(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(51) Int. Cl.[6]
G06F 13/00

(45) 공고일자 2001년08월07일
(11) 등록번호 10-0296633
(24) 등록일자 2001년05월12일

(21) 출원번호	10-1998-0701672	(65) 공개번호	특1999-0044427
(22) 출원일자	1998년03월05일	(43) 공개일자	1999년06월25일
번역문제출일자	1998년03월05일		
(86) 국제출원번호	PCT/US1996/11716	(87) 국제공개번호	WO 1997/11418
(86) 국제출원일자	1996년07월15일	(87) 국제공개일자	1997년03월27일
(81) 지정국	국내특허 : 아일랜드 알바니아 오스트레일리아 바베이도스 불가리아 브라질 캐나다 중국 체코 에스토니아 그루지야 헝가리 이스라엘 아이슬란드 일본 북한 AP ARIPO특허 : 케냐 레소토 말라위 수단 스와질랜드 케냐 EA 유라시아특허 : 아르메니아 아제르바이잔 벨라루스 키르기즈 EP 유럽특허 : 오스트리아 벨기에 스위스 독일 덴마크 스페인 프랑스 영국 그리스 이탈리아 룩셈부르크 모나코 네덜란드 포르투칼 스웨덴 오스트리아 스위스 독일 덴마크 스페인 핀랜드 영국		

(30) 우선권주장 08/523,385 1995년09월05일 미국(US)

(73) 특허권자 인텔 코오퍼레이션 피터 엔. 데트킨
미합중국 캘리포니아 산타클라라 미션 칼리지 블러바드 2200
(72) 발명자 아자노빅 재스민
미국 캘리포니아 95630 폴섬 캠버웰 웨이 119
멀도 로버트 엔.
미국 캘리포니아 95818 새클라멘토 8 애비뉴 1294
도빈스 티모시 엠.
미국 캘리포니아 95630 폴섬 윌로우 크리크 드라이브 409
스레니바스 에디탸
미국 캘리포니아 95030 폴섬 시블리 스트리트 넘버 7 1
세일러 스튜어트 이.
미국 캘리포니아 95630 폴섬 폴섬 랜춰 드라이브 넘버 204 1015
래버 제프리 엘.
미국 캘리포니아 95670 랜초 코르도바 머더 로드 서클 11459
(74) 대리인 박종혁, 장두현, 장용식

심사관 : 오홍수

(54) 동적지연트랜잭션메카니즘

명세서

기술분야

<1> 본 발명은 CPU 버스상에서의 대기시간을 감소시키는 디바이스에 관한 것이다. 더 상세히는, 본 발명은 컴퓨터 시스템에서 분할 또는 지연 트랜잭션을 조절하는 방법 및 장치에 관한 것이다.

배경기술

<2> 버스는 일반적으로 CPU 버스 또는 I/O 버스로 분류된다. 프로세서 버스는 짧고, 일반적으로 고속이고, 메모리-프로세서 대역폭을 최대화하기 위해 메모리 시스템에 연결되어 있다. 대조적으로, I/O버스는 길게 될 수 있고, I/O버스에 접속된 여러 유형의 디바이스를 가질 수 있고, 종종 I/O 버스에 접속된 디바이스의 데이터 대역에서 광대역을 가지고 있다. I/O 버스는 I/O 브리지 메모리 제어기를 통해서 메모리와 인터페이스할 수 있거나, 메모리에 접속한 CPU 버스를 사용할 수 있다.

<3> CPU 버스상의 전형적인 트랜잭션은 몇몇 단계로 분류될 수 있다. 버스상의 에이전트, 프로세서나 I/O 인터페이스중 하나는 버스에 대하여 요청을 만드므로써, 트랜잭션을 개시한다. 중재 단계시에, 버스 에이전트는 그 버스의 다음 점유자를 결정하기 위해 서로를 중재한다. 에이전트가 중재 단계시 중재에 도달했다면, 에이전트는 요청 단계에서 버스상의 요청을 위치시킨다. 트랜잭션의 에러 단계시, 모든 에이전트는 올바른 패리티에 대한 요청을 체크하고, 가능성 있는 에러를 신호할 수 있다. 모든 프로세서는 그것의 캐시의 내용에 반하여 트랜잭션의 요청의 어드레스를 체크하고, 트랜잭션의 스누프(Snoop)단계시의 결과를 지시한다. 스누프 단계는 트랜잭션내의 포인트를 프로세서의 메모리 액세스의 글로벌 관측

포인트와 캐시데이터의 점유권이 결정되는 곳에 마크한다. 그러면, 그 트랜잭션은 트랜잭션의 완료를 간주하는 추가정보가 신호될 수 있는 곳에서 응답 단계로 완료한다. 트랜잭션의 유형을 근거로 하여, 개시된 요청, 개시된 응답, 또는 개시된 스누프 데이터 단계는 트랜잭션에 표현될 수 있다.

<4> 다수의 트랜잭션이 동시에 CPU 버스상에 발행될 수 있지만, 하나의 트랜잭션만이 동시에 그 단계의 각각에 있을 수 있다. 트랜잭션이 긴 주기 시간동안 단계중 임의의 하나를 차지할 때, 이것은 후속 트랜잭션이 CPU 버스상에서 실행하는 것을 연기할 것이다. 외부버스로의 데이터 전달을 포함하는 프로세서 트랜잭션이 수행되었을 때 연속적인 트랜잭션을 발행하도록 파이프라인된 단일 프로세서나 복수의 프로세서를 가진 컴퓨터 시스템은 지연을 겪는다. 예를 들어, 프로세서 I/O 액세스 트랜잭션은 정상적으로 완료하는 데 몇몇의 클록주기가 걸린다. I/O 액세스 트랜잭션은 플레쉬 메모리와 같은 I/O 버스상의 메모리 또는 데이터 기억 디바이스와 같은 I/O 디바이스로의 액세스를 포함하고 있다. 시간신호의 주기가 I/O 디바이스에 전파되거나, 데이터가 I/O 디바이스로부터 검색되는 동안에, 프로세서는 계속해서 데이터 버스 라인을 점유한다. 이것은 다른 트랜잭션이 데이터 버스라인을 사용하는 것을 방지하고, 대기 시간 문제를 발생시킨다.

<5> 그러므로, 필요한 것은 프로세서 I/O 액세스 트랜잭션에 의해 CPU 버스에서 병목됨으로써 생기는 대기시간을 줄이는 장치와 방법이다.

발명의 상세한 설명

<6> 프로세서에 의해 버스상에 발행된 트랜잭션의 지연을 조절하는 트랜잭션 지연 메카니즘이 나타나 있다. 버스 트랜잭션 리코더는 버스상에 발행된 트랜잭션으로부터 인코드된 신호를 처리하며 버스에 연결되어 있다. 트랜잭션 프로세서 유니트는 버스 트랜잭션 리코더와 버스에 연결되어 있다. 트랜잭션 프로세서 유니트는 보류 트랜잭션이 버스상에 발행되도록 대기하고 있다는 것을 지시하는 지시신호를 버스로부터 수신할 때, 그리고 버스상에 발행된 트랜잭션으로부터의 인코드된 신호가 버스상에 발행된 트랜잭션이 지연후보라는 것을 지시할 때 트랜잭션 프로세서 유니트는 버스상에 발행된 트랜잭션을 지연시킨다.

<7> 현재 트랜잭션의 지연을 조절하는 방법이 설명되어 있다. 먼저, 보류 트랜잭션이 존재하는지를 결정한다. 다음, 현재의 트랜잭션이 지연 가능하는지를 결정한다. 최종적으로, 보류 트랜잭션이 존재하고 현재의 트랜잭션이 지연가능하다면 현재의 트랜잭션을 지연시킨다.

도면의 간단한 설명

<8> 본 발명은 아래에 주어진 상세한 설명과 본 발명에 예시된 요소와 여러 특성의 첨부도면으로 충분히 이해할 수 있을 것이다. 명세서와 도면이 본 발명을 특정 실시예에 제한되는 것을 의미하지 않는다. 명세서와 도면은 설명과 이해를 위해 제공된다.

<9> 도 1은 컴퓨터 시스템에 구현된 본 발명의 실시예,

<10> 도 2는 본 발명에 따른 트랜잭션 지연 유니트의 일실시예의 블록도,

<11> 도 3은 본 발명에 따른 호스트 버스 트랜잭션 리코더의 일실시예,

<12> 도 4는 본 발명에 따른 트랜잭션 프로세서의 일실시예,

<13> 도 5는 본 발명에 따른 I/O 버스 인터페이스의 일실시예,

<14> 도 6a는 트랜잭션이 종래의 컴퓨터 시스템에서 발행되는 방법,

<15> 도 6b는 지연 트랜잭션에 걸리는 패널티,

<16> 도 6c는 본 발명의 일실시예에 따라서 지연 트랜잭션이 트랜잭션 지연 유니트로 발행되는 방법,

<17> 도 7은 본 발명의 일실시예에 따라서 I/O 액세스 트랜잭션을 프로세서에서 동적으로 지연시키는 방법을 설명하는 흐름도이다.

실시예

<18> 새로운 트랜잭션 지연 유니트가 설명되어 있다. 다음의 상세한 설명에서 본 발명의 완전한 이해를 돕기 위해 특정하게 설명되어 있다. 그러나, 당업자들은 본 발명이 특정하게 설명되지 않아도 실행될 수 있다는 것을 이해할 수 있을 것이다. 다른 경우에, 공지된 방법, 과정, 부품 및 회로는 본 발명을 모호하지 않게 하기 위해서 설명되지 않았다.

<19> 도 1은 본 발명의 일실시예의 컴퓨터 시스템을 블록도 형태로 설명하고 있다. 이 컴퓨터 시스템은 디지털 데이터를 처리하는 프로세서(101)로 구성되어 있다. 프로세서(101)는 복합명령세트계산(CISC) 마이크로프로세서, 간략화된 명령 세트 계산(RISC) 마이크로프로세서, 매우 긴 명령워드(VLIW) 마이크로프로세서, 또는 다른 프로세서 디바이스일수 있다. 도 1은 본 발명을 사용한 단일 프로세서 컴퓨터 시스템의 한예를 도시하고 있다. 그러나, 본 발명은 또한 복수의 프로세서를 가진 컴퓨터 시스템에 구현될 수 있다는 것을 알고 있다. 프로세서(101)는 컴퓨터 시스템에서 다른 부품과 프로세서(101) 사이에 신호를 전송하는 CPU 버스(110)에 연결되어 있다. 메모리(113)는 CPU 버스(110)에 연결되어 있고, 동적 임의 접근 메모리(DRAM) 디바이스, 정적 임의 접근 메모리(SRAM)디바이스, 또는 다른 메모리 디바이스로 구성되어 있다. 메모리(113)는 프로세서(101)에 의한 실행시의 정보 또는 다른 중간 데이터를 저장한다. 브리지 메모리 제어기(111)는 CPU버스(110)와 메모리(113)에 연결되어 있다. 브리지 메모리 제어기(111)는 컴퓨터 시스템내의 다른 부품 프로세서(101) 및 메모리(113) 사이의 데이터 소통량을 조정하고, 이 부품으로 부터 고속 I/O 버스(120)로 신호를 브리지한다.

<20> I/O 버스(120)는 높은 데이터 처리속도로 작동하는 주변장치를 지원한다. 버스(120)는 단일 버스 또는 복수의 버스 조합일 수 있다. 예를 들어, 버스(120)는 주변 구성요소 내부접속(PCI)버스, 퍼스널 컴퓨터 메모리 카드 국제협회(PCMCIA)버스, VL버스 또는 다른 버스로 구성될 수 있다. 버스(120)는 컴퓨터 시스템내의 부품 사이에 통신링크를 제공한다. 네트워크 제어기(121)는 컴퓨터의 네트워크를 서로 연결하고, 장치간에 통신한다. 디스플레이 디바이스 제어기(122)는 고속 I/O 버스(120)에 연결되어 있다. 디스플레이 디바이스 제어기(122)는 컴퓨터 시스템에 디스플레이 디바이스를 연결시키고, 디스플레이 디바이스와 컴퓨터 시스템 사이에서 인터페이스로서 작용한다. 디스플레이 디바이스 제어기(122)는 단색 디스플레이 접합기(MDA) 카드, 컬러 그래픽 접합기(CGA)카드, 고화질 그래픽 접합기(EGA)카드, 다색 그래픽 접합기(MCGA)카드, 비디오 그래픽 어레이(VGA)카드, 확장 그래픽 어레이(XGA)카드 또는 다른 디스플레이 디바이스 제어기가 될 수 있다. 그 디스플레이 디바이스는 텔레비젼 세트, 컴퓨터 모니터, 평면 디스플레이 또는 다른 디스플레이 디바이스가 될 수 있다. 그 디스플레이 디바이스 는 프로세서(101)에서 디스플레이 디바이스 제어기(122)까지 정보와 데이터를 수신하고, 컴퓨터 시스템의 사용자에게 정보와 데이터를 디스플레이한다.

<21> I/O 버스(130)는 저 처리속도로 작동하는 주변장치사이에 정보를 전달하는데 사용된다. I/O 버스(130)는 단일 버스나 복수버스의 조합이 될 수 있다. 예를 들어, 버스(130)는 업계 표준 아키텍쳐(ISA)버스, 확장업계 표준 아키텍쳐(EISA)버스, 또는 마이크로 채널구조(MCA)버스로 구성될 수 있다. 버스(130)는 컴퓨터 시스템내의 부품사이에 통신링크를 제공한다. 키보드 인터페이스(132)는 키보드 제어기 또는 다른 키보드 인터페이스가 될 수 있다. 키보드 인터페이스(132)는 전용 디바이스가 되거나, 버스제어기나 다른 제어기와 같은 다른 디바이스내에 내장될 수 있다. 키보드 인터페이스(132)는 키보드를 컴퓨터 시스템에 연결시키고, 키보드에서 컴퓨터 시스템으로 신호를 전송한다. 대용량 기억 디바이스(131)는 하드 디스크 드라이브, 플로피 디스크 드라이브, CD-ROM 디바이스, 플레쉬 메모리 디바이스, 또는 다른 대용량 기억 디바이스가 될 수 있다. 사운드의 기록과 재생을 조정하도록 작동하는 오디오 제어기(133)는 I/O 버스(130)에 또한 연결되어 있다. 버스 브리지(123)는 버스(120)를 버스(130)에 연결시킨다. 버스 브리지(123)는 버스(120, 130)사이에 신호를 브리지하기 위해 트랜스레이터를 포함한다.

<22> 본 발명의 일실시예에서, 브리지 메모리 제어기(111)는 트랜잭션 지연 유니트(112)로 구성되어 있다. 트랜잭션 지연 유니트(112)는 보류 트랜잭션이 발행되기를 대기하고 있을 때, 완료하는데 소정량의 시간보다 더 걸리는 현재의 트랜잭션을 지연시키도록 작동한다. 트랜잭션 지연 유니트(112)는 또한 현재의 트랜잭션이 분할 트랜잭션의 성능에 의해 지연되었음을 프로세서(101)에 지시하도록 작동한다. 지연 또는 분할 트랜잭션은 복수의 부품이 동시에 CPU 버스(110) 상에서 통신하고자 할 때, CPU 버스(110)의 효과적인 버스 대역을 증가시킨다. 본 발명의 일실시예에서, 복수의 부품은 CPU 버스(110)상에서 통신하고자 하는 복수의 프로세서가 될 수 있다. 트랜잭션이 지연될 때, 그 트랜잭션은 기록의 경우에, 어드레스, 제어정보 및 데이터를 포함하고 있는 요청 트랜잭션과 트랜잭션의 완료를 신호하는 프로세서에 메시지를 포함하고 있는 응답 트랜잭션으로 분류된다. 그 응답 메시지는 트랜잭션이 성공적이면, 그 트랜잭션이 완료된다는 메시지나 트랜잭션이 성공적이지 않다면, 트랜잭션을 재시도하라는 메시지중 하나가 될 수 있다. 처음부터 지연된 CPU 버스 트랜잭션을 완료하기 위해서, 응답 트랜잭션은 타겟에 의해 전형적으로 발행된다. 판독의 경우에, 응답 트랜잭션은 요청된 데이터를 또한 포함하고 있다. 트랜잭션의 각각은 프로세서(101)가 트랜잭션을 확인하는 순서로 태그(tag)된다. 요청 트랜잭션 다음에, CPU 버스(110)가 해제된다. 이것은 요청 트랜잭션이 I/O 디바이스로 전파되는 시간의 주기동안에 CPU 버스(110)를 위로 개방하고, 다른 요청자가 CPU 버스(110)를 사용할 수 있게 한다. 전형적으로 CPU 버스(110)상의 데이터 버스 라인은 실제 데이터 전송이 일어나지 않을 때 조차, I/O 액세스도안의 프로세서에 의해 유지된다. 분할 트랜잭션에서, CPU 버스(110)상의 데이터 버스라인은 요청 트랜잭션이 판독에 대하여 완료된 후에만 다시 액세스된다.

<23> 분할 트랜잭션은 트랜잭션 또는 하나의 전송을 완료하는 시간을 증가시킨다. CPU 버스(110)는 요청 트랜잭션에 대하여 한번, 그리고 응답 트랜잭션에 대하여 한번, 즉 두 번 요청되어야 한다. 분할 트랜잭션 프로토콜은 다른 부품의 트랙이 통신으로 유지하는 것이 요청된다. 브리지 메모리 제어기(111)는 버스 트랜잭션의 응답부분을 개시하기 위해 요청자와 교신하여야 하고, 그래서 요청자의 신원이 메모리 시스템에 의해 전송되어 보유되어야 한다. 트랜잭션 지연 유니트(112)는 분할 트랜잭션이 전체 컴퓨터 시스템에 대하여 효과적인 CPU 버스(110) 대역을 개선시킬 수 있을 때 만, 분할 트랜잭션을 동적으로 개시한다. 트랜잭션 지연 유니트(112)는 CPU 집중 성능과 I/O 집중 성능간의 시스템 성능 균형과 벤치마크를 일으킨다.

<24> 도 2는 본 발명의 트랜잭션 지연 유니트(112)의 일실시예를 설명하고 있다. 트랜잭션 지연 유니트(112)는 호스트 버스 트랜잭션 리코더(221), 트랜잭션 프로세서(222), CPU 대기시간 타이머(223), 지연 대기열 상태 머신(224), 및 I/O 버스 인터페이스 유니트(225)로 구성되어 있다. 호스트 버스 트랜잭션 리코더 유니트(221)는 CPU 버스(110)상에 현재 발행된 트랜잭션을 기록하도록 작동한다. 라인(231)은 호스트 버스 트랜잭션 리코더를 CPU 버스(110)와 연결시킨다. 본 발명의 일실시예에서, 호스트 버스 트랜잭션 리코더(221)는 CPU 버스(110)로부터 어드레스신호, 바이트 인에이블 신호, 및 캐시 간섭신호와 같은 요청 그룹 신호를 수신한다. 호스트 버스 트랜잭션 리코더(221)는 요청이 만들어지는 시간에 따라서, CPU 버스(110)상의 대기열을 모니터하고, CPU 버스(110)상에 발행되어 있는 트랜잭션으로부터 정보를 기록한다. 호스트 버스 트랜잭션 리코더(221)는 CPU 버스(110)로부터의 요청 그룹 신호를 처리하고, 급송제어신호를 트랜잭션 프로세서(222)에 출력한다.

<25> 트랜잭션 프로세서(222)는 호스트 버스 트랜잭션 리코더(221)에 연결되어 있다. 트랜잭션 프로세서(222)는 라인(233)을 통해 호스트 버스 트랜잭션 리코더(221)로부터 급송제어신호를 수신한다. 트랜잭션 프로세서(222)는 트랜잭션 대기열로부터의 급송제어신호를 호스트 버스 트랜잭션 리코더(221)에서 판독함으로써 현재 CPU 버스(110)상에 발행된 트랜잭션의 유형을 확인한다. 또한, 트랜잭션 프로세서는 라인(232)을 통해 CPU 버스(110)로부터 직접 신호를 수신한다. 트랜잭션 프로세서(222)는 CPU 버스(110)로부터 수신된 신호를 처리함으로써 발행되기를 기다리고 있는 보류 CPU 버스 트랜잭션이 있는지를 결정한다. 트랜잭션 프로세서(222)는 발행되기를 기다리고 있는 보류 CPU 버스 트랜잭션이 있을 때와 현재의 CPU 버스 트랜잭션이 지연후보일 때, 현재의 트랜잭션을 지연시키도록 작동한다. 트랜잭션 프로세

서(222)는 어느 트랜잭션이 지연후보이고, 어느 트랜잭션이 지연되지 않는지를 결정하기 위해 필터로서 작동하도록 프로그램될 수 있다. 본 발명의 일 실시예에서, 프로세서 I/O 트랜잭션은 지연가능후보로서 트랜잭션 프로세서(222)의 상태 머신에 의해 확인된다. 본 발명의 다른 실시예에서, 프로세서 I/O 사이클, 프로세서 인터럽트 긍정 응답 사이클, 및 포스트가 불가능할 때의 프로세서에서 고속 I/O 버스로의 특정 사이클과 프로세서에서 고속 I/O 버스로의 메모리 판독 및 기록 사이클은 지연 가능한 후보로서 트랜잭션 프로세서(205)의 상태 머신에 의해 확인된다. 전형적으로, 시간의 긴 주기동안 데이터 버스 라인을 유지하는 트랜잭션은 지연에 대한 좋은 후보이다. 메모리 액세스 트랜잭션은 고정 대기시간을 가지고 있고, 일반적으로 완료하는데 짧은 주기시간이 걸린다. 그러므로, 메모리 액세스 트랜잭션은 분할 트랜잭션에 대하여 좋은 후보가 아니고 지연되지 않는다.

<26> CPU 대기시간 타이머(223)는 트랜잭션 프로세서(222)에 연결되어 있다. 보류 CPU 버스 트랜잭션이 CPU 버스(110)의 데이터 버스상에서 처리되기를 기다리고 있을 때와 현재의 트랜잭션이 지연후보일 때, 트랜잭션 프로세서는 CPU 대기시간 타이머(223)를 작동시키기 위해 라인(234)을 통해 신호를 보낸다. CPU 대기시간 타이머(223)는 다른 보류 트랜잭션의 존재로 지연되기 전에 완료하기 위해 현재의 트랜잭션에 주어진 시간의 양을 한정하도록 작동한다. CPU 대기시간 타이머(223)는 일부 보류 CPU 버스 트랜잭션의 대기시간을 동시에 기록할 수 있다. 보류 CPU 버스(110) 트랜잭션의 대기시간의 소정의 설정시간을 초과한 후에, CPU 대기시간 타이머(223)는 그 타이머가 종료되었음을 지시하는 신호를 라인(234)을 통해서 트랜잭션 프로세서(222)에 출력한다.

<27> I/O 버스 인터페이스 유니트(225)는 라인(238)을 통해 트랜잭션 프로세서(222)에, 그리고 라인(239)을 통해 I/O 버스(120)에 연결되어 있다. I/O 버스 인터페이스 유니트(225)는 라인(239)을 통해 CPU 버스(110)에 액세스하는 I/O 버스(120)상에 주변장치로부터의 요청을 수신한다.

<28> 트랜잭션 프로세서(222)는 보류 CPU 버스 트랜잭션이 데이터에 존재하는지를 고려하는 정보를 CPU 버스(110)로부터, 보류 CPU 버스 트랜잭션의 대기시간이 소정의 설정시간을 초과했는지를 고려하는 정보를 CPU 대기시간 타이머(223)로부터, 현재의 CPU 버스 트랜잭션이 지연 트랜잭션에 대한 후보인 지를 고려하는 정보를 호스트 버스 트랜잭션 리코더(221)로부터, 그리고 I/O 버스상의 트랜잭션이 메모리(113)를 타겟으로 하는 지를 고려하는 정보를 I/O 버스 인터페이스(225)로부터 수신한다. 그 수신된 정보를 근거로 하여, 트랜잭션 프로세서(222)는 분할 트랜잭션을 발행함으로써 CPU 버스(110)상에 발행된 현재의 트랜잭션을 동적으로 지연시키는지를 결정한다. 트랜잭션 프로세서(222)는 현재의 트랜잭션이 지연될 것임을 지시하는 신호를 라인(235)을 통해서 프로세서(101)에 출력한다.

<29> 지연 대기열 상태 머신(224)은 라인(237)을 통해 트랜잭션 프로세서(222)에 연결되어 있다. 지연 대기열 상태 머신(224)은 CPU 버스(110)와 I/O 버스(120)상에 현재 발행된 트랜잭션의 기록을 트랜잭션 프로세서로부터 수신하고, 발행될 수 있는 새로운 트랜잭션을 한정하도록 작동한다. 발행될 수 없는 트랜잭션에 있어서, 지연대기열 상태 머신(224)은 그 트랜잭션이 재시도되어야 함을 명령하는 신호를 보낸다. 본 발명의 일 실시예에서, 지연 대기열 상태 머신(224)은 I/O 액세스 트랜잭션이 요청되어 다른 I/O 액세스 트랜잭션이 현재 발행될 때마다, 재시도 명령을 발행하도록 작동한다.

<30> 도 3은 본 발명에 따른 호스트 버스 트랜잭션 리코더(221)의 일실시예의 블록도를 설명하고 있다. 호스트 버스 트랜잭션 리코더는 디코더 유니트(331), 트랜잭션 대기열(332), 및 버스 트래커 유니트(333)로 구성되어 있다. 버스 트래커 유니트(333)는 트랜잭션이 CPU 버스(110)상에 발행되었음을 지시하는 신호를 라인(231)을 통해 CPU 버스(110)로부터 수신한다. 버스 트래커 유니트(333)는 트랜잭션 대기열(332)이 어드레스신호, 바이트 인에이블 신호, 및 캐시 간섭신호와 같은 정보를 라인(231)을 통해 CPU 버스(110)로부터 래치하도록 명령하여 트랜잭션 대기열(332)에 보내는 신호를 발생시킨다. 그 제어신호는 라인(344)을 통해서 보내진다. 디코더 유니트(331)는 액세스의 목적지에 대한 인코딩된 정보를 CPU 버스(110)로부터 디코딩하도록 동작한다. 디코더 유니트(331)는 이 정보를 라인(345)을 통해 트랜잭션 대기열(322)에 보낸다. 트랜잭션 대기열(332)은 디코더 유니트(331)로부터 처리된 정보와 CPU 버스(110)로부터 검색된 정보를 각각의 트랜잭션으로부터 저장하는 저장 유니트이다. 트랜잭션 대기열(332)은 단일 트랜잭션로부터 정보를 저장하는 저장 유니트 또는 복수의 트랜잭션으로부터 정보를 기억하는 FIFO와 같은 복수의 저장소자를 가진 저장 유니트일 수 있다. 트랜잭션 대기열(332)은 각각의 트랜잭션을 고려한 정보를 급송된 제어신호로서 급송버스(233)를 통해 트랜잭션 프로세서(222)에 보낸다. 호스트 버스 트랜잭션 리코더(221)는 다른 공지된 논리회로에 의해 또한 이행될 수 있다.

<31> 도 4는 본 발명에 따른 트랜잭션 프로세서(222)의 일실시예의 블록도를 설명하고 있다. 트랜잭션 프로세서(222)는 배열 사이클 상태 머신(432)으로 구성되어 있다. 배열사이클 상태 머신(432)은 보류 CPU 버스 트랜잭션이 CPU 버스(110)로부터 데이터에 존재하는지를 고려한 정보를 라인(232)으로부터, 보류 CPU 버스트랜잭션의 대기시간이 CPU 대기시간 타이머(223)로부터 소정의 설정시간을 초과하였는지를 고려한 정보를 라인(234)으로부터, 현재의 CPU 버스 트랜잭션이 지연된 트랜잭션의 후보인지를 호스트 버스 트랜잭션 리코더(221)로부터 고려한 정보를 급송 버스(233)로부터, I/O 버스상의 트랜잭션이 메모리(113)를 타겟으로 하는지를 I/O 버스 인터페이스 유니트(225)로부터 고려한 정보를 라인(238)으로부터, 그리고 그 보류 트랜잭션이 지연 대기열 상태 머신(224)으로부터 재시도될 수 있는지 아닌지를 고려한 정보를 라인(237)으로부터 수신한다. 수신된 정보를 근거로 하여, 배열사이클 상태 머신(432)은 분할 트랜잭션을 발행함으로써 CPU 버스(110)상에 발행된 현재의 트랜잭션을 동적으로 지연시키는 지를 결정한다. 배열사이클 상태 머신(432)은 현재의 트랜잭션이 지연될 수 있음을 지시하는 신호를 라인(235)을 통해 프로세서(101)에 출력한다. 배열 사이클 상태 머신(432)은 라인(236)을 통해 데이터 경로와 DRAM 제어기에 마이크로명령을 출력한다.

<32> 본 발명의 일실시예에서, 배열 사이클 상태 머신(432)은 현재의 트랜잭션이 I/O 액세스 트랜잭션 일때만 분할 트랜잭션을 발행한다. I/O 액세스 트랜잭션은 전형적으로 완료하는데 긴 주기시간이 걸린다. 트랜잭션이 분할 또는 지연될 때, 프로세서(101)는 액세스 메모리(113)와 같이 다른 트랜잭션을 수행하는 데이터 버스라인을 사용하게 된다. 본 발명의 다른 실시예에서, 배열 사이클 상태 머신(432)은 현재의 트랜잭션이 프로세서 I/O 사이클, 프로세서 인터럽트 긍정 응답사이클, 및 포스트가 불가능할때의 프로세서에서 고속 I/O 버스로의 특정 사이클과 프로세서에서 고속 I/O 버스로의 메모리 판독

및 기록 사이클일 때, 분할 트랜잭션을 발행한다. 본 발명의 다른 실시예에서 배열사이클 상태 머신(432)은 현재의 트랜잭션이 지연후보이고 보류 CPU 버스 트랜잭션이 CPU 버스 대기열에 있을때만 분할 트랜잭션을 발행한다. 본래 지연된 CPU 버스 트랜잭션을 완료하기 위해서 지연응답 트랜잭션을 발행시키기 위한 타겟이 필요하기 때문에 트랜잭션이 지연될때마다 추가 부담이 초래된다. 전형적으로 패널티는 6 호스트 클록이다. CPU 버스에 보류 CPU 버스 트랜잭션이 있는지 와는 무관하게 트랜잭션의 목적지만을 근거로하여 CPU 버스 트랜잭션이 지연되었다면, 그 시스템 I/O 성능은 패널티가 불필요하게 발생되는 상황에서 심각하게 강등될 것이다. 배열 사이클 상태 머신(432)은 컴퓨터 시스템 조건이 보장될 때만 분할 트랜잭션을 발행한다. 라인(232)을 통해 CPU 버스(110)상의 통신량을 모니터함을 근거로 하여 결정이 이루어진다.

<33> 본 발명의 다른 실시예에서, 배열사이클 상태 머신(432)은 현재의 트랜잭션이 지연후보이고 보류 CPU 버스 트랜잭션이 소정의 주기시간을 초과하여 CPU 버스대기열에서 대기하였을 때만 분할 트랜잭션을 발행한다. 본 발명의 이러한 실시예에서, 배열사이클 상태 머신(432)은 CPU 대기시간 타이머를 종료하게 하는 소정의 설정시간보다 큰 보류 CPU 버스 트랜잭션의 대기시간을 기록한다. 그 소정의 설정시간은 프로그램가능 값이거나 하드코드 값 일수 있다. 그 CPU 대기시간 타이머는 완료신호를 보내기 전에 완료하기 위해 트랜잭션에 대해 주어진 양의 시간을 상태 엔진(202)에 배정한다. 시간종료주기를 늘림으로써, I/O 사이클은 덜 빈번하게 지연될 것이고, 이것은 전체 CPU에서 I/O 로의 액세스 대기시간을 낮출 것이다. 시간종료주기를 줄임으로써, CPU 버스대기시간은 감소될 것이다. 프로그램가능 CPU 대기시간 타이머는 모든 사람들이 선택된 컴퓨터 시스템 환경에서 효과적으로 트랜잭션 지연 유니트(112)를 이용하게 한다. 예를 들어, I/O 디바이스로 접근하는 데스크톱 컴퓨터 시스템은 프로세서 계열 서버에 대하여 CPU 버스상의 짧은 대기시간이 더 중요한 까닭으로 더 중요하다. 그러므로 CPU 대기시간 타이머(223)는 데스크톱 컴퓨터 시스템에 대하여 I/O 사이클을 완료하는데 더 많은 시간을 주도록 프로그램될 수 있다. 컴퓨터 시스템 서버에서, 전형적으로 I/O 디바이스보다 더 자주 메모리에 액세스하는 일부 에이전트가 있다. 그러므로 서버환경에서 CPU 대기시간 타이머는 I/O 사이클을 완료하는데 더 적은 시간을 주도록 프로그램될 수 있다.

<34> 본 발명의 다른 실시예에서, 배열 사이클 상태 머신(432)은 현재의 CPU 버스트랜잭션이 발행되고, 그 이후에 I/O 버스 트랜잭션이 메모리(113)를 타겟로 할 때, 분할 트랜잭션을 발행한다. 트랜잭션 프로세서(222)는 다른 공지된 논리회로에 의해 또한 이행될 수 있다.

<35> 도 5는 본 발명에 따른 I/O 버스 인터페이스 유니트(225)의 일실시예의 블록도를 설명하고 있다. I/O 버스 인터페이스 유니트(225)는 개시기 상태 머신(531)과 타겟 상태 머신(532)으로 구성되어 있다. 개시기 상태 머신(531)은 I/O 버스(120)에 액세스하는 CPU 버스(110)로부터 요청을 수신한다. 개시기 상태 머신(531)은 트랜잭션이 완료되었을 때 I/O 버스(120)로의 액세스를 얻고 CPU 버스(110)에 역으로 응답을 보내도록 작동한다. 타겟 상태 머신(532)은 CPU 버스(110)에 액세스하는 I/O 버스(120)로부터 요청을 수신한다. 타겟 상태 머신(532)은 그 요청을 트랜잭션 프로세서(222)에 보내도록 작동한다. 개시기 상태 머신(531)과 타겟 상태 머신(532)은 마이크로 명령을 매개로 하여 트랜잭션 프로세서(222)와 통신한다. 개시기 상태 머신(531)은 I/O 버스(120)에 보내질 정보를 CPU 버스(110)로부터 또한 인코드한다. 타겟 상태 머신은 CPU 버스(110)에 보내질 정보를 I/O 버스(120)로부터 디코드한다. 버스 인터페이스 유니트(225)는 다른 공지된 논리회로에 의해 또한 이행될 수 있다.

<36> 도 6a 는 종래의 컴퓨터 시스템으로 트랜잭션이 CPU 버스상에 발행되는 방법의 예를 설명하고 있다. 도 6a에서, 트랜잭션(X)는 클록주기(0)에서 요청된다. 트랜잭션(X)은 외부 I/O 버스상에서 I/O 디바이스로부터 데이터를 액세스하는 외부 버스 트랜잭션이다.

<37> 클록주기(0)에서 CPU 버스상에 발행되는 다른 트랜잭션이 없기 때문에, 트랜잭션(X)은 클록주기(0-5)시에 이 요청단계와 스누프 단계를 완료한다. 이것은 X_1 시간으로 지시된다. 그 스누프 단계 이후에, 트랜잭션(X)은 클록주기(6)에서 데이터 단계를 시작한다. 클록주기(6-36)시에 트랜잭션(X)은 프로세서가 요청된 데이터를 수신할 때까지, 데이터 버스 라인을 CPU 버스상에 유지한다. 트랜잭션(X)이 그 데이터 버스라인을 유지하는 시간 주기는 X_1'시간으로 도시되어 있다.

<38> 트랜잭션(Y)은 클록주기(8)에서 요청된 메모리 액세스 트랜잭션이다. 클록주기(8)에서, 트랜잭션(Y)은 요청버스라인과 스누프 버스라인을 CPU 버스상에 액세스하고, 요청과 스누프 단계를 시작한다. 트랜잭션(X)은 클록주기(6-35)에서 데이터 버스라인을 유지하기 때문에, 트랜잭션(Y)은 스누프 단계에서 멈춰지고 클록주기가 36일때까지 데이터 단계를 시작할 수 없다. Y_1 시간은 트랜잭션(Y)이 요청단계와 스누프 단계에 있는 시간을 도시한다. Y_1' 시간은 트랜잭션(Y)이 데이터 단계에 있는 시간을 도시한다. 메모리 액세스 트랜잭션이 전형적으로 짧은 주기 시간으로 완료될 수 있지만, 트랜잭션(Y)은 트랜잭션이 I/O 버스로 향한 후에 실행될 때 완료하는데 44클록 주기가 걸린다. CPU 버스상에 데이터 버스 라인을 해제시키기 위해 트랜잭션(X)을 기다리는 것이 필요하기 때문에, 트랜잭션(Y)을 완료하는데 추가시간이 필요하다. 단지 하나의 트랜잭션이 한시간에 데이터 버스라인을 유지할 수 있기 때문에 대기가 필요하다.

<39> 도 6b는 트랜잭션을 분할하는데 초래되는 패널티와 분할 트랜잭션이 CPU 버스상에 발행되는 방법의 예를 설명하고 있다. 도 6b에서, 트랜잭션(X)는 클록주기(0)에서 요청된다. 트랜잭션(X)은 외부 I/O 버스상의 I/O 디바이스로부터 데이터를 액세스하는 외부 버스 트랜잭션이다. 클록주기(0)에서 CPU 버스상에 발행되는 다른 트랜잭션은 없기 때문에, 트랜잭션(X)은 클록주기(0-5)시에 요청 단계와 스누프 단계를 완료한다. 이것은 X1 시간으로 지시된다. 스누프 단계 이후에 트랜잭션(X)은 클록주기(7)에서 데이터 단계를 시작한다. 그 스누프 단계 이후에 트랜잭션(X)은 데이터 버스라인을 유지함으로써 데이터 단계를 시작한다. 그 데이터 단계가 시작한 후의 어느 시간에도 트랜잭션(X)은 지연될 수 있다. 지연시키는 단계에서, 프로세서는 데이터가 현재 이용 불가능하지만 나중에 이용 가능하게 될 것이다는 것을 알린다. 이것에 응답하여, 프로세서는 CPU 버스상에 데이터버스라인을 해제하고, 차후 요청된 트랜잭션이 CPU 버스상의 데이터라인상에 발행되게 한다. 그 데이터 라인이 다른 트랜잭션에 이용가능하게 되는 시간동안에 브리지 메모리 제어기는 독립적으로 그 요청된 데이터를 트랜잭션(X)으로부터 검색한다. 그

요청된 데이터를 검색하는 그 메모리 제어기에 요청되는 시간의 주기는 X_1'시간으로서 설명되어 있다. 그 브리지 메모리 제어기가 트랜잭션(X)로부터 요청된 데이터를 검색한 후에, 그 메모리 제어기는 그 요청된 데이터를 프로세서에 보내는 CPU 버스상에 응답을 발행한다. 응답을 발행시키는 CPU 버스상에 요청되는 시간의 주기는 X_2시간으로 설명되어 있다. 그 트랜잭션의 끝점에 단독으로 CPU 버스 트랜잭션을 자동으로 지연시키는 컴퓨터 시스템은 보류 CPU 버스 트랜잭션이 CPU 버스를 사용할 필요가 있는 것과 무관하게 X_2 시간의 패널티를 초래한다. CPU 버스를 사용하길 기다리는 보류 CPU 버스 트랜잭션이 없을 때, 존재하는 CPU 버스 트랜잭션이 지연없이 완료되게 하는 것이 더 효과적이다.

<40> 도 6c는 지연된 트랜잭션이 본 발명 일실시예에 따른 트랜잭션 지연 유니트로 발행되는 방법의 예를 설명하고 있다. 도 6c에서, 트랜잭션(X)은 클록주기(0)에서 요청된다. 트랜잭션(X)은 I/O 디바이스로부터의 데이터를 외부 I/O 버스상에 액세스하는 외부 버스 트랜잭션이다. 클록주기(0)에서 CPU 버스상에 발행되는 다른 트랜잭션이 없기 때문에, 트랜잭션(X)은 클록주기(0-5)시에 요청단계와 스누프 단계를 완료한다. 이것은 X_1 시간으로 지시된다. 스누프 단계 이후에 트랜잭션(X)은 클록주기(6)에서 데이터 단계를 시작한다. 스누프 단계 이후에, 트랜잭션(X)은 데이터 버스 라인을 유지함으로써 데이터 단계를 시작한다. 이것은 X_1' 시간영역내의 입체 라인에 의해 지시된다.

<41> 트랜잭션(Y)은 클록주기(8)에서 요청된 메모리 액세스 트랜잭션이다. 그 요청 버스라인과 스누프 버스라인은 이용가능하기 때문에, 트랜잭션(Y)은 클록주기(8)에서 요청과 스누프 단계를 시작한다. 트랜잭션(Y)가 클록주기(14)에 의해 요청과 스누프 단계를 완료할 수 있지만, 트랜잭션(X)가 클록주기(14)에서 데이터 단계라인을 유지하고 있기 때문에, 트랜잭션(Y)은 추가 클록주기동안 스누프 단계에 멈춰진다. 본 발명의 트랜잭션 지연 유니트는 트랜잭션(Y)이 보류 CPU 트랜잭션이다는 것과 트랜잭션(X)이 지연후보인 I/O 액세스 트랜잭션이다는 것을 인식한다. 트랜잭션(Y)이 소정된 양의 시간동안 스누프 단계에 스톨된 후에, 트랜잭션(X)이 지연된다. 도 3c는 클록주기(18)에서 트랜잭션(X)이 지연되는 것을 도시하고 있다. 트랜잭션(X)을 지연시키는 단계에서, 그 트랜잭션 지연 유니트는 데이터가 현재 이용가능하지 않지만, 나중에 이용가능하게 될 것이라는 것을 프로세서에 알린다. 이것에 응답하여, 그 프로세서는 CPU 버스상에 데이터 버스라인을 릴리스하고, 트랜잭션(Y)이 CPU 버스상에서 데이터 단계를 시작하게 한다. 그 데이터 버스라인이 트랜잭션(Y)에 대하여 이용가능하게 되는 시간에 브리지 메모리 제어기는 독립적으로 그 요청된 데이터를 트랜잭션(X)으로부터 검색한다. Y_1은 트랜잭션(Y)이 요청과 스누프 단계에 있는 시간주기를 도시한다. Y_1' 은 트랜잭션(Y)이 데이터 단계에 있는 시간주기를 도시한다. 그 요청된 데이터를 검색하는 브리지 메모리 제어기에 의해 필요되는 시간의 주기는 X_1' 시간으로서 설명되어 있다. 브리지 메모리 제어기가 트랜잭션(X)으로부터 요청된 데이터를 검색한 후에, 브리지 메모리 제어기는 요청된 데이터를 프로세서에 보내는 CPU 버스상에 응답을 발행한다. 응답을 발행시키는 CPU 버스상에 요청된 시간의 주기는 X_2 시간에 의해 설명되어 있다.

<42> 도 7은 본 발명의 일실시예에 따라 외부 버스 트랜잭션을 동적으로 지연시키는 방법을 설명하는 흐름도를 설명하고 있다. 먼저, 보류 버스 트랜잭션이 있는지를 결정한다. 이것은 CPU 버스로부터 신호를 판독하여 획득될 수 있다. 이것은 블록(701)에 도시되어 있다. 보류 버스 트랜잭션이 존재하지 않으면, 블록(701)으로 되돌아가라. 보류 버스 트랜잭션이 존재하면, 현재의 요청된 트랜잭션이 자연가능 트랜잭션인지를 결정한다. 이것은 블록(702)에 도시되어 있다. 현재의 버스 트랜잭션의 확인은 기록된 정보를 트랜잭션 리코더에서 판독함으로써 결정될 수 있다. 본 발명의 일 실시예에서, 트랜잭션은 프로세서 I/O 사이클, 프로세서 인터럽트 긍정응답 사이클, 프로세서에서 고속 I/O 버스로의 특정 사이클, 포스트가 불가능할때의 프로세서에서 고속 I/O 버스로의 메모리 판독 및 기록 사이클이라면 지연가능하다. 현재의 버스 트랜잭션에 지연가능하지 않다면, 블록(703)에 도시된 바와 같이 지연시킴이 없이 현재의 트랜잭션을 완료한다. 현재의 CPU 버스 트랜잭션이 지연가능하다면, CPU 대기시간 타이머를 동작시킨다. 그 CPU 대기시간 타이머는 현재의 버스 트랜잭션이 완료하는데 걸리는 시간의 양을 타이밍한다. 이것은 블록(704)에 도시되어 있다. 다음, 블록(705)에 도시된 바와 같이, CPU 대기시간 타이머가 종료되었는지를 결정한다. CPU 대기시간 타이머가 종료되었다면, 블록(706)에 도시된 바와 같이, 현재의 버스 트랜잭션을 지연시킨다. CPU 대기시간 타이머가 종료되지 않았다면, 보류 버스 트랜잭션이 데이터 버스라인을 기다리는 고속 I/O 버스에서 메모리로의 액세스 트랜잭션인 지를 결정한다. 이것은 블록(707)에 도시되어 있다. 데이터 버스라인을 기다리는 고속 I/O 버스에서 메모리로의 액세스 트랜잭션이 있으면, 블록(706)에 도시된 바와 같이, 현재의 버스 트랜잭션을 지연시킨다. 보류 버스 트랜잭션이 고속 I/O 버스에서 메모리로의 액세스 트랜잭션이 아니면, 블록(708)에 도시된 바와 같이 현재의 CPU 버스 트랜잭션이 완료되는지를 결정한다. 현재의 CPU 버스 트랜잭션이 완료하면, 블록(709)에 도시된 바와 같이, 보류 트랜잭션을 시작한 다음, 블록(701)으로 되돌아간다. 현재의 트랜잭션이 완료하지 않으면, 시작으로 되돌아간다.

<43> 상기 설명한 바와 같이, 본 발명은 특정 실시예를 기준으로 하여 설명되어 있다. 그러나, 첨부된 청구범위에 설명된 바와 같이 본 발명의 사상과 범위를 벗어남이 없이 여러 수정과 변경이 이루어질 수 있다는 것을 알 수 있다. 따라서, 상세한 설명과 도면은 제한적이기 보다는 예시적이다. 상기 설명을 참조한 후에 당업자는 본 발명의 여러 수정과 변경을 이해할 수 있기에, 예시적으로 설명되고 도시된 특정 실시예는 제한될 어떤 의도도 없다는 것을 알 수 있다. 그러므로, 특정 실시예의 상세한 설명의 내용은 청구범위를 제한하는 것으로 의도되지 않으며, 이것은 자체로 본 발명에 필수적인 것으로 간주되는 특징만을 상술한다.

(57) 청구의 범위

청구항 1

프로세서에 의해 버스상에 발행된 트랜잭션의 지연을 조절하는 트랜잭션 지연 메카니즘에 있어서,

상기 버스에 연결되어 인코딩된 신호를 상기 버스에 발행된 상기 트랜잭션으로부터 처리하는 버

스 트랜잭션 리코더; 및

상기 버스 트랜잭션 리코더와 상기 버스에 연결된 트랜잭션 프로세서 유니트를 포함하며, 상기 트랜잭션 프로세서 유니트는 보류 트랜잭션이 상기 버스상에 발행되기를 대기하고 있다는 것을 지시하는 지시신호를 상기 버스로부터 상기 트랜잭션 프로세서 유니트가 수신할 때, 및 상기 버스상에 발행된 상기 트랜잭션으로부터의 상기 인코딩된 신호가 상기 버스상에 발행된 상기 트랜잭션이 지연후보임을 지시할 때, 상기 버스상에 발행된 상기 트랜잭션을 지연시키는 것을 특징으로 하는 트랜잭션 지연 메카니즘.

청구항 2

제 1 항에 있어서, 상기 트랜잭션 프로세서는 I/O 액세스 트랜잭션이 상기 지연후보임을 인식하도록 프로그램된 상태 머신을 포함하는 것을 특징으로 하는 트랜잭션 지연 메카니즘.

청구항 3

제 1 항에 있어서, 상기 버스 트랜잭션 리코더는,

상기 버스에 연결되어 상기 인코딩된 신호를 상기 버스로부터 수신하고 상기 인코딩된 신호를 상기 트랜잭션 프로세서 유니트에 의해 판독가능한 목적지 정보로 디코딩하는 디코더 유니트; 및

상기 디코더 유니트에 연결되어 상기 목적지 정보를 저장하는 트랜잭션 대기열을 포함하는 것을 특징으로 하는 트랜잭션 지연 메카니즘.

청구항 4

제 3 항에 있어서, 상기 트랜잭션 대기열은 복수의 보류 트랜잭션으로부터 복수의 목적지 정보를 저장할 수 있는 FIFO인 것을 특징으로 하는 트랜잭션 지연 메카니즘.

청구항 5

제 4 항에 있어서, 상기 트랜잭션 프로세서는 상기 버스상에 발행된 상기 트랜잭션의 상기 목적지 정보를 저장하는 상기 FIFO 내의 제1 저장소자를 판독하기 위한 포인터를 포함하는 것을 특징으로 하는 트랜잭션 지연 메카니즘.

청구항 6

프로세서에 의해 버스상에 발행된 트랜잭션의 지연을 조절하는 트랜잭션 지연 메카니즘에 있어서,

상기 버스에 연결되어 인코딩된 신호를 상기 버스상에 발행된 상기 트랜잭션으로부터 처리하는 버스 트랜잭션 리코더;

보류 트랜잭션이 상기 버스상에서 대기하고 있을 때 상기 버스상에 발행된 상기 트랜잭션을 타이밍하고, 상기 트랜잭션을 완료하는데 소정량의 시간보다 더 걸릴 때 종료 신호를 출력하는 CPU 대기시간 타이머; 및

상기 버스 트랜잭션 리코더, 상기 버스 및 상기 CPU 대기시간 타이머에 연결된 트랜잭션 프로세서 유니트를 포함하고, 상기 트랜잭션 프로세서 유니트는 보류 트랜잭션이 상기 버스상에 발행되기를 기다리고 있음을 지시하는 지시신호를 상기 버스로부터 수신할 때, 상기 버스상에 발행된 상기 트랜잭션으로부터의 상기 인코딩된 신호가 상기 버스상에 발행된 상기 트랜잭션이 지연후보임을 지시할 때, 그리고 상기 CPU 대기시간 타이머가 상기 종료신호를 출력할 때, 상기 버스상에 발행된 상기 트랜잭션을 지연시키는 것을 특징으로 하는 트랜잭션 지연 메카니즘.

청구항 7

제 6 항에 있어서, 상기 CPU 대기시간 타이머는 프로그램가능 타이머인 것을 특징으로 하는 트랜잭션 지연 메카니즘.

청구항 8

제 6 항에 있어서, 상기 보류 트랜잭션이 상기 버스상에 발행될 수 있는지를 결정하는 지연 대기열 상태 머신을 더 포함하는 것을 특징으로 하는 트랜잭션 지연 메카니즘.

청구항 9

제 6 항에 있어서, 상기 호스트 버스 트랜잭션 리코더는,

상기 버스에 연결되어 상기 인코딩된 신호를 상기 버스로부터 수신하고 상기 인코딩된 신호를 상기 트랜잭션 프로세서에 의해 판독가능한 목적지 정보로 디코딩하는 디코더 유니트; 및

상기 디코더 유니트에 연결되어 상기 목적지 정보를 저장하는 트랜잭션 대기열을 포함하는 것을 특징으로 하는 트랜잭션 지연 메카니즘.

청구항 10

프로세서에 의해 버스상에 발행된 트랜잭션의 지연을 조절하는 트랜잭션 지연 메카니즘에 있어서,

상기 버스에 연결되어 인코딩된 신호를 상기 버스로부터 발행된 상기 트랜잭션으로부터 처리하는 버스 리코더 수단;

보류 트랜잭션이 상기 버스상에서 기다리고 있을 때 상기 버스에 발행된 상기 트랜잭션을 타이밍하고, 상기 트랜잭션이 완료하는데 소정량의 시간보다 더 걸릴 때 종료신호를 출력하는 타이밍수단; 및

상기 버스 리코더수단, 상기 버스, 및 상기 타이밍수단에 연결된 프로세서 수단을 포함하고, 상기 프로세서 수단은 보류 트랜잭션이 상기 버스상에 발행되기를 기다리고 있음을 지시하는 지시 신호를 상기 프로세서 수단이 상기 버스로부터 수신할 때, 상기 버스상에 발행된 상기 트랜잭션으로부터의 상기 인코딩된 신호가 상기 버스상에 발행된 상기 트랜잭션이 지연후보임을 지시할 때, 그리고 상기 타이밍 수단이 상기 종료신호를 출력할 때 상기 버스상에 발행된 상기 트랜잭션을 지연시키는 것을 특징으로 하는 트랜잭션 지연 메카니즘.

청구항 11

컴퓨터 시스템내의 구성요소들 사이에 링크를 제공하는 버스;

상기 버스에 연결되어 디지털 데이터를 처리하는 프로세서;

상기 버스에 연결되어 디스플레이 디바이스를 상기 컴퓨터 시스템에 연결시키는 디스플레이 디바이스 제어기;

상기 버스에 연결되어 정보를 저장할 수 있는 외부 메모리; 및

상기 버스에 연결되어 인코딩된 신호를 상기 버스상에 발행된 상기 트랜잭션으로부터 처리하는 버스 트랜잭션 리코더, 보류 트랜잭션이 상기 버스상에서 기다리고 있을 때 상기 버스상에 발행된 상기 트랜잭션을 타이밍하며 상기 트랜잭션이 완료하는데 소정량의 시간보다 더 걸릴 때 종료신호를 출력하는 CPU 대기시간 타이머, 및 상기 트랜잭션 리코더와 상기 버스와 상기 CPU 대기시간 타이머에 연결된 트랜잭션 프로세서 유니트를 포함하는, 상기 프로세서에 의해 상기 버스상에 발행된 트랜잭션의 지연을 조절하는 트랜잭션 지연 메카니즘을 포함하고,

상기 트랜잭션 프로세서 유니트는 보류 트랜잭션이 상기 버스상에서 발행되기를 기다리는 것을 지시하는 지시신호를 상기 버스로부터 수신할 때, 상기 버스상에 발행된 상기 트랜잭션으로부터의 상기 인코딩된 신호가 상기 버스상에 발행된 상기 트랜잭션이 지연후보임을 지시할 때, 그리고 상기 CPU 대기시간 타이머가 상기 종료신호를 출력할 때 상기 버스상에 발행된 상기 트랜잭션을 지연시키는 것을 특징으로 하는 컴퓨터 시스템.

청구항 12

현재 트랜잭션의 지연을 조절하는 방법에 있어서,

보류 트랜잭션이 존재하는지를 결정하는 단계;

상기 현재 트랜잭션이 지연가능한 지를 결정하는 단계; 및

상기 보류 트랜잭션이 존재하고 상기 현재 트랜잭션이 지연가능하다면 상기 현재 트랜잭션을 지연시키는 단계를 포함하는 것을 특징으로 하는 방법.

청구항 13

제 12 항에 있어서, 상기 보류 트랜잭션이 존재하는지를 결정하는 단계는 CPU 버스로부터 직접 신호를 처리함으로써 달성되는 것을 특징으로 하는 방법.

청구항 14

제 12 항에 있어서, 상기 현재 트랜잭션이 지연가능한 지를 결정하는 단계는 트랜잭션 리코더로부터 급송제어신호를 처리함으로써 달성되는 것을 특징으로 하는 방법.

청구항 15

현재 트랜잭션의 지연을 조절하는 방법에 있어서,

보류 트랜잭션이 존재하는지를 결정하는 단계;

상기 현재 트랜잭션이 지연가능한지를 결정하는 단계;

상기 현재 트랜잭션이 완료하는데 소정의 주기시간보다 더 오래 걸리는지를 결정하는 단계; 및

상기 보류 트랜잭션이 존재하고, 상기 현재 트랜잭션이 지연가능하며, 상기 현재 트랜잭션이 완료하는데 소정의 주기시간보다 더 오래 걸리면, 상기 현재 트랜잭션을 지연시키는 단계를 포함하는 것을 특징으로 하는 방법.

청구항 16

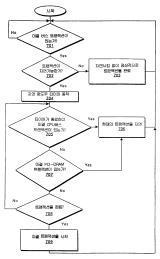
제 15 항에 있어서, 상기 현재 트랜잭션이 완료하는데 소정의 주기시간보다 더 오래걸리는 지를 결정하는 단계는 상기 현재 트랜잭션을 CPU 대기시간 타이머로 타이밍함으로써 달성되는 것을 특징으로 하는 방법.

요약

상기 버스상에 발행된 트랜잭션(#702)의 지연을 컴퓨터 시스템내의 프로세서에 의해 조절하는 장치 및 방법이 설명되어 있다. 버스에 연결된 버스 트랜잭션 리코더는 버스상에 발행된 트랜잭션으로부터 인코드된 신호를 처리한다. 버스에 연결된 라인은 보류 트랜잭션(#701) 요청이 버스상에 발행될 때 지시 신호를 보낸다. CPU 대기시간 타이머는 새로운 보류 트랜잭션이 버스상에 대기하고 있을 때 버스상에 현

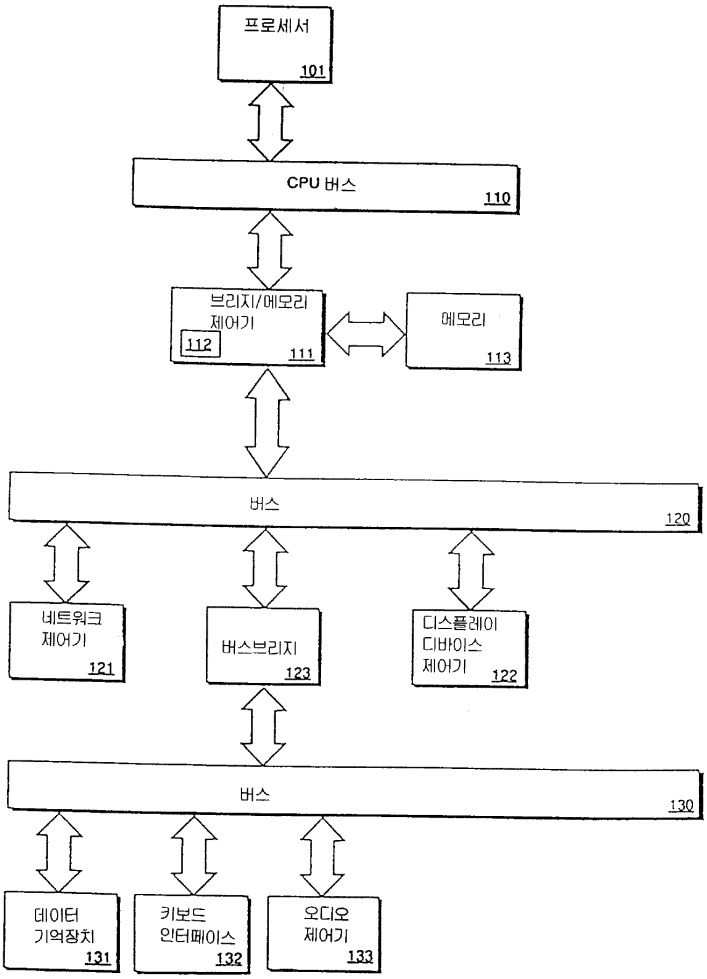
재의 트랜잭션을 타이밍한다. CPU 대기시간 타이머는 트랜잭션이 완료하는데 소정량의 시간보다 더 걸릴 때 종료신호(#705)를 출력한다. 트랜잭션 프로세서 유니트는 버스 트랜잭션 리코더, 라인 및 CPU 대기시간 타이머에 연결되어 있다. 트랜잭션 프로세서 유니트는, 버스상에 발행된 트랜잭션으로부터의 인코딩된 신호가 버스상에 발행된 트랜잭션이 지연후보임을 지시할 때, 그리고 CPU 대기시간 타이머가 종료신호(#708)를 출력할 때 버스상에 발행된 트랜잭션을 지연시킨다.

대표도

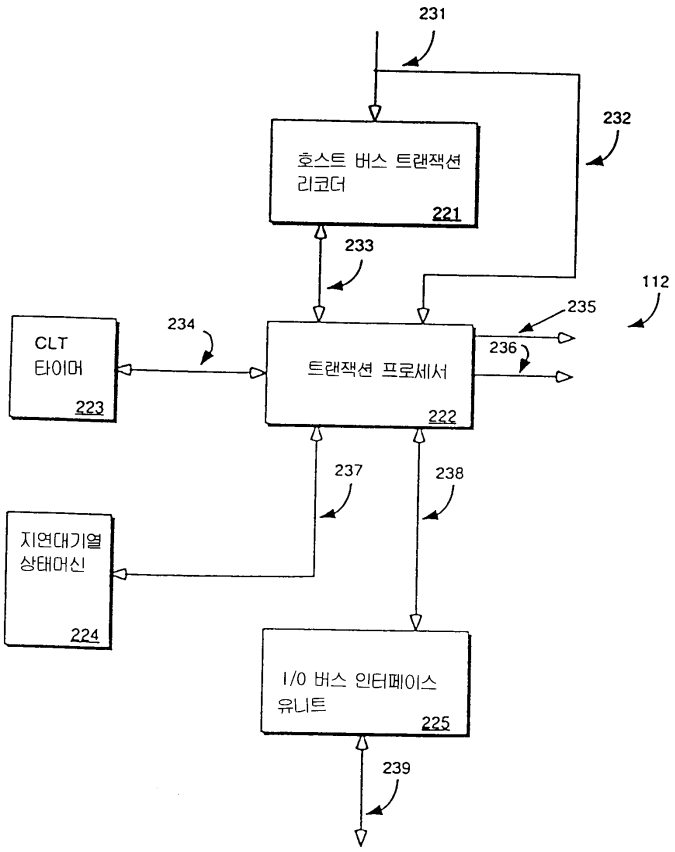


도면

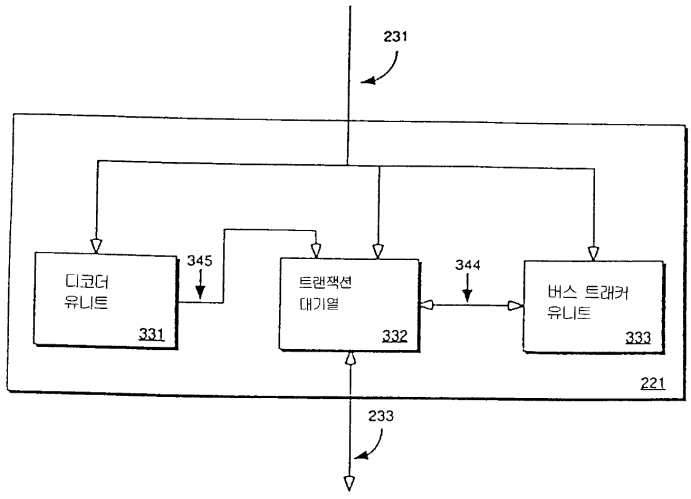
도면1



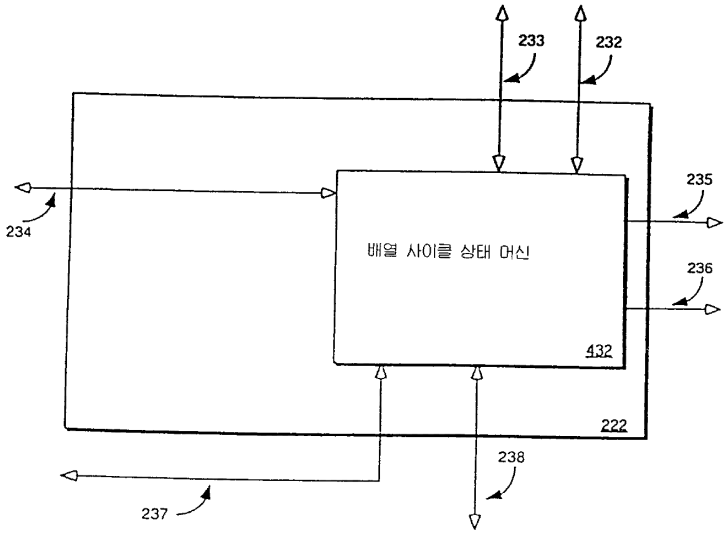
도면2


231
232
호스트 버스 트랜잭션
리코더
221
233
112
235
234
CLT
타이머
223
트랜잭션 프로세서
222
236
237
238
지연대기열
상태머신
224
I/O 버스 인터페이스
유니트
225
239

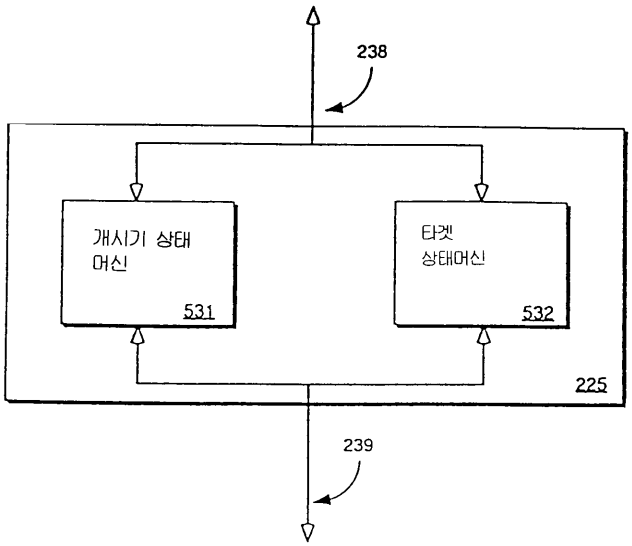
도면3


231
345
344
디코더
유니트
331
트랜잭션
대기열
332
버스 트래커
유니트
333
221
233

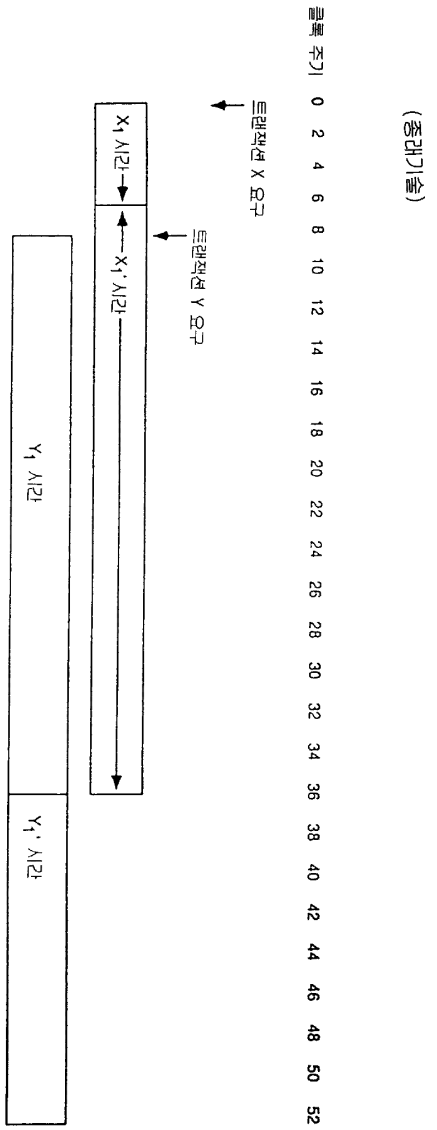
도면4


233
232
234
235
배열 사이클 상태 머신
236
432
222
238
237

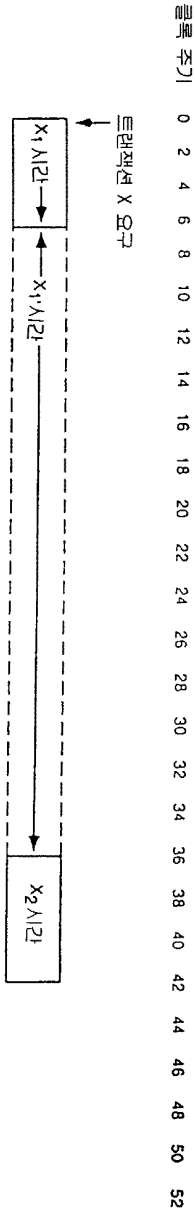
도면5


238
개시기 상태
머신
531
타겟
상태머신
532
225
239

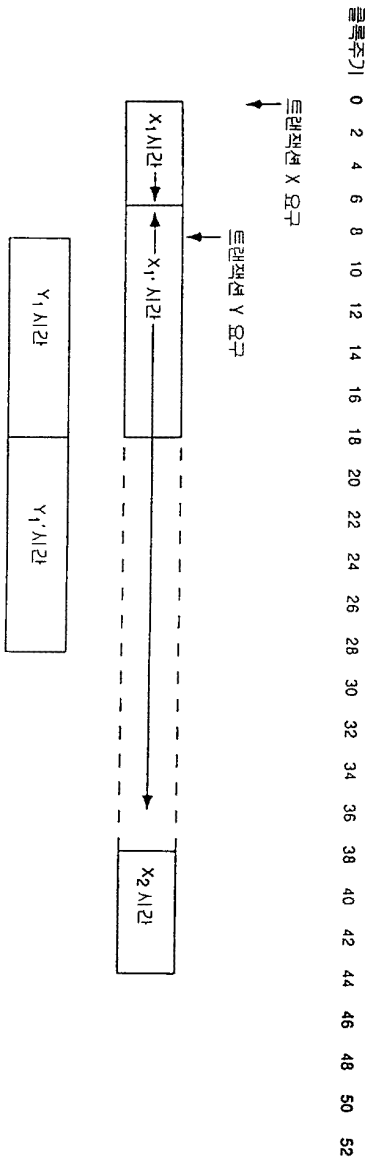
도면6a


(종래기술)
클록 주기
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52
트랜잭션 X 요구
트랜잭션 Y 요구
X1 시간
X1' 시간
Y1 시간
Y1' 시간

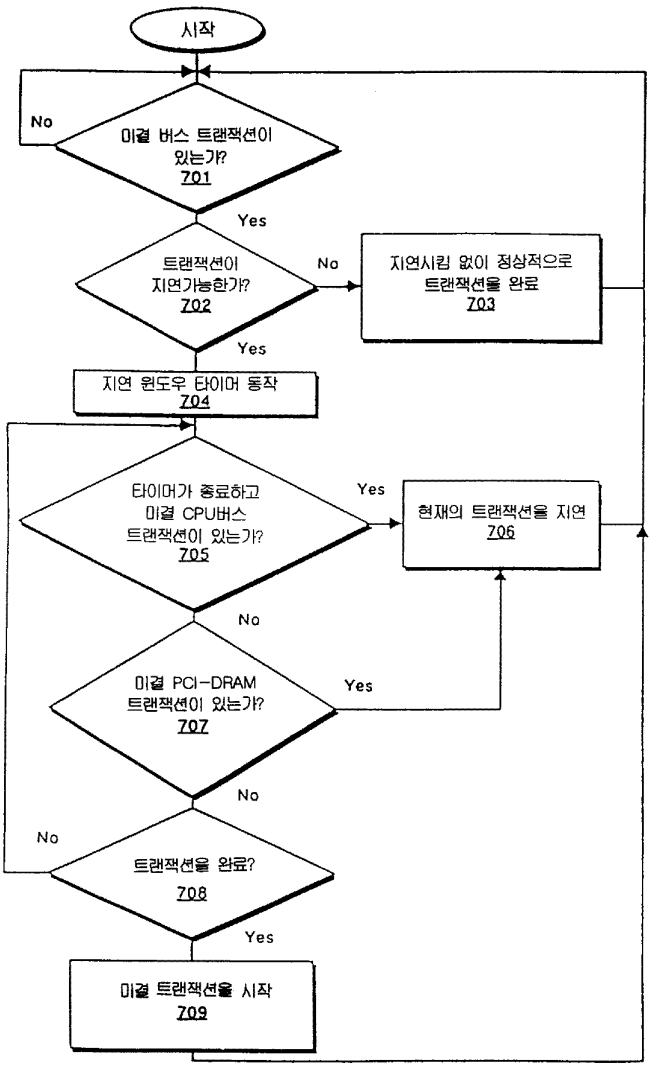
도면6b


클록 주기
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52
트랜잭션 X 요구
x_1 시간
x_1 시간
x_2 시간

도면6c


클록주기
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52
트랜잭션 X 요구
트랜잭션 Y 요구
X1시간
X'1시간
X2시간
Y1시간
Y'1시간

도면7


시작
No
미결 버스 트랜잭션이 있는가?
701
Yes
트랜잭션이 지연가능한가?
702
No
지연시킴 없이 정상적으로 트랜잭션을 완료
703
Yes
지연 윈도우 타이머 동작
704
타이머가 종료하고 미결 CPU버스 트랜잭션이 있는가?
705
Yes
현재의 트랜잭션을 지연
706
No
미결 PCI-DRAM 트랜잭션이 있는가?
707
Yes
No
No
트랜잭션을 완료?
708
Yes
미결 트랜잭션을 시작
709