링 구조 다중프로세서 시스템에서 링 대역폭 확장을 위한 효율적인 방안

(Efficient Schemes for Scaling Ring Bandwidth in Ring-based Multiprocessor System)

장 병 순 † 정 성 우 †† 장 성 태 ††† 전 주 식 ††††

(Byoung Soon Jang) (Sung Woo Chung) (Seong Tae Jhang) (Chu Shik Jhon)

요 약 최근 몇 년간 클러스터링 기반 다중 프로세서 시스템에서의 상호 연결망으로서의 버스의 제약을 극복하기 위한 단방향 지점간 링크를 이용한 링 구조가 제안되었다. 하지만 계속되는 프로세서의 고속화와 지역 버스 및 메모리의 고성능화로 인해 지점간 링크의 현재 표준 대역폭으로는 시스템 확장성에 한계를 보이며 이에 따라 대역폭 확장에 대한 연구가 필요하다.

본 논문에서는 클러스터링 기반 다중프로세서 시스템으로 개발된 PANDA 시스템을 기본 모델로 채택한다. 최근 대중화된 프로세서 및 지역 버스의 사양을 반영한 모의실험을 통해 현재의 지점간 링크가 전체 시스템 성능에 병목이 됨을 보여주고 두 배 이상의 대역폭 확장이 필요함을 보인다.

상호 연결망의 대역폭을 확장하기 위해, 두 배 증가된 대역폭을 지닌 새로운 링크를 개발하는 것은 과다한 설계비용과 개발시간이 요구된다. 이에 대한 대안으로 본 논문에서는 상용화되어있어 쉽게 적용 가능한 기존 IEEE 표준 대역폭을 가진 링크를 이용해 이중으로 링을 구성하는 몇 가지 방법 - 단순 이중 링, 트랜잭션 분리 이중 링, 방향 분리 이중 링 - 을 제시하고 모의실험을 통해 두 배 대역폭 단일 링과 더블어 각각의 방식에 대한 장단점을 분석한다.

Abstract In the past several years, many systems which adopted ring topology with high-speed unidirectional point-to-point links have emerged to overcome the limit of bus for interconnection network of clustered multiprocessor system. However, rapid increase of processor speed and performance improvement of local bus and memory system limit scalability of system with point-to-point link of standard bandwidth. Therefore, necessity to extend bandwidth is emphasized.

In this paper, we adopt PANDA system as base model, which is clustering-based multiprocessor system. By simulating a model adopting commercial processor and local bus specification, we show that point-to-point link is bottleneck of system performance, and bandwidth expansion by more than 200% is needed.

To expand bandwidth of interconnection network, it needs excessive design cost and time to develop new point-to-point link with doubled bandwidth. As an alternative to double bandwidth, we propose several ways to implement dual ring -simple dual ring, transaction-separated dual ring, direction-separated dual ring- by using off-the-shelf point-to-point links with IEEE standard bandwidth. We analyze pros. and cons. of each model compared with doubled-bandwidth single ring by simulation.

† 비 회 원 : 서울대학교 컴퓨터공학과
jangbs@panda.snu.ac.kr
†† 학생회원 : 서울대학교 컴퓨터공학과
sungwoo@superman.snu.ac.kr
††† 종신회원 : 수원대학교 전자계산학과 교수
stjhang@mail.suwon.ac.kr
†††† 종신회원 : 서울대학교 컴퓨터공학과 교수
csjhon@riact.snu.ac.kr
논문접수 : 1999년 4월 20일
심사완료 : 1999년 11월 24일

1. 서 론

현재 공유 메모리 다중 프로세서 시스템의 상호 연결망 중에서 가장 대중적인 기술은 공유 버스이다. 공유 버스는 구현상의 복잡도가 낮고 비용이 적게 드는 장점이 있으나, 빠르게 성능 향상이 되고 있는 프로세서의 속도를 따르지 못하고 있다. 또한 버스의 물리적 특성으로 인한 확장성과 버스 사용량의 증가로 인한 버스 대역폭에 대해 문제점을 갖고 있다. 버스의 대역폭을 늘리

는 문제는 버스의 중재와 주소 지정에 필요한 고정된 오버헤드 때문에 실제 대역폭이 그만큼 향상되는 효과를 보지 못하고 있다. 오버헤드를 줄이기 위해서 라인 전송의 크기를 늘리는 방법은 캐쉬 라인의 크기를 넘을 때는 그 효과가 거의 없게 된다.

이러한 버스 구조 한계를 극복하고자 진행된 여러 연구 중 단방향 지점간 링크를 이용한 SCI가 IEEE에 의해 표준으로 확정되었다[1]. Sequent에서는 4개의 펜티엄 프로 프로세서가 P6버스에 의해 UMA구조로 묶인 클러스터를 SCI 링크를 사용해 연결한 STiNG[2]을 발표했다. Toronto 대학에서 만든 NUMAchine은 디렉토리를 사용한 캐쉬 일관성 유지방법을 사용하는 링 구조의 네트워크를 계층적 구조로 확장한 것으로, 64개의 프로세서를 연결한 프로토타입이 제작되었다[3].

한편, 스누핑 방식은 단방향 지점간 링크를 사용할 때 방송의 어려움으로 인해 적합하지 않으리라 믿어져 왔으나 그 비용이 크지 않을 뿐 아니라 항상 일정한 응답 지연시간에서 이득을 볼 수 있다는 점에 착안해 Michel Dubois는 스누핑 방식의 캐쉬 일관성 유지방법을 사용한 Express Ring[4]을 제안하였다. 하지만 Express Ring은 슬롯링(Slotted Ring) 방식으로 구성되어있기 때문에, 전송하고자 하는 패킷의 크기와 다른 슬롯이 도착할 때는 그 슬롯을 사용할 수 없고 따라서 링의 이용률이 낮아지게 된다.

서울대학교. PANDA 연구실에서는 스누핑 방식의 캐쉬 일관성 유지방법을 사용하고 하나의 트랜잭션을 연속된 여러 개의 패킷으로 나누어 전송하는 레지스터 삽입 방식의 단방향 링크를 사용한 PANDA (Progressive Approach of NUMA model based on Distributed shared memory Architecture) 시스템을 제안하였고[5], 응답 지연시간의 단축으로 디렉토리 캐쉬 일관성 유지 방법을 사용한 시스템에 비해 성능이 우수함이 발표되었다[6].

최근 Data general에서는 500MByte/sec.의 대역폭의 링크로써 회전 방향을 다르게 하여 이중으로 링을 구성한 AViiON AV 25000이라는 제품을 발표했다[7]. 이 제품에서는 linked list를 사용하여 디렉토리 방식으로 캐쉬 일관성을 유지한다.

현재 IEEE 표준 SCI 링크는 16bit 데이터 폭으로 1GByte/sec. (500MHz)의 대역폭을 가진다[1]. 한편 최근 제조업체들이 발표하는 프로세서들은 이미 500MHz 이상의 클럭 속도를 나타내고[8], 메모리의 고속화[9]에 따르는 100MHz의 버스 사양이 상용화되고 있다. 이에 따라 이러한 사양의 프로세서와 지역 버스를 채택한 클러스터링 기반 다중 프로세서 시스템에서 기존의 SCI 기반 단방향 지점간 링크의 링을 전역 연결망으로 사용한다면 대역폭의 한계에 의한 확장성의 문제가 야기될 것으로 보여진다.

본 논문에서는 고속화된 프로세서 및 지역 버스의 사양에 맞추어 모델링한 PANDA 시스템에서의 모의실험에 근거해 현재의 단일 지점간 링크의 한계를 보여주고 두 배 이상의 대역폭 확장이 필요함을 보인다.

현재 IEEE 표준의 두 배 대역폭을 지닌 상용 지점간 링크는 존재하지 않기 때문에 이를 기존 시스템에 채택하기 위해서는 새로운 링크를 직접 개발하여야 하는데, 그러하기에는 시스템의 설계비용 및 개발시간이 과도해질 것이다. 이에 대한 대안으로 본 논문에서는 상용화되어있어 쉽게 적용 가능한 기존 IEEE 표준 대역폭을 가진 링크를 이용해 이중으로 링으로 구성하는 방식을 제안한다. 이중 링으로 구성할 수 있는 몇 가지 변형을 제시하고 각각의 방식에 대한 장단점을 분석해 본다.

이하 본 논문의 구성은 다음과 같다. 2장에서는 논문의 연구 대상이 될 이중 링을 사용한 PANDA 시스템의 전체 구조, 노드 제어기의 구조, 캐쉬 일관성 프로토콜 등에 대해 살펴본다. 3장에서는 이중 링으로 구성하는 몇 가지 방식을 제안하고 정성적 분석을 한다. 4장에서는 대역폭 확장에 의한 효과와 3장에서 제시한 각 방식들의 특성을 모의실험 결과를 통해 보여주고 이를 근거로 5장에서 결론을 맺는다.

2. 시스템의 구조와 동작

2.1 시스템의 기본 구조

본 논문에서 모델로 하고 있는 시스템은 강결합성 분산 메모리 다중 프로세서 시스템(tightly-coupled distributed shared memory multiprocessor system)으로 그림 1과 같이 각 프로세서 노드를 지점간 이중 링크(point-to-point dual link)로 연결해서 링 토폴로지를 구성한다.

각 노드는 Intel(R)의 SHV 모델[10]과 유사하게 네 개의 프로세서 모듈과 지역 메모리, I/O 제어기 및 노드 제어기가 지역 버스에 연결 되어있다. 노드의 지역 버스는 요청 트랜잭션과 응답 트랜잭션을 분리하는 split 버스 프로토콜을 지원하며 스누핑 방식에 의한 캐쉬 일관성 유지 프로토콜을 사용한다. 각 프로세서 모듈은 1차 캐쉬(L1 cache) 및 2차 캐쉬(L2 cache)를 내장하고 있으며 지역 메모리는 분산형 공유 메모리로서 전체 시스템 메모리 물리 주소 영역의 일부를 구성하고 있다.

IO 제어기는 프로세서에서 요구한 I/O 디바이스에

대한 읽기 및 쓰기를 수행해 준다. 모든 I/O 관련 트랜잭션은 지역버스에 연결된 PCI 버스를 통해 처리하게 된다.

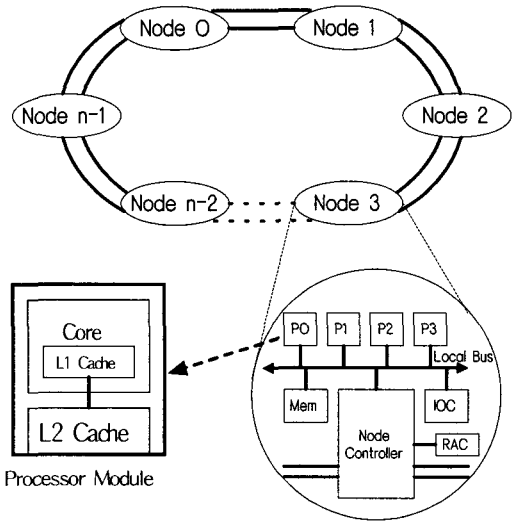

그림 1 전체 시스템 구성도

노드 제어기 원격 노드로의 접근 지연 시간을 단축하기 위한 원격 접근 캐쉬(RAC : Remote Access Cache)를 유지하며 지역 버스에서 발생한 요청에 대해 RAC로부터 또는 전역 링크를 통한 원격 노드로부터의 데이터 제공을 담당하고 마찬가지로 전역 링크에서 발생한 요청에 대해 응답할 책임을 진다. 이와 동시에 노드 제어기는 지역 버스 및 전역 링크에서 발생하는 모든 트랜잭션을 스누핑하여 메모리-캐쉬 일관성 유지를 위해 필요한 제어를 수행한다.

2.2 노드 제어기의 구조

노드 제어기는 지역 트랜잭션 제어기(Local Transaction Controller), 전역 트랜잭션 제어기(Global Transaction Controller), RAC 및 태그, 지역 메모리 디렉토리(Local Memory Directory), 그리고 링 인터페이스로 이루어져 있다. 그림 2는 노드 제어기의 구성도이다.

2.2.1 RAC

일반적으로 네트워크를 통한 원격 메모리 접근은 시간이 오래 걸리는 작업이므로, RAC는 원격 메모리 접근을 줄이는 역할을 함으로써 평균적인 메모리 응답 지연 시간을 줄여준다[11][12]. 또한 프로세서 캐쉬와 원격 메모리 영역에 대해 MLI(Multi-Level Inclusion Property)를 만족시키므로 스누핑 여과의 기능을 통해 전역 연결망에서의 스누핑 요구가 지역 버스에까지 접근하는 것을 막아 준다.

본 시스템에서는 전역 트랜잭션 제어기 및 지역 버스로부터의 요청을 지역 트랜잭션 제어기가 받아들여 요청에 해당하는 동작을 RAC에 행하도록 모델링 하였다.

2.2.2 링 인터페이스

링 인터페이스는 이전 링을 통해 전달된 트랜잭션 및 전역 트랜잭션 제어기로부터의 트랜잭션을 지역 노드 또는 다음 노드로 전송해주는 역할을 한다. 기존 단일 링을 사용한 PANDA 시스템의 링 인터페이스 부분과 비교해볼 때 헤더 디코더, bypass 큐, inbound 큐 및 outbound 큐가 두 개씩 중복되어 있는 형태로 구성된다.

링 상에서 하나의 트랜잭션은 여러 개의 심볼로 나뉘어져 연속적으로 전송된다. 심볼이란 지점간 링크를 통해 한 번에 전송할 수 있는, 링크의 폭의 크기를 가지는 전송 단위이다.

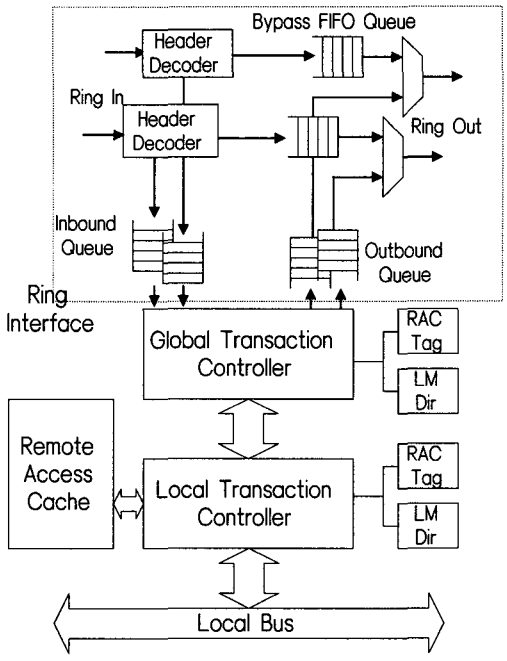

그림 2 노드 제어기의 세부 구조

헤더 디코더는 링을 통해 들어온 트랜잭션의 헤더부분을 검사해 현재 전달되는 트랜잭션이 자신의 노드에 대한 요청을 내포하고 있는지를 판단한다. 만약 내포하고 있다면 inbound 큐에 삽입을 하고, 내포하고 있지

않거나 방송 요청(broadcast request)이라면 bypass FIFO 큐에 삽입을 하는 역할을 한다.

다음 노드로의 전송은 bypass FIFO 큐의 내용과 outbound 큐의 내용 중 선택된다. 우선 순위가 높은 것은 bypass FIFO에 이미 들어 온 패킷이다. Outbound 큐에서 다음 노드로의 전송이 이미 진행되고 있을 동안 이전 노드로부터 들어오는 트랜잭션이 지점간 링크 상에서 스톨(stall)되지 않게 하기 위해 bypass FIFO 큐는 최대 길이를 가진 트랜잭션의 심볼 개수에 해당하는 큐 엔트리를 유지한다. Bypass FIFO 큐의 입구에서 들어온 심볼은 큐의 각 엔트리를 모두 지나야 출구로 빠져나갈 수 있다.

2.2.3 지역 메모리 디렉토리 및 RAC 태그

지역 메모리의 상태 및 RAC의 상태를 캐쉬 라인별로 유지하는 부분이다.

지역 메모리 디렉토리와 RAC 태그는 지역 트랜잭션 제어기와 전역 트랜잭션 제어기에서 각각 독립적으로 동작할 수 있도록 이중화시킨다. 이중화를 시키는 이유는 본 논문에서 모델로 사용하는 시스템이 지역 버스 및 전역 연결망 모두를 스누핑 방식으로 캐쉬 일관성을 유지하고 있기 때문에 상태 검색 및 갱신 요구에 의한 디렉토리 및 태그의 접근이 전체 시스템의 병목 지점이 되고 있으므로 지역 버스에서의 요구와 전역 연결망에서의 요구를 각각 독립적으로 처리해 빠른 스누핑을 보장하기 위함이다.

2.3 캐쉬 일관성 프로토콜

프로세서 모듈에 포함된 L1 및 L2 캐쉬는 Write-once 프로토콜[13]을 축약한 MSI 프로토콜을 사용하고 RAC의 일관성 유지를 위해서는 지역 메모리로의 갱신을 줄일 수 있는 Berkeley 프로토콜[14]을 사용한다.

메모리 디렉토리는 C(clean), G(gone), S(shared)의 세 가지 상태를 유지해 외부 노드의 RAC와 캐쉬 일관성을 유지하게 된다. 각 상태의 의미는 다음과 같다.

- Clean : 해당 라인을 공유하고 있는 외부 노드가 없다는 의미이다. 지역 노드에서 라인에 대한 읽기 및 쓰기 권한을 가짐을 나타낸다.
- Gone : 해당 라인의 최신 정보가 외부 노드에 존재한다는 의미이다. 지역 노드에서 라인에 대한 읽기, 쓰기 권한 모두 없음을 나타낸다.
- Shared : 해당 라인의 복사본이 외부 노드에 존재할 가능성이 있다는 의미이다. 지역 노드에서 라인에 대한 읽기 권한만이 있음을 나타낸다.

링에서 스누핑 방식의 캐쉬 일관성 프로토콜을 사용하여 트랜잭션을 처리하는 방법[9]을 간단히 요약하면 다음과 같다. 요청 노드가 링으로 요청 패킷을 방송(broadcast)하면, 요청 패킷이 링을 순회하는 동안 각 노드는 이 요청에 대한 스누핑을 하게 된다. 이때 스누핑 결과 자신이 응답에 대한 책임을 가진다고 판단한 노드는 응답 패킷을 요청 노드로 단일전송(unicast)하게 된다. 요청 노드는 응답 패킷을 확인한 후 이를 노드 내 지역 버스로 보내어 줌과 동시에 응답 노드에는 Ack 패킷을 단일전송 하여 준다.

3. 이중 링의 구성과 정성적 분석

이 장에서는 이중 링을 구성하는 몇 가지 방법들을 소개하고 이들의 정성적인 분석을 통해 두 배의 대역폭을 가진 단일 링과 비교를 하고 성능을 예측하여 본다.

3.1 단순 이중 링

이 모델에서는 단일 링을 사용한 스누핑 방식 시스템을 단순히 확장하여 두 개의 링을 똑같은 방식으로 구성한다. 확률적으로 양 링을 균등하게 사용하도록 하기 위한 손쉬운 방법으로 요청 패킷의 목적지 캐쉬 라인 주소에 따라 홀수 주소용 링과 짝수 주소용 링으로 이분하여 각각 독립적으로 구동시키는 방식을 사용한다.

이 모델의 가장 큰 이점은 기존 단일 링 시스템의 구조에 약간의 수정만을 가하면 바로 적용할 수 있기 때문에 구현의 측면에서 상당한 이득을 볼 수 있다는 점이다. 그러나 이 모델은 두 배의 대역폭을 가진 단일 링에 비교해 볼 때 대역폭의 크기는 같지만 두 개의 자원(홀수 주소용 링 및 짝수 주소용 링)을 독립적으로 접근하고 있으므로 두 개의 자원에 대한 요구가 정확히 번갈아 가면서 발생하지 않는 이상 어느 한 쪽이 유휴 상태에 있을 때 다른 한 쪽에선 자원에 대한 경쟁이 일어나게 되곤 한다. 이로 인해 링의 사용권을 획득하기 위해 outbound 큐에서 대기하는 시간이 전체적으로 길어질 것을 예측할 수 있다.

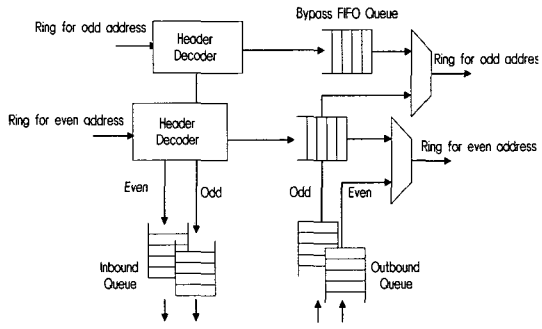

그림 3 단순 이중 링을 위한 링 인터페이스

그리고 응답 지연 시간도 단일 링에 비해 길어지게 된다. 링크의 폭이 절반이 되면 bypass FIFO 큐의 엔트리 개수는 두 배로 늘어나게 될 것이고 이 때문에 링을 순회하는데 걸리는 시간이 길어져 성능에 저하를 가져올 것으로 보인다.

위와 같은 이유로 이 모델은 두 배의 대역폭을 가진 단일 링에 비해 좋지 못한 성능을 나타낼 것으로 보여진다.

3.2 트랜잭션 분리 이중 링

이 모델에서는 하나의 링을 데이터를 포함하지 않은 트랜잭션용으로 사용하고 다른 하나는 데이터를 포함하는 트랜잭션(되쓰기, 쓰기 요청 및 읽기 요청에 대한 응답)용으로 구분하여 사용한다.

이러한 구성의 기본적인 착상은 트랜잭션이 노드를 거쳐 다음 노드로 넘겨질 때 노드 제어기의 링 인터페이스에 포함된 bypass 큐에서 늘 일정한 지연 시간을 거쳐야 하는 특성에 기인한다. 최대 길이의 트랜잭션(일반적으로 주소와 데이터를 모두 포함한 되쓰기(write-back)가 가장 긴 트랜잭션이다.)의 심볼을 모두 담을 수 있도록 bypass 큐의 엔트리 개수가 결정되기 때문에 길이가 짧은 트랜잭션일 경우 쓸데없이 많은 시간을 bypass 큐에서 보내게 된다. 이러한 bypass 큐 내에서의 쓸모 없는 지연은 시스템의 성능을 저해하는 요소가 된다. 특히 데이터를 포함하지 않은 트랜잭션은 대부분이 방송(broadcast) 트랜잭션이기 때문에 빠르게 다음 노드로 전달될 수 있다면 성능향상에 크게 기여할 수 있을 것으로 보인다.

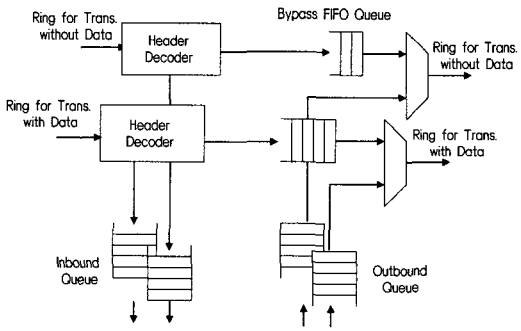

그림 4 트랜잭션 분리 이중 링을 위한 링 인터페이스

이를 위해 이 모델에서는 데이터를 포함하지 않은 트랜잭션의 심볼 개수에 해당하는 엔트리를 가진 bypass 큐를 채택하여 링을 구성함으로써, 데이터 불포함 트랜잭션이 bypass 큐를 통과하기 위한 시간을 줄일 수 있도록 구성한다. 그렇게 하면 트랜잭션이 링을 통해 순회하는 데 걸리는 시간을 크게 단축시킬 수 있을 것이고 이로 인한 응답 지연시간의 단축이 시스템의 성능 향상을 가져올 것으로 보여진다.

3.3 방향 분리 이중 링

이 모델에서는 링의 전송 회전 방향을 분리하여 하나는 시계 방향으로 다른 하나는 반시계 방향으로 트랜잭션을 전달하도록 구성한다.

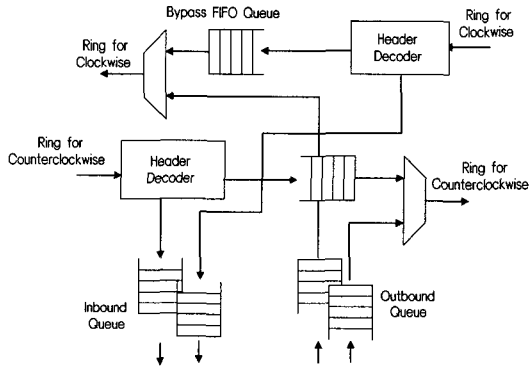

그림 5 방향 분리 이중 링을 위한 링 인터페이스

이러한 구성으로는 방송 트랜잭션에 대해선 동일한 결과를 나타내지만 응답 트랜잭션, 쓰기 요청 트랜잭션, ack 등 단일전송 트랜잭션일 경우 좀더 가까운 방향을 결정하여 해당 링을 사용한다면 응답 지연을 줄이는 효과를 보일 것이다. 비록 스누핑 방식의 시스템은 방송에 기반하고 있지만 2.3절에서 설명한 것처럼 PANDA 시스템의 프로토콜 특성상 전체 트랜잭션 중 많은 비율이 단일전송이기 때문에 응답 지연의 단축에 따르는 시스템 성능 향상을 크게 기대할 수 있을 것이다.

특히 프로세서의 개수가 증가하여 노드의 개수가 많아지면 응답 지연시간의 단축이라는 장점은 더욱 선명하게 드러날 것으로 보인다.

4. 시스템 성능 모의실험 및 분석

4.1 모의실험 환경

본 논문에서는 앞에서 제안한 여러 가지의 링 구성 방식을 적용한 다중 프로세서 시스템을 모의실험하기 위해 모의실험 도구로 SES/Workbench를 사용한다. SES/Workbench는 큐잉 모델에 기반하여 모델링 되어 있는 시스템의 성능을 분석하기 위해 과학이나 공학 분야에서 쓰이고 있는 모의실험 도구이다[15].

모의 실험에서 SES/Workbench의 입력으로는

TPC-A[16]를 사용한다. TPC-A 벤치마크는 집약적 갱신의 특징을 나타내는 OLTP(on-line transaction processing) 응용 데이터베이스 서비스 시스템의 환경에서 실제 궤적(trace)을 따라 추출한 통계자료이다. OLTP 환경은 다음과 같은 특징을 갖는다.

- 다중 온라인 터미널 세션
- 많은 디스크 입출력
- 적정한 시스템과 응용 프로그램의 실행 시간
- 트랜잭션의 무결성

4.2 시스템 관련 인자

모의실험에서 각 노드는 최근 발표된 상용 제품의 성능에 근거하여 500MHz의 클럭을 가지는 4개의 CPU가 100MHz의 지역 버스에 클러스터링 되어 있는 것으로 모델링 하였다. 그리고 전역 연결망을 위한 링은 IEEE 표준을 따라 500MHz로 동작하게 한다. 사용된 시스템 관련 인자를 다시 표 1에 정리하였다.

본 논문에서는 두 배 대역폭 단일 링, 단순 이중 링, 트랜잭션 분리 이중 링, 방향 분리 이중 링의 순서로 링의 폭을 16bit, 32bit으로 변화시켜 가면서(이때 두 배 대역폭 단일 링은 32bit, 64bit 순서로 변화한다.) 그리고 프로세서의 개수를 16개, 32개, 64개, 128개 순서로 변화시켜 가면서 각 경우에 따른 시스템의 성능을 비교하도록 한다.

표 1 시스템 관련 인자

프로세서 클럭	500 MHz
지역 버스 클럭	100 MHz
노드당 프로세서	4 개
메모리 버스 클럭	100 MHz
링(지점간 링크) 클럭	500 MHz
I/O 버스 클럭	100 MHz
2차 캐쉬 크기	512 KB
3차 캐쉬 크기	32 MB
캐쉬 라인 크기	32 Byte

4.3 모의실험 결과 및 분석

모의실험 결과에 따른 성능은 초당 트랜잭션 수행 수(TPS : Transaction Per Second)의 수치로 비교를 한다.

각 모델들의 비교에 앞서 PANDA 시스템에서 대역폭 확장의 필요성에 대해 설명하기 위해 일단 단일 링으로 구성한 시스템의 전역 연결망 대역폭 및 프로세서 개수에 따른 성능 향상을 살펴본다.

그림 6은 500MHz의 클럭을 가진 고속의 프로세서와 100MHz의 고속 지역 버스 및 메모리 버스를 PANDA 시스템에 적용했을 때 프로세서 개수 별 및 대역폭 별 시스템 성능의 변화를 나타낸 그래프이다. 이 실험을 통해 기존 500MHz의 16bit 단일 링으로는 프로세서 16개를 장착한 시스템이 한계임을 알 수 있다. 프로세서의 개수를 32개로 증가시켜 보아도 성능 향상은 거의 발생하지 않고 있다. 이는 16bit 단일 링에서 프로세서를 16개만 장착하더라도 전역 연결망의 이용률이 이미 임계점에 거의 다가서 있기 때문이다.(그림 7)

프로세서 32개의 시스템일 경우 최소 32 bit의 대역폭은 보장해야 충분한 성능을 나타낼 것이다. 그러나 여전히 프로세서의 개수가 64개로 증가함에 따라 곧 전역 연결망이 포화상태에 이르게 되고 32bit 대역폭 링크로는 더 이상의 성능 향상은 기대할 수 없게 된다. 64 bit의 링으로도 프로세서의 개수가 128개로 늘어나면 임계점을 넘어서게 되어 더 큰 대역폭을 요구함을 여기서 볼 수 있다.

지금까지 지점간 링크의 대역폭 확장은 PANDA 시스템의 확장성을 위해 필수적임을 알아보았다. 대역폭 확장을 위한 방법으로 두 배의 대역폭을 가진 단일 링의 사용은 현재 PANDA 시스템에서는 바로 적용하기 어려운 방법임을 앞서 언급했다. 이후부터는 3장에서 제시한 기존 대역폭 링크 이중 링 구조의 각 모델과 두 배 대역폭 단일 링 시스템에 대한 모의 실험 결과를 분

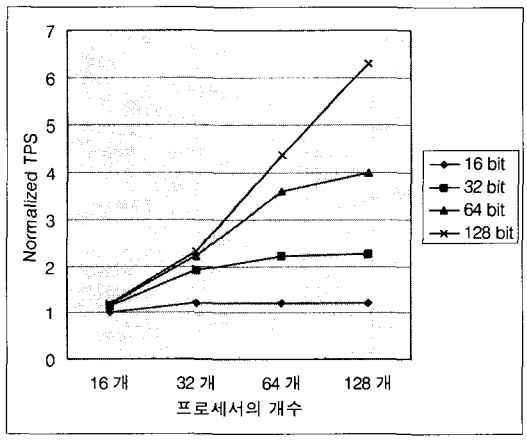

그림 6 시스템에 따른 TPS의 변화(16 bit 지점간 링크로 단일 링을 구성하고 프로세서 32개를 사용한 시스템의 TPS로 정규화)

석하고 비교를 통해 각 모델의 특성을 드러내고자 한다. (이후 단일 링은 모두 두 배 대역폭 단일 링을 이른다.)

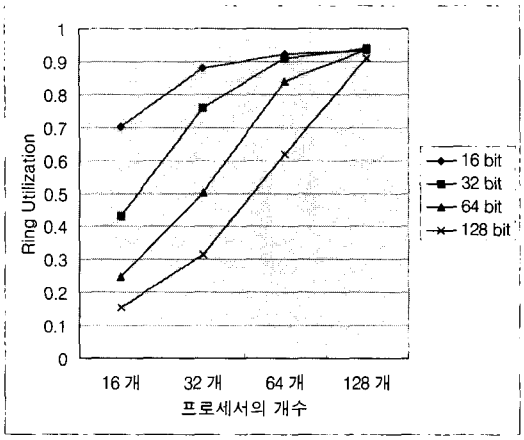

그림 7 시스템에 따른 링 이용률의 변화

4.3.1 각 시스템별 응답 지연 시간의 변화

그림 8과 그림 9는 프로세서의 개수 및 전역 연결망의 대역폭의 변화에 따르는 각 시스템별 전역 요청 트랜잭션의 평균 응답 지연 시간의 변화를 나타낸다.

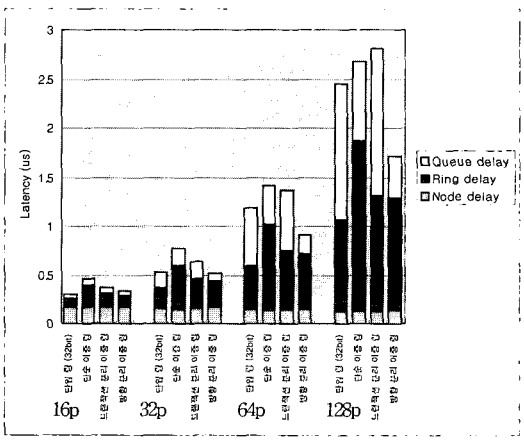

그림 8 각 모델의 응답 지연 시간의 변화(32bit 링크 단일 링 및 16bit 링크 두 개를 사용한 이중 링)

응답 지연 시간은 링 사용권의 획득을 위해 outbound 큐에서 대기하는 시간, 트랜잭션이 링을 순회하는데 드는 시간, 그리고 목적지 노드에서 해당 트랜잭션을 처리하는 데 걸리는 시간으로 나누어 측정하였다.

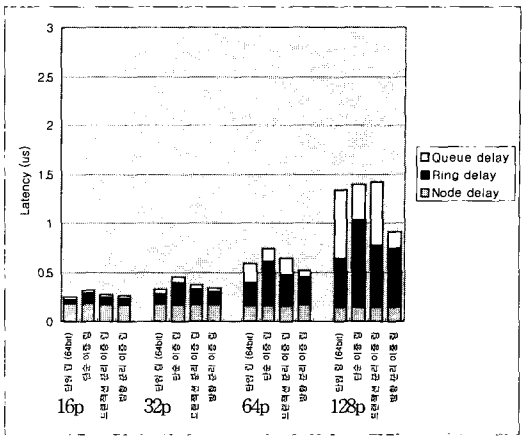

그림 9 각 모델의 응답 지연 시간의 변화(64bit 링크 단일 링 및 32bit 링크 두 개를 사용한 이중 링)

목적지 노드에서의 트랜잭션 처리 시간은 링 대역폭의 변화 및 링 구성 방식의 변화와는 거의 무관함을 알 수 있다. 프로세서의 개수가 많아짐에 따라서는 노드에서의 처리 시간이 전반적으로 줄어들고 있음을 알 수 있는데 이는 다음과 같이 지역 버스 이용률의 감소현상으로 설명되어 질 수 있다. 프로세서의 개수가 증가하게 되면 원격 노드를 접근해야 하는 트랜잭션의 처리시간이 길어진다. 따라서 이 트랜잭션에 종속성(dependency)이 있는 이후 다른 트랜잭션들의 처리 요구도 그만큼 지연되고 이로 인해 프로세서가 유휴상태에 놓여있을 경우가 많아진다. 그리하여 지역 버스의 이용률은 낮아지게 되고 전역 연결망으로부터 inbound 큐를 통해 전달된 전역 트랜잭션이 지역 버스의 사용권을 획득하는데 걸리는 시간이 줄어들게 된다.

시스템에 따른 링 순회 시간의 변화는 다음과 같다. 프로세서의 개수를 두 배 증가시키면 그에 따라 링 순회 시간도 두 배로 증가하게 되고 링의 대역폭을 두 배 증가시키면 링 순회 시간은 반으로 감소하게 된다. 링 순회 시간은 다시 bypass 큐에서 보내는 시간과 링크상에서 보내는 시간으로 나누어 볼 수 있다. 이중 링 시스템에서는 단일 링일 때에 비해 bypass 큐의 엔트리 개수가 두 배로 증가하게 되고(단 트랜잭션 분리 이중 링에서 데이터 불포함 링은 예외) 이로 인해 링 순회 시간이 단일 링에 비해 길어지게 된다. 단순 이중 링에서는 단일 링에 비해 링 순회 시간이 거의 두 배에 이른다. 트랜잭션 분리 이중 링의 경우는 데이터를 포함하지 않은 트랜잭션의 심볼 개수만큼의 엔트리를 가지도

록 bypass 큐를 구성함으로써 이들 트랜잭션의 링 순회 시간 단축으로 인하여 전체 트랜잭션의 링 순회 시간이 단순 이중 링에 비해 최소 30%, 최대 37%까지 줄어든다. 방향 분리 이중 링의 경우는 단일 전송의 회전 방향 최적화를 통한 트랜잭션 전송 거리의 단축으로 단순 이중 링에 비해 링 순회 시간이 최소 33%, 최대 44%까지 줄어든다.

프로세서의 수가 증가함에 따라 링의 사용권 획득을 위해 outbound 큐에서 대기하는 시간은 길어져 전체 응답 지연 시간에서 큐 대기 시간이 차지하는 비율이 차츰 높아지게 됨을 알 수 있다. 특히 이용률이 임계점을 넘어서는 시점부터는 큐 대기 시간이 매우 급격하게 증가하고 있다. 이중 링 시스템에서는 링 인터페이스 부분의 outbound 큐가 각각의 링에 장착되어 있으므로 트랜잭션이 두 개의 outbound 큐에 분산되어 대기하기 때문에 전체적으로 큐 대기 시간이 단일 링에 비해 짧다. 큐 대기 시간은 링의 이용률과 밀접한 관련이 있다.

4.3.2 각 시스템별 링 이용률의 변화

그림 10과 그림 11은 프로세서의 개수 및 전역 연결망의 대역폭의 변화에 따르는 각 시스템별 링 이용률의 변화를 나타낸다.

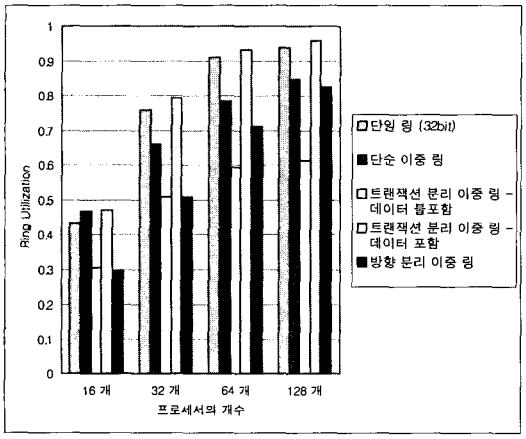

그림 10 각 모델의 링 이용률의 변화(32bit 링크 단일 링 및 16bit 링크 두 개를 사용한 이중 링)

단순 이중 링의 경우 프로세서의 개수가 적을 때는 단일 링에 비해 이용률이 높지만 프로세서의 개수를 늘이면 단일 링에 비해 이용률이 낮아진다. 이는 다음과 같이 설명되어 질 수 있다. 링을 순회하는 트랜잭션의 수가 아주 적을 경우에서의 링 이용률은 트랜잭션이 전송되는 속도에 크게 영향을 받게 된다. 따라서 단일 링

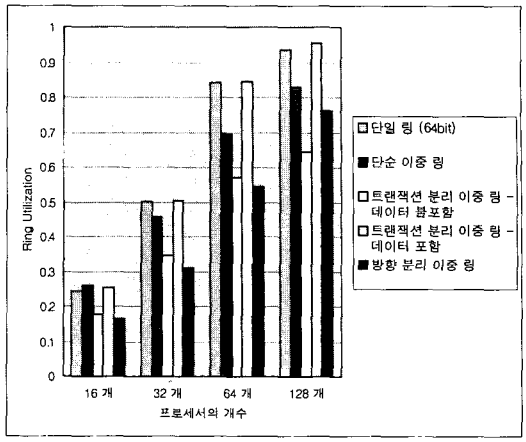

그림 11 각 모델의 링 이용률의 변화(64bit 링크 단일 링 및 32bit 링크 두 개를 사용한 이중 링)

의 절반의 대역폭을 가지는 단순 이중 링에서는 트랜잭션이 링을 점유하고 있는 기간이 길어지게 된다. 전역 트랜잭션의 수가 많아짐에 따라 링의 사용권 획득을 위한 경쟁이 빈번하게 발생하게 되는데 이때 단순 이중 링은 하나의 링이 경쟁 상태에 있을 때 다른 하나의 링은 유휴 상태에 놓여있을 경우가 많아질 것이며 이러한 링의 비효율적인 사용으로 인해 단일 링에 비해 전체 링 이용률이 저하되는 효과를 가져온다.

트랜잭션 분리 이중 링의 경우 데이터 포함 트랜잭션용의 링의 이용률이 불포함 트랜잭션용 링에 비해 매우 높음을 알 수 있다. 모의 실험에서 구현한 데이터 포함 트랜잭션은 불포함 트랜잭션에 비해 발생 빈도는 낮지만 전송해야 할 크기가 최대 세 배이므로 그만큼 링 이용률이 높아져서 프로세서 개수가 증가함에 따라 곧 임계점을 넘어서게 된다. 그림 10 및 그림 11에서 살펴볼 수 있듯이 프로세서의 개수가 64개 및 128개일 경우에 데이터 불포함 트랜잭션용 링은 아직 불포화 상태에 있음에도 데이터 포함 트랜잭션용 링이 포화상태에 놓여있게 된다. 이로 인해 전체 트랜잭션의 지연시간 중에서 큐 대기 시간을 급격히 증가시키는 결과를 가져옴을 그림 8 및 그림 9에서 확인할 수 있다.

방향 분리 이중 링의 경우 단순 이중 링에 비해 단일 전송 트랜잭션이 거쳐야 하는 링의 수가 줄어들기 때문에 전체 링 이용률을 낮추어 줄 수 있다. 그리하여 트랜잭션의 큐 대기 시간을 감소시켜 준다.

4.3.3 각 시스템별 성능의 변화

그림 12와 그림 13은 프로세서의 개수 및 전역 연결

망의 대역폭의 변화에 따라 각 시스템별 TPS를 단일 링의 TPS에 정규화 시켜 나타내었다.

단일 링은 방향 분리 이중 링을 제외한 다른 이중 링 모델에 비해 우수한 성능을 보인다. 이는 앞서 언급했듯이 이중 링에 비해 절반의 bypass 큐 엔트리 개수를 가지며 자원을 효율적 사용하는 단일 링의 특성에 기인한다고 분석된다.

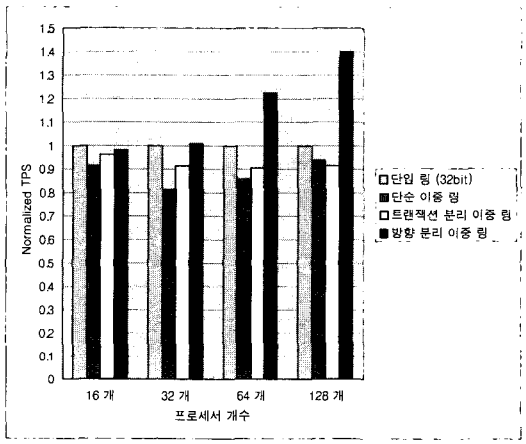

그림 12 각 모델의 성능의 변화(32bit 링크 단일 링 및 16bit 링크 두 개를 사용한 이중 링- 두 배 대역폭 단일 링의 TPS로 정규화)

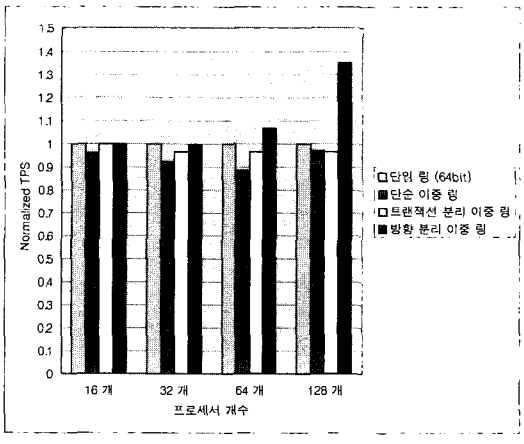

그림 13 각 모델의 성능의 변화(64bit 링크 단일 링 및 32bit 링크 두 개를 사용한 이중 링- 두 배 대역폭 단일 링의 TPS로 정규화)

단순 이중 링의 경우 단일 링 및 다른 이중 링 모델에 비해 저조한 성능을 보인다. Bypass 큐 엔트리 개수의 증가에 따라 링 순회 시간이 길어진 것이 성능 저하의 가장 직접적인 원인으로 분석된다.

트랜잭션 분리 이중 링의 경우 프로세서의 개수가 적을 때는 데이터 불포함 트랜잭션의 링 순회 시간의 단축으로 인하여 단순 이중 링에 비해 성능이 최대 12% 이상 증가한다. 프로세서의 개수가 많아지게 되면 데이터 포함 트랜잭션용 링의 이용률이 포화상태에 이르게 되고 이에 따라 링 사용권 획득을 위한 큐 대기 시간이 매우 길어지게 되어 성능 저하가 발생해 단순 이중 링보다 성능이 더 나빠지는 경우도 발생한다.

방향 분리 이중 링의 경우 단일 전송 방향의 최적화로 인한 링 순회 속도의 단축과 링을 순회하는 트랜잭션 수를 줄임으로써 얻어지는 링 이용률의 감소를 통해 단일 링 및 다른 이중 링에 비해 높은 성능을 보인다. 특히 프로세서의 개수가 증가할수록 성능 향상은 두드러진다.

5. 결 론

버스 구조의 전기적, 물리적 한계를 극복하기 의해 지점간 링크를 이용한 링 구조의 시스템이 많이 보편화 되었다. 하지만 끊임없는 프로세서의 속도 향상과 메모리 및 지역 버스의 고속화로 인해 현재 IEEE 표준으로 되어 있는 500MHz, 16bit 대역폭의 단방향 링으로는 자원 이용률의 포화상태로 인해 더 이상 성능향상을 기대하기 힘들어진다. 본 논문에서는 클러스터링 기반 다중 프로세서 시스템으로 제작된 PANDA 시스템에서의 대역폭 확장에 대한 필요성 및 확장을 위한 몇 가지 모델을 제시하였다. 각 모델들 간의 특성을 분석하고 여러 가지 환경에 따른 성능을 비교하였다.

두 배 대역폭을 가진 단일 링의 경우 bypass 큐의 엔트리 개수를 절반으로 줄임으로 인한 링 순회 시간의 단축의 효과 및 링의 효율적인 사용으로 인해 이중 링 방식에 비해 전반적으로 우월한 성능을 보인다. 하지만 상용 제품 채택이 불가능하기 때문에 새로운 링크를 개발해야만 하고 이는 시스템의 설계비용과 개발시간을 크게 증가시키는 요소가 된다.

단순 이중 링의 경우 응답 지연시간의 증가와 자원의 비효율적인 사용으로 인해 대부분 좋지 못한 성능을 나타내지만 기존 PANDA 시스템의 설계에 약간의 수정만 거쳐 적용할 수 있어 구현이 용이하다는 장점을 지니며 전역 연결망의 이용률이 낮은 경우에 적용하면 두

배 대역폭 단일 링만큼의 성능을 보일 수 있다. 트랜잭션 분리 이중 링의 경우 데이터 포함 트랜잭션용 링이 포화되기 전에는 응답 지연시간의 단축에 따르는 성능 향상이 보여지나 링이 포화상태에 근접하게 되면 매우 좋지 못한 결과를 나타낸다. 방향 분리 이중 링은 전반적으로 좋은 결과를 나타내는데 이는 응답 지연시간의 단축 및 링 이용률의 감소에 의한 결과임을 본 논문에서는 보여준다.

본 논문에서 제안한 스누핑 방식의 이중 링 시스템과 AV25000과 같은 디렉토리 방식의 이중 링 시스템과의 성능 비교를 현재 진행 중에 있으며 차기 논문의 주제로서 연구해 나갈 예정이다. 그리고 프로세서와 지역버스의 속도 및 캐쉬 크기에 따라 제안한 시스템의 성능에 얼마나 영향을 미치는가에 대한 연구도 필요하다.

전역 연결망의 대역폭을 두 배 확장함에도 불구하고 여전히 시스템에서 전역 연결망이 병목지점이 되는 경우가 발견되고 있다. 이의 해결을 위해 삼중 링 및 다중 링에 대한 연구도 앞으로 필요할 것으로 보여지며 이때 본 논문에서 제시한 여러 모델들을 기반하여 다양한 방식을 섞어 구현한다면 보다 성능을 많이 향상시킬 수 있을 것으로 보여진다. 마지막으로 포화 상태에 놓인 링의 이용률을 감소시키기 위한 방법으로 전송 속도의 고속화에 따른 시스템의 성능에 대한 연구도 앞으로의 과제가 될 것이다.

참 고 문 헌

[1] IEEE Computer Society, IEEE Standard for Scalable Coherent Interface(SCI), Institute of Electrical and Electronics Engineers, August 1993.

[2] Tom Lovett and Russell Clapp, "STiNG : A CC-NUMA Computer System for the Commercial Marketplace," the 23th ISCA(International Symposium on Computer Architecture), pp. 308-317, May 1996.

[3] Z. Vranesic, et al., "The NUMAchine Multiprocessor," Department of Computer Science, Toronto Univ., 1995

[4] L. Barroso and M. Dubois, "The Performance of Cache-Coherent Ring-based Multiprocessors," the 20th ISCA, pp.268-277, May 1993.

[5] 김형호, 지점간 링크를 이용한 스누핑 버스의 설계 및 성능 분석, 서울대학교 컴퓨터 공학과 석사학위 논문, 1996.

[6] Sung Woo Chung, Seong Tae Jhang and Chu Shik Jhon, "PANDA : Ring-Based Multiprocessor System using New Snooping Protocol," ICPADS'98, pp. 10-17, Dec. 1998.

[7] AViiON AV 25000 ccNUMA Server, Available in *http://www.dg.com/aviion/html/av_25000_enterprise_server.html*

[8] Intel Announces Fastest Pentium(R)II Xeon(TM) Processor, Available in *http://www.intel.com/pressroom/archive/releases/wp100698.htm*

[9] Memory-Specifications, Available in *http://developer.intel.com/design/pcisets/memory/Index.htm*

[10] Standard High-Volume Servers, Available in *http://www.intel.com/procs/SERVERS/feature/shv*

[11] Daniel Lenoski, et al., "The Stanford Dash Multiprocessor," IEEE Computer, March 1992.

[12] Zhang, Z. and J. Torrellas, "Reducing Remote Conflict Misses: NUMA with Remote Cache versus COMA," In Proc. of the 3rd IEEE Symp. on High Performance Computer Architecture(HPCA-3), pp. 272-281, Feb. 1997.

[13] J. R. Goodman, "Using Cache Memory to Reduce Processor-Memory Traffic," the 10th ISCA, pp. 124-131, June 1983.

[14] R. H. Katz, S. J. Eggers, D. A. Wood, C. L. Perkins, and R. G. Sheldon, "Implementing a Cache Consistency Protocol," the 12th ISCA, pp. 276-283, June 1985.

[15] SES/Workbench Technical Reference, Scientific and Engineering Software, 1995.

[16] Transaction Processing Performance Council, Overview of the TPC Benchmark A:, Available in *http://www.tpc.org/adetail.html*

장 병 순

1997년 2월 서울대학교 공학사. 1999년 2월 서울대학교 대학원 컴퓨터공학과 석사. 1999년 3월 ~ 현재 서울대학교 대학원 컴퓨터공학과 박사과정. 관심분야는 컴퓨터 구조, 병렬 처리 시스템

정 성 우

1996년 2월 서울대학교 컴퓨터공학과 공학사. 1998년 2월 서울대학교 대학원 컴퓨터공학과 석사. 1998년 3월 ~ 현재 서울대학교 대학원 컴퓨터공학과 박사과정. 관심분야는 컴퓨터 구조, 병렬 처리 시스템, 컴퓨터 시스템 성능 분석.

장 성 태

1986년 2월 서울대학교 전자계산기공학과 공학사. 1988년 2월 서울대학교 대학원 컴퓨터공학과 석사. 1994년 2월 서울대학교 대학원 컴퓨터공학과 박사. 1994년 3월 ~ 현재 수원대학교 전자계산학과 교수. 관심분야는 다중 프로세서컴퓨터구조, 병렬처리, 캐쉬 일관성 유지 프로토콜

전 주 식

1975년 2월 서울대학교 응용수학과 학사. 1977년 2월 한국과학기술원 전산학과 석사. 1983년 2월 미국 Univ. of Utah 박사. 1983년 ~ 1985년 Univ. of Iowa 조교수. 1985년 ~ 현재 서울대학교 컴퓨터공학과 교수. 1994년 ~ 현재 컴퓨터신기술공동연구소 소장. 관심분야는 컴퓨터 구조, 병렬처리, VLSI/CAD