(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2008년07월04일
(11) 등록번호 10-0843536
(24) 등록일자 2008년06월27일

(51) Int. Cl.
G06F 12/10 (2006.01) *G06F 12/08* (2006.01)
G06F 12/00 (2006.01)
(21) 출원번호 10-2005-7008348
(22) 출원일자 2005년05월11일
심사청구일자 2006년09월06일
번역문제출일자 2005년05월11일
(65) 공개번호 10-2005-0088077
(43) 공개일자 2005년09월01일
(86) 국제출원번호 PCT/GB2003/005108
국제출원일자 2003년11월21일
(87) 국제공개번호 WO 2004/053698
국제공개일자 2004년06월24일
(30) 우선권주장
10/318,541 2002년12월12일 미국(US)
(56) 선행기술조사문헌
EP1182569 A1*
US 2002/053006 A1
EP 0856797 A
US 4727485 A
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
인터내셔널 비지네스 머신즈 코포레이션
미국 10504 뉴욕주 아몬크 뉴오차드 로드
(72) 발명자
데이 마이클 노르만
미국 78681 텍사스주 라운드 락 메이필드 드라이브 2201
호프스테 함 피터
미국 78703 텍사스주 오스틴 프레슬러 스트리트 704
(뒷면에 계속)
(74) 대리인
김원준, 김창세, 장성구

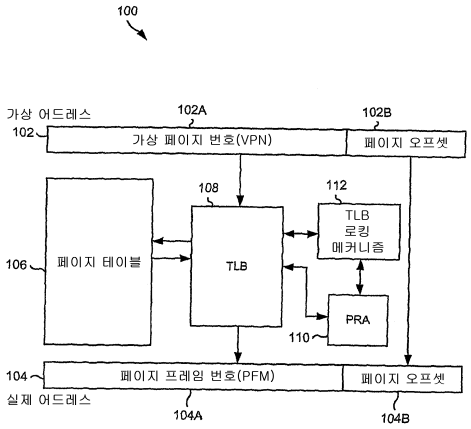
전체 청구항 수 : 총 8 항 심사관 : 이종익

(54) 컴퓨터 시스템 내의 메모리 관리 향상 방법, 메모리 관리 메커니즘 및 컴퓨터 판독 가능한 기록 매체

(57) 요 약

어드레스 변환 정보(address translation information)의 서브셋(subset)이, CPU에 의한 보다 빠른 액세스를 위해 이러한 어드레스 변환 정보를 저장하기 위해 확보된 캐시 메모리(cache memory) 내에 저장된 다른 어드레스 변환 정보로 대체되는 것을 방지하는 것에 의해 컴퓨터 시스템 내에서의 메모리 관리가 개선된다. 이러한 방법으로, CPU는 캐시 내에 저장된 어드레스 변환 정보의 서브셋을 식별할 수 있다.

대표도



(72) 발명자

존스 찰스 레이

미국 78759 텍사스주 오스틴 캐시아 드라이브 10703

칼 제임스 알란

미국 78731 텍사스주 오스틴 후로크 드라이브 5402

트루옹 쑤옹 쿠앙

미국 78727 텍사스주 오스틴 피켓 로프 레인 12612

쉬피 데이비드

미국 78746 텍사스주 오스틴 브라이언스 메도우 씨브이 1313

특허청구의 범위

청구항 1

처리 로직(processing logic)과 제 1 및 제 2 메모리를 구비하는 컴퓨터 시스템 내에서 메모리 관리를 향상시키는 방법으로서,

가상 어드레스(virtual addresses)와 실제 어드레스 사이의 어드레스 변환 정보(address translation information)를 상기 제 1 메모리 내에 저장하는 단계와,

상기 처리 로직으로부터의 더 빠른 액세스를 위해 상기 어드레스 변환 정보의 적어도 일부분을 상기 제 2 메모리 내에 캐싱(caching)하는 단계-상기 어드레스 변환 정보의 일부분은 어드레스 변환 정보의 서브셋(subset)을 포함하고, 상기 제 2 메모리는 상기 제 1 메모리보다 상기 처리 로직에 더 가깝게 위치됨-와,

상기 어드레스 변환 정보의 상기 서브셋이 상기 제 1 메모리 내에 저장된 다른 어드레스 변환 정보로 대체되는 것을 방지하는 단계

를 포함하되,

상기 어드레스 변환 정보의 상기 서브셋이 상기 제 1 메모리 내에 저장된 다른 어드레스 변환 정보로 대체되는 것을 방지하는 상기 단계는,

레인지 레지스터(range register)에 대한 하나 이상의 클래스 ID(Class ID)를 생성하는 단계-각각의 클래스 ID는 유효 어드레스의 주어진 어드레스 레인지(address range)를 나타냄-와,

상기 하나 이상의 클래스 ID를 인덱스(indices)로 이용하여 대체 관리 테이블(replacement management table)(RMT)을 액세스하는 단계와,

상기 어드레스 변환 정보의 적어도 일부분의 하나 이상의 세트에 대해 상기 하나 이상의 클래스 ID를 매핑(mapping)하는 단계와,

상기 어드레스 변환 정보의 어떠한 세트가 상기 RMT에 기초하는 대체를 위해 적합한지 판정하는 단계와,

대체를 위해 적합한 것으로 판정된 상기 어드레스 변환 정보의 세트에 대해서만 대체 알고리즘을 수행하는 단계

를 더 포함하는 컴퓨터 시스템 내의 메모리 관리 향상 방법.

청구항 2

제 1 항에 있어서,

상기 제 1 및 제 2 메모리는 각각 상기 컴퓨터 시스템의 주 메모리(main memory) 및 변환 룩어사이드 버퍼(translation lookaside buffer)(TLB)인 컴퓨터 시스템 내의 메모리 관리 향상 방법.

청구항 3

제 1 항에 있어서,

상기 어드레스 변환 정보는 하나 이상의 페이지 테이블 엔트리(page table entries)(PTE)를 갖는 하나 이상의 페이지 테이블(page tables) 내에 저장되는 컴퓨터 시스템 내의 메모리 관리 향상 방법.

청구항 4

제 1 항에 있어서,

상기 어드레스 변환 정보의 상기 서브셋은 실시간(real-time) 애플리케이션(들)에 대해 할당되는 컴퓨터 시스템 내의 메모리 관리 향상 방법.

청구항 5

제 3 항에 있어서,

상기 대체 알고리즘은 페이지 대체 알고리즘을 포함하는 컴퓨터 시스템 내의 메모리 관리 향상 방법.

청구항 6

제 5 항에 있어서,

상기 페이지 대체 알고리즘은 LRU(least-recently-used) 알고리즘을 포함하는 컴퓨터 시스템 내의 메모리 관리 향상 방법.

청구항 7

처리 로직과 제 1 및 제 2 메모리를 구비하는 컴퓨터 시스템 내의 향상된 메모리 관리 장치로서,

제 1 항 내지 제 6 항 중 어느 한 항에 기재된 방법을 실행하는 메모리 관리 장치.

청구항 8

처리 로직과 제 1 및 제 2 메모리를 구비하는 컴퓨터 시스템 내에서 메모리 관리를 향상시키기 위한 컴퓨터 판독 가능한 기록 매체로서,

컴퓨터 시스템 상에서 실행되면, 제 1 항 내지 제 6 항 중 어느 한 항에 기재된 방법을 실행하는 컴퓨터 프로그램을 포함하는 컴퓨터 판독 가능한 기록 매체.

명 세 서

기 술 분 야

<1> 본 발명은 일반적으로 컴퓨터 시스템 내에서의 메모리 관리에 관한 것이고, 보다 구체적으로는 컴퓨터 시스템 상에서 실행되는 실시간(real-time) 애플리케이션의 성능을 증대시키기 위해 개선된 메모리 어드레스 변환 메커니즘(memory address translation mechanism)에 관한 것이다.

배 경 기 술

<2> 현대의 컴퓨터 시스템에서, 가상 메모리(virtual memory)로 알려진 기법을 사용하여 컴퓨터 시스템의 물리적 메모리(physical memory) 또는 실제 메모리(real memory)의 리소스를 관리한다. 이러한 기법은 애플리케이션에게 있어서 메모리 양이 매우 크다는 착각을 일으키게 한다. 본래, 이러한 가상 메모리 시스템은 너무 작은 물리적 또는 실제 메모리를 갖는 문제를 해결하기 위해 개발되었다. 이들이 풍부한 메모리를 갖고 있는 대부분의 현대적 시스템에 있어서 더 이상 중요한 인자가 될 수 없다고 해도, 가상 메모리 메커니즘 그 자체는 그 외의 이유로 인해 매우 가치가 있는 것으로 증명되었다. 이러한 이유들 중에서 특히, 가상 메모리 메커니즘은 애플리케이션 프로그램 설계에서뿐만 아니라 메모리 할당(memory allocation) 및/또는 메모리 보호를 단순화한다.

<3> 가상 메모리 시스템에서, 가상 어드레스는 복수의 페이지 테이블 엔트리(page table entries)(PTE)를 갖는 페이지 테이블(page tables)을 가지고 실제 어드레스로 변환될 수 있다. 전형적으로, 이러한 페이지 테이블은 DRAM 칩 등과 같은 컴퓨터 시스템의 주 메모리(main memory) 내에 저장된다. 그러나, CPU와 주 메모리 사이에서 접속하는 메모리 버스(memory bus)가 CPU의 작동 속도보다 매우 느리기 때문에 CPU와 주 메모리 사이에는 병목 현상(bottleneck)이 존재한다. 실시간 응답을 필요로 하지 않는 소정의 애플리케이션에 있어서, 이러한 병목 현상은 그에 합당한 문제점을 유발하지 않을 수 있다. 그러나, 여러 실시간 애플리케이션에서는 이러한 병목 현상이 이러한 애플리케이션의 작동에 있어서 뚜렷한 지연을 유발한다는 것이 확인된 바 있다.

<4> 이러한 병목 현상의 문제점을 극복하기 위한 공지된 기법은, 중앙 처리 장치(Central Processing Unit)(CPU)에 근접하고, 변환 룩어사이드 버퍼(translation lookaside buffer)(TLB)로 지칭되는 특수한 캐시 내에 PTE를 저장하는 것이다. 전형적으로, TLB는 CPU 내부에서 온-칩(on-chip) 상태로 상주하여, CPU에 의해 검색된 PTE(들)의 특정한 세트가 TLB 내에 저장되는 한, 주 메모리 내에서 페이지 테이블을 탐색하는 것과 연관된 임의의 지연을 제거하거나 실질적으로 감소시킨다. 특정한 PTE(들)의 세트가 TLB 내에서 발견되지 않으면, 이러한 이벤트는 전형적으로 TLB 미스(TLB miss)로 규정된다.

발명의 상세한 설명

<5> 일반적으로, 메모리 판독 및/또는 기록 요청이 이루어지면, 가상 메모리로부터 실제 메모리로의 매핑(mapping)

은 TLB 내에서 캐싱(caching)된다. TLB 미스가 존재하면, 전용 하드웨어 또는 소프트웨어가 정확한 PTE를 가지고 TLB를 리로딩(reload)해야 한다. TLB의 하드웨어 리로딩뿐만 아니라 소프트웨어 리로딩은, 이러한 리로딩이 주 메모리로부터 PTE를 판독하고 그에 해당되는 일치(match) 대상을 탐색하도록 요구할 것이므로 성능과 관련하여 상당히 소모적일 수 있다. 전형적으로, 최대 3개의 캐시 라인 미스(cache line misses)를 취하여 일치 대상을 찾을 수 있다. 그러므로 실시간 애플리케이션에서, TLB 미스가 존재하는 것은 바람직하지 않다.

<6> 본 발명은 제 1 항에 기재된 바와 같이 컴퓨터 시스템 내에서 메모리 관리를 향상시키는 방법을 제공하고, 다른 청구항에 기재된 바와 같이 그에 대응하는 장치 및 컴퓨터 프로그램 제품을 제공한다.

<7> 본 발명 및 그 이점에 대한 보다 완전한 이해를 위하여, 다음으로 첨부된 도면과 함께 이하의 설명에 대해 참조하기로 한다.

실 시 예

<11> 다르게 표시되어 있지 않는 한, 본 명세서에 개시된 모든 기능은 하드웨어나 소프트웨어, 또는 그 소정의 조합으로 실행될 수 있다는 것을 유의해야 한다. 그러나, 바람직한 실시예에서, 이러한 기능은 다르게 표시되어 있지 않는 한, 컴퓨터 프로그램 코드 등과 같은 코드에 따라서 컴퓨터 또는 전자 데이터 프로세서(electronic data processor) 등과 같은 프로세서, 소프트웨어 및/또는 이러한 기능을 수행하도록 코딩된 집적 회로에 의해 수행된다.

<12> 도면 중에서 도 1을 참조하면, 참조 번호 100은 일반적으로 실시간 애플리케이션을 위한 개선된 메모리 관리 메커니즘을 도시하는 상위 레벨 블록도를 지정한다. 일반적으로, 상위 레벨 블록도(100)는 가상 메모리를 이용하는 컴퓨터 시스템에 적용될 수 있다. 대부분의 현대적 컴퓨터 시스템이 이러한 카테고리 내에 포함된다. 가상 어드레스(102)는 페이지 테이블(106) 및 테이블 룩어사이드 버퍼(TLB)(108) 내에 저장된 페이지 테이블 엔트리(PTE)(도시하지 않음)를 이용하여 실제 또는 물리적 어드레스(104)로 변환된다. 특히, 가상 어드레스(102)는 가상 페이지 번호(virtual page number)(VPN)(102A) 및 페이지 오프셋(page offset)(102B)을 포함한다. 이와 유사하게, 실제 어드레스(104)는 페이지 프레임 번호(page frame number)(PFN)(104A) 및 페이지 오프셋(104B)을 포함한다. 바람직하게는, 가상 어드레스(102)의 VPN(102A)만이 실제 어드레스(104)의 PFN(104A)로 변환된다. 페이지 오프셋(102B)은 수정없이 페이지 오프셋(104B)으로 복제된다.

<13> TLB 미스(TLB miss)가 존재하고, 이러한 TLB 내의 PTE가 페이지 테이블(106)로부터의 새로운 PTE(손실된 페이지를 포함함)로 대체되어야 하는 경우에 하드웨어 및/또는 소프트웨어 페이지 대체 알고리즘(page replacement algorithm)(PRA)(110)을 또한 사용하여 TLB를 보조한다. TLB 로킹 메커니즘(locking mechanism)(112)은 TLB(108) 및 PRA(110)와 상호 작용하여 실시간 애플리케이션을 위한 메모리 관리 메커니즘(100)을 개선한다.

<14> 적어도 하나의 중앙 처리 장치(central processing unit)(CPU)(도시하지 않음)는 다르게 표시되지 않는 한, 이러한 관계(context)에서 필요한 대부분의 동작 및/또는 계산을 직접 또는 간접적으로 제어한다. 또한, CPU에 접속된 메모리(도시하지 않음)는 페이지 테이블(106)을 저장한다. 하드웨어로 구현되는 경우에, PRA(110)는 TLB(108) 및 TLB 로킹 메커니즘(112)에 결합된다. 소프트웨어로 구현되는 경우에, PRA(110)는 메모리(도시하지 않음) 내에 저장되고, TLB 미스가 발생되는 경우에 CPU에 의해 실행될 것이다.

<15> 일반적으로, CPU가 페이지 테이블(106)을 저장하는 메모리에 대한 액세스를 획득하기 위해서는, (CPU의 추정 가능한 짧은 클록 사이클(clock cycle)에 비해서) 비교적 긴 시간 주기가 소요될 것이다. TLB(108)는 원래 페이지 테이블(106)로부터 입수 가능한 PTE의 일부분을 동적으로 저장하는 데 사용되는 특수한 캐시이다. 특히, TLB(108)는 빠른 액세스를 위해 비교적 적은 PTE의 세트를 저장하는 연합 메모리(associative memory)이다. 몇몇 필드(들)의 값이 주어지면, 연합 메모리 내의 하드웨어 메커니즘은 모든 레코드를 검색하고, 주어진 값을 포함하는 필드를 갖는 레코드를 리턴(return)한다. VPN(102A)이 주어지면, TLB(108) 내의 하드웨어 메커니즘은 TLB 내의 모든 PTE를 검색하고, TLB 미스가 존재하지 않는 경우에 PFN(104A)을 리턴한다. 본 발명의 진정한 정신을 벗어나지 않으면서 TLB(108) 대신에 다른 타입의 연합 메모리를 이용할 수 있다는 것을 유의하라.

<16> 전형적으로 TLB(108)는 CPU에 의해 가까운 미래에 사용될 가능성이 있는 몇몇 PTE를 동적으로 캐싱하기 위해 사용된다. PRA(110)는 사실상 CPU가 가까운 미래에 어떤 페이지를 사용할 가능성이 있는지 예측하고, 그에 따라서 주어진 시점에서 TLB(108) 내에 어떤 PTE가 저장되어야 하는지 판정한다. 일반적으로 본 기술 분야에서는 LRU(least-recently-used) 알고리즘(도시하지 않음)을 포함하는 여러 다른 페이지 대체 알고리즘이 알려져 있으나 이것으로 한정되지 않는다. TLB 미스가 존재할 때, LRU 알고리즘은 가장 오랜 시간 동안 사용되지 않은 PTE(즉, 최근 최소 사용된(least recently used) PTE)를 새로운 PTE(예를 들면, TLB 내에서 미싱된 PTE)로 대

체한다.

<17> 이러한 페이지 대체 기법은 시간에 걸쳐서 메모리 관리 메커니즘(100)의 전체적 성능을 최적화할 수 있기는 하지만, 이러한 기법은 실시간 애플리케이션에 대해 발생될 가능성이 있는 역효과에 대해 고려하지 않는다. TLB 로킹 메커니즘(112)은 TLB(108) 및 PRA(110)와의 작업에 의해 이러한 문제를 극복한다. 일반적으로, TLB 로킹 메커니즘(112)은 실시간 애플리케이션에 대해 할당된 PTE(들)의 세트가 페이지 테이블(106) 내에 속하는 다른 PTE의 세트로 대체되는 것을 방지한다. 다시 말해서, TLB 로킹 메커니즘(112)은 PRA(110)가 판정하는 것에 무관하게 실시간 애플리케이션을 위해 할당된 PTE(들)의 세트를 로킹함으로써 이러한 PTE(들)의 세트가 TLB(108)에서 제거되지 않도록 한다.

<18> 실시간 애플리케이션이 본 발명으로부터 가장 큰 이득을 받을 수 있기는 하지만, 본 발명은 반드시 실시간 애플리케이션으로 한정되지 않는다는 것을 유의하는 것은 중요하다. 일반적으로, 본 발명은 TLB 미스가 발생될 때 임의의 타입의 애플리케이션 및/또는 임의의 특정한 애플리케이션(들)이 대체되는 것을 방지하기 위해서 적용 가능한 것으로 고려되어야 한다.

<19> 다음으로 도 2를 참조하면, 실시간 애플리케이션에 대한 개선된 메모리 관리 메커니즘의 바람직한 실시예를 나타내는 하위 레벨 블록도(200)가 도시되어 있다. 일반적으로, 메모리 요청은 유효 어드레스(effective address)(EA)(202)의 형태로 CPU(도시하지 않음)로부터 생성된다. EA(202)는 가상 어드레스(virtual address)(VA)(204)로 변환되고, 이는 그 후에 실제 어드레스(real address)(RA)(206)로 변환된다. 유효 어드레스(EA)는 유효 세그먼트 ID(ESID)(206A), 페이지 번호(PN)(206B) 및 오프셋(206C)을 포함한다. 이와 유사하게, VA(204)는 가상 세그먼트 ID(VSID)(204A), 페이지 번호(PN)(204B) 및 오프셋(204C)을 포함한다. 바람직하게는, 가상 페이지 번호(VPN)(204D)는 VSID(204A)와 PN(204B)을 연접함으로써 생성된다.

<20> EA(202)는 세그먼트 룩어사이드 버퍼(segment lookaside buffer)(SLB)(208) 및 변환 룩어사이드 버퍼(TLB)(210)에 전달된다. SLB(208)는 EA(202)를 VA(204)로 변환하도록 구성된다. 바람직하게는, SLB(208)는 ESID(202A)를 VSID(204A)로 변환하고, PN(202B) 및 오프셋(202C)은 각각 PN(204B) 및 오프셋(204C)으로 복제된다.

<21> VPN(204D) 및 EA(202)가 주어지면, CPU는 EA(202)에 의해 인덱싱된 모든 PTE를 액세스하고, VPN(204A)을 비교함으로써 TLB(210)에서 페이지 테이블 엔트리(PTE) 내의 일치 대상(즉, PTE 히트(hit))을 검색한다. 히트가 존재하는 경우에, 실제 페이지 번호(212)가 리턴된다. 바람직하게는, 로직 소자(logic element)(216)와 함께 TAG 비교기(214)를 이용하여 이러한 비교를 수행한다. 특히, TAG 비교기(214)는 VPN(204D)이 TLB(210) 내의 VPN(218)와 일치될 때 로직 소자(216)가 RPN(212)을 전달할 수 있게 한다. VPN(204D)이 TLB(210) 내의 VPN(218)과 일치되지 않으면, TAG 비교기(214)는 로직 소자(216)가 RPN(212)을 전달하는 것을 방지한다.

<22> 미스가 존재하는 경우에, TLB 리로딩 메커니즘(TLB reload mechanism)(도시하지 않음)은 메모리(도시하지 않음) 내의 페이지 테이블(도시하지 않음)을 검색하고, TLB(210)를 리로딩한다. 통상적인 TLB 구조에서, 미스 이후에 TLB(210)가 리로딩될 때, LRU(least-recently-used) 알고리즘 등과 같은 페이지 대체 알고리즘(PRA)(220)을 사용하여, 일치 클래스(congruence class) 내의 어떤 PTE의 세트가 대체되어야 하는지 표시한다.

<23> TLB 리로딩이 발생한 경우에, 레인지 레지스터(range registers)(222)의 세트, 클래스 ID, 대체 세트(replacement set)(224), 대체 관리 테이블(replacement management table)(RMT)(226) 및 대체 세트 제어 블록(replacement set control block)(230)을 사용하여 어떤 세트를 대체할 것인지 선택한다. 특히, 레인지 레지스터(222)는 RMT(226)에 결합되어 클래스 ID를 RMT(226)에 전달한다. TLB 리로딩은 EA(202)를 레인지 레지스터(222)에 전달함으로써 이루어진다. 레인지 레지스터(222) 내에 히트가 존재한다면, 클래스 ID가 생성되고 RMT(226)로 전달된다. RMT는 RMT 대체 제어 비트를 생성하고, 이것은 RMT 대체 제어 버스(RMT replacement control bus)(232)를 통해서 대체 세트 제어 블록(230)으로 전달된다. 대체 세트 제어 블록(230)은 TLB(210)로부터의 RMT 대체 제어 비트 및 대체 세트(224)를 이용하여 어떤 세트를 대체할 것인지 결정한다.

<24> 도 3은 RMT(226)의 일례를 도시한다. 이 예에서, RMT(226)는 8개의 클래스 ID(행)×8개의 TLB 세트(열)의 매트릭스를 포함한다. 바람직하게는, RMT(226)는 TLB(210) 내의 세트가 주어진 클래스 ID와 상관될 때, 이러한 세트의 대체 관리 상태에 대한 운영 시스템 소프트웨어의 결정을 나타낸다.

<25> 바람직한 실시예에서, RMT(226)는 소프트웨어 관리형 테이블(software-managed table)이다. 소프트웨어는 구조를 유지하고, RMT 엔트리의 의미를 해석한다. 일반적으로, RMT(226)는 TLB의 하나의 세트 또는 세트들에 대한 유효 어드레스 레인지의 매핑에 사용된다. 또한, 클래스 ID(즉, TLB(226)의 행)는 주어진 어드레스 범위를 나

타낸다. 이 예에서, 클래스 ID 0은 다른 클래스 ID에 의해 특별하게 제공되지 않은 임의의 어드레스 레인지에 해당한다. 그러면 주어진 클래스 ID가 RMT로 전달되고, RMT에 의해 인덱스로서 사용된다. 클래스 ID를 인덱스로서 사용하면, RMT(226)가 액세스되고, 해당되는 정보는 RMT 대체 제어 버스(232) 상에서 전달된다.

<26> 이 예에서, RMT(226)는 8×8 매트릭스이다. 다시 말해서, TLB(210)는 8-웨이(8-way) 세트 연합체(즉, 8개의 세트를 가짐)이다. 그러므로, 각각의 RMT 엔트리는 8 비트를 갖는다. 특히, RMT(226)는 TLB 세트 0-2로부터 클래스 ID 행 0 내에 정의된 "1"과, 행의 나머지 부분에서 정의된 "0"을 갖는다. 그러므로, 클래스 ID 0에 있어서, 데이터가 TLB로 대체되어야 한다면, 이러한 데이터는 TLB의 처음 3개의 세트, 즉 세트 0-2 내에서 업데이트될 수 있다. 또한, 클래스 ID 0은 이러한 세트에 대해 한정적으로 사용된다. 그러므로, TLB의 세트 0-2는 클래스 ID 0에 대해 한정된다. 클래스 ID 0은 레인지 레지스터 내에서 히트(hit)되지 않는 어드레스에 대한 디폴트 클래스 ID를 제공한다.

<27> 클래스 ID 1에 있어서, RMT는 세트 0-2 및 세트 4-7 내에서 정의된 "0"과, 세트 3에서 정의된 "1"을 갖는다. 클래스 ID 1에 대응하는 임의의 데이터는 세트 0-2 또는 세트 4-7 내에서 대체되지 않는다. 그 대신에, 데이터는 세트 3 내에서 대체된다. 클래스 ID 2 및 클래스 ID 3은 모두 동일한 세트인 세트 7에서 대체한다. 그러므로, 클래스 ID 2 및 클래스 ID 3은 오로지 세트 7에서만 대체한다. 클래스 ID 4는 대체를 위한 여러 개의 유효한 후보 세트를 갖는다. 이러한 세트는 세트 4, 세트 5 및 세트 6이다.

<28> 세트 대체의 최종 선택은 LRU 알고리즘 등과 같은 임의의 페이지 대체 알고리즘(PRA)에 의해 이루어진다. 다시 말해서, 소정 세트는 RMT에 의한 대체가 바람직한 것으로 마킹되고, 다음에 도 2의 대체 세트 선택 블록(230) 내에 포함된 LRU 알고리즘 제어 로직(도시하지 않음)을 사용하여 대체될 세트의 선택을 마무리한다.

도면의 간단한 설명

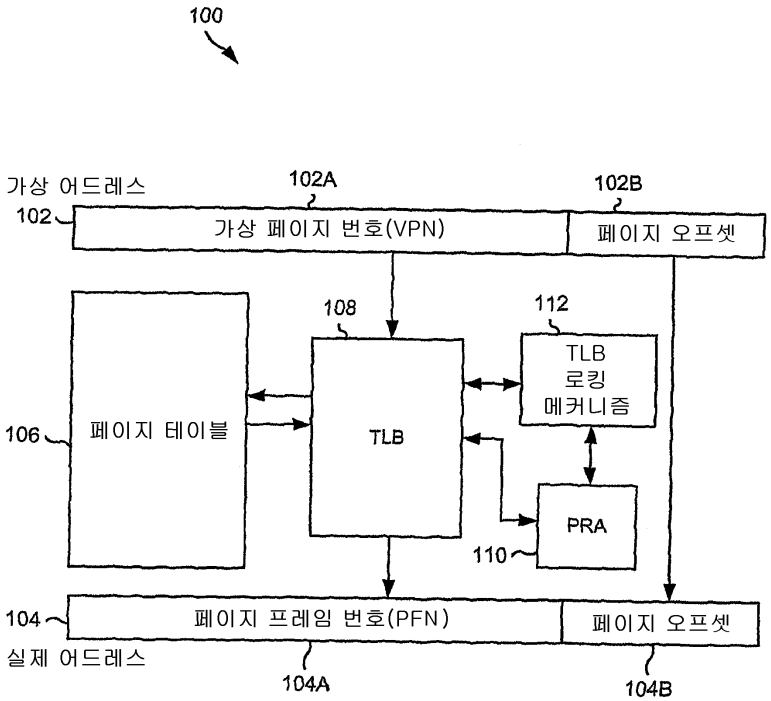
<8> 도 1은 실시간 애플리케이션에 있어서 개선된 메모리 관리 메커니즘을 도시하는 상위 레벨 블록도(high-level block diagram).

<9> 도 2는 실시간 애플리케이션을 위한 개선된 메모리 관리 메커니즘의 바람직한 실시예를 도시하는 하위 레벨 블록도(high-level block diagram).

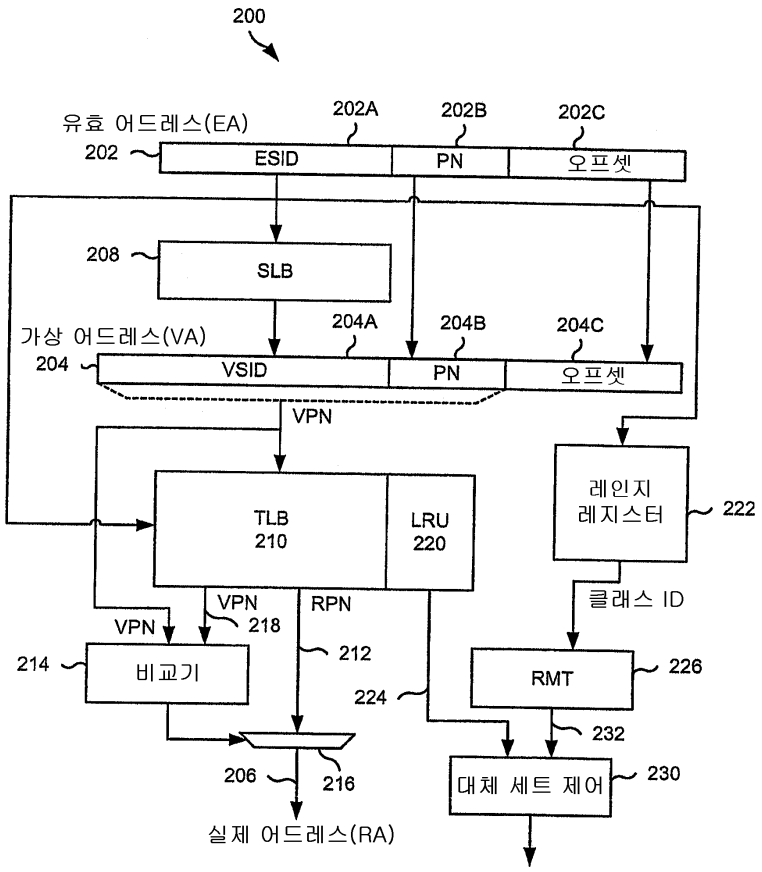
<10> 도 3은 도 2의 개선된 메모리 관리 메커니즘에서 사용되는 대체 관리 테이블(replacement management table)(RMT)의 일례를 도시하는 도면.

도면

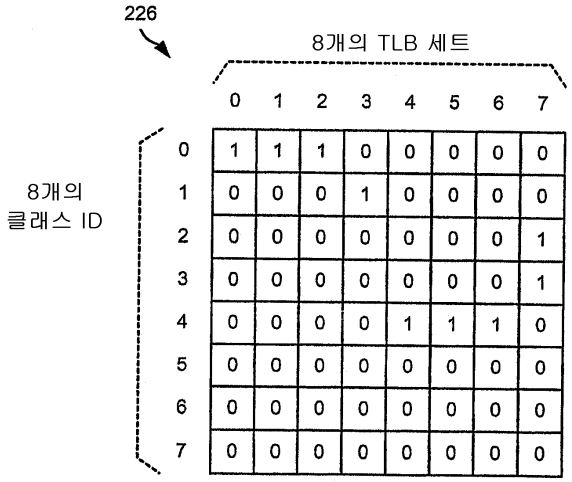
도면1


100
가상 어드레스
102
102A
가상 페이지 번호(VPN)
102B
페이지 오프셋
108
112
TLB
로킹
메커니즘
106
페이지 테이블
TLB
PRA
110
104
페이지 프레임 번호(PFN)
페이지 오프셋
실제 어드레스
104A
104B

도면2


200
유효 어드레스(EA)
202
202A
202B
202C
ESID
PN
오프셋
208
SLB
가상 어드레스(VA)
204
204A
204B
204C
VSID
PN
오프셋
VPN
TLB
210
LRU
220
레인지
레지스터
222
클래스 ID
VPN
VPN
218
RPN
212
214
비교기
224
RMT
226
232
206
216
대체 세트 제어
230
실제 어드레스(RA)

도면3


226
8개의 TLB 세트
0 1 2 3 4 5 6 7
8개의
클래스 ID
0 1 1 1 0 0 0 0 0
1 0 0 0 1 0 0 0 0
2 0 0 0 0 0 0 0 1
3 0 0 0 0 0 0 0 1
4 0 0 0 0 1 1 1 0
5 0 0 0 0 0 0 0 0
6 0 0 0 0 0 0 0 0
7 0 0 0 0 0 0 0 0