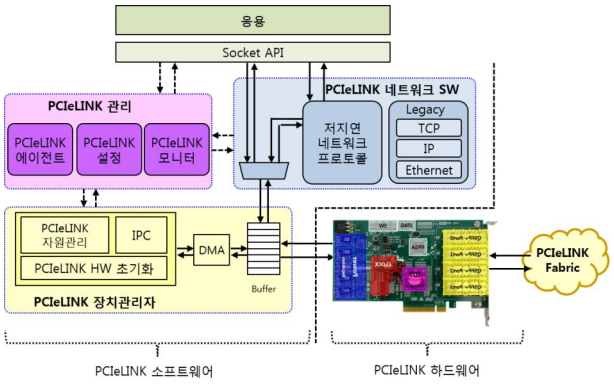


그림 14. PCIeLINK 소프트웨어 구조도
Fig. 14. Block diagram of PCIeLINK software.

DMA 및 데이터 송수신을 위한 버퍼관리 등을 수행한다. 네트워크 SW 스택은 기존 Linux TCP/IP 스택 지원을 위하여 가상 Ethernet 장치 인터페이스 모듈과 TCP/IP 대신사용 가능한 저지연 특성의 경량 네트워크 프로토콜로 구성된다. PCIeLINK 네트워크 관리 모듈은 PCIeLINK로 구성되는 인터커넥트 및 하드웨어 장치의 상태와 동작을 모니터링하기 위한 관리 모듈로 구성되어 있다.

나. PCIeLINK 장치 드라이버 및 메커니즘

PCIeLINK 소프트웨어는 일반 네트워크 통신의 지원을 위하여 장치 드라이버와 가상 Ethernet 장치 인터페이스 모듈을 기본으로 제공한다. PCIeLINK 장치 드라이버의 하드웨어 초기화 및 기타 관리를 제외한 핵심적인 역할은 IPC(Inter Processor Communication)를 통한 상대 시스템의 정보 획득 및 메모리 영역 설정 기능이다. 이를 통하여 PCIeLINK 소프트웨어 및 하드웨어는 로컬 PCI 어드레스를 액세스하여 로컬 PCI 도메인에 속하지 않은 상대 호스트의 메모리 영역에 대한 액세스를 가능케 한다(그림 15).

호스트의 TCP/IP 계층을 통하여 전달되는 TCP/IP 패킷은 가상 Ethernet 인터페이스를 통하여 기본 장치 드라이버로 전달되며, 기본 장치 드라이버는 전달된 TCP/IP 패킷을 PCI Express 패킷 payload로 encapsulation하여 원격 호스트로 DMA 전송한다.

기본적인 연결형상은 두 호스트 간의 back-to-back

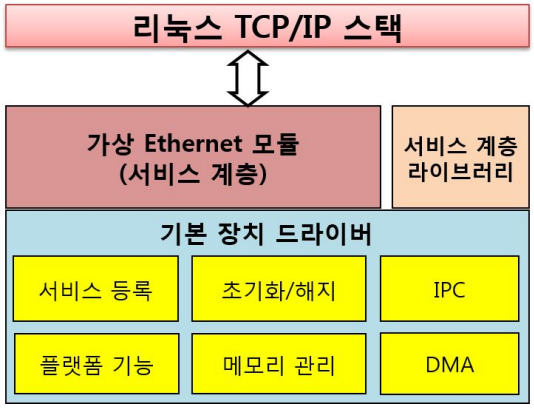


그림 15. PCIeLINK 장치 드라이버 기본 구조
Fig. 15. Basic structure of PCIeLINK Device Driver.

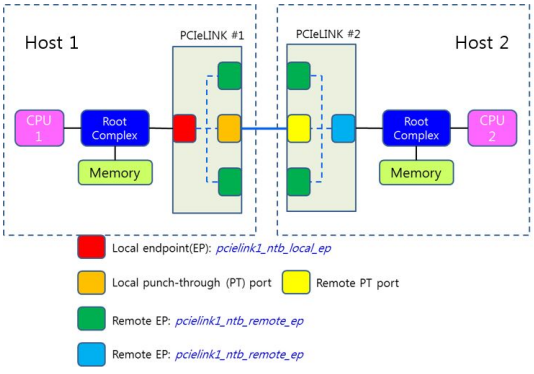


그림 16. Back-to-back 기본 연결 형상
Fig. 16. The implementation of basic back-to-back connection in PCIeLINK.

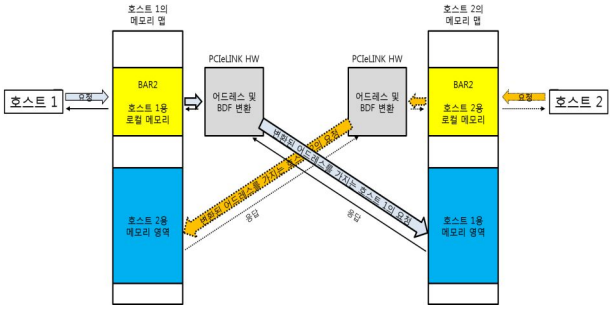


그림 17. PCIeLINK 어드레스 및 BDF 변환 메커니즘
Fig. 17. Address and ID translation mechanism in PCIeLINK system interconnect.

형상의 연결을 기본으로 한다(그림 16). 호스트 1에서 호스트 2로 PCI Express 패킷을 전달 할 경우, 호스트 1은 자신의 로컬영역에 등록되어 있는 로컬 엔드포인트로 패킷을 전달하고, 로컬 스위치 내부의 NT 엔드포인트(그림에서 local punch-through port, PT)를 통해 호스트 2로 전달한다. 호스트 2는 리모트 PT 포트를 통해 전달받은 패킷을 호스트 2의 로컬 엔드포인트(호스트 1에서 볼 때 리모트 엔드포인트)를 통하여 패킷을 전달받는 구조이다[17, 18].

이와 같은 과정에서 서로 다른 호스트의 PCI 도메인을 액세스하기 위한 메모리 어드레스 변환 및 ID(BDF 번호) 변환을 통해, 번호체계 충돌문제를 해결한다. 그림 17은 이와 같은 어드레스 및 ID 변환 메커니즘을 설명하는 개념적인 그림이다. 어드레스 및 ID는 각 NT 포트를 지날 때 변환된다.

다. 저지연 특성의 경량 네트워크 프로토콜 구현

PCI Express는 자체 프로토콜을 사용함으로 인하여, 일반 네트워크/인터커넥트 장치와 달리 네트워크 패킷을 위한 헤더, 데이터 처리 등을 위한 하드웨어 지원이 없다. 이로 인하여, TCP/IP와 같은 패킷을 PCI Express를 통하여 송수신 할 경우, TCP/IP 패킷은 PCI Express의 payload에 encapsulate 되며 TCP/IP 패킷의 생성과 처리는 순수하게 소프트웨어적으로 처리된다. 따라서 소프트웨어적인 오버헤드가 존재하게 되어 PCI Express 인터커넥트의 성능 손실이 예상된다. PCI Express 기반 인터커넥트의 성능향상과 CPU 처리의 부담을 경감하기 위하여 PCI Express 인터커넥트를 위한 저지연 경량 네트워크 프로토콜을 제안, 설계하였다[23].

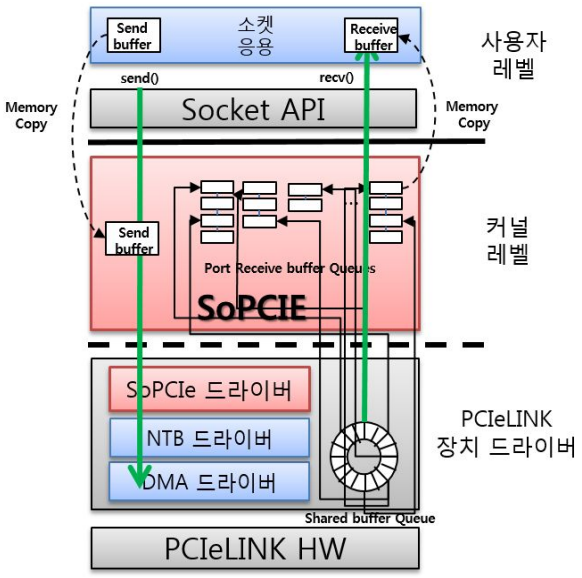


그림 18. SoPCIe 모듈의 데이터 전송 처리 구조
Fig. 18. Packet data processing diagram in SoPCIe.

제안한 경량 네트워크 프로토콜(Socket over PCIe, SoPCIe)은 리눅스 소켓에 기반 하여 새로운 소켓 프로토콜 패밀리를 정의하고, 이를 리눅스 커널 모듈로 개발하였다. SoPCIe 모듈은 기존 TCP/IP 스택을 거치지 않는 일종의 SDP(Socket Direct Protocol)의 일종으로[11] 통신 수행 시 발생하는 프로토콜 처리 오버헤드를 줄이며, 데이터 페이로드에 대한 커널 복사 수를 줄임으로써 PCIeLINK의 인터커넥트 대역폭 사용을 최대화 하도록 설계 하였다. 특히, SoPCIe는 PCI Express 프로토콜의 특성인 메모리 액세스 방식(일종의 RDMA 기능)을 지원함으로 인하여, 기존 프로토콜 처리에 비하여 경량의 구현이 가능하다. 그림 18은 SoPCIe의 데이터 전송 처리 구조를 나타낸 그림이다.

Ⅳ. 실 험

이번 장에서는 구현된 PCIeLINK 하드웨어 장치와 소프트웨어에 대한 실험 및 결과를 제시한다.

가. PCIeLINK 실험 환경

구현된 PCIeLINK 인터커넥트 하드웨어와 소프트웨어의 실험을 위하여, x86 기반 컴퓨팅 노드간의 back-to-back 연결 형상을 통한 하드웨어 동작 및 성능 실험을 수행하였다.

실험에 사용된 컴퓨팅 노드는 Intel Core I5-3570K에 기반 한 마더보드에 개발한 PCIeLINK 카드를 장착하여 사용하였으며, 이들 노드간의 연결은 InfiniBand용 QDR 케이블을 사용하여 시험하였다. 소프트웨어의 개발 및 실험은 CentOS 6.4(Kernel 2.6.32) 기반의 Linux 환경 하에서 개발한 장치 드라이버와 기타 소프트웨어를 사용하여 수행하였다.

표 4. PCIeLINK 시스템 인터커넥트 실험 환경
Table 4. Experiment environment for PCIeLINK.

테스트베드 하드웨어 환경	
프로세서	인텔 Core i5-3507K CPU @ 3.4GHz
메모리	DDR3 1333MHz, 4 GB
마더보드	ASUS P8H77-M Pro
PCIeLINK 하드웨어	
보드	4 포트 PCIeLINK, PCIe Gen2
외부 연결 케이블	InfiniBand QDR 케이블(copper)
드라이버 버전	20141028
테스트베드 소프트웨어 환경	
운영체제	리눅스, CentOS 6.4, Kernel-2.6.32.sl6.i686
벤치마크	NetPIPE 3.7.2 벤치마크 Netperf 2.6.0 벤치마크 자체제작 테스트(ping-pong)
테스트 SW	vsftp, ssh, scp, etc.

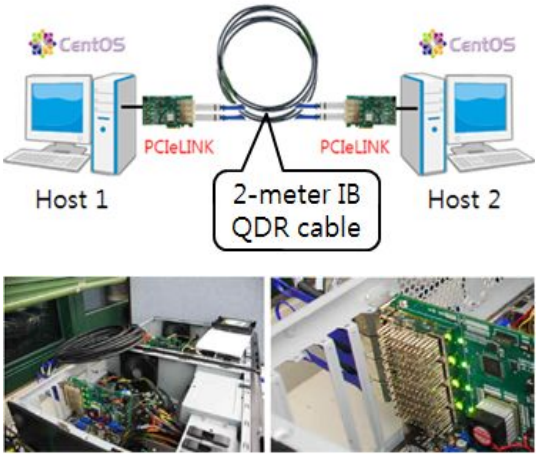


그림 19. PCIeLINK 인터커넥트 실험 형상 및 시험 사진
Fig. 19. Configuration for experiment.

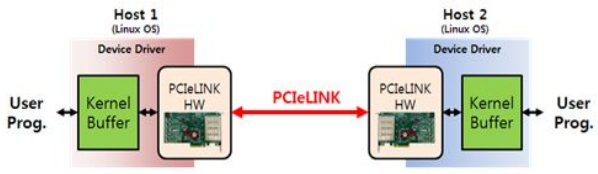


그림 20. Ping-pong 성능 시험 환경
Fig. 20. Configuration for ping-pong test.

나. 벤치마크를 통한 PCIeLINK 성능 평가

하드웨어적 시험을 통과한 PCIeLINK 장치 기반으로 노드 간 데이터 전송 성능 확인을 위한 pin-pong, 벤치마크, 응용 프로그램 등의 시험을 수행하였다.

(1) Ping-Pong 성능 시험

Ping-pong 시험은 두 노드의 여타 관련 모듈을 제거한 최소한의 장치 드라이버를 구현하고, 이 장치 드라이버상의 메시지 송수신을 위한 버퍼에 메시지 크기별로 데이터의 읽기, 쓰기를 반복하며 데이터가 전송되는 시간을 측정하는 방식으로 수행하였다. Ping-pong 측정 시험 결과, 시간의 측정 방식에 일부 오차는 있으나 PCIeLINK 인터커넥트는 단방향으로 최대 8.6 Gbps, 약 1GB/s의 메시지 전송 성능을 나타내었다.

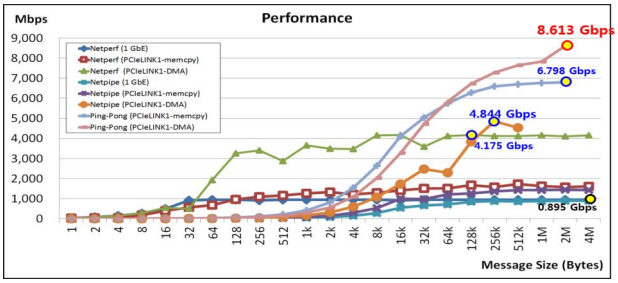


그림 21. PCIeLINK 성능 시험 결과
Fig. 21. Performance benchmark results of PCIeLINK.

(2) 벤치마크 성능 시험

벤치마크에 기반을 둔 성능 시험은 TCP/IP 네트워크에 기반을 둔 PCIeLINK 장치 드라이버를 설치하고, 네트워크의 성능평가에 사용되는 NetPIPE 3.7.2[24]와 Netperf 2.6.0[25]을 사용하여 두 호스트 시스템 간 메시지 전송 성능을 평가하였다.

초기에 구현된 memcpy 기반 장치 드라이버의 경우,

메시지의 전송에 Linux memcpy 명령을 사용함에 따라, 1.0~1.1 Gbps의 낮은 메시지 전송 성능을 나타내었다. 반면 DMA에 기반한 장치 드라이버의 경우, NetPIPE와 Netperf에서 최대 4.84 Gbps와 4.17 Gbps의 메시지 전송 성능을 가짐을 확인하였다. Ethernet 과의 성능 비교를 위하여 마더보드에 내장된 1G 이더넷 장치에 대한 NetPIPE 및 Netperf 결과는 각각 최대 0.89 Gbps, 0.94 Gbps의 성능을 보임을 확인하였다(그림 21).

(3) Linux 응용프로그램 시험

PCIeLINK에 기반 한 TCP/IP 네트워크의 성능을 확인하기 위하여, Linux vsftp에 기반 한 파일 송수신 성능 실험을 수행하였다. 실험 과정에서 HDD 상의 파일 읽기 및 쓰기로 인한 성능손실을 배제하기 위하여, Linux 상에 램 드라이브를 설정하고 램 드라이브에 대한 FTP를 수행하는 방식으로 PCIeLINK HW 및 SW를 실험하였다.

실험결과 PCIeLINK 장치를 이용한 TCP/IP 기반 FTP의 경우 순시 Peak의 경우 최대 5.1 Gbps, 평균 4.0~4.5 Gbps의 파일 전송 성능을 확인하였다. 또한 DMA에 기반 한 장치 드라이버는 Ethernet 장치를 이용한 FTP에서와 동일한 수준의 CPU 사용률을 나타냄을 확인하였다(그림 22).

그림 22. Linux vsftp 시험 결과
(좌: memcpy 기반, 우: DMA 기반)
Fig. 22. vsftp test result.
(left: memcpy based, right: DMA based)

다. 성능 측정 결과 분석 및 시사점

PCIeLINK 시스템 인터커넥트에 대한 초기 성능 평가 결과, ping-pong 및 벤치마크가 각각 8.6 Gbps, 4.7 Gbps 정도의 성능을 나타냄을 확인하였다(그림 21). 이와 같은 측정 결과는 포트의 성능 규격인 20Gbps 에 미치지 못하는 측정결과로서 다음과 같은 요인들이 성능 저하의 원인으로 작용한 것으로 분석된다.

(1) DMA 엔진의 활용

PCIeLINK에 사용된 PCI Express 스위치 칩셋은 내부에 DMA 엔진 당 2개의 전송 채널을 가지는 DMA 엔진을 2개 제공한다. 구현된 PCIeLINK 장치 드라이버는 제공되는 DMA 엔진 중 하나만을 사용하고 있으며, 사용되는 DMA의 채널 중 1개 채널은 로컬 호스트 시스템과의 DMA 전송, NT 포트의 DMA 전송에 사용된다. 개별 포트 자체는 cut-through 방식의 전송을 지원하나, DMA 전송 과정에서 store and forward 동작으로 인하여 메시지 전송성능이 저하되는 것으로 추정된다.

(2) Linux TCP/IP 프로토콜 처리

본 논문의 성능 평가는 제안한 SoPCIe에 기반한 측정 결과가 아닌 기존 Linux TCP/IP 스택에 기반한 성능 평가 방식으로 수행되었다. 기존 이더넷 장치와 달리 PCIeLINK는 PCI Express의 종단장치 자체를 사용함으로 널리 사용되는 TCP/IP offload와 같은 기능을 제공하지 않는다. 따라서 모든 TCP/IP의 처리를 소프트웨어로 처리됨에 따른 오버헤드가 존재하며 그에 따른 성능 저하가 있는 것으로 분석되었다.

V. 결 론

본 논문에서는 기존 이더넷, 인피니밴드와 같은 시스템 인터커넥트에 대한 대안 인터커넥트로써의 PCI Express 기술을 분석, 평가하고 PCI Express에 기반한 시스템 인터커넥트 하드웨어 및 소프트웨어를 개발, 평가 하였다. PCIeLINK는 기존 PCI Express 시스템에서 네트워크로의 적용 시 문제가 되는 전기적, 논리적인 충돌 문제를 방지하면서, PCI Express Gen2 규격에 기반 한 20 Gbps급의 x4 연결을 동시에 4개 지원한다.

PCIeLINK 시스템 인터커넥트에 대한 초기 성능 평가 결과, ping-pong 및 벤치마크 결과 각각 8.6 Gbps, 4.7 Gbps 정도의 성능을 나타냄을 확인하였으며, 기존 1G 이더넷 대비 높은 성능과 10 G 이더넷에 준하는 성능을 나타냄을 확인하였다.

평가 및 분석 결과, 하드웨어 장치의 제어와 네트워크 프로토콜의 처리 등에 있어서의 오버헤드로 인하여, 아직은 이론 성능 대비 상대적으로 낮은 성능을 보이는

점은 추후 개선되어야 할 것으로 판단된다.

추후 DMA 방식의 개선, SoPCIe 통신 모듈의 적용과 장치 드라이버 내부 메커니즘의 간소화를 통한 성능개선이 필요하며, 이를 통하여 향후 14~15 Gbps급의 성능을 나타낼 수 있을 것으로 기대 된다.

REFERENCES

[1] Young Woo Kim, et. al., “Technology and Trends of High Performance Processors,” Electronics and telecommunications trends, vol.25, no.5, pp. 123-136, 2010.

[2] Interconnect Family Statistics, TOP500. org accessed May 14, 2015, http://www.top500.org/statistics/list/

[3] “The 2015 Ethernet Roadmap,” white paper from Ethernet alliance accessed May 13, 2015, http://www.ethernetalliance.org/roadmap/

[4] “InfiniBand Roadmap,” InfiniBand Trade Association accessed May 13, 2015, http://www.infinibandta.org/content/pages.php?pg=technology_overview

[5] “Annual Meeting of Members Presentation,” PCI-SIG accessed May 13, 2015, https://www.pcisig.com/members/downloads/PCI-SIG_Annual_Meeting_of_Members_2013_Final.pdf

[6] “Why FPU Generator?” Floating-Point Unit Generator accessed May 15, 2015, https://sites.google.com/a/stanford.edu/fpgen/why

[7] Wonok Kwon, et. al., “PCI Express Gen3 System Design using High-speed Signal Integrity Analysis,” Journal of the Institute of Electronics and Information Engineers, vol.52, no.4, pp.125-132, 2015.

[8] Wonok Kwon, et. al., “ALTERA Embedded Gigabit Transceiver Measurement for PCI Express Protocol,” Journal of the Institute of Electronics and Information Engineers, vol.41, no.4, pp.359-367, 2004.

[9] Vijay Medury, “PCI Express in Clustering,” High Speed Interconnects Seminar, Linley Group, Nov., 2010.

[10] John Byrne, et. al., “Power-Efficient Networking for Balanced System Designs: Early Experiences with PCIe,” HotPower '11 Proceedings of the 4th Workshop on Power-Aware Computing and Systems, Article No. 3, 2011.

[11] Ahmed Bu-Khamsin, “Socket Direct Protocol over PCI Express Interconnect: Design, Implementation and Evaluation,” MS Thesis, Simon Fraser University, 2012.

[12] T. Hanawa, T. Boku, S. Miura, T. Okamoto, M. Sato, et al., “Low-Power and High-Performance Communication Mechanism for Dependable Embedded Systems,” International Workshop on Innovative Architecture for Future Generation High-Performance Processors and Systems, pp. 67-73, 2008.

[13] T. Hanawa, Y. Kodama, T. Boku, M. Sato, “Interconnection Network for Tightly Coupled Accelerators Architecture,” IEEE 21st Annual Symposium on High-Performance Interconnects, San Jose, USA, pp. 79-82, Aug. 2013.

[14] Young Woo Kim, et. al., “Design of MAHA Supercomputing System for Human Genome Analysis,” KIPS transactions on software and data engineering, vol.2, no.2, pp.81-90, 2013.

[15] Young Woo Kim, “MAHA-Manycore HPC System for Bio-Applications,” The Second KIISE-KOCSEA HPC SIG Joint Workshop, Denver, USA, Nov., 2013.

[16] Young Woo Kim, et. al., “Implementation of System Interconnection Device Using PCI Express,” Proceedings on 2014 IEIE Summer Conference, pp. 515-518, Jeju Island, Korea, 2014.

[17] Ye Ren, et. al., “A Preliminary Implementation of PCIeLINK and Its Performance Evaluation,” Proceedings on 2014 IEIE Summer Conference, pp. 519-522, Jeju Island, Korea, 2014.

[18] Ye Ren, et. al., “Implementation of System Interconnection Devices Using PCI Express,” IEEE International Conference on Consumer Electronics, Las Vegas, USA, pp. 300-301, Jan. 2015.

[19] D. Percival, “PCI Express Clustering,” PCI-SIG Developers Conference, Israel, 2011.

[20] IDT's PCIe Gen2 Switch family Non-Trans patent Operation, Application Note, IDT, 2009

[21] “SFF-8436 Specification for QSFP+ 10 Gbs 4X PLUGGABLE TRANSCEIVER” SFF Committee accessed Apr. 13, 2012, ftp://ftp.seagate.com/sff/SFF-8436.PDF

[22] IDT® 89HPES24NT6AG2 PCI Express® Switch User Manual, IDT accessed May 25, 2012.

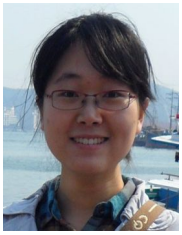
[23] WonHyuk Choi, et. al., “A Design of Dedicated Communication module for PCI Express network device,” KIISE Proceedings on Korea Computer Congress 2015, pp 95-97, Jeju Island, Korea,

2015.
[24] NetPIPE, http://bitspjoule.org/netpipe/
[25] Netperf, http://www.netperf.org/netperf/

저 자 소 개

김 영 우(정회원)
1994년 고려대학교 전자공학과 학사 졸업.
1996년 고려대학교 전자공학과 석사 졸업.
2001년 고려대학교 전자공학과 박사 졸업.
2014년～현재 과학기술연합대학원대학교 겸임교수(부교수)
2001년～현재 한국전자통신연구원, 책임연구원
<주관심분야 : 컴퓨터 구조, 반도체 회로설계, 고속 시스템 인터커넥트>

최 원 혁(정회원)
1999년 경북대학교 컴퓨터공학과 학사 졸업
2001년 경북대학교 컴퓨터공학과 석사 졸업
2001년～현재 한국전자통신연구원, 선임연구원
<주관심분야 : SW 가상화, SW 서비스, HPC>



런 예(정회원)
2010년 하얼빈공업대학 학사 졸업.
2012년 한국과학기술원 전기 및 전자공학부 석사 졸업.
2013년～현재 한국전자통신연구원, 연구원
<주관심분야 : 컴퓨터 네트워킹, 고성능 컴퓨팅>