(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(51) Int. Cl.[7]
G06F 11/00

(11) 공개번호 특2001-0041519
(43) 공개일자 2001년05월25일

(21) 출원번호 10-2000-7009689
(22) 출원일자 2000년09월01일
번역문제출일자 2000년09월01일
(86) 국제출원번호 PCT/US1999/04234 (87) 국제공개번호 WO 1999/45470
(86) 국제출원출원일자 1999년02월26일 (87) 국제공개일자 1999년09월10일
(81) 지정국 AP ARIPO특허 : 가나 감비아 케냐 레소토 말라위 수단 시에라리온 스와질랜드 우간다 짐바브웨

EA 유라시아특허 : 아르메니아 아제르바이잔 벨라루스 키르기즈 카자흐스탄 몰도바 러시아 타지키스탄 투르크메니스탄

EP 유럽특허 : 오스트리아 벨기에 스위스 사이프러스 독일 덴마크 스페인 핀랜드 프랑스 영국 그리스 아일랜드 이탈리아 룩셈부르크 모나코 네덜란드 포르투칼 스웨덴

OA OAPI특허 : 부르키나파소 베넹 중앙아프리카 콩고 코트디브와르 카메룬 가봉 기네 기네비쏘 말리 모리타니 니제르 세네갈 차드 토고

국내특허 : 알바니아 아르메니아 오스트리아 오스트레일리아 아제르바이잔 보스니아-헤르체고비나 바베이도스 불가리아 브라질 벨라루스 캐나다 스위스 중국 쿠바 체코 독일 덴마크 에스토니아 스페인 핀랜드 영국 그레나다 그루지야 가나 감비아 크로아티아 헝가리 인도네시아 이스라엘 인도 아이슬란드 일본 케냐 키르기즈 북한 대한민국 카자흐스탄 세인트루시아 스리랑카 라이베리아 레소토 리투아니아 룩셈부르크 라트비아 몰도바 마다가스카르 마케도니아 몽고 말라위 멕시코 노르웨이 뉴질랜드 폴란드 포르투칼 루마니아 러시아 수단 스웨덴 싱가포르 슬로베니아 슬로바키아 시에라리온 타지키스탄 투르크메니스탄 터어키 트리니다드토바고 우크라이나 우간다 미국 우즈베키스탄 베트남 유고슬라비아 짐바브웨

(30) 우선권주장 9/034,624 1998년03월04일 미국(US)
(71) 출원인 인텔 코오퍼레이션 피터 엔. 데트킨
미합중국 캘리포니아 산타클라라 미션 칼리지 블러바드 2200
(72) 발명자 데이비스베리알.
미국오레곤97229포틀랜드노스웨스트슬로컴웨이1418
에스칸데리닉지.
미국아리조나85226챈들러노스퍼스트리트1021
(74) 대리인 특허법인 신성 박해천, 특허법인 신성 원석희, 특허법인 신성 정지원

심사청구 : 있음

(54) 브리지에서의 성능 최적화를 위한 트리거 포인트

요약

PCI 로컬 버스 브리지 (또는 호스트 브리지)뿐만 아니라 PCI-PCI 브리지를 포함하는 브리지 장치(100)의 성능을 튜닝하기 위한 방법 및 장치가 개시된다. 본 발명의 실시예는 다중-버스 컴퓨터 시스템이 애플리케이션 및 브리지 큐 크기에 비추어 튜닝되도록 허가한다. 이러한 애플리케이션은 (디스크 저장 장치(disk storage)에서와 같은) 원 대역폭(raw bandwidth)에 관한 것과 (네트워킹 및 비디오회의(videoconferencing)에서와 같은) 레이턴시에 관한 것을 포함한다. 본 발명의 실시예는 브리지(100)의 판독 및 기록 큐(110, 114, 118, 120)에 의해 충족되는 저장 조건을 지정하는 제어 레지스터(130)를 특징짓는다. 프로그래밍된 저장 조건은 브리지(100)로부터 원하는 성능(스루풋 또는 레이턴시)을 증진시키기 위해 판독 및 기록 트랜잭션 동안에 브리지(100)로 하여금 큐(110, 114, 118, 120)에서 데이터를 전송하거나 또는 큐로부터 데이터를 제거하도록 야기하는 트리거 포인트이다.

대표도

도1

색인어

브리지, 트랜잭션, 기록, 판독, PCI, 스루풋, 레이턴시, 큐, 트리거, 포인트

명세서

기술분야

본 발명은 컴퓨터 버스간 통신에 관한 것으로서, 특히 버스-버스 브리지 장치(bus-to-bus bridge devices)에 관한 것이다.

배경기술

지난 30년에 걸쳐 컴퓨터 버스 아키텍처(computer bus architecture)의 개발은 컴퓨터를 연구 툴(research tool)에서 실용적인 다용도 머신(practical, multi-purpose machine)으로 변천시키는데 영향을 주었다. 이제, 버스는 집적회로 처리 유닛(integrated circuit(IC) processing unit) 내에서 모두 발견될 수 있고, 처리 유닛을 외부 메모리 및 주변 기기와 같은 다른 에이전트(agents)에 연결시킬 수 있다. 하지만, 단일 버스(single bus)의 물리적인 특성은 단일 버스에 부착될 수 있는 에이전트(주변 기기 및 프로세서)의 갯수에 제한을 둔다. 그러므로, 컴퓨터 시스템의 많은 최신 애플리케이션은 기능을 더 확장하기 위해 물리적으로 분리된 다수의 버스를 구비한 다중-버스 아키텍처(multiple-bus architectures)에 의존한다.

가끔, 물리적으로 분리된 버스는 브리지를 이용함으로써 단일 논리 버스(single logical bus)에서 결합된다. 브리지는 물리적으로 분리된 적어도 2개의 버스간 데이터 트래픽을 모니터링하고 제어하는 하드웨어(디지털 하드와이어드 회로(digital hardwired circuitry)), 소프트웨어(하나 또는 그 이상의 프로세서에 의해 실행되는 상위 또는 하위 레벨 명령(commands) 및 명령어(instructions)), 펌웨어(다른 유형의 ROM(read-only memory)에 통상 저장되어 있는 소프트웨어) 또는 이들의 결합을 포함할 수 있다. 브리지는 다른 버스 상의 에이전트간 통신을 용이하게 하도록 하나의 버스 프로토콜을 다른 버스 프로토콜에 인터페이스시킨다.

2개의 버스를 연결시키는 브리지는 통상적으로 투명하도록 구성되기 때문에, 물리적으로 분리된 버스들이 에이전트 및 시스템에 의해 하나의 버스로서 취급될 수 있다. 이러한 결과를 획득하기 위해, 어드레스 공간(address space)이 양쪽 버스 상의 에이전트에 의해 공유된다. 공유된 어드레스 공간 내의 어드레스 범위(address range)를 가지는 요구(request)(판독 또는 기록)는 개시 버스(initiating bus) 상의 개시자 에이전트(initiator agent)에 의해 생성된다. 브리지는 어드레스 범위를 인식하고, 요구를 타깃 버스(target bus) 상의 타깃 에이전트(target agent)로 전송할 수 있다. 따라서, 브리지가 개시자 에이전트를 대신하여 타깃 버스 상에서 요구를 자동적으로 수행한다고 말할 수 있다.

다른 버스 및 브리지 아키텍처가 현재 상태의 컴퓨터 기술(computer technology)에 많이 있다. 최신 컴퓨터 버스의 일례는 PCI(Peripheral Components Interconnect) 버스이다. PCI 버스는 "PCI Special Interest Group (SIG) in PCI Local Bus Specification, Revision 2.1, October 21, 1994"에 의해 통상적으로 정의된 산업 표준, 고성능, 짧은 레이턴시(low latency) 시스템 버스이다. PCI 버스는 이러한 개시를 통해 본 발명의 다양한 실시예의 일정한 원리 및 동작을 설명하는데 이용될 것이다. 하지만, 이들 원리는 다른 다중-버스 아키텍처에 적용될 수도 있다.

또한, PCI SIG는 "PCI-to-PCI Bridge Architecture Specification, Revision 1.0, April 5, 1994"에서 설명된 브리지 아키텍처를 유지한다. 또한, PCI 브리지가 본 발명의 다양한 실시예의 일정한 원리 및 동작을 설명하는 이러한 개시에서 언급된다. 하지만, 이들 원리는 다른 브리지 설계에서도 적용될 수 있다.

브리지를 이용한 트랜잭션(transactions)

트랜잭션은 개시자(initiator)와 타깃(target)간 데이터의 완전한 전송으로서 정의되고, 여기에서 개시자 및 타깃은 브리지에 의해 연결된 서로 다른 물리적인 버스 상에 존재한다. 한 버스에서 다른 버스로 데이터를 전송할 경우에, 통상적으로 브리지는 데이터를 획득 또는 전송하기 위한 타깃 버스에 대한 액세스 요구 및 획득 관련 지연(delay)을 숨기기 위해 다수의 데이터 큐(data queues)를 통상적으로 구현한다. 트랜잭션이 완료될 경우에, 각 트랜잭션이 배포되는 논리 큐(logical queue)를 통상적으로 할당받는다.

통상적으로, 큐는 FIFO(First-In-First-Out) 데이터 구조를 구현하는 메모리(memory) 또는 버퍼(buffer)의 일부분이 될 것이다. FIFO는 데이터 항목(items)이 입력된 동일한 순서로 출력되는 데이터 구조이다. 또한, 이것은 데이터 항목이 한 끝단(end)에서 떨어지도록 셸프(shelf)의 다른 끝단 상으로 데이터 항목을 밀어넣는 것과 유사한 셸프로서 공지되어 있다. 통상적으로, FIFO는 동시에 기록 및 판독될 수 있다. 브리지에서의 FIFO는 동기되지 않는 즉, 정확하게 동일한 속도(same rate)로 전송 및 수신하지 못하는 개시자와 타깃간 데이터 스트림(stream of data)을 버퍼링하는데 유용하다.

여기에서, 정의된 트랜잭션은 타깃에 의해 청구(claim)되는 주어진 어드레스 범위로부터 판독하거나 주어진 어드레스 범위에 기록하기 위해 개시자로부터의 요구를 수반한다. 요구가 브리지에 의해 수락될 경우에, 트랜잭션이 시작하고, 적합한 액세스(access)가 시작된다. 액세스는 셋업(setup)을 위한 유휴 단계(idle phase), 특정 요구를 위한 어드레스 정보가 교환되는 어드레스 단계(address phase) 및 때때로, 데이터가 교환되는 데이터 단계(data phase)를 통상적으로 포함한다.

선택적으로, 요구가 브리지에 의해 거부될 수 있다. 이러한 경우에, 브리지는 재시도 신호(retry

signal)로서 공지된 종료 신호(termination)를 개시자로 발행한다. 이것은 할당된 브리지 큐가 충만된 경우 또는 전송 데이터를 가지고 있지 않은 경우에 발생할 수 있다. 때때로, 신규 요구는 할당에 이용가능한 공백 큐(free queue)가 존재하지 않을 경우에 거부될 수 있고, 여기서 모든 큐는 다른 계류중인 트랜잭션(pending transactions)에 이용된다. 요구가 거부될 경우에, 개시자는 진행중인 트랜잭션(ongoing transaction)을 완료하기 위한 요구를 반복할 수 있거나 또는 신규 트랜잭션을 시작하기 위한 시도를 반복할 수 있다.

요구가 수락되고 제1 액세스가 시작될 경우에, 액세스는 다양한 이유로 개시자, 타깃 또는 브리지에 의해 너무 이른 시기에 종료될 수 있다. 이것이 발생할 경우에, 요구가 반복될 수 있거나 또는 후속 요구(subsequent request)가 트랜잭션을 완료하고 요구된 모든 데이터를 전송하도록 개시자에 의해 발행될 수 있다. 하지만, 원하는 데이터가 다중 액세스에서 전송되도록 트랜잭션을 분할하는 것(splitting)은 추가적인 유휴 및 어드레스 단계를 가지고 있는 추가적인 액세스의 형태로 증가된 오버헤드(overhead)를 도입한다. 증가된 오버헤드는 스루풋(throughput)을 감소시킬 수 있고, 여기서 스루풋은 단위 시간당 브리지를 가로질러 전송되거나 또는 주어진 기간(given period) 동안 평균된 데이터양이다. 반대로, 레이턴시는 개시자 또는 타깃으로 다중-블록 트랜잭션(multiple-block transaction)의 제1 데이터 블록을 제공하는데 필요한 시간으로서 정의된다. 이들 두 성능 표준(criteria)은 이러한 개시를 통해 본 발명의 서로 다른 실시예의 일정한 장점을 설명하는데 이용될 것이다. 브리지가 특정 애플리케이션에 맞추어지도록 스루풋에서의 증가 또는 레이턴시에서의 감소를 허가하는 기술(technique)을 가지는 것이 바람직하다.

기록 트랜잭션

기록 트랜잭션은 PCI 모델에서 포스팅된 기록 트랜잭션(posted write transaction)으로서 통상 수행된다. 이러한 트랜잭션에서, 개시자는 브리지가 개시자의 요구를 수락한 후에 브리지에서 데이터를 큐에 전송한다. 이후, 브리지는 타깃 버스의 제어(control)를 요구하고, 제어(control)를 수신한 후에 데이터를 큐로부터 타깃으로 전송한다. 하지만, 트랜잭션이 타깃 버스 상에서 완료되기 전에 개시 버스 상에서 완료된다.

기록 트랜잭션은 통상 메모리 기록(typical memory write) 및 MWI(memory write and invalidate)를 포함한다. 두 기록 트랜잭션은 MWI가 정수개의 캐시 메모리 라인(cache memory lines)을 위한 것인 반면, 통상 메모리 기록(plain memory write)은 더 적은 데이터양을 기록하는데 이용될 수 있다는 점에서 서로 상이하다.

평균적으로, 브리지의 어느 한 측면 상의 버스가 통상 메모리 기록과 함께 보다는 MWI 트랜잭션과 함께 장시간 동안 활발한 상태(busy)로 유지될 것이다. 이것은 MWI 요구에 응답하도록 구성되지 않은 에이전트를 가지고 있는 타깃 버스를 불필요하게 결합(tie up)시킬 수 있다. 예를 들어, 개시자는 구세대 타깃(older generation target)을 구비한 구 다중-버스 컴퓨터 시스템에 플러그인되는 차세대 주변기기(newer generation peripheral device)일 수 있고, 여기서 개시자는 MWI를 지원하지만, 타깃은 MWI를 지원하지 않는다. 이러한 시스템에서 성능을 개선시키기 위해, 개시자에서의 소프트웨어는 MWI를 지원하지 않는 타깃에 목표된 트랜잭션을 수행하도록 MWI를 발행하지 않고, 대신 통상 메모리 기록을 이용하도록 수정될 수 있다. 하지만, 이러한 변화는 각 장치 상에서 구현될 필요가 있을 것이고, 많은 새로운 장치가 시스템의 존재 기간(lifetime) 동안에 추가될 수 있기 때문에 시스템 운용자에게 성가신 작업(cumbersome task)을 제공할 수 있다. 그러므로, 그 존재 기간 동안에 추가될 수 있는 각 새로운 장치에서 소프트웨어를 수정할 필요없이 다중-버스 컴퓨터 시스템에서 MWI 요구를 처리하기 위한 기술(technique)을 가지는 것이 바람직할 것이다.

판독 트랜잭션

기록 트랜잭션에 덧붙여, 브리지에서의 성능 최적화의 다른 영역은 판독 트랜잭션에 있다. 브리지를 가로지르는 판독 트랜잭션이 포스팅된 트랜잭션으로서 보다는 지연된 트랜잭션으로서 통상 수행된다는 점에서 기록 트랜잭션보다 더 많은 것을 수반한다. 예를 들어, PCI 모델에서 지연된 트랜잭션에서의 PCI 브리지는 초기 요구(initial request)를 완료하는데 필요한 정보를 래치하고, 이때 개시자는 재시도 신호를 수신한다. 이후, 브리지는 개시자를 대신하여 타깃 버스를 통해 초기 요구를 수행한다. 타깃으로부터 리턴하는 데이터 또는 응답은 브리지 큐에 저장된다. 이후, 개시자는 큐로부터 데이터를 검색하고 트랜잭션을 완료하도록 최초의 요구(original request)를 반복해야 한다.

PCI 시스템에서 판독 트랜잭션과 함께, 브리지가 타깃 버스를 통해 판독하는 정확한 데이터양은 지정되지는 않지만, 특정 PCI 명령 유형 및 판독되는 메모리 어드레스 공간이 프리페치 가능한지 여부에 따라 결정될 수 있다. 개시자는 판독하는데 필요한 정확한 데이터양을 알 수는 있지만, PCI 모델 하에서 이러한 데이터양을 지정할 수는 없다.

메모리 공간이 프리페치 가능할 경우에, 판독 요구에 응답하여 브리지는 고정된 소정의 블록 갯수까지 또는 할당된 브리지 큐가 충만될 때까지 타깃으로부터의 데이터를 판독 및 저장한다. 개시자를 대신하여 이러한 투기적 동작(speculative operation)이 개시자로부터의 후속 또는 반복 판독 요구(repeated read requests)의 예측에서 이행된다. 반복 판독 요구의 도착시에, 판독 데이터는 브리지 큐로부터 개시자로 전송되기 시작하지만, 언제든지 개시자에 의해 정지될 수 있다. 이때, 브리지가 타깃으로부터 판독하였지만 개시자에게 전송되지 않은 큐에서의 데이터는 폐기된다.

이러한 판독 프리페칭 방식의 단점은 타깃 버스를 활발한 상태(busy)로 유지시키는 것이 귀중한 버스 시간(valuable bus time)을 낭비하는 것이다. 액세스 동안에, 개시 및/또는 타깃 버스는 데이터가 브리지에서 큐 내부 또는 외부로 전송되면서 점유될 수 있거나 또는 활발한 상태(busy)로 될 수 있다. 점유될 경우에, 버스는 다른 에이전트에 의해 정상적으로 이용될 수 없다. 판독 액세스에서의 일정한 프리페치된 데이터가 차후에 폐기될 경우에, 타깃 버스는 브리지를 통해 데이터 전송을 초래하지 않고 활발한 상태(busy)로 유지되었다. 그러므로, 최적화 방식(optimization scheme)은 이러한 방식에서 타깃 버스가

활발한 상태(busy)로 불필요하게 유지되는 시간을 감소시키기 위해 바람직하다.

브리지 성능 최적화의 다른 영역은 개시자와 타깃간 데이터 전송 속도를 제어하는데 있다. PCI 다중-버스 아키텍처에서, 브리지의 어느 한 측면 상의 개시자 및 타깃 에이전트는 그들간 데이터 흐름 속도를 "스로틀(throttle)"하도록 허용된다. 스로트링(throttling)은 트랜잭션을 위한 데이터가 브리지에 저장되면서, 어느 한 에이전트가 브리지로부터 대기 상태(wait states)를 요구 및 획득할 경우에 발생한다. 대기 상태는 데이터가 브리지에서 수락되거나 또는 브리지로부터 제거되는 속도를 감소시킨다.

하지만, 브리지의 어느 한 측면에서의 데이터 속도가 아주 장시간 동안 한 방향에서 아주 많이 상이할 경우에, 브리지 큐는 충만되거나(기록 트랜잭션에서 개시자를 블로킹함) 또는 비어있게 될 것이고(데이터를 타깃으로 제공하지 않음), 브리지의 내부 및 외부로의 동시적인 데이터 흐름을 정지(halting)시킬 것이다. 그러므로, 전술한 설명(above observation)에 따르면, 스루풋 또는 레이턴시(브리지의 애플리케이션에 종속됨)를 개선시키고 동시적인 흐름의 가능성을 증가시키기 위해 브리지를 통해 데이터 흐름의 최적화를 허용하는 그 이상의 기술(techniques)을 가지는 것이 바람직할 것이다.

발명의 상세한 설명

발명의 요약

일실시예에서, 본 발명은 제1 컴퓨터 버스를 제2 컴퓨터 버스에 연결시키기 위한 브리지에 관한 것이다. 브리지는 제1 및 제2 버스 인터페이스, 트랜잭션의 일부분으로서 한 버스 인터페이스로부터 다른 버스 인터페이스로 전송되는 데이터를 임시 저장하기 위한 큐 및 트랜잭션을 수행하면서 큐에 의해 충족되는 하나 또는 그 이상의 조건을 나타내는 제어 레지스터를 구비한다.

도면의 간단한 설명

도1은 본 발명의 실시예에 따른 브리지를 구비한 컴퓨터 시스템의 일부분을 도시한 도면.

도2는 본 발명의 다른 실시예에 따른 큐의 비-충만 상태(queue non-full state)를 구현하는데 있어 브리지에 의해 수행되는 단계의 흐름도.

도3은 본 발명의 다른 실시예에 따른 기록 요구 제어(write request control)를 구현하는데 있어 브리지에 의해 수행되는 단계의 흐름도.

도4는 본 발명의 다른 실시예에 따른 판독 완료 제어(read completion control)를 구현하는데 있어 브리지에 의해 수행되는 단계의 흐름도.

도5는 본 발명의 다른 실시예에 따른 성능-튜닝가능 브리지(performance-tunable bridge)를 특징짓는 지능형 I/O 서브시스템(intelligent I/O subsystem)의 블록도.

실시예

본 발명의 실시예는 도면을 참조하여 더 상세히 하기에서 설명된다. 설명 목적을 위해, 특정 실시예가 본 발명의 완전한 이해를 제공하기 위해 하기에서 설명된다. 하지만, 이 기술분야에서 통상의 지식을 가진 자는 본 발명이 특정 세부사항 없이도 실시될 수 있다는 것을 이해할 것이다. 또한, 공지된 요소, 장치, 프로세스, 단계 등은 본 발명을 모호하게 만들지 않기 위해 생략 또는 간략화되었다.

도1은 브리지 또는 브리지 장치(100)에 의해 연결된 제1 및 제2 버스(134 및 138)를 구비한 컴퓨터 시스템으로서 본 발명의 실시예를 설명한다. 브리지(100)는 두 버스간 데이터 전송을 버퍼링하는데 이용되는 다수의 로직 큐(110, 114, 118 및 122)를 구비한다. 각 큐는 개시될 경우에 각 신규 트랜잭션이 분리 큐를 할당 받는다는 점에서 매 트랜잭션(per transaction)에 기초하여 이용된다. 또한, 브리지(100)는 큐를 버스에 연결시키는 제1 버스 인터페이스(104) 및 제2 버스 인터페이스(108)를 구비한다. 하기에 상세히 설명되는 본 발명의 다른 실시예에 따른 트리거 포인트(trigger points)를 식별하는 제어 레지스터(130)가 제공된다.

하기의 설명에서, 제1 버스(134)는 트랜잭션이 개시자(140)에 의해 개시될 경우에 개시 버스가 되도록 취해지는 반면, 제2 버스(138)는 타깃(144)에 연결된 타깃 버스이다. 또한, 제1 버스(134)에 연결된 호스트 프로세서(148)는 적당한 값을 가지고 있는 제어 레지스터를 구성하는 프로그램을 실행하는데 이용될 수 있다.

메모리 기록 비-충만 상태

이러한 본 발명의 제1 실시예는 브리지가 후속 기록 요구에서 데이터를 수락하지 않도록 하거나 또는 단일 액세스에서 데이터를 수락하지 않도록 할 경우에 기록 트랜잭션 동안 브리지 큐를 완전히 충만하는 것이 바람직하지 않다고 하는 관념(idea)에 기초한다. 브리지는 특정 트랜잭션을 위한 기록 큐가 충만되기 때문에 데이터를 거부(reject)해야 하는 경우에, 브리지의 내부 및 외부로의 동시적인 흐름이 정지(halted)될 것이다. 또한, 이것은 타깃 버스 또는 에이전트가 활발한 상태(busy)이기 때문에 큐에서의 데이터를 타깃으로 이동시킬 수 없을 경우에 스루풋을 감소시킨다. 또한, 스루풋은 큐가 충만되어 있어 후속 요구가 단일 액세스에서 수락될 수 없을 경우에 개시 버스 상에서 요구되는 다중 액세스의 오버헤드로 인해 감소된다.

이러한 제1 실시예에서, 브리지(100)는 "비-충만" 상태의 기록 큐(110)와 구성된다. 비-충만 상태는 완전히 충만된 상태와 완전히 비어있는 상태 사이의 큐(110)의 조건이다. 큐(110)는 비-충만 상태에 있어야 한다. 즉, 큐(110)는 기록 요구가 브리지에 의해 수락될 수 있게 되기 전에 이용가능한 최소의 빈 공간(free space)(제어 레지스터에서 지정된 하나 또는 그 이상의 데이터 블록)을 가지고 있어야 한다. 그

상태는 버스(134) 또는 버스(138) 상의 호스트 프로세서(148)에 의해 실행되는 사용자 명령 또는 구성 프로그램에 응답하여 제어 레지스터(130)에서 정의될 수 있다. 예를 들어, 도1에 도시된 바와 같이, 두 비트(two bits)는 서로 다른 4개의 비-충만 상태까지 부호화되도록 제어 레지스터에서 이용될 수 있다.

본 발명의 제1 실시예의 동작은 도2의 흐름도 및 도1의 시스템에 따라 하기와 같이 설명될 수 있다. 브리지(100)는 도2에서 동작을 수행하도록 (전술한 바와 같은 하드웨어 및 소프트웨어에서) 구성된다. 블록(204)과 함께 시작하는 개시자(140)로부터의 기록 트랜잭션을 위한 요구는 개시 버스(134) 및 인터페이스(104)를 통해 수신된다. 이후, 브리지는 큐(110)를 트랜잭션에 할당하고, 큐를 개시자로부터의 기록 데이터로 채우기 위해 액세스를 시작한다. 이후, 개시자는 액세스를 종료하고, 후속 기록 요구를 발행함으로써 나중에 트랜잭션을 재개할 수 있다.

후속 기록 요구가 동일한 트랜잭션을 위해 수신될 경우에, 동작은 제어 레지스터(130)가 큐(110)에 대한 비-충만 상태를 식별하도록 단계(208)를 계속한다. 단계(208)는 특정 구현에 따라 각 기록 요구의 검출에 응답하여 수행될 수 있거나, 또는 트랜잭션이 시작되기 전에 또는 시작된 후에 수행될 수 있다. 어느 한 경우에서, 동작은 단계(212)를 계속한다.

단계(212)에서, 비교는 큐(110)가 비-충만 상태에 있는지를 판단하기 위해 이루어진다. 비-충만 상태는 완전히 충만된 상태 또는 완전히 비어있는 상태와 다른 큐의 저장 조건(storage condition)이다. 기록 큐(110)가 비-충만 상태에 도달한 경우에, 큐는 "충만된" 것으로 간주되고, 동작은 단계(220)를 계속한다. 브리지가 후속 기록 요구에 응답하여 재시도를 신호하고, 그 요구는 수락되지 않는다. 한편, 큐에서의 데이터는 개시자에 의한 요구와 무관하게 타깃(144)으로 전송될 수 있다.

하지만, 기록 큐(110)는 후속 요구가 수신될 때까지 비-충만 상태보다 더 비어있을 경우에, 큐가 "비-충만(non-full)"되고, 동작은 단계(220) 대신 단계(216)를 계속한다. 브리지가 기록 요구를 수락하도록 허용된다. 수락될 경우에, 개시 버스를 통한 기록 액세스는 단계(220)에서 시작할 것이고, 트랜잭션을 위한 더 많은 데이터가 큐(110)에 전송될 수 있다. 전술한 후속 요구는 기록되는 전체 데이터양, 큐의 크기 및 큐로부터 타깃으로 데이터를 전송하는 능력에 따라 트랜잭션 동안에 수회(several times) 반복될 수 있다. 이러한 방식으로, 브리지를 동작시키는 장점은 하기의 성능 튜닝 일례에 의해 설명될 수 있다.

네트워크 서버(network server) 또는 워크스테이션(workstation)과 같은 컴퓨터가 도1의 브리지 및 다중-버스 아키텍처와 설계될 수 있다. 브리지를 가로지르는 예측된 트래픽 패턴(expected traffic patterns) 및 버스 상의 서로 다른 에이전트의 데이터 처리 능력이 있을 경우에, 타깃 버스(138)가 비교적 활발한 상태(busy)로 머무를 것이고, 또한/또는 비교적 큰 버스트 트랜잭션(burst transaction)(비교적 대량의 데이터를 가짐)이 개시 버스(134) 상에서 요구되는 것이 수락 가능할 때, 큐의 비-충만 상태는 공백 상태(empty)에 더 가깝게 설정된다. 이것은 브리지가 기록 요구를 수락하기 전에 큐(110)가 공백 상태에 가까워야 한다는 것을 의미한다. 이러한 방식으로, 큐가 단일 데이터 단계에서 모든 기록 요구 데이터를 저장하기 위한 충분한 공간을 가지고 있어 트랜잭션을 개시 버스 상의 다중 액세스로 패킷화하거나 또는 분할하는 것과 관련된 오버헤드를 감소시키는 것이 더 가능해질 것이다. 비교에서, 비-충만 상태가 충만 상태(full)에 가깝게 설정되었을 경우에, 큐가 활발한 상태(busy)의 타깃 버스 또는 요구된 대량의 데이터로 인해 이른 시기에 채워지는 것(또한 그에 따라 개시자가 데이터 전송을 중지하도록 블로킹 또는 강요하는 것)이 더 가능해질 것이다.

한편, 타깃 버스가 비교적 활발하지 않은 상태(quiet)에 있고, 또한/또는 대개 소량의 기록 데이터가 요구된다고 예측될 경우에, 비-충만 상태가 충만 상태(full)에 가깝게 설정된다. 이러한 경우에, 큐가 완전히 충만될(그에 따라 개시자를 블로킹함) 가능성은 (심지어, 비-충만 상태가 충만 상태에 가깝게 설정되더라도) 적으며, 그 이유는 타깃 버스가 큐로부터 데이터를 제거하기에 충분히 빠르고, 또한/또는 소량의 데이터만이 개시 버스 상의 각 액세스에서 전송되기 때문이다. 동시에, 개시자는 거의 큐의 전체 용량을 이용하도록 허용되어 충만된 큐로 인해 개시자를 블로킹할 가능성을 더 감소시킨다.

각 특정 경우에 대해, 큐 동작(queue behavior)은 트래픽 시뮬레이션 또는 개시 및/또는 타깃 버스 상의 실제 트래픽의 모니터링과 큐 크기에 기초하여 판단될 수 있다. 확실히, 시뮬레이션 결과는 브리지에서 기록 큐의 비-충만 상태를 변화시키는 것은 스루풋에 상당한 영향을 준다는 것을 나타내었다.

메모리 기록 요구 제어

본 발명의 제2 실시예는 부분적으로 기록 데이터를 큐(110)로부터 타깃(144)으로 제거 및 전송하기 위해 타깃 버스(138)의 제어를 얼마나 신속하게 요구하는지에 초점을 맞춘다. 이러한 실시예는 버스가 이용가능하게 되면 브리지가 타깃 버스의 제어를 요구하도록 허용하는 것이 항상 바람직한 것은 아니라는 관념에 기초한다. 도1 및 도3에 관련하여 하기에 설명되는 바와 같이, 본 발명의 이러한 실시예는 특히, 개시 및 타깃 버스가 다른 대역 조건을 가지고 있을 경우에 브리지의 레이턴시의 성능 튜닝에 유용할 수 있고, 예를 들어, 개시 버스는 64비트의 범위를 가지고 있는 반면, 타깃 버스는 단지 32비트의 범위를 가지고 있다.

도1에서의 실시예는 버스(134)를 통해 개시자로부터 수신된 데이터 및 기록 트랜잭션의 일부분으로서 버스(138)를 통해 타깃으로 전송될 데이터를 임시 저장하기 위한 큐(110)를 가지고 있는 브리지(100)를 구비한다. 제어 레지스터(130)는 타깃 버스가 개시 버스로부터 수신되었고 큐(110)에 저장되었던 데이터를 전송하기 위한 인터페이스(108)에 의해 요구될 수 있을 경우에 트리거 포인트를 판단하기 위한 메모리 기록 요구 제어 비트(Memory Write Request Control bits)를 포함한다. 그 비트는 브리지(100)가 타깃 버스의 제어를 요구할 수 있게 되기 전에 (기록 트랜잭션의 일부분으로서) 큐에 수신 및 저장되어야 하는 (하나의 데이터 블록보다 더 큰) 제로가 아닌 데이터양(non-zero amount of data)을 지정할 수 있다. 예를 들어, 두 비트는 큐에서 적어도 서로 다른 3개의 QWORD의 양을 부호화하는데 이용될 수 있고, 여기서 각 QWORD는 64비트의 폭(width)을 가지고 있는 데이터 블록이다. 전술한 제1 실시예와 함께 하는 것으로서, 큐(110)는 PCI 브리지에서 포스팅된 메모리 기록 큐일 수 있다.

기록 트랜잭션에서 레이턴시를 감소하는 것이 바람직할 경우에, 기록 요구 제어 저장 조건이 비교적 적

은 데이터 블록으로 설정된다. 지정된 블록 갯수가 수신 및 저장되었을 경우에, 데이터는 큐로부터 타깃으로 전송되기 시작할 수 있다.

하위 저장 조건(lower storage condition)을 설정하는 것이 이로울 수 있는 다른 상황은 큐가 충만되고, 타깃 버스가 획득될 수 있게 되기 전에 개시자를 블로킹하도록 타깃 버스가 비교적 활발한 상태(busy)가 된다고 예측될 경우이다. 이들 조건 하에서, 하위 저장 조건이 큐를 비움에 있어 브리지로 헤드 시작(head start)을 줄 수 있고, 브리지가 타깃을 획득하고, 큐가 충만되기 전에 큐로부터 기록 데이터를 언로딩(unloading)하는 것을 시작하도록 충분한 시간을 제공할 수 있다.

레이턴시보다 스루풋을 개선시키는 것이 바람직할 경우에, 브리지의 주위 환경은 타깃 버스가 비교적 신속하게 획득될 수 있도록 즉, 비교적 활발하지 않은 상태(quiet)의 타깃 버스일 수 있도록, 기록 요구 제어 저장 조건이 큐의 완전한 용량(full capacity)보다 더 소량으로서 대량의 데이터를 지시하기 위해 설정된다. 이것은 타깃 버스가 획득되어 스루풋을 개선시킬 경우에 대량의 데이터가 단일 액세스에서 타깃으로 전송되도록 이용가능하다는 것을 보증하도록 도울 것이다.

또한, 기록 요구 트리거 포인트를 대량으로 설정하는 것이 유익할 수 있는 다른 상황은 개시자(140)가 브리지(100)로부터의 예를 들어, 대기 상태의 형태로, 트랜잭션 동안, 지연을 요구할 수 있는 경우이다. 이것은 개시자가 브리지를 통해 동시적인 데이터 흐름의 가능성을 증가시켜 원하는 스루풋을 증가 또는 유지시키기 위해 그 이상의 데이터의 도착을 지연시키면서, 큐(110)가 트랜잭션 동안에 비어있지 않도록 하고, 타깃 버스를 통해 전송되도록 충분한 데이터를 유지시키는 것을 보증하도록 도울 것이다.

도3은 본 발명의 브리지 실시예에서 기록 요구 제어 메커니즘을 이용하여 기록 트랜잭션을 수행하기 위한 예시적인 방법을 도시한다. 단계(304)에서, 브리지(100)는 기록 트랜잭션을 시작하기 위해 개시자로부터 초기 기록 요구를 수신한다. 동작은 데이터가 개시 버스(308)를 통해 큐(110)로 전송될 경우에 단계(308)를 계속한다. 단계(312)에서, 큐가 제어 레지스터(130)에서 지정된 기록 요구 제어 저장 조건을 충족하는지 즉, 충분한 데이터가 큐에 전송되었는지를 판단한다. "예"이면, 단계(320)에서 타깃 버스가 요구될 것이다. 이후, 타깃 버스가 획득될 경우에, 브리지가 큐로부터 타깃으로 기록 데이터를 전송하기 시작한다. 반면, 신규 기록 데이터(fresh write data)가 개시자로부터 큐에 계속 수신될 것이다.

단계(312)로 리턴하면, 저장 조건이 충족되지 않았을 경우에, 브리지는 타깃 버스를 요구하도록 허가되지 않지만, 큐에서 신규 기록 데이터를 계속 수신할 것이다. 그렇지 않으면,큐 내부 및 외부로의 전송은 서로 무관하게 발생할 것이다.

MWI

기록 요구를 처리하는 본 발명의 또다른 실시예에서, 브리지(100)는 타깃 버스가 MWI 요구를 처리하기 위해 구비되지 않은, 도1에서의 타깃(144)과 같은, 에이전트를 연결하는 것이 예측될 경우에 타깃 버스를 통해 종래의 메모리 기록과 같은 개시 버스 상에서 MWI 요구를 전송하도록 구성된다. 예를 들어, 이러한 에이전트는 구세대의 메모리 제어기 및 브리지를 구비한다. 이러한 방식에서, 각 새로운 개시자의 소프트웨어 및 하드웨어 구성이 변경될 필요가 없다. 컴퓨터 시스템 동작이 MWI 변환 기능(MWI conversion feature)을 인에이블시키기 위해 제어 레지스터(130)를 구성함으로써 통상 메모리 기록 트랜잭션을 위해 최적화될 수 있다. 인에이블될 경우에, 브리지에서의 로직 회로는 타깃 버스 상에서 트랜잭션을 수행하기 전에 MWI 트랜잭션의 명령부(command portion)를 통상 메모리 기록의 명령부로 변경하도록 지시된다.

메모리 판독 프리페치 크기

판독 트랜잭션과 동작하는 본 발명의 실시예는 하기와 같다. 판독 트랜잭션을 처리하는 본 발명의 제1 실시예는 제어 레지스터(130)에서 판독 트랜잭션 동안에 타깃으로부터 프리페치된 데이터의 최대 크기가 비트에 의해 조절되도록 허가한다. 이러한 개선은 브리지가 브리지 판독 큐에 잔류하는 프리페치된 데이터의 요구되지 않은 부분을 폐기하도록 구성될 경우에 브리지 성능을 개선시키는데 특히 효과적일 수 있다.

개시 버스 상에서 초기 판독 요구를 수신하고, 재시도를 신호한 후에, 브리지는 제어 레지스터(130)에서 적합한 비트를 검사함으로써 프리페치되는 최대 데이터 크기를 판단한다. 브리지가 타깃으로부터 (제어 레지스터(130)에서 지정된 최대량까지) 가능한 많은 데이터를 획득하고, 판독 큐(122)에 데이터를 저장한 후에만, 브리지가 개시 버스 상에서 반복 판독 요구를 수락할 수 있고, 큐(122)로부터 개시자(140)로 판독 데이터의 전송을 야기할 수 있다.

시스템 운용자 또는 설계자는 제어 레지스터(130)에서의 프리페치 크기를 데이터 블록의 갯수에 맞춤으로써 차후에 폐기될 수 있는 요구되지 않은 데이터양을 감소시킨다. 따라서, 브리지는 특정 애플리케이션에 튜닝될 수 있다. 예를 들어, 애플리케이션이 대량 저장 애플리케이션(mass storage)이고, 개시자가 자기 디스크 저장 제어기(magnetic disk storage controller)인 타깃(144)으로부터 큰 버스트 판독 트랜잭션을 요구할 수 있을 경우에, 프리페치 크기는 비교적 대량의 데이터가 개시자에 의해 차후에 요구될 것이라는 확신을 갖고 대량으로 설정될 수 있다. 프리페치 크기는 특정 브리지 애플리케이션에 비추어 폐기될 요구되지 않은 데이터양을 감소시켜 타깃 버스를 더 효율적으로 이용하도록 맞추어져야 한다.

메모리 판독 완료 제어

판독 트랜잭션 관련 본 발명의 제2 실시예는 도1 및 도4를 이용하여 설명된다. 실시예는 지연된 메모리 판독 트랜잭션을 위한 개시 버스 인터페이스(104)의 동작을 제어하는 제어 레지스터에서 비트를 제공한다. 비트는 개시자(140)가 큐(122)에 저장된 판독 데이터를 액세스하도록 언제 허용하는지를 판단하는 큐(122)를 위한 판독 완료 제어 저장 조건을 식별한다. 제어 레지스터(130)는 브리지가 개시 버스 상에서 반복 판독 요구를 수락하고 큐(122)로부터의 데이터를 리턴하기 전에 타깃으로부터 수신되고 큐(122)에 저장되어야 하는 최소 데이터 블록의 갯수를 변경하기 위해 프로그래밍될 수 있다. 예를 들어, 두 비

트는 반복 요구가 수락될 수 있게 되기 전에 큐에 수신 및 저장되어야 하는 서로 다른 4개의 QWORDS의 양(64비트 범위의 데이터 블록)을 부호화하는데 이용될 수 있다.

약간 다른 실시예에서, 요구된 모든 판독 데이터가 큐(122)에 수신 및 저장되었을 경우에 즉, 트랜잭션이 타깃 버스 상에서 완료될 경우에, 저장된 데이터는 제어 레지스터(130)에서 지정된 완료 제어 저장 조건이 큐에 의해 충족되었는지 여부와 무관하게 개시자에게 전달될 수 있다.

도4는 본 발명의 실시예를 구현함에 있어 브리지에 의해 수행되는 일련의 단계를 도시한다. 단계(404)와 함께 시작되는, 제1 판독 요구가 판독 트랜잭션을 시작하기 위해 개시 버스(134)로부터 수신된다. PCI 실시예에서, 판독 트랜잭션은 메모리 판독, 메모리 판독 라인(memory read line)(MRL) 및 메모리 판독 멀티플(memory read multiple)(MRM) 중 하나일 수 있다.

(물론, 공백 판독 큐가 이용가능하면) 판독 큐(122)를 트랜잭션에 할당하고, 판독 어드레스 정보를 획득한 후에, 브리지는 단계(406)에서 개시 버스(134) 상에서 재시도를 신호한다. 이후, 브리지는 타깃 버스(138)를 획득하기 위해 시도한다. 성공하면, 타깃(144)으로부터의 판독 데이터의 전송은 단계(408)에서 시작할 것이다. 그렇지 않으면, 브리지는 타깃 버스 상에서 트랜잭션을 시작하기 위해 타깃 버스를 획득하기 위해 계속 시도할 수 있다.

단계(412)에서와 같이 반복 요구가 타깃 버스 상에서의 판독 완료 전에 개시 버스 상에서 수신될 경우에, 단계(416)에서 충분한 판독 데이터가 제어 레지스터(130)에서 지정된 완료 제어 저장 조건을 충족시키기 위해 큐에 수신되었는지를 판단한다. 수신되지 않았으면, 개시자가 단계(406)에서 재시도 신호를 수신한다. 반면, 데이터가 단계(408)에서 타깃으로부터 큐(122)로 계속 전송될 것이다.

하지만, 판단 단계(416)에서 큐가 저장 조건을 충족시키기 위해 타깃으로부터 충분한 판독 데이터를 수신하였을 경우에, 반복 요구가 수락될 수 있고, 판독 트랜잭션이, 바람직하게도 단일 판독 액세스에서, 큐로부터 개시자로 판독 데이터를 전송하기 위해 시도함으로써 개시 버스 상에서 완료될 수 있다.

판독 큐 및 브리지 트래픽 조건의 깊이의 함수로서 완료 제어 저장 조건을 변화시킴으로써, 사용자는 주위 환경에 브리지를 양호하게 정합시키기 위해 브리지 성능을 튜닝할 수 있다. 예를 들어, 개시자가 메모리 제어기인 타깃(144)으로부터 데이터를 판독하려고 하는 디스크 제어기일 경우에, 데이터 백업(data backup)과 같은 대량 저장 애플리케이션을 고려하자. 이러한 경우의 개시자는 레이턴시와 관련될 수 없지만, 높은 스루풋을 원할 것이다. 그러므로, 사용자는 비교적 대량의 데이터를 지시하기 위해 제어 레지스터(130)의 메모리 판독 완료 제어부를 구성할 것이다. 이러한 방식에서, 많은 데이터가 레이턴시를 증가시키면서 트랜잭션을 완료하기 위해 단일 액세스에서 개시자로 전송될 수 있다.

제어 레지스터(130)를 특징짓는 성능-튜닝가능 브리지(performance-tunable bridge)(100)가 도5에서 지능형 I/O 서브시스템(510)의 구성요소로서 이용될 수 있다. I/O 서브시스템(510)은 제3 버스(로컬 버스(516)) 상의 서브시스템 프로세서(512) 및 메모리 제어기(514)를 특징짓는다. 로컬 버스는 각 어드레스 변환 유닛(address translation units)(ATUs)(526 및 528)를 통해 1차 PCI 버스(521) 및 2차 PCI 버스(522)에 연결된다. I/O 서브시스템(510)은 단일 IC로서 구현될 수 있고, 네트워크 서버 머더보드(network server motherboard)와 같은 시스템 애플리케이션의 일부분으로서 이용될 수 있다. I/O 서브시스템(510) 및 PCI 버스(521 및 522)에 덧붙여서, 머더보드는 1차 버스(521)에 연결된 호스트 프로세서 및 메모리와, 2차 버스(522)에 연결된 하나 또는 그 이상의 네트워크 인터페이스 제어기를 구비한다. 네트워크 제어기는 도1에서 타깃(144)의 몇몇 일례에 의해 도시될 수 있다. I/O 서브시스템(510)에서의 브리지(100)는 네트워크 서버 머더보드의 성능을 개선시키기 위해 레이턴시를 최적화시키기 위해 전술한 바와 같이 튜닝될 수 있다.

요약하면, 전술한 바와 같은 본 발명의 실시예는 브리지 및 컴퓨터 시스템에 관한 것이고, 여기에서 브리지의 성능은 데이터 큐 및 특정 시스템 애플리케이션의 깊이(depth)에 비추어 튜닝될 수 있다. 물론, 본 발명의 실시예는 구조 및 구현에서의 다른 변화에 종속된다. 예를 들어, 제어 레지스터(130)에서의 비트는 프로그래머블(programmable)(판독 및 기록), 또는 선택적인 원-타임 프로그래머블(one-time programmable)(프로그래머블 ROM(read-only-memory)), (플래시 메모리와 같은) 인-서킷 프로그래머블(in-circuit programmable), 또는 심지어 (IC의 테스트 배치(batch) 동안 프로그래머블하지만, 판독-전용 제품으로서 고정되는) 테스트 비트일 수 있다. 제어 레지스터(130)는 호스트 프로세서(148)(도1 참조) 또는 도5의 I/O 서브시스템(510)의 서브시스템 프로세서(510)에 의해 액세스될 수 있다.

또한, 전술한 제어 레지스터(130)의 실시예는 서로 다른 데이터 큐의 서로 다른 4개의 트리거 포인트/저장 조건을 나타내는 두 비트를 가지고 있지만, 그 이상의 많은 다수의 큐가 선택적으로 이용될 수 있어 브리지 및 주위 컴퓨터 시스템의 성능 튜닝에서 정밀성(finer granularity)을 허용한다.

그러므로, 본 발명의 범위는 전술된 실시예가 아니라 첨부된 청구항 및 그들의 법적 등가물에 의해서 판단되어야 한다.

(57) 청구의 범위

청구항 1

제1 및 제2 버스 인터페이스;

상기 제1 버스 인터페이스를 통해 개시된 트랜잭션의 일부분으로서 상기 제1 및 제2 버스 인터페이스간에 전송될 데이터를 저장하기 위한 큐; 및

상기 트랜잭션 동안에 상기 큐에 의해 충족되는 적어도 하나의 저장 조건을 나타내는 제어 레지스터

를 포함하는 브리지.

청구항 2

제1항에 있어서,

상기 저장 조건은 상기 브리지가 기록 요구를 수락하기 전에 상기 큐에 의해 충족되는

브리지.

청구항 3

제2항에 있어서,

상기 제어 레지스터는 상기 기록 요구가 수락될 수 있게 되기 전에 상기 큐에서 이용가능한 공간의 양을 지정하는 다수의 비트를 가지고 있는

브리지.

청구항 4

제1항에 있어서,

상기 저장 조건은 상기 브리지가 상기 제2 버스 인터페이스를 통해 제2 버스의 제어를 요구할 수 있게 되기 전에 상기 큐에 의해 충족되는

브리지.

청구항 5

제4항에 있어서,

상기 제어 레지스터는 상기 제2 버스가 기록 트랜잭션의 일부분으로서 상기 데이터를 전송하도록 요구받기 전에 상기 큐에 저장될 데이터의 양을 나타내는 다수의 비트를 가지고 있는

브리지.

청구항 6

제1항에 있어서,

상기 저장 조건은 상기 데이터가 상기 제1 버스 인터페이스를 통해 개시되는 판독 요구를 뒤따르는 후속 판독 요구에 응답하여 상기 큐로부터 상기 제1 버스 인터페이스로 전송될 수 있게 되기 전에 상기 큐에 의해 충족되는

브리지.

청구항 7

제6항에 있어서,

상기 제어 레지스터는 상기 데이터가 상기 후속 판독 요구에 응답하여 상기 개시자로 전송될 수 있게 되기 전에 상기 큐에 저장될 데이터의 양을 나타내는 적어도 하나의 비트를 포함하는

브리지.

청구항 8

제6항에 있어서,

상기 브리지는 PCI 브리지이고, 반복 판독 요구는 지연된 판독 트랜잭션의 일부분인

브리지.

청구항 9

제1항에 있어서,

상기 브리지는 PCI 브리지이고, 상기 큐는 포스팅된 메모리 기록 큐인

브리지.

청구항 10

제1항에 있어서,

상기 큐는 상기 데이터를 저장하기 위한 FIFO 장치를 포함하는

브리지.

청구항 11

제1항에 있어서,

상기 제어 레지스터는 부분적으로 판독가능하고 기록가능한

브리지.

청구항 12

브리지가 제2 버스 인터페이스를 통해 메모리 기록 요구로서 제1 버스 인터페이스에서 수신된 MWI(memory write and invalidate) 요구를 전송하도록 하기 위한 로직 회로

를 포함하는 브리지.

청구항 13

제12항에 있어서,

상기 로직 회로는 레지스터 내의 비트를 포함하는

브리지.

청구항 14

제13항에 있어서,

상기 레지스터는 부분적으로 판독가능하고 기록가능한

브리지.

청구항 15

제1 컴퓨터 버스 상의 개시자를 제2 컴퓨터 버스 상의 타깃에 연결시키기 위한 브리지 - 상기 브리지는 제1 판독 요구에 응답하여 상기 타깃으로부터 판독된 데이터를 저장하기 위한 큐를 구비하고, 또한 상기 브리지는 상기 브리지가 상기 타깃으로부터 판독했던 것보다 더 적은 데이터를 요구하는 상기 개시자에 응답하여 상기 데이터의 요구되지 않은 부분을 폐기하도록 구성됨 - 에 있어서,

상기 제1 판독 요구에 응답하여 상기 타깃으로부터 상기 브리지에 의해 판독되는 데이터의 최대량을 나타내기 위한 제어 레지스터

를 포함하는 브리지.

청구항 16

제15항에 있어서,

상기 제어 레지스터는 부분적으로 판독가능하고 기록가능한

브리지.

청구항 17

데이터 큐를 통해 개시 버스를 타깃 버스로 연결시키는 브리지에서의 연결 방법에 있어서,

상기 개시 버스 상에서 기록 요구를 수신하는 단계;

상기 큐의 비-충만 상태를 판단하기 위해 제어 레지스터의 내용을 검사하는 단계; 및

상기 큐가 상기 비-충만 상태에 있지 않을 경우에 상기 개시 버스 상에서 재시도를 신호하고, 상기 큐가 상기 비-충만 상태에 있을 경우에 데이터를 상기 개시 버스로부터 상기 큐로 전송하기 위해 기록 액세스를 시작하는 단계

를 포함하는 방법.

청구항 18

제17항에 있어서,

상기 제어 레지스터의 내용을 검사하는 단계는 상기 기록 요구의 수신에 응답하여 수행되는

방법.

청구항 19

데이터 큐를 통해 개시 버스를 타깃 버스로 연결시키는 브리지에서의 연결 방법에 있어서,

상기 개시 버스 상에서 기록 요구를 수신하는 단계;

제어 레지스터를 검사함으로써 상기 큐의 기록 요구 제어 조건을 판단하는 단계;

상기 큐가 상기 기록 요구 제어 조건을 충족할 때까지 상기 개시 버스로부터의 데이터를 상기 큐로 전송하는 단계; 및

상기 데이터를 상기 큐로부터 상기 타깃 버스로 전송하기 위해 상기 타깃 버스를 요구하는 단계

를 포함하는 방법.

청구항 20

제19항에 있어서,

상기 기록 요구 제어 조건을 판단하는 단계는 상기 기록 요구의 수신에 응답하여 수행되는

방법.

청구항 21

데이터 큐를 통해 개시 버스 상의 개시자를 타깃 버스 상의 타깃으로 연결시키는 브리지에서의 연결 방법에 있어서,

상기 개시 버스 상에서 판독 요구를 수신하는 단계;

반복 판독 요구가 수신될 때까지 상기 타깃으로부터의 판독 데이터를 상기 큐로 전송하는 단계;

제어 레지스터를 검사함으로써 상기 큐의 완료 제어 조건을 판단하는 단계; 및

상기 큐가 상기 완료 제어 조건을 충족하였을 경우에 상기 판독 데이터를 상기 큐로부터 상기 개시자로 전송함으로써 상기 반복 판독 요구를 완료시키는 단계

를 포함하는 방법.

청구항 22

제21항에 있어서,

상기 완료 제어 조건을 판단하는 단계는 상기 판독 요구의 수신에 응답하여 수행되는

방법.

청구항 23

1차 PCI 버스;

상기 1차 버스에 연결된 호스트 프로세서 및 호스트 메모리;

2차 PCI 버스;

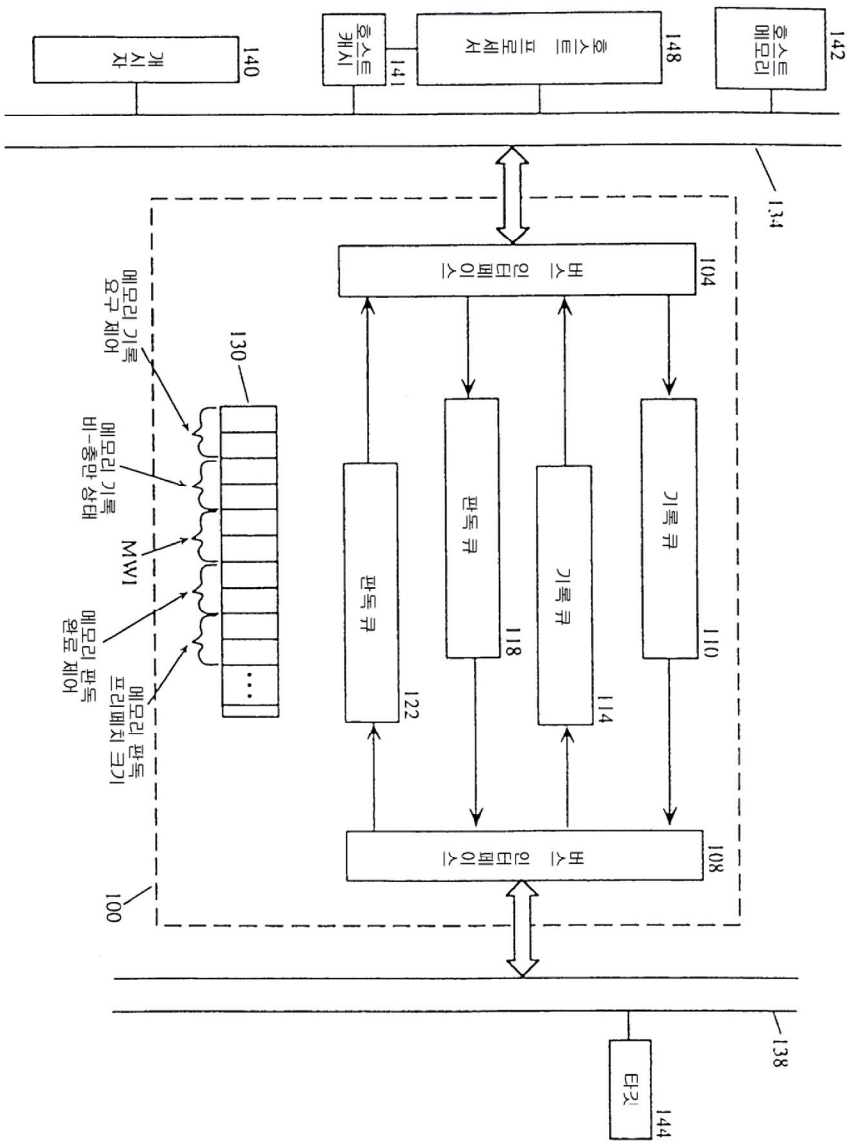
상기 2차 버스에 연결된 네트워크 인터페이스 제어기(network interface controller)(NIC); 및

상기 1차 버스를 상기 2차 버스에 연결시키는 브리지 - 상기 브리지는 호스트 프로세서와 상기 NIC간의 트랜잭션의 일부분으로서 상기 1차 및 2차 버스간에 전송되는 데이터를 임시 저장하기 위한 큐 및 상기 트랜잭션 동안에 상기 큐에 의해 충족되는 적어도 하나의 저장 조건을 나타내는 제어 레지스터를 구비함 -

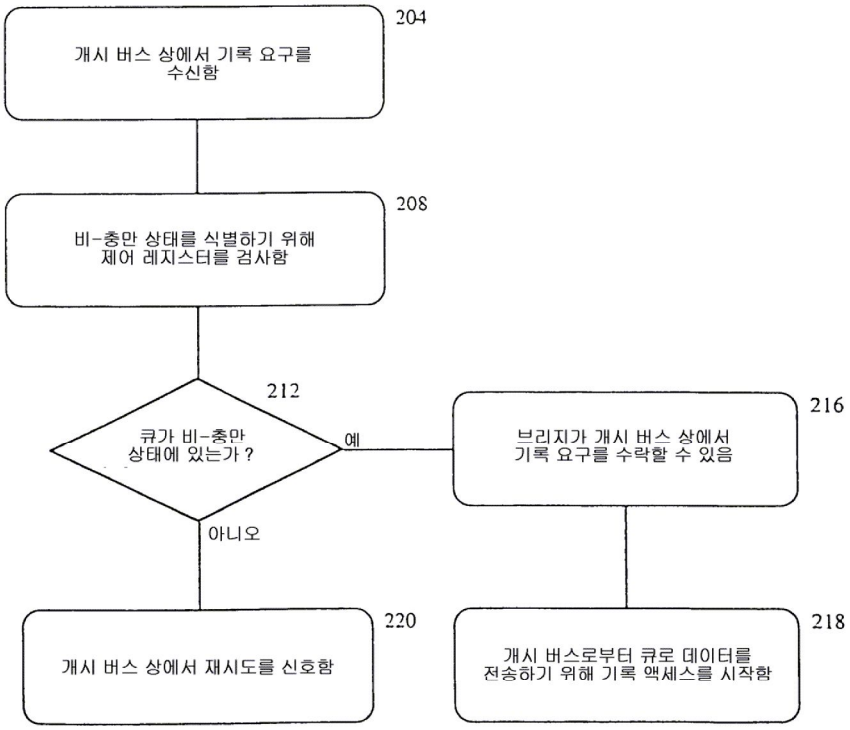
를 포함하는 네트워크 서버 시스템.

도면

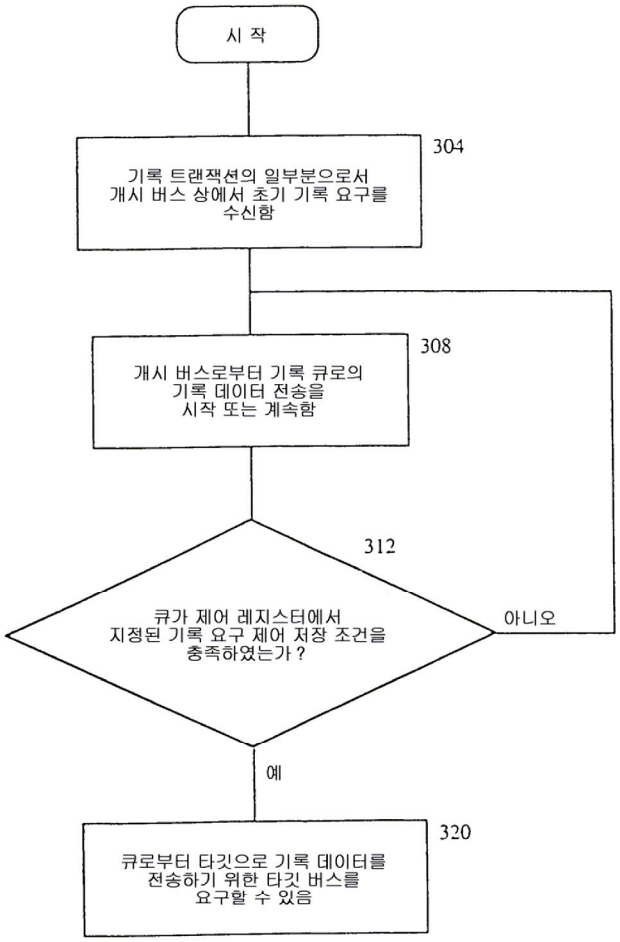
도면1


개시자
140
호스트 캐시
141
호스트 프로세서
148
호스트 메모리
142
134
버스 인터페이스
104
메모리 기록 요구 제어
130
메모리 기록 비-충만 상태
MW1
메모리 판독 완료 제어
메모리 판독 프리페치 크기
판독 큐
122
판독 큐
118
기록 큐
114
기록 큐
110
버스 인터페이스
108
100
138
타깃
144

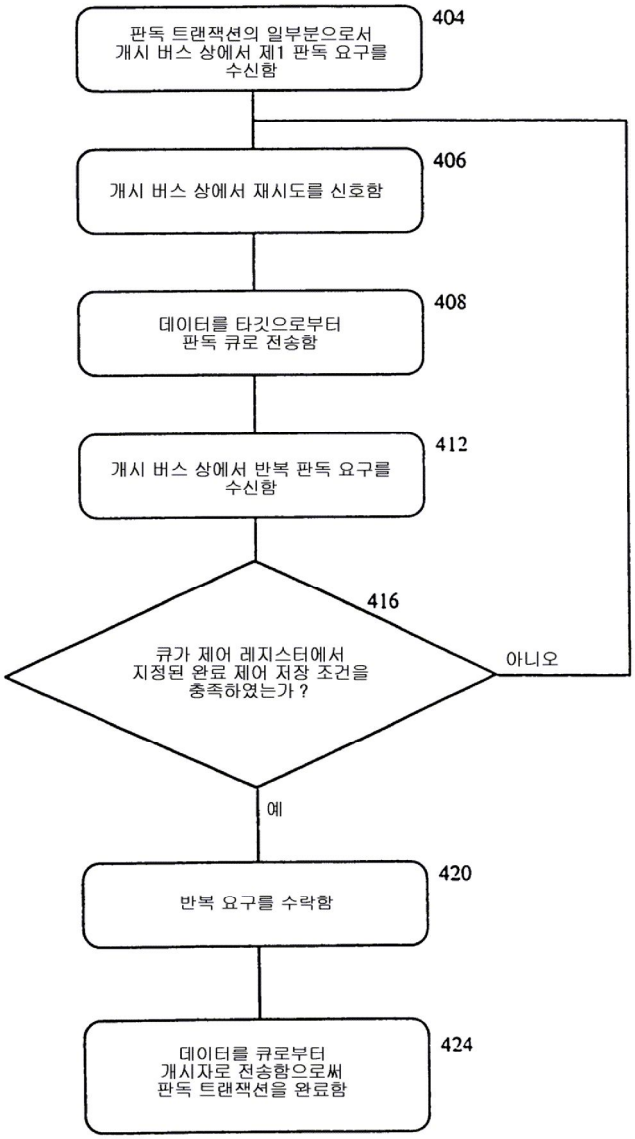
도면2


204
개시 버스 상에서 기록 요구를 수신함
208
비-충만 상태를 식별하기 위해 제어 레지스터를 검사함
212
큐가 비-충만 상태에 있는가 ?
예
216
브리지가 개시 버스 상에서 기록 요구를 수락할 수 있음
아니오
220
개시 버스 상에서 재시도를 신호함
218
개시 버스로부터 큐로 데이터를 전송하기 위해 기록 액세스를 시작함

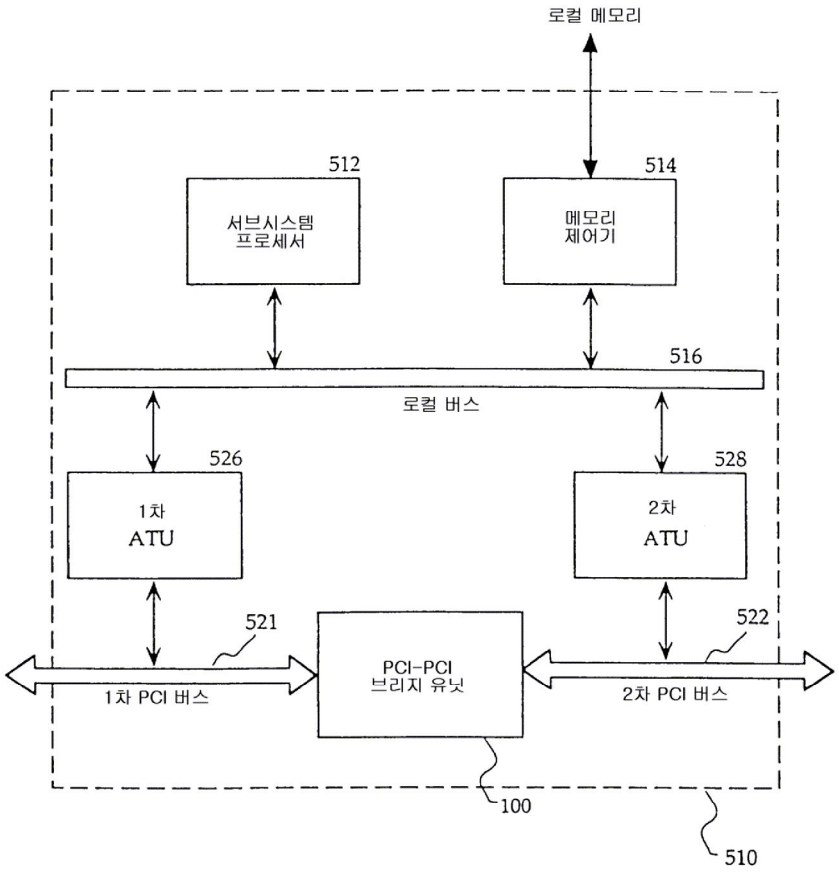
도면3


시 작
304
기록 트랜잭션의 일부분으로서 개시 버스 상에서 초기 기록 요구를 수신함
308
개시 버스로부터 기록 큐로의 기록 데이터 전송을 시작 또는 계속함
312
큐가 제어 레지스터에서 지정된 기록 요구 제어 저장 조건을 충족하였는가 ?
아니오
예
320
큐로부터 타깃으로 기록 데이터를 전송하기 위한 타깃 버스를 요구할 수 있음

도면4


판독 트랜잭션의 일부분으로서
개시 버스 상에서 제1 판독 요구를
수신함
404
개시 버스 상에서 재시도를 신호함
406
데이터를 타깃으로부터
판독 큐로 전송함
408
개시 버스 상에서 반복 판독 요구를
수신함
412
416
큐가 제어 레지스터에서
지정된 완료 제어 저장 조건을
충족하였는가?
아니오
예
반복 요구를 수락함
420
데이터를 큐로부터
개시자로 전송함으로써
판독 트랜잭션을 완료함
424

도면5


로컬 메모리
512
서브시스템
프로세서
514
메모리
제어기
516
로컬 버스
526
1차
ATU
528
2차
ATU
521
522
PCI-PCI
브리지 유닛
1차 PCI 버스
2차 PCI 버스
100
510