(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2009년02월23일
(11) 등록번호 10-0885277
(24) 등록일자 2009년02월17일

(51) Int. Cl.
G06F 12/08 (2006.01)
(21) 출원번호 10-2003-7014899
(22) 출원일자 2003년11월14일
심사청구일자 2007년03월21일
번역문제출일자 2003년11월14일
(65) 공개번호 10-2003-0097871
(43) 공개일자 2003년12월31일
(86) 국제출원번호 PCT/US2002/008804
국제출원일자 2002년03월21일
(87) 국제공개번호 WO 2002/93385
국제공개일자 2002년11월21일
(30) 우선권주장
09/859,290 2001년05월16일 미국(US)
(56) 선행기술조사문헌
KR1019930010733 A
JP 03154948 A
전체 청구항 수 : 총 44 항

(73) 특허권자
어드밴스드 마이크로 디바이시즈, 인코포레이티드
미국 캘리포니아 94088-3453 서니베일 원 에이엠디 플레이스 메일 스톱68
(72) 발명자
탄테익-충
미국텍사스78733오스틴오타와드라이브1114
샌더벤자민티.
미국텍사스78735오스틴크리스탈워터드라이브5109
(74) 대리인
박장원

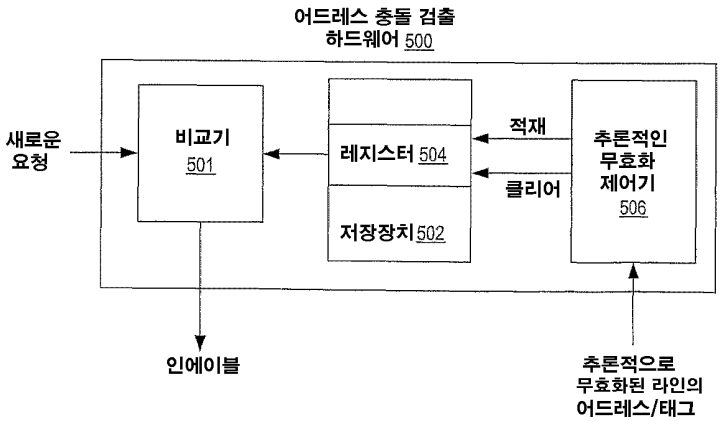
심사관 : 안철용

(54) 캐시에서의 라인들을 추론적으로 무효화하는 방법 및 시스템

(57) 요 약

캐시 라인을 추론적으로 무효화하도록 구성되는 캐시 제어기(419)는 에러 검사가 완료되는 것을 기다리는 대신에 즉시 무효화 요청 또는 명령에 응답할 수 있다. 상기 에러 검사가 상기 무효화에 에러가 있으므로 수행되어서는 안된다고 판단하는 경우에, 상기 캐시 제어기는 에러 검사가 완료될 때까지 상기 추론적으로 무효화된 캐시 라인을 변경으로부터 보호한다. 이러한 방식으로, 상기 무효화가 나중에 에러가 있다가 판명된다면, 상기 추론적인 무효화는 번복될 수 있다. 어떠한 에러들도 검출되지 않고 에러 검사가 완료되면, 상기 추론적인 무효화는 비-추론적이 된다.

대 표 도 - 도3



특허청구의 범위

청구항 1

프로세서와;

시스템 메모리와;

상기 프로세서에 결합되어 있으며, 다수의 캐시 라인 저장 위치들을 포함하는 캐시와;

제 1 요청을 수신하도록 결합된 캐시 제어기- 상기 캐시 제어기는 상기 제 1 요청에 응답하여 제 1 캐시 라인 저장 위치 내의 제 1 캐시 라인을 추론적으로 무효화하도록 구성되고, 상기 제 1 캐시 라인을 추론적으로 무효화하는 것에 응답하여, 상기 캐시 제어기는 상기 제 1 캐시 라인의 무효화가 비-추론적이 될 때까지 상기 제 1 캐시 라인의 대체를 막도록 더 구성되며 -와; 그리고

상기 제 1 요청에 대응하는 적어도 하나의 검사를 수행하도록 구성된 에러 검출 유닛을 포함하며, 만일 상기 검사가 수행되어 그 어떤 에러도 검출하지 않는 경우, 상기 제 1 캐시 라인의 무효화는 비-추론적이 되는 것을 특징으로 하는 컴퓨터 시스템.

청구항 2

제 1항에 있어서,

상기 캐시 제어기는 상기 제 1 캐시 라인과 관련된 유효성 비트를 토글링함으로써 상기 제 1 캐시 라인을 추론적으로 무효화하도록 구성되는 것을 특징으로 하는 컴퓨터 시스템.

청구항 3

제 2항에 있어서,

만일 상기 검사가 수행되어 에러를 검출한다면, 상기 캐시 제어기는 상기 제 1 캐시 라인과 관련된 상기 유효성 비트를 토글링함으로써 상기 추론적인 실행을 번복하도록 더 구성되는 것을 특징으로 하는 컴퓨터 시스템.

청구항 4

제 1항에 있어서,

상기 캐시 제어기는 상기 제 1 캐시 라인의 무효화가 비-추론적이 될 때까지 상기 제 1 캐시 라인 내의 데이터 또는 상기 제 1 캐시 라인의 상태에 따르는 요청들을 수락하지 않도록 더 구성되는 것을 특징으로 하는 컴퓨터 시스템.

청구항 5

제 1항에 있어서,

상기 캐시 제어기는 상기 제 1 캐시 라인의 상기 무효화가 비-추론적이 될 때까지, 각각의 추가적인 요청의 타입, 그리고 각각의 추가적인 요청이 상기 추론적으로 제 1 캐시 라인에 의존하는지 또는 이러한 제 1 캐시 라인을 수정하는지에 근거하여 추가적인 요청들을 수락하지 않도록 구성되는 것을 특징으로 하는 컴퓨터 시스템.

청구항 6

제 1항에 있어서,

상기 제 1 캐시 라인의 무효화가 비-추론적이 될 때까지, 상기 캐시 제어기는 추가적인 요청들을 수락하지 않도록 구성되며, 상기 추가적인 요청들은:

상기 제 1 캐시 라인을 적중하는 더 높은 레벨의 캐시로부터의 채움 요청들;

상기 제 1 캐시 라인에 대한 프로브 또는 상태 변경 요청들; 또는

대체될 상기 제 1 캐시 라인을 선택하는, 상기 더 높은 레벨 캐시로부터의 카피 백(copy backs)인 것을 특징으로 하는 컴퓨터 시스템.

청구항 7

제 1항에 있어서,

상기 캐시 제어기는 상기 제 1 캐시 라인의 상기 무효화가 비-추론적이 될 때까지, 상기 제 1 캐시 라인의 제 1 태그의 제 1 부분과 매칭하는 추가적인 요청들을 수락하지 않도록 구성되는 것을 특징으로 하는 컴퓨터 시스템.

청구항 8

제 1항에 있어서,

상기 추론적인 무효화의 일부로서, 상기 캐시 제어기는 상기 제 1 캐시 라인과 관련된 이전의 추론적인 무효화 대체 상태를 세이브하고, 그리고 마치 상기 제 1 캐시 라인이 무효화 된 것처럼, 이후의 추론적인 무효화 대체 상태를 업데이트하도록 구성되는 것을 특징으로 하는 컴퓨터 시스템.

청구항 9

제 8항에 있어서,

상기 검사가 수행되어 에러를 검출한 경우, 상기 캐시 제어기는 세이브된 이전의 추론적인 무효화 대체 상태를 복구하도록 더 구성되는 것을 특징으로 하는 컴퓨터 시스템.

청구항 10

제 1항에 있어서,

상기 캐시 제어기는 상기 제 1 캐시 라인의 상기 무효화가 비-추론적이 된 이후에, 상기 제 1 캐시 라인을 수정하는 추가적인 요청들을 수락하도록 더 구성되는 것을 특징으로 하는 컴퓨터 시스템.

청구항 11

제 1항에 있어서,

상기 적어도 하나의 검사는 오더링(ordering) 검사를 포함하는 것을 특징으로 하는 컴퓨터 시스템.

청구항 12

제 1항에 있어서,

상기 적어도 하나의 검사는 에일리어스(alias) 검사를 포함하는 것을 특징으로 하는 컴퓨터 시스템.

청구항 13

제 1항에 있어서,

상기 적어도 하나의 검사는, 액세스 대상인 캐시 라인의 상태를 고려할 때 상기 액세스 대상인 캐시 라인이 액세스될 수 있는지 여부에 대한 검사를 포함하는 것을 특징으로 하는 컴퓨터 시스템.

청구항 14

제 1항에 있어서,

상기 캐시는 배타적인 캐시를 포함하며, 상기 제 1 요청은 상기 캐시가 배타적이라는 것에 관해서 더 높은 레벨의 캐시로부터의 채움 요청인 것을 특징으로 하는 컴퓨터 시스템.

청구항 15

캐시에서의 라인을 추론적으로 무효화하는 방법으로서,

상기 캐시에서의 라인을 무효화하는 요청을 수락하는 단계와;

상기 무효화에 에러가 있는지 여부를 판단하는 검사들을 시작하는 단계와;

상기 수락 단계에 응답하여, 상기 라인을 추론적으로 무효화하는 단계- 상기 추론적으로 무효화하는 단계는 상

기 라인이 무효임을 표시하는 단계 및 상기 검사들이 완료될 때까지 후속하는 수정으로부터 상기 라인을 보호하는 단계를 포함하며, 여기서 상기 보호하는 단계는 상기 수락 단계에 응답하여, 상기 검사들이 완료될 때까지 상기 추론적으로 무효화된 라인 내의 데이터 또는 상기 추론적으로 무효화된 라인의 상태에 따르는 추가적인 요청들을 수락하지 않는 단계를 더 포함하며 -와; 그리고

상기 무효화에 에러가 있다고 판단한 상기 검사들 중 하나에 응답하여, 상기 라인이 유효임을 표시함으로써 상기 추론적인 무효화를 번복하는 단계를 포함하는 것을 특징으로 하는 캐시에서의 라인을 추론적으로 무효화하는 방법.

청구항 16

제 15항에 있어서,

상기 추론적으로 무효화하는 단계는 상기 라인과 관련된 유효성 비트를 토글링하는 것을 포함하는 것을 특징으로 하는 캐시에서의 라인을 추론적으로 무효화하는 방법.

청구항 17

제 16항에 있어서,

상기 번복 단계는 상기 라인과 관련된 유효성 비트를 토글링하는 것을 포함하는 것을 특징으로 하는 캐시에서의 라인을 추론적으로 무효화하는 방법.

청구항 18

제 15항에 있어서,

상기 보호 단계는 상기 추론적으로 무효화된 라인의 태그의 제 1 부분과 매칭하는 추가적인 요청들을 수락하지 않는 것을 포함하는 것을 특징으로 하는 캐시에서의 라인을 추론적으로 무효화하는 방법.

청구항 19

제 15항에 있어서,

상기 추론적으로 무효화하는 단계는 상기 라인과 관련된 이전의 추론적인 무효화 대체 상태 정보를 세이브함과 아울러 마치 상기 제 1 캐시 라인이 무효화된 것처럼, 이후의 추론적인 무효화 대체 상태 정보를 업데이트하는 단계를 더 포함하는 것을 특징으로 하는 캐시에서의 라인을 추론적으로 무효화하는 방법.

청구항 20

제 19항에 있어서,

상기 무효화에 에러가 있다고 판단하는 검사들 중 하나에 응답하여, 상기 세이브된 이전의 추론적인 무효화 대체 상태 정보를 복구하는 단계를 더 포함하는 것을 특징으로 하는 캐시에서의 라인을 추론적으로 무효화하는 방법.

청구항 21

제 15항에 있어서,

상기 검사들이 완료된 이후에 상기 라인을 보호하는 단계를 중단하는 단계를 더 포함하는 것을 특징으로 하는 캐시에서의 라인을 추론적으로 무효화하는 방법.

청구항 22

배타적인 캐시에서 제 1 캐시 라인을 추론적으로 무효화하는 방법으로서,

높은 레벨의 캐시로부터의 채움 요청을 수락하는 단계와;

상기 채움 요청들이 상기 배타적인 캐시에서 적중하는지를 판단하는 단계와;

상기 채움 요청이 잘못 개시되었는지를 판단하는 검사들을 개시하는 단계와;

상기 배타적인 캐시에서의 채움 요청 적중에 응답하여, 상기 배타적인 캐시로부터의 상기 제 1 캐시 라인을 상기 높은 레벨의 캐시에 제공함과 아울러 상기 검사들이 아직 완료되지 않은 경우에, 상기 배타적인 캐시에서의 상기 제 1 캐시 라인을 추론적으로 무효화하는 단계- 상기 추론적으로 무효화하는 단계는 상기 제 1 캐시 라인이 무효임을 표시함과 아울러 상기 검사들이 완료될 때까지 후속하는 수정으로부터 상기 제 1 캐시 라인을 보호하는 것을 포함하며 -와; 그리고

상기 채움 요청이 잘못 개시되었다고 판단하는 상기 채움 요청과 관련된 상기 검사들 중 하나에 응답하여, 상기 제 1 캐시 라인의 상기 추론적인 무효화를 번복하는 단계를 포함하는 것을 특징으로 하는 배타적인 캐시에서 제 1 캐시 라인을 추론적으로 무효화하는 방법.

청구항 23

제 22항에 있어서,

상기 제 1 캐시 라인이 무효임을 표시하는 단계는 상기 제 1 캐시 라인과 관련된 유효성 비트를 토글링하는 것을 포함하는 것을 특징으로 하는 배타적인 캐시에서 제 1 캐시 라인을 추론적으로 무효화하는 방법.

청구항 24

제 23항에 있어서,

상기 번복 단계는 상기 제 1 캐시 라인과 관련된 유효성 비트를 토글링하는 것을 포함하는 것을 특징으로 하는 배타적인 캐시에서 제 1 캐시 라인을 추론적으로 무효화하는 방법.

청구항 25

제 22항에 있어서,

상기 추론적으로 무효화하는 단계는 상기 검사들이 완료될 때까지 상기 제 1 캐시 라인 내의 데이터 또는 상기 제 1 캐시 라인의 상태에 따르는 추가적인 요청들을 수락하지 않는 것을 더 포함하는 것을 특징으로 하는 배타적인 캐시에서 제 1 캐시 라인을 추론적으로 무효화하는 방법.

청구항 26

제 22항에 있어서,

상기 보호 단계는 상기 추론적으로 무효화된 라인의 제 1 태그의 제 1 부분과 매칭하는 추가적인 요청들이 수락되지 않게 하는 것을 포함하는 것을 특징으로 하는 배타적인 캐시에서 제 1 캐시 라인을 추론적으로 무효화하는 방법.

청구항 27

제 22항에 있어서,

상기 보호 단계는 카피 백이 상기 추론적으로 무효화된 라인을 덮어쓰기할 것임을 검출함과 아울러 상기 검출에 응답하여, 상기 카피백이 다른 라인을 덮어쓰게 하는 것을 포함하는 것을 특징으로 하는 배타적인 캐시에서 제 1 캐시 라인을 추론적으로 무효화하는 방법.

청구항 28

제 22항에 있어서,

상기 추론적으로 무효화하는 단계는 상기 라인과 관련된 이전의 추론적인 무효화 대체 상태 정보를 세이브함과 아울러 마치 상기 제 1 캐시 라인이 무효화된 것처럼, 이후의 추론적인 무효화 대체 상태 정보를 업데이트하는 것을 더 포함하는 것을 특징으로 하는 배타적인 캐시에서 제 1 캐시 라인을 추론적으로 무효화하는 방법.

청구항 29

제 28항에 있어서,

상기 채움 요청이 잘못 개시된 것으로 판단하는 검사들 중 하나에 응답하여, 상기 세이브된 이전의 추론적인 무

효화 대체 상태 정보를 복구하는 단계를 더 포함하는 것을 특징으로 하는 배타적인 캐시에서 제 1 캐시 라인을 추론적으로 무효화하는 방법.

청구항 30

다수의 캐시 라인 저장 위치들을 포함하는 캐시와;

제 1 요청을 수신하도록 연결되고, 상기 제 1 요청의 수신에 응답하여 제 1 캐시 라인 저장 위치 내의 제 1 캐시 라인을 추론적으로 무효화하도록 구성된 캐시 제어기- 상기 캐시 제어기는 상기 추론적인 무효화에 에러가 있음이 검출된 경우에, 상기 제 1 캐시 라인의 상기 추론적인 무효화를 복구하도록 더 구성되며, 상기 추론적인 무효화가 잘못된 것인지를 검출하는데에 소정수의 주기들을 취하며 -와; 그리고

상기 제 1 캐시 라인을 추론적으로 무효화하는 상기 캐시 제어기에 응답하여, 상기 소정수의 주기 동안 상기 제 1 캐시 라인을 수정으로부터 보호하도록 구성된 추론적인 무효화 제어기를 포함하며, 여기서 상기 추론적인 무효화는 상기 소정수의 주기 이후에 비-추론적이 되며, 상기 추론적인 무효화 제어기는 상기 캐시 제어기가 상기 소정수의 주기 동안 상기 추론적으로 무효화된 제 1 캐시 라인을 수정하거나 이에 따르는 추가적인 요청들을 수락하는 것을 막도록 더 구성되는 특징으로 하는 캐시 서브시스템.

청구항 31

제 30항에 있어서,

상기 캐시 제어기는 상기 제 1 캐시 라인과 관련된 유효성 비트를 토글링함으로써 상기 제 1 캐시 라인을 추론적으로 무효화하도록 구성되는 것을 특징으로 하는 캐시 서브시스템.

청구항 32

제 31항에 있어서,

상기 캐시 제어기는 상기 제 1 캐시 라인과 관련된 상기 유효성 비트를 토글링함으로써 상기 추론적인 실행을 번복하도록 더 구성되는 것을 특징으로 하는 캐시 서브시스템.

청구항 33

제 30항에 있어서,

상기 추론적인 무효화 제어기는 상기 제 1 캐시 라인을 추론적으로 무효화하는 상기 캐시 제어기에 응답하여, 상기 제 1 캐시 라인의 태그의 제 1 부분을 레지스터에 적재하도록 더 구성되는 것을 특징으로 하는 캐시 서브시스템.

청구항 34

제 33항에 있어서,

상기 레지스터에 저장된 상기 태그의 제 1 부분과 상기 캐시 제어기에 송신된 추가적인 요청내의 다른 태그를 비교하도록 구성되는 비교기를 더 포함하며, 여기서 상기 비교기는 태그들이 매칭한다면 인에이블 신호를 디어서트하도록 구성되며, 상기 비교기는 상기 인에이블 신호를 상기 캐시 제어기에 제공하도록 더 구성되며, 그리고 상기 캐시 제어기는 상기 인에이블 신호가 디어서트(deassert)된 때에, 수신된 요청을 수락하지 않도록 구성되는 것을 특징으로 하는 캐시 서브시스템.

청구항 35

제 30항에 있어서,

상기 추론적인 무효화 제어기는 상기 추론적인 무효화가 비-추론적이 된 이후에 상기 제 1 캐시 라인을 후속적인 수정으로부터 보호하지 않도록 더 구성되는 것을 특징으로 하는 캐시 서브시스템.

청구항 36

제 30항에 있어서,

상기 추론적인 무효화의 일부로서, 상기 캐시 제어기는 상기 제 1 캐시 라인과 관련된 이전의 추론적인 무효화

대체 상태를 세이브하고, 아울러 마치 상기 제 1 캐시 라인이 무효화 된 것처럼, 이후의 추론적인 무효화 대체 상태를 업데이트하도록 더 구성되는 것을 특징으로 하는 캐시 서브시스템.

청구항 37

제 36항에 있어서,

상기 제 1 캐시 라인의 추론적인 무효화에 에러가 있다는 표시의 수신에 응답하여, 상기 캐시 제어기는 상기 세이브된 이전의 추론적인 무효화 대체 상태를 복구하도록 더 구성되는 것을 특징으로 하는 캐시 서브시스템.

청구항 38

제 30항에 있어서,

상기 소정수의 주기들은 상기 제 1 요청을 개시한 이벤트에 관한 에일리어스 검사를 완료하기 위한 개수들의 주기를 포함하는 것을 특징으로 하는 캐시 서브시스템.

청구항 39

제 30항에 있어서,

상기 캐시 제어기는 후속 요청이 상기 제 1 캐시 라인을 덮어쓰기 한다는 것을 검출하고, 이에 응답하여, 상기 후속 요청이 다른 캐시 라인을 덮어쓰게 함으로써 상기 제 1 캐시 라인을 보호하도록 더 구성되는 것을 특징으로 하는 캐시 서브시스템.

청구항 40

프로세서와;

시스템 메모리와;

상기 프로세서와 연결되어 있으며, 다수의 캐시 라인 저장 위치들을 포함하는 캐시와; 그리고

제 1 요청을 수신하도록 연결된 캐시 제어기- 상기 캐시 제어기는 상기 제 1 요청에 응답하여 제 1 캐시 라인 저장 위치 내의 제 1 캐시 라인을 추론적으로 무효화하도록 구성되고, 상기 제 1 캐시 라인을 추론적으로 무효화하는 것에 응답하여, 상기 캐시 제어기는 상기 제 1 캐시 라인의 무효화가 비-추론적이 될 때까지 상기 제 1 캐시 라인의 수정을 막도록 더 구성되며 -와; 그리고

상기 제 1 요청에 대응하는 적어도 하나의 검사를 수행하도록 구성된 에러 검출 유닛을 포함하며, 여기서, 만일 상기 검사가 수행되어 임의의 에러들을 검출하지 않는다면, 상기 제 1 캐시 라인의 무효화는 비-추론적이 되며,

여기서, 상기 캐시 제어기는 상기 제 1 캐시 라인의 무효화가 비-추론적인 것으로 될 때까지, 상기 제 1 캐시 라인의 상태에 따르거나 상기 제 1 캐시 라인 내의 데이터에 따르는 요청들을 수락하지 않도록 더 구성되는 것을 특징으로 하는 컴퓨터 시스템.

청구항 41

프로세서와;

시스템 메모리와;

상기 프로세서와 연결되어 있으며, 다수의 캐시 라인 저장 위치들을 포함하는 캐시와; 그리고

제 1 요청을 수신하도록 연결된 캐시 제어기- 상기 캐시 제어기는 상기 제 1 요청에 응답하여 제 1 캐시 라인 저장 위치 내의 제 1 캐시 라인을 추론적으로 무효화하도록 구성되고, 상기 제 1 캐시 라인을 추론적으로 무효화하는 것에 응답하여, 상기 캐시 제어기는 상기 제 1 캐시 라인의 무효화가 비-추론적이 될 때까지 상기 제 1 캐시 라인 저장 위치의 수정을 막도록 더 구성되며 -와; 그리고

상기 제 1 요청에 대응하는 적어도 하나의 검사를 수행하도록 구성된 에러 검출 유닛을 포함하며, 여기서, 만일 상기 검사가 수행되어 임의의 에러들을 검출하지 않는다면, 상기 제 1 캐시 라인의 무효화는 비-추론적이 되며,

여기서, 상기 캐시 제어기는 상기 제 1 캐시 라인의 상기 무효화가 비-추론적이 될 때까지, 각각의 추가적인 요청의 타입, 그리고 각각의 추가적인 요청이 상기 추론적으로 제 1 캐시 라인에 의존하는지 또는 이러한 제 1 캐

시 라인을 수정하는지에 근거하여 추가적인 요청들을 수락하지 않도록 구성되는 것을 특징으로 하는 컴퓨터 시스템.

청구항 42

캐시에서의 라인을 추론적으로 무효화하는 방법으로서,

상기 캐시에서의 라인을 무효화하는 요청을 수락하는 단계와;

상기 무효화에 에러가 있는지 여부를 판단하는 검사들을 시작하는 단계와;

상기 수락 단계에 응답하여, 상기 라인을 추론적으로 무효화하는 단계- 상기 추론적으로 무효화하는 단계는 상기 라인이 무효임을 표시하는 단계 및 상기 검사들이 완료될 때까지 후속하는 수정으로부터 상기 라인을 보호하는 단계를 포함하며, 그리고 상기 보호 단계는 상기 검사들이 완료될 때까지 상기 추론적으로 무효화된 라인의 태그의 제 1 부분과 매칭하는 추가적인 요청들을 수락하지 않는 단계를 포함하며 -와; 그리고

상기 무효화에 에러가 있다고 판단한 상기 검사들 중 하나에 응답하여, 상기 라인이 유효임을 표시함으로써 상기 추론적인 무효화를 번복하는 단계를 포함하는 것을 특징으로 하는 캐시에서의 라인을 추론적으로 무효화하는 방법.

청구항 43

다수의 캐시 라인 저장 위치들을 포함하는 캐시와;

제 1 요청을 수신하도록 연결되고, 상기 제 1 요청의 수신에 응답하여 제 1 캐시 라인 저장 위치 내의 제 1 캐시 라인을 추론적으로 무효화하도록 구성된 캐시 제어기- 상기 캐시 제어기는 상기 추론적인 무효화에 에러가 있음이 검출된 경우에, 상기 제 1 캐시 라인의 상기 추론적인 무효화를 복구하도록 더 구성되며, 상기 추론적인 무효화가 잘못된 것인지를 검출하는데에 소정수의 주기들을 취하며, 상기 소정수의 주기들은 상기 제 1 요청을 개시한 이벤트에 관한 에일리어스 검사를 완료하기 위한 개수들의 주기를 포함하며 -와; 그리고

상기 제 1 캐시 라인을 추론적으로 무효화하는 상기 캐시 제어기에 응답하여, 상기 소정수의 주기 동안 상기 제 1 캐시 라인을 수정으로부터 보호하도록 구성된 추론적인 무효화 제어기를 포함하며, 여기서 상기 추론적인 무효화는 상기 소정수의 주기 이후에 비-추론적이 되는 특징으로 하는 캐시 서브시스템.

청구항 44

다수의 캐시 라인 저장 위치들을 포함하는 캐시와;

제 1 요청을 수신하도록 연결되고, 상기 제 1 요청의 수신에 응답하여 제 1 캐시 라인 저장 위치 내의 제 1 캐시 라인을 추론적으로 무효화하도록 구성된 캐시 제어기- 상기 캐시 제어기는 상기 추론적인 무효화에 에러가 있음이 검출된 경우에, 상기 제 1 캐시 라인의 상기 추론적인 무효화를 복구하도록 더 구성되며, 상기 추론적인 무효화가 잘못된 것인지를 검출하는데에 소정수의 주기들을 취하며, 상기 캐시 제어기는 후속 요청이 상기 제 1 캐시 라인을 덮어쓰기 한다는 것을 검출하고, 이에 응답하여, 상기 후속 요청이 다른 캐시 라인을 덮어쓰게 함으로써 상기 제 1 캐시 라인을 보호하도록 더 구성되며 -와; 그리고

상기 제 1 캐시 라인을 추론적으로 무효화하는 상기 캐시 제어기에 응답하여, 상기 소정수의 주기 동안 상기 제 1 캐시 라인을 수정으로부터 보호하도록 구성된 추론적인 무효화 제어기를 포함하며, 여기서 상기 추론적인 무효화는 상기 소정수의 주기 이후에 비-추론적이 되는 특징으로 하는 캐시 서브시스템.

명 세 서

기 술 분 야

<1> 본 발명은 캐시들에 관한 것으로, 특히, 캐시에서의 라인들을 무효화하는 것에 관한 것이다.

배 경 기 술

<2> 메인 시스템 메모리는 전형적으로 속도보다는 밀도에 치중하여 설계되기 때문에, 마이크로프로세서 설계자들은 그들의 설계들에 캐시들을 추가함으로써 메인 메모리에 직접 액세스하는 마이크로프로세서의 요구를 감소시켰다. 캐시는 상기 메인 메모리보다 훨씬 빨리 액세스할 수 있는 소형 메모리이다. 컴퓨터 시스템들은 다

수의 서로 다른 레벨의 캐시들을 가질 수 있다. 예를 들면, 하나의 컴퓨터 시스템은 하나의 "레벨 1"(L1) 캐시와 하나의 "레벨 2"(L2) 캐시를 가질 수 있다. 이 캐시들은 전형적으로 상기 마이크로프로세서와 집적된다. 캐시들은 전형적으로 메인 시스템 메모리에 이용되는 메모리들(전형적으로 동적 랜덤 액세스 메모리들(DRAM들) 또는 동기식 동적 랜덤 액세스 메모리들(SDRAM들))보다 고속의 액세스 시간을 갖는 정적 랜덤 액세스 메모리들(SRAM들)과 같은 고속 메모리 셀들로 구성된다. 저밀도 및 고비용 때문에 상기 고속의 SRAM들은 전형적으로 메인 시스템 메모리에 이용되지 않는다.

<3> 많은 다른 타입의 캐싱(caching)이 또한 가능하다. 예를 들면, 상기 메인 시스템 메모리는 시스템의 저속 직접 액세스 저장 장치들(예를 들면, 하드 디스크 드라이브들)에 대한 캐시로서 동작할 수 있다. 하드 드라이브들과 같은 기타 다른 장치들은 또한 내부 캐시들을 포함할 수 있어 그들의 성능을 개선시킬 수 있다.

<4> 마이크로프로세서가 메모리로부터 데이터를 요구할 때, 마이크로프로세서는 전형적으로 먼저 그의 L1 캐시를 검사하여, 상기 요구된 데이터가 캐시되었는지를 확인한다. 상기 요구된 데이터가 캐시되지 않았다면, 상기 L2 캐시를 검사한다. 그와 동시에, 상기 L2 캐시에 미스(miss)가 존재하는 경우 상기 데이터는 메모리로부터 요청될 수 있다. 상기 L2 캐시가 상기 데이터를 저장하고 있다면, 상기 L2 캐시는 (전형적으로 상기 메인 시스템 메모리보다 훨씬 더 높은 속도 및 더 낮은 레이턴시(latency)에서) 상기 데이터를 상기 마이크로프로세서에 제공하고, 상기 데이터가 메모리로부터 요청되었다면, 그 요청은 취소될 수 있다. 상기 데이터가 상기 L1 캐시 또는 L2 캐시에 캐시되지 않은 경우("캐시 미스(cache miss)"라 칭함), 상기 데이터는 메인 시스템 메모리 또는 어떤 타입의 대용량 저장 장치(예를 들면, 하드 디스크 드라이브)로부터 판독된다. 상기 L1 캐시로부터의 데이터에 액세스하는 것과 관련하여, 메모리에의 액세스들은 더 많은 클럭 주기들을 취한다. 이와 유사하게, 상기 데이터가 상기 메인 시스템 메모리에 있지 않은 경우, 대용량 저장 장치로부터의 데이터에 액세스하는 것은 훨씬 더 많은 주기들을 취한다.

<5> 캐시들은 전형적으로 참조의 지역성(locality of reference)의 원리로 동작을 행하는 바, 이것은 가장 최근에 이용된 데이터(그리고 그 참조 지역성의 데이터)가 그 데이터의 나머지보다 액세스되기 쉽다는 것을 말한다. 이러한 원리는 컴퓨터 소프트웨어가 전형적으로 이전에 실행된 코드로 하여금 재실행되게 하는 루프들 및 브랜치들을 갖기 때문에 유지된다. 캐시에 최근에 액세스된 명령들 및 데이터를 저장함으로써, 상기 마이크로프로세서는 상기 명령들 및 데이터가 메인 메모리로부터 판독되기를 기다릴 필요가 없기 때문에 시스템 성능이 증가될 수 있다.

<6> 마이크로프로세서 및 컴퓨터 시스템 설계자들은 브랜치 예측과 같은 기술들을 이용하여 상기 캐시에 명령들 및 데이터를 사전행동적으로 저장함으로써 이 명령들 및 데이터가 상기 마이크로프로세서에 의해 실제로 요구되기 전에, 상기 참조 지역성 원리의 1단계를 행하였다. 또한, 명령 또는 데이터 바이트가 메모리로부터 판독될 때, 상기 명령 또는 데이터 다음에 오는 추가적인 바이트들이 판독 및 캐시된다. 또다시, 상기 참조 지역성의 최고는 이들 명령 및 데이터 바이트들이 보통 다른 데이터 또는 명령들보다 상기 프로세서에 의해 더 요구되기 쉽도록 지시한다.

<7> 상기 시스템 메모리를 상기 캐시로 매핑하는 여러 서로 다른 방법들이 존재한다. 하나의 일반적인 방식은 n-웨이 세트 어소시에이티브 캐시(n-Way set-associative cache)를 이용하며, 이 방식으로 상기 캐시는 세트들로 세그먼트된다. 각 세트는 n개의 캐시 라인을 포함한다. 하나의 캐시 라인은 순차적인 그룹의 바이트들(예를 들면, 32 또는 64)이다. 효율성의 목적으로, 캐시 메모리 트랜잭션들은 전형적으로 단일 바이트들 보다는 캐시 라인들에 있다. 메인 메모리 내의 캐시가능한 위치들은 각각 캐시 라인들의 세트들 중 하나에 할당될 수 있다. 결과적으로, 각 위치는 그의 할당된 세트내의 n 위치들 중 어느 하나에서 캐시될 수 있다. 상기 n-웨이 세트 어소시에이티브 캐시의 하나의 특별한 경우는 직접 매핑 캐시(direct-mapped cache)이다. 직접 매핑 캐시에 있어서, n=1이고, 이에 따라 각 메모리 위치는 상기 캐시 안의 오직 하나의 위치에 매핑된다. 상기 n-웨이 세트 어소시에이티브 캐시의 다른 특별한 경우는 완전 어소시에이티브 캐시(fully associative cache)이다. 이 경우, n=m이고, 여기서 m은 상기 캐시 내의 라인들의 수이다(그리고 이에 따라 오직 하나의 "세트"가 존재하게 된다). 이 경우, 각 메모리 위치는 상기 캐시 위치들 중 어느 위치에 매핑할 수 있다.

<8> 캐시들에 대한 2가지 기본적인 성능 판단기준은 적중률(hit ratio)(즉, 메모리 액세스들의 전체 수에 대한 상기 캐시에서 발견된 메모리 액세스들의 비) 및 서치 속도(search speed)(즉, 적중 또는 미스 판단이 얼마나 신속하게 이루어질 수 있는지)이다. 직접 매핑 캐시에 있어서, 서치 속도는 적중률을 희생하여 최적화된다. 이것은 적중들/미스들을 판단하기가 비교적 용이하지만(하나의 메모리 위치는 오직 하나의 캐시 라인에 매핑하기 때문에, 오직 그 라인이 검사되어야 한다) 다중 메모리 위치들이 단일 캐시 라인에 매핑하기 때문에 높은 적중률을 갖기

가 어렵기 때문이다. 반대로, 완전 어소시에이티브 캐시들은 서치 속도를 희생하여 적중률을 최적화한다. 모든 메모리 위치들이 임의의 캐시 라인에 매핑할 수 있게 함으로써, 적중이 존재할 확률을 개선시키지만, 모든 캐시 라인들이 각 메모리 위치에 대해 서치되어야 하기 때문에 서치들의 복잡성을 매우 증대시키게 된다. 세트 어소시에이티브 캐시들은 직접 매핑 캐시들보다 더 나은 결합성(이에 따라 더 높은 적중률)을 제공하고, 또한 완전 어소시에이티브 캐시들보다 더 고속의 서치 속도를 제공함으로써 그 둘 사이를 절충시키고자 한다.

<9> 캐시 크기가 다수의 인자들(다이(die) 크기, 전력 소비 및 비용 포함)에 의해 한정되기 때문에, 상기 캐시에 정보를 적재할 때 주의해야 한다. 설계자의 특정 관심 분야는 캐시 내의 기존 명령들 및 데이터를 겹쳐쓰기 또는 무효화하여 신규 명령들 및 데이터의 자리를 마련하는 정책을 판단할 때 발생한다. 따라서, 세트 어소시에이티브 캐시들(여기서, n>1(이에 따라 특정 메모리 위치를 캐시하는 라인에 관한 선택들이 존재함))에 있어서, 신규 데이터를 채우는 가능한 캐시 라인들을 선택하는 어떤 방법이 있어야 한다. 일반적인 해결책은, 각각의 캐시된 메모리 위치에 대한 액세스의 상대적인 순서를 추적하고, 그 다음에 최근 최소 이용 명령들 또는 데이터를 신규 명령들 또는 데이터로 대체하는 것이다. 이러한 해결책은 최근에 액세스된 캐시 라인들이 다시 액세스되기 쉽다는 원리에 근거한다. 다른 해결책들은 랜덤 대체 및 선입선출 기술들을 포함한다.

<10> 평균적으로, 최근 최소 이용(LRU: least-recently used) 캐시 대체 알고리즘들은 다른 알고리즘들보다 더 우수한 성능을 제공한다. 그러나, n-웨이 세트 어소시에이티브 캐시에서 상기 최근 최소 이용(LRU) 캐시 라인을 판단하기 위해서, 통상적인 방식들은 상기 LRU 알고리즘을 실시하기 위해서 카운터들 및 n-웨이 멀티플렉서들을 포함하는 상당량의 복잡한 하드웨어를 필요로 한다. 추가적으로, 각 캐시 엔트리에 대한 상태 비트들이 각 엔트리의 이용을 추적한다. 상기 세트에서 새로운 엔트리가 이루어질 때, 상기 상태 비트들은 상기 캐시 라인들 중 어느 캐시 라인이 최근 최소 이용 또는 무효인지를 판단하기 위해 스캔(scan)된다. 그 다음, 상기 최근 최소 이용 또는 무효 라인은 상기 새로운 엔트리의 자리를 마련하기 위해서 퇴출된다. 통상의 LRU 대체 알고리즘의 결점 중에는 상기 알고리즘을 구현하는데 필요한 하드웨어 및 상태 비트 시간의 양 뿐만 아니라 상기 세트 내의 무효 기입들을 스캔하는데 필요한 시간 및 하드웨어가 있다.

<11> 일반적으로, 캐시 서브시스템들의 성능을 개선시키는 것이 바람직하다. 예를 들면, 프로세서 속도가 개선됨에 따라, 더 많은 데이터를 더 신속하게 공급할 수 있는 캐시 서브시스템들을 제공하는 것이 바람직하다.

발명의 상세한 설명

<12> 캐시에서 라인을 무효화하는 것이 종종 유용할 수 있다. 그러나, 캐시 라인 무효화는 많은 근본적인 인자들(factors)에 좌우된다. 많은 상황들에서, 에러 검사가 수행되어, 이들 근본적인 인자들이 정확하였는지 여부를 판단한다. 이 인자들이 정확하지 않다면, 상기 무효화에 에러가 있으므로, 수행되어서는 안된다. 에러 검사는 완료하는데 상당량의 시간이 걸리기 때문에, 무효화에 에러가 있는지 여부에 관한 판단은 상기 무효화 요청이 캐시 제어기에 의해 실제로 수락될 때 이용가능하지 않을 수 있다. 그 결과, 무효화 요청들은 캐시 제어기로 하여금 에러 해결을 위한 검사를 기다리는 시간을 소비하도록 함으로써, 상기 캐시 제어기가 다른 계류중인 작업으로 이동하지 못하게 한다. 동시에, 캐시 라인 무효화에 에러가 있게 되는 것은 드문 일이므로, 상기 캐시 제어기가 에러 검사를 완료하기 위해 기다리는데 시간이 종종 낭비된다.

<13> 캐시 제어기가 캐시 라인을 추론적으로 무효화하도록 구성된다면, 상기 캐시 제어기는 에러 검사가 완료되기를 기다리는 대신에 무효화 요청에 즉시 응답할 수 있다. 상기 무효화에 에러가 있으므로 수행되어서는 안되는 보기드문 상황을 처리하기 위해서, 상기 캐시 제어기는 또한 에러 검사가 완료될 때까지 추론적으로 무효화된 캐시 라인을 변경으로부터 보호한다. 이러한 방식으로, 나중에 상기 무효화에 에러가 있다고 판명된다면, 상기 추론적인 무효화는 번복(reverse)될 수 있다.

<14> 따라서, 캐시에서의 라인을 추론적으로 무효화하는 방법들 및 시스템들의 다양한 실시예들이 개시된다. 일 실시예에 있어서, 컴퓨터 시스템은 프로세서, 시스템 메모리, 캐시 제어기, 캐시 및 에러 검출 유닛을 포함한다. 상기 캐시는 상기 프로세서와 연결되어 있으며, 다수의 캐시 라인 저장 위치들을 포함한다. 상기 캐시 제어기는 제 1 캐시 라인을 무효화하는 제 1 요청을 수신하도록 연결된다. 상기 제 1 요청의 수신에 응답하여, 상기 캐시 제어기는 상기 제 1 캐시 라인을 추론적으로 무효화하도록 구성된다. 상기 추론적인 무효화가 추후 번복되어야 하는 경우에 상기 제 1 캐시 라인을 보호하기 위하여, 상기 캐시 제어기는 상기 제 1 캐시 라인의 무효화가 비-추론적이 될 때까지 상기 제 1 캐시 라인 저장 위치의 변경을 막도록 더 구성된다. 상기 에러 검출 유닛은 상기 제 1 요청에 대응하는 적어도 하나의 검사를 수행하도록 구성된다. 예를 들면, 상기 에러 검출 유닛은 상기 캐시 제어기 자체일 수도 있고, 상기 검사는 상기 추론적인 무효화를 유발시키는 동작(예를 들면, 더 높은 레벨의 캐시로부터의 채움 요청(fill request)에 응답하여 배타적인 캐시 내의 적중)이 상기 제 1 캐시 라인의 상태를

적절하게 주어지도록 하는 검사를 포함할 수 있다. 상기 에러 검출 유닛이 상기 검사를 수행하고, 임의의 에러들을 검출하지 않는다면, 상기 제 1 캐시 라인의 무효화는 비-추론적이 된다.

<15> 일 실시예에 있어서, 상기 캐시 제어기는 상기 제 1 캐시 라인과 관련된 유효성 비트를 토글링(toggling)함으로써 상기 제 1 캐시 라인을 추론적으로 무효화하도록 구성될 수 있다. 이에 따라, 상기 추론적인 무효화를 번복하는 것은 상기 제 1 캐시 라인이 다시 유효함을 나타내기 위해 상기 유효성 비트를 재토그링(re-toggling)하는 것을 포함할 수 있다. 또한, 어떤 실시예들에 있어서, 상기 캐시 제어기는 상기 제 1 캐시 라인의 상기 무효화가 비-추론적이 될 때까지 상기 제 1 캐시 라인에서의 상태 또는 데이터에 따르는 요청들을 수락하지 않도록 구성될 수 있다. 이러한 방식으로, 이들 요청들은 상기 추론적인 무효화가 비-추론적이 되거나 번복될 때까지 지연될 수 있다. 일반적으로, 상기 제 1 캐시 라인의 상기 무효화가 비-추론적이 될 때까지, 상기 캐시 제어기는 요청의 타입, 그리고 상기 요청이 상기 추론적으로 무효화된 제 1 캐시 라인에 의존하는지 또는 이 제 1 캐시 라인을 변경하는지에 근거하여 추가적인 요청들을 수락하지 않도록 구성될 수 있다. 예를 들면, 상기 캐시 제어기는 상기 제 1 캐시 라인을 적중하는 더 높은 레벨의 캐시로부터의 채움 요청들, 상기 제 1 캐시 라인에 대한 프로브(probe) 또는 상태 변경 요청들 또는 상기 더 높은 레벨의 캐시로부터의 카피 백(copy backs)으로서 상기 제 1 캐시 라인을 대체하기 위해 상기 제 1 캐시 라인을 선택하는 카피 백인 상기 추가적인 요청들을 수락하지 않도록 구성될 수 있다. 상기 캐시 제어기는 상기 제 1 캐시 라인의 태그의 부분을 매칭하는 추가적인 요청들을 수락하지 않도록 구성될 수 있다.

<16> 어떤 실시예들에 있어서, 상기 추론적인 무효화의 부분으로서, 상기 캐시 제어기는 상기 제 1 캐시 라인과 관련된 이전의 추론적인 무효화 대체 상태(예를 들면, 대체하기 위해 라인을 선택하는데 이용된 상태)를 세이브(save)하고, 상기 제 1 캐시 라인이 무효화되었다면 상기 제 1 캐시 라인의 이후의 추론적인 무효화 대체 상태를 업데이트하도록 구성될 수 있다. 상기 추론적인 무효화에 에러가 있다고 추후에 판단된다면, 상기 캐시 제어기는 상기 추론적인 무효화를 번복할 때 세이브된 이전의 추론적인 무효화 대체 상태를 복구할 수 있다.

<17> 다른 실시예에 있어서, 캐시에서의 라인을 추론적으로 무효화하는 방법이 개시되어 있다. 상기 방법은 상기 캐시에서 상기 라인이 무효화되게 하는 요청을 수락하는 단계와, 상기 무효화에 에러가 있는지 여부를 판단하는 검사들을 시작하는 단계와, 그리고 상기 라인을 추론적으로 무효화하는 단계를 포함한다. 상기 라인을 추론적으로 무효화하는 단계는 상기 라인이 무효임을 표시하는 단계와, 상기 검사들이 완료될 때까지 상기 라인을 후속의 변경으로부터 막는 단계를 포함한다. 상기 검사들 중 하나가 상기 무효화에 에러가 있다고 판단한다면, 상기 방법은 또한 상기 라인이 다시 유효임을 표시함으로써 상기 추론적인 무효화를 번복하는 단계를 포함한다.

<18> 다른 실시예에 있어서, 배타적인 캐시에서 제 1 캐시 라인을 추론적으로 무효화하는 방법이 개시되어 있다. 상기 방법은 더 높은 레벨의 캐시로부터의 채움 요청을 수락하는 단계와, 상기 채움 요청이 상기 배타적인 캐시에서 적중하는지 여부를 판단하는 단계와, 상기 채움 요청이 에러가 있게 시작되었는지 여부를 판단하는 검사를 시작하는 단계와, 그리고 상기 채움 요청이 상기 배타적인 캐시에서 적중한다면, 상기 배타적인 캐시로부터 상기 더 높은 레벨의 캐시에 제 1 캐시 라인을 제공하는 단계를 포함할 수 있다. 상기 제 1 캐시 라인이 상기 더 높은 레벨의 캐시에 제공될 때 상기 검사들이 아직 완료되지 않았다면, 상기 제 1 캐시 라인은 추론적으로 무효화될 수 있다. 추론적으로 무효화하는 단계는 상기 제 1 캐시 라인이 무효화임을 표시하는 단계와, 상기 검사들이 완료될 때까지 상기 제 1 캐시 라인을 후속의 변경으로부터 막는 단계를 포함할 수 있다.

<19> 또 다른 실시예에 있어서, 캐시 서브시스템이 개시되어 있다. 상기 캐시 서브시스템은 캐시와 캐시 제어기를 포함한다. 상기 캐시 제어기는 제 1 캐시 라인을 추론적으로 무효화하도록 구성된다. 상기 추론적인 무효화에 에러가 있다고 검출되면, 상기 캐시 제어기는 상기 추론적인 무효화를 번복하도록 구성될 수 있다. 상기 캐시 서브시스템은 또한 상기 무효화가 비-추론적이 될 때까지 상기 제 1 캐시 라인을 변경으로부터 보호하도록 구성된 추론적인 무효화 제어기를 포함할 수 있다. 상기 추론적인 무효화에 에러가 있는지를 검출하는 것은 어떤 수의 주기를 취할 수 있고, 이에 따라 상기 추론적인 무효화가 그 어떤 수의 주기 후까지 비-추론적이 되지 않게 된다.

실 시 예

<30> **캐시 서브시스템들**

<31> 컴퓨터 시스템들(하기의 도 6 및 도 7에 예시된 것과 같음)은 전형적으로 캐시 서브시스템들을 이용한다. 전형적으로 이 캐시 서브시스템들은 프로세서와 집적된 L1 캐시 및 집적되지 않은 L2 캐시를 포함한다. 그러나, 프로세서 기술의 진보가 반도체 디바이스들의 고집적을 가능하게 함에 따라, 마이크로프로세서 설계자들은 이제

성능을 또한 개선하기 위한 방법으로서 칩내장의 레벨 2(L2) 캐시들을 포함할 수 있다. 상기 L2 캐시를 집적함으로써, 더이상 핀들을 통해 통신이 일어날 필요가 없기 때문에 상기 L1과 L2 캐시들 간의 전송 레이턴시 및 전송 대역폭이 개선될 수 있다.

<32> 전형적인 방식들은 추가적인 액세스 레이턴시를 발생시키지 않고 상기 L1 캐시를 가능한 한 크게 만들고자 하는 목적을 가지고 제1의 (L1) 캐시를 설계하였다. 이러한 큰 L1 캐시는 전형적으로 보통 상기 L1 캐시의 레이턴시보다는 더 크지만,시스템 메모리의 레이턴시보다는 더 작은 레이턴시를 가지며, 보통 상기 L1 캐시만큼 크거나 상기 L1 캐시보다 큰 유사한 큰 L2 캐시와 통신한다.

<33> L2 캐시는 포괄적이거나 배타적이거나 또는 어느 쪽도 아니도록 설계될 수 있다. 이상적인 포괄적인 L2 캐시에 있어서, 도 1A의 벤다이어그램에 의해 표시된 바와 같이, 상기 L1 캐시의 모든 라인이 또한 상기 L2 캐시에 있다. 이와 반대로, 이상적인 포괄적인 L2 캐시에 있어서, 도 1B의 벤다이어그램에 도시된 바와 같이, 상기 L1 캐시의 라인들이 상기 L2 캐시에 있지 않다. 어떤 실시예들에 있어서, 상기 포괄적인 L2 캐시와 상기 배타적인 L2 캐시는 둘 다 (도 1A 및 도 1B에 도시된 바와 같이) 시스템 메모리에 대해 포괄적이고, 이에 따라 "배타적인" L2 캐시는 오직 L1 캐시에 대해서만 배타적이라는 것에 주목할 필요가 있다. 포괄적이지도 배타적이지도 않은 캐시 시스템에 있어서, 상기 L2 캐시에 저장된 라인들은 상기 L1 캐시에 저장된 라인들에 의존하지 않고, 이에 따라 상기 L1 캐시에 라인이 존재하는지 또는 상기 L2 캐시에 라인이 존재하는지에 관한 어떠한 보증도 존재하지 않는다.

<34> 이러한 원리들 각각에 대해서, 장점들과 단점들 모두 존재한다. 전형적으로, 대부분의 L2 캐시들은 포괄적이다. 포괄적인 캐시에 있어서, L1 대 L2 데이터 전송들의 수가 감소될 수 있다. 전송수에 있어 이러한 감소는 변경되지 않은 L1 희생(victim)이 상기 L2 캐시에 카피백될 필요가 없기 때문에 발생한다. 희생은 그것이 재사용되기 전에 메인 메모리에 돌아가서 기입되는 데이터 블록이다. 예를 들면, 상기 L1 캐시 내의 채움이 기존의 라인이 대체되는 것을 요구한다면, 상기 기존의 라인, 즉 희생이 유입 라인의 자리를 마련하기 위해 상기 L1 캐시로부터 방출된다. 포괄적인 캐시에 있어서, L1 희생의 카피(copy)가 상기 L2 캐시에 이미 존재한다. 따라서, 상기 L1 희생이 변경되지 않았으며, 상기 L1 캐시에 있는 경우라면, 동일한 라인이 이미 존재하기 때문에 상기 L2 캐시에 그 라인을 카피할 필요가 없다. 오프칩(off-chip) L2 캐시에 대해서, 포괄성은 또한 매우 중요할 수 있는데, 이는 상기 L1 캐시 내의 라인이 상기 L2 캐시에 있도록 보장됨으로써 L1 대 L2 카피백(L1 to L2 copy back) 동안에 외부 L2 태그 룩업(lookup)을 수행할 필요성을 배제할 수 있기 때문이다. 추가적으로, 상기 L2 캐시는 (예를 들면, 인-라인(in-line) L2 구성이 이용될 때) 상기 L1 캐시에 대한 스누프 필터(snoop filter)일 수 있다.

<35> 포괄적인 캐시의 하나의 주요 결점은 포괄적인 캐시가 효율적인 캐시 크기를 감소시킨다는 점이다. 상기 L1 내의 모든 라인이 또한 상기 L2에 있기 때문에, 상기 L1 및 L2 캐시들의 효율적인 캐시 크기는 도 1A에 도시된 바와 같이, 오직 상기 L2 캐시의 크기이다. 이것은 L2 캐시들이 전형적으로 L1 캐시들보다 더 크기 때문이다. 다른 결점은 포괄성을 유지해야 하기 때문에 상기 캐시 제어기들에 요구되는 더 복잡한 제어기 설계이다. 예를 들면, 잠재적인 L2 희생이 선택될 때, 상기 포괄적인 캐시 제어기는 상기 L1 캐시를 다시 백-프로브(back-probe)하여 상기 L2 희생이 상기 L1 캐시에 현재 존재하지 않게 해야 한다. 잠재적인 L2 희생이 상기 L1 캐시에 존재한다면, 포괄성을 유지하기 위해 다른 희생이 선택되어야 한다. 대안으로, 어떤 실시예들에 있어서, 상기 잠재적인 L2 희생이 또한 상기 L1 캐시에 존재한다면, 상기 L1 캐시 내의 대응하는 라인을 무효화함으로써 포괄성이 유지될 수 있다(다른 L2 희생을 선택하는 것과 대조적임).

<36> 캐시들이 전형적으로 저밀도이기 때문에, 특히, L2 캐시가 전형적으로 포괄적이라서 L1 캐시보다 더 컸기 때문에, L2 캐시들을 집적한 고가의 메모리들은 이력상 가격과 온-칩 리엘 에스테이트(real estate) 면에서 선호되지 않는 설계 선택이었다.

<37> 포괄적인 캐시들과 대조적으로, 배타적인 캐시들은 도 1B에 도시된 바와 같이, 상기 L1 캐시와 L2 캐시의 혼합된 크기와 동일한 더 큰 효율적인 캐시 크기들을 제공할 수 있다. 이러한 더 큰 효율적인 캐시 크기는 적중률을 더 좋게 할 수 있다. 또한, L2 희생들을 방출시킬 때 상기 L1 캐시를 백-프로브할 필요가 없기 때문에, 상기 L2 제어기의 복잡성을 감소시킬 수 있다.

<38> 배타적인 캐시를 이용할 때, 상기 L1 캐시와 상기 L2 캐시 간의 전송 수가 증가한다. 예를 들면, L1 대 L2 데이터 전송들의 수는 L1 희생들이 변경되었는지 여부에 상관없이 참조 최고 지역성에 따라 상기 L2 캐시에 복사될 수 있다. 상기 L2 캐시에 L1 희생들을 저장시키면, L1 희생이 아마도 상기 L2 캐시 내의 다른 라인들 중 어느 라인보다 더 최근에 이용되었기 때문에 상기 참조 지역성을 만족시킬 수 있다. 어떤 경우들에서, 증가된 L1 대

L2 트래픽은 포괄적인 캐시를 더 바람직하게 함으로써, 상기 L1 캐시와 L2 캐시 간의 통신 링크에 의존하게 된다.

<39> 배타적인 캐시들은 또한 다른 캐시 응용들에서도 유용하다. 예를 들면, 배타적인 캐시는 그래픽 시스템 또는 하드 디스크 드라이브와 함께 사용하기 위해 멀티-레벨 캐시 서브시스템을 설계하는데 유용하다. 이들 실시예들에 있어서, 배타적인 캐시를 이용하여 포괄적인 캐시에 필요한 만큼의 추가적인 캐시 메모리를 요구하지 않고 상기 캐시 서브시스템의 효율적인 크기를 증대시키는 것이 바람직하다. 어떤 실시예들에 있어서, 상기 배타적인 캐시는 더 높은 레벨의 캐시로서 동일한 기판상에 집적되지 않는다 해도 실용적인 설계 선택일 수 있다.

<40> 도 2는 실행 코어(409)에 의해 사용하기 위해 시스템 메모리(425)로부터의 데이터 및/또는 명령 라인들을 저장하도록 구성되는 캐시 서브시스템(407)의 일 실시예를 도시한다. 상기 실행 코어(409)는 상기 캐시 서브시스템 제어기(419)에게 어드레스 버스(411)에서 어드레스를 제공함으로써 데이터를 요청한다. 상기 어드레스는 가상 어드레스일 수 있다. 이 실시예에서는 동일한 캐시 서브시스템 제어기(419)가 L1 캐시와 L2 캐시를 둘다 제어하지만은, 다른 실시예들에서는 별개의 L1 및 L2 제어기들이 이용될 수 있다. 상기 L2 캐시(423)는 배타적이거나, 포괄적이거나, 또는 배타적이지도 포괄적이지도 않을 수 있다.

<41> 상기 실행 코어(409)로부터 어드레스를 수신하는 것에 응답하여, 상기 캐시 서브시스템 제어기(419)는 상기 L1 캐시(417)에 상기 어드레스를 제공할 수 있다. 어떤 실시예들에 있어서, 상기 L1 캐시(417)는 선형적으로 또는 가상적으로 어드레스될 수 있으며, 그래서 상기 실행 코어(409)로부터 수신된 어드레스가 가상 어드레스인 경우, 그 어드레스를 상기 L1 캐시(417)에 제공하기 전에 상기 어드레스를 변환할 필요가 없다. 이와는 대조적으로, 상기 L2 캐시(423)는 어떤 실시예들에서 물리적으로 어드레스될 수 있으며, 그래서 가상 어드레스들은 상기 L1 캐시(417)에 미스가 존재할 때 상기 L2 캐시(423)에 제공하기 전에 변환되어야 한다. 다른 실시예들에 있어서, 두 캐시들은 동일한 방식으로(예를 들면, 가상적으로 또는 물리적으로) 어드레스될 수 있다.

<42> L1 채움 요청이 상기 L2 캐시(423)에서 미스할 때, 상기 라인은 시스템 메모리(425)로부터 요청되어 상기 L1 캐시(417)로 가져오게 되거나(예를 들면, 상기 L2 캐시(423)가 배타적인 캐시인 경우) 또는 상기 L2 캐시와 L1 캐시로 가져오게 된다(예를 들면, 상기 L2 캐시(423)가 포괄적인 캐시인 경우). 상기 L1 캐시(417)에 상기 요청된 라인을 채우기 위해서, L1 희생이 생성되어 (상기 L2 캐시(423)가 배타적인 경우 상기 L1 희생이 지워지는지 변경되는지 또는 상기 L2 캐시(423)가 포괄적인 경우 상기 L1 희생이 변경되는지에 상관없이) 상기 L2 캐시(423)로 카피백된다. 상기 L2 캐시(423)가 배타적인 경우, 이러한 카피백은 L2 라인이 방출될 필요가 있어, L2 희생이 생성되어, 상기 희생이 변경되는 경우 메모리(425)에 되돌아가서 기입된다. 상기 L2 캐시(423)가 배타적인 일 실시예에 있어서, 상기 L1 캐시와 L2 캐시는 희생 라인들이 상기 배타적인 L2 캐시(423)(희생 라인들이 L1 희생들인 경우) 또는 시스템 메모리(425)(희생 라인들이 변경된 L2 희생들인 경우)에 카피백될 때 희생 라인들을 버퍼링하는 하나 또는 그 이상의 희생 버퍼들(421)을 공유할 수 있다.

<43> L1 채움 요청이 L2 캐시(423)에서 적중할 때, 그 라인은 상기 L2 캐시(417)에 복사된다. 상기 L1 요청이 L2 배타적인 캐시(423)에서 적중하면, 배타성을 유지하는 것과 관련된 여러 이유들을 위해 (상기 적중 라인을 유효한 채로 두는 것과 반대로) 상기 적중 라인을 무효화하는 것이 바람직하다. 예를 들면, 상기 라인이 무효화되면, 상기 캐시 서브시스템(407)에 상기 라인의 오직 하나의 카피만이 존재하게 된다. 이것은 상기 캐시 서브시스템(407)에 대한 제어 로직을 매우 간단하게 하는바, 그 이유는 그 라인의 어떤 카피가 임의의 소정의 시간에서 더 최근의 것인지를 추적할 필요가 없기 때문이다. 또한, 상기 라인을 무효화하는 것은 상기 L2 캐시(423) 내의 위치를 비움으로써, 배타적인 캐시 시스템에 의해 제공된 더 효율적인 캐시 크기가 달성될 수 있게 해준다. 그러나, 라인이 무효화되기 전에, 상기 무효화를 일으키는 동작(예를 들면, 상기 L2 캐시에 대한 채움 요청을 발생시키는 L1 캐시 내의 미스)이 에러가 있게 수행되었는지 여부를 판단하기 위하여 여러 검사들이 수행될 수 있다. 상기 검사들이 에러를 밝히면, 그 동작은 취소되고 그리고/또는 추후에 재시도된다. 따라서, 상기 무효화는 오직 이 검사들이 상기 무효화를 일으키는 상태들에 에러가 없음을 표시하는 경우에만 발생해야 한다.

<44> 포괄적인 캐시 또는 배타적이지도 포괄적이지도 않은 캐시에서의 라인들이 무효화되는 많은 상황들이 또한 존재한다. 상기 배타적인 캐시 무효화들에 대하여, 이 검사들은 결국 상기 무효화 명령이 에러가 있는지 여부를 판단하는 다양한 검사들에 의존한다. 예를 들면, 명령들이 잘못 실행중일 때, 브랜치 예측(branch prediction)의 결과로서 명령이 인출될 수 있다. 이러한 브랜치 예측이 정확하지 않다고 판명되면, 상기 인출된 명령의 실행이 취소될 필요가 있고/있거나 상기 인출된 명령의 실행의 효과들이 번복될 필요가 있다. 대안으로, 비-특권 모드(non-privileged mode)로 잘못 실행하는 명령이 특권을 갖는 명령일 수 있다. 이 명령들 중 하나가 캐시 라인 또는 블록을 상기 캐시로부터 상기 라인을 실제로 플러싱(flushing)함이 없이 무효화하는 명령인 경우, 포괄적

인 L2 캐시 내의 그 라인의 카피를 또한 무효화하는 것이 바람직하다. 그러나, 어떤 검사들이 나중에 잘못 예측된 브랜치 또는 독점 위반을 검출한 경우, 예외가 발생될 수 있고, 상기 예외 발생 명령들로부터 생기는 상기 무효화들은 번복될 필요가 없다. 따라서, 캐시에서의 무효화들은 예외 검사에 의존한다.

<45> 검사의 다른 예로는 오더링(ordering) 요건들에 응하기 위해서 상기 컴퓨터 시스템에서 데이터의 다른 버전들을 검사하는 것을 포함할 수 있다. 예를 들면, 요청된 데이터 라인의 변경된 카피가 버퍼(예를 들면, 더 높은 레벨의 캐시로부터의 기입 버퍼 또는 희생 버퍼)에 저장될 수 있다. 이러한 변경된 카피는 더 낮은 레벨의 캐시에 저장된 카피보다 더 최근에 변경되었으며, 이에 따라 이러한 카피가 검출되면, 상기 더 낮은 레벨의 캐시로의 채움 요청이 취소되어, 요청하는 디바이스는 요청된 데이터의 더 최근에 변경된 카피를 수신할 수 있다.

<46> 수행될 수 있는 다른 검사들로는 에일리어스(alias) 검사들이 있다. 에일리어스 검사는 가상적으로 어드레스된 캐시들에서 일어날 수 있는 문제점들을 발견하는 것을 포함한다. 가상 메모리 시스템들은 하나 이상의 가상 어드레스에 메모리의 물리적 페이지를 매핑할 수 있다. 이들 서로 다른 가상 어드레스들(즉, 에일리어스들)은 가상적으로 어드레스된 캐시에서 하나 이상의 위치에 캐시될 수 있다. 예를 들면, 단일 물리적 페이지로부터의 데이터는 그 데이터가 서로 다른 캐시 라인들 및 서로 다른 가상 페이지들에 매핑하는 2개 또는 그 이상의 서로 다른 가상 어드레스들에 매핑한다면, 가상적으로 어드레스된 캐시내의 다중 위치들에서 잠재적으로 캐시될 수 있다. 결과로서, 하나의 가상 어드레스를 이용하는 어떤 라인을 요청하는 것은 적중을 발생시키는 다른 에일리어스를 이용하여 동일한 라인을 요청함에도 불구하고 미스를 발생시킬 수 있다. 에일리어스 검사는 명백한 미스가 실제로 가상적으로 어드레스된 캐시에서 에일리어스 라인에서 적중함을 검출할 수 있다. 예를 들면, 일 실시예에 있어서, 가상적으로 어드레스된 L1 캐시는 독립 명령 및 데이터 캐시들을 포함할 수 있다. 상기 L1 캐시는 동일한 데이터의 카피가 상기 명령 및 데이터 캐시들 모두에 동시에 존재하도록 구성될 수 있다. 상기 L1 제어기는 상기 데이터 캐시에서 L1 희생을 선택할 때 에일리어스 카피를 검사하도록 구성되지 않는다. 그 결과, L1 희생은 그 라인의 에일리어스 카피가 상기 명령 캐시(또는 상기 희생이 상기 명령 캐시로부터 선택된 경우, 상기 데이터 캐시)에 여전히 존재함에도 불구하고 배타적인 L2 캐시에 카피백된다. 결과로서, 그 데이터에 대한 추후의 채움 요청은 (에일리어스 검사들이 완료되기 전에) 상기 L1에서 미스하지만, 상기 L2에서는 적중할 수 있다. 상기 에일리어스 검사가 상기 L1 캐시에서 상기 에일리어스 카피를 검출해야 하기 때문에, 상기 라인은 상기 L2 캐시에서 무효화되어서는 안된다. 이에 따라, 이러한 실시예에 있어서, 배타적인 캐시에서의 무효화는 상기 에일리어스 검사의 결과에 따를 수 있다.

<47> 또 다른 예의 검사는 라인의 상태에 근거하여 그 라인이 액세스될 수 있는지 여부를 판단할 수 있다. 많은 시스템은 각 라인의 MESI 또는 MOESI(변경(Modified), 소유(Owned), 배타적(Exclusive), 공유(Shared), 무효(Invalid)) 상태를 추적한다. 라인이 속한 상태에 따라서, 그 라인에서 어떤 동작들을 수행하는 것이 허용가능할 수 있거나 또는 허용가능할 수 없다. 예를 들면, L1 명령 캐시로부터의 채움 요청이 상기 L2 캐시에서의 변경된 라인에서 적중하면, 상기 변경된 L2 라인이 메모리에 기입된 후까지 상기 채움 요청을 제공하는 것은 적절하지 않다(예를 들면, 상기 명령 캐시가 캐시 일관성 상태들을 저장하지 않는 경우에 이러한 제한이 이용될 수 있다). 따라서, 어떤 검사는 상기 라인의 현재 상태를 고려하여 특정 액세스가 적절한지 여부를 판단하는 것을 포함할 수 있다.

<48> 따라서, 배타적인 캐시 또는 포괄적인 캐시에서, 캐시 라인들은 다른 캐시 동작의 부분으로서 또는 결과로서 무효화될 수 있다. 이와 동시에, 이들 무효화들은 완료하는데 비교적 긴 시간이 걸리는 다양한 검사들의 결과들에 의해 번복될 수 있다. 상기 검사들이 완료된 후까지 상기 무효화들이 지연된다면, 상기 캐시들은 바람직하지 않게 상기 검사들을 기다리는 시간 양을 소비하게 된다. 상기 캐시 제어기가 상기 라인을 무효화하기 전에 상기 검사들이 완료하기를 기다리고 있다면, 상기 캐시 제어기는 다른 동작들을 수행할 수 없으며, 따라서 이러한 지연은 캐시 성능에 부정적으로 영향을 미칠 수 있다. 대안으로, 상기 캐시 제어기가 상기 검사들이 완료되기 전에 상기 무효화를 수행한다면, 상기 무효화가 나중에 에러가 있다고 밝혀지면, 상기 무효화된 라인을 복구할 방법이 존재하지 않을 수 있다. 그 결과, 상기 검사들을 수행하면, 상기 캐시 서브시스템의 정확도를 개선할 수 있고, 이러한 개선된 정확도는 성능을 희생하여 달성할 수 있다.

<49> **캐시 제어기**

<50> 도 2에 도시된 것과 같은 캐시 서브시스템의 성능을 개선하기 위해서, 캐시 제어기는 명령 또는 요청에 응답하여, 상기 요청에 대한 검사들이 완료되지 않았음에도 불구하고, 캐시 라인들을 추론적으로 무효화하도록 구성될 수 있다. 예를 들면, 더 높은 레벨의 배타적인 캐시의 액세스 레이턴시를 최소화하기 위해서, 채움 요청(이 채움 요청은 이 채움 요청이 상기 배타적인 캐시에서 적중하는 경우, 결국 상기 배타적인 캐시에서의 라인을 무효

화함)은 상기 더 높은 레벨의 캐시에서 미스가 검출되는 즉시 상기 더 낮은 레벨의 배타적인 캐시 제어기에 전송될 수 있다. 이것은 에일리어싱 및 오더링 요건들과 같은 다양한 검사들이 수행되기 전에 상기 더 낮은 레벨의 배타적인 캐시 제어기에 요청을 전송하는 것을 포함할 수 있다. 이들 검사들은 완료하는데 여러 주기가 걸릴 수 있지만, 이 검사들이 실제로 실패하는 것은 드문 일이다. 따라서, 상기 검사들이 완료되기 전에 캐시 라인을 추론적으로 무효화하는 것은 일반적으로 정확한 결과를 발생시키고, 이와 동시에 임의의 불필요한 지연들을 피할 수 있게 한다. 가능하게는 더 나은 효율성을 제공하는 것에 더하여, 캐시 제어기가 추론적인 무효화들을 수행하도록 구성함으로써 생길 수 있는 하나의 이득은 더 나은 자원 이용이다. 예를 들면, 배타적인 캐시 제어기는 각각의 명령(command)을 개별적으로 실시해야 하는 대신에, 단일 명령으로서 태그 룩업 및 추론적인 무효화를 실시할 수 있다.

<51> 상기 검사들이 실제로 실패하는 드문 경우에, 무효화-유발 명령 또는 요청이 상기 검사들을 실패하게 한 문제점이 바로잡힌 후에 취소, 지연 및/또는 재시도될 수 있다. 예를 들면, 명령 캐시로부터의 채움 요청이 배타적인 L2 캐시에서 적중한다면, 상기 요청은 상기 L2 캐시 내의 적중 라인이 변경된 상태에 있는 경우 취소될 수 있다. 상기 적중라인이 추론적으로 무효화되었다면, 상기 검사들이 완료된 때에 상기 무효화를 번복하는 것이 불가능할 수 있다. 이러한 상황을 피하기 위해서, 추론적인 무효화들을 수행하도록 구성되는 캐시 제어기는 또한 상기 검사들이 후속하여 실패하는 경우 상기 추론적으로 무효화된 라인이 복구될 수 있게 해주는 복구 방법을 제공하도록 구성될 수 있다.

<52> 일 실시예에 있어서, 캐시 제어기는 상기 검사들이 완료된 후까지 상기 추론적으로 무효화된 라인을 보호함으로써 이러한 복구 메커니즘을 제공할 수 있다. 따라서, 상기 라인을 추론적으로 무효화함으로써, 상기 캐시 제어기들로 하여금 상기 추론적으로 무효화된 라인을 변경하지 않는 다른 동작들로 이동할 수 있게 된다. 예를 들면, 어떤 실시예들에 있어서, 캐시 제어기는 라인이 추론적으로 무효화되는 시간과 상기 검사들이 완료되는 시간 사이에 여러개의 요청들을 넘겨받는다. (상기 추론적인 무효화가 추후 번복되는 경우) 이들 일련의 명령들 또는 요청들과 상기 추론적인 무효화 사이의 충돌들을 막기 위하여, 상기 캐시 제어기는 잠재적인 충돌들을 나타내는 임의의 요청들을 수락하지 않도록 구성될 수 있다. 상기 캐시 제어기는 그들의 요청들이 실제로 상기 라인을 변경하지 않았음에도 불구하고 상기 추론적으로 무효화된 라인에 관한 요청들을 차단하도록 구성될 수 있다. 예를 들면, 상기 추론적인 무효화가 결국 번복된다면(예를 들면, 상기 검사들 중 하나가 에러를 밝혀내기 때문), 상기 검사들이 완료된 후까지 이들 변경되지 않은 요청들을 수락하지 않음으로써, 요청이 마치 상기 라인이 무효한 것처럼 에러가 있게 처리되지 못하게 할 수 있다. 따라서, 일 실시예에 있어서, 상기 캐시 제어기는 상기 추론적으로 무효화된 라인의 태그의 일부분을 매칭하는 태그들을 갖는 임의의 요청들 또는 명령들을 수락하지 않도록 구성될 수 있다.

<53> 다소 더 상세한 실시예에 있어서, 상기 캐시 제어기는 제시된 요청 또는 명령의 타입 및/또는 상기 명령의 잠재적인 효과에 근거하여 요청들을 수락하지 않도록 구성될 수 있다. 예를 들면, 상기 캐시 제어기는 상기 추론적으로 무효화된 라인의 태그를 매칭하는 채움 요청들, 프로브들 또는 상태 변경들을 차단하도록 구성될 수 있다. 또한, 상기 캐시 제어기는 태그 매칭과 상기 라인의 대체 상태 둘에 근거하여 상기 추론적으로 무효화된 라인을 대체하는 카피백들 또는 채움들을 수락하지 않도록 구성될 수 있다. 예를 들면, LRU 대체가 이용된다고 가정하면, 카피백이 상기 추론적으로 무효화된 라인의 태그를 매칭하고, 그 라인이 세트에서 최소 최근 사용 라인인 경우, 그 라인은 보통 카피백에 의해 대체될 것이다. 상기 추론적으로 무효화된 라인을 보호하기 위하여, 상기 캐시 제어기는 상기 카피백을 수락하지 않도록 구성될 수 있다. 대안으로, 상기 캐시 제어기는 상기 카피백을 수락하지만 상기 추론적으로 무효화된 라인 대신에 (현재의 대체 상태를 무시하는) 다른 라인을 대체하도록 구성될 수 있다. 상기 추론적으로 무효화된 라인의 대체 상태가, 상기 카피백 또는 채움이 상기 추론적으로 무효화된 라인을 덮어쓰기하지 않음(예를 들면, LRU 대체 방식이 이용되고, 상기 추론적으로 무효화된 라인이 상기 LRU 라인이 아님)을 표시한다면, 상기 캐시 제어기는 상기 요청을 수락하도록 구성될 수 있다.

<54> 일 실시예에 있어서, 상기 캐시 제어기는 도 3에 도시된 바와 같이, 어드레스 충돌 검출 하드웨어(500)를 포함한다. 상기 어드레스 충돌 검출 하드웨어는 라인을 추론적으로 무효화되게 하는 요청과 상기 라인이 무효화되는 시간과 상기 추론적인 무효화에 대한 검사들이 완료되는 시간 사이에 수신된 임의의 후속 요청들 또는 명령들 간의 임의의 상호작용을 검출하도록 구성될 수 있다.

<55> 상기 어드레스 충돌 검출 하드웨어(500)는 상기 추론적으로 무효화된 라인에 대응하는 어드레스의 전체 또는 부분을 저장하도록 구성된 어드레스 레지스터(504)를 포함할 수 있다. 예를 들면, 상기 어드레스 레지스터(504)는 추론적으로 무효화된 라인의 태그를 저장한다. 다중 라인들을 추론적으로 무효화하는 능력을 제공하기 위하여, 여러개의 이러한 레지스터들(504)이 저장 유닛(502)에 제공될 수 있다. 비교기(501)는 계류중인 요청 내의 어드

레스와 상기 어드레스 레지스터(504) 및/또는 저장 유닛(502) 내의 어드레스(들)를 비교한다. 상기 어드레스들이 매칭하지 않으면, 상기 비교기(501)는 인에이블 신호를 어서트(assert)한다. 상기 인에이블 신호의 상태에 근거하여, 상기 캐시 제어기는 상기 계류중인 요청을 수락하거나 수락하지 않을 수 있다. 예를 들면, 상기 계류중인 요청의 어드레스가 상기 추론적으로 무효화된 라인에 대응하는 상기 어드레스(또는 상기 어드레스의 부분)와 매칭한다면, 상기 비교기(501)는 상기 인에이블 신호를 디어서트(deassert)함으로써, 상기 캐시 제어기로 하여금 상기 계류중인 요청을 수락하지 않게 한다.

<56> 추론적인 무효화 제어기(506)는 어떤 실시예들에 있어서 상기 어드레스 충돌 검출 하드웨어(500)를 제어한다. 상기 추론적인 무효화 제어기(506)는 추론적으로 무효화된 라인의 어드레스 또는 태그의 일부분으로 상기 저장 유닛(502) 내의 상기 레지스터들(504) 중 하나를 적재하도록 구성될 수 있다. 상기 추론적인 무효화 제어기(506)는 어떤 검사들이 완료하는데 걸리는 최대 주기 수 동안 상기 추론적으로 무효화된 라인(들)을 보호하도록 구성될 수 있다. 예를 들면, 상기 추론적인 무효화 제어기(506)는 에일리어스 검사가 완료하는데 걸리는 주기 수 동안 상기 추론적으로 무효화된 라인(들)을 보호하도록 구성될 수 있다. 다른 실시예에 있어서, 상기 추론적인 무효화 제어기(506)는 예외 검사가 완료되는데 걸리는 주기 수 동안 상기 추론적으로 무효화된 라인(들)을 보호할 수 있다.

<57> 일 실시예에 있어서, 상기 추론적인 무효화 제어기(506)는 상기 추론적으로 무효화된 라인의 태그 또는 어드레스의 부분을 레지스터(504)에 적재하고, 상기 비교기(501)에 의해 그 레지스터 내의 값을 유입 요청들과 비교되게 함으로써 어떤 수의 주기 동안 상기 추론적으로 무효화된 라인을 보호할 수 있다. 특정 수의 주기 후에, 상기 추론적인 무효화 제어기(506)는 상기 레지스터(504)를 클리어(clear)하거나, 또는 상기 비교기(501)에게 레지스터(504) 내의 값과 유입 요청들을 더이상 비교하지 않도록 명령할 수 있다. 상기 추론적인 무효화 제어기는 또한 보호할 추론적으로 무효화된 라인들이 존재하지 않는 경우 상기 인에이블 신호를 어서트함으로써 상기 비교기(501)를 바이패스(bypass)하도록 구성될 수 있다.

<58> 일 실시예에 있어서, 상기 검사들이 완료될 때까지, 상기 어드레스 충돌 검출 하드웨어(500)는 상기 캐시 제어기로 하여금, 추가적인 채움 요청들을 상기 추론적으로 무효화된 라인과 관련이 없는 한 수락할 수 있게 한다. 이와 유사하게, 추가적인 프로브 또는 상태-변경 요청들은 상기 추론적으로 무효화된 라인을 포함하지 않는 한 수락될 수 있고, L1 대 L2 카피백은 대체를 위해 상기 추론적으로 무효화된 라인을 선택하지 않는 한 수락될 수 있다. 이에 따라, 추가적인 비교들은 어떤 타입들의 명령들을 검출할 수 있고, 오직 그들의 명령들만이 상기 저장 유닛(502)내의 상기 어드레스 레지스터들(504) 내의 값들과 비교될 수 있다.

<59> L1 대 L2 카피백이 상기 추론적으로 무효화된 라인을 선택하는지 여부를 검출하기 위하여, 상기 어드레스 충돌 검출 하드웨어는 대체를 위해 어느 라인이 선택되는지를 검출하도록 추가적인 로직을 포함할 수 있다. 예를 들면, LRU 대체를 이용하는 N-웨이 세트 어소시에이티브 캐시에서, 카피백은 일반적으로 세트 내의 최근 최소 이용을 대체할 것이다. 적절한 세트 내의 최근 최소 이용 라인이 상기 추론적으로 무효화된 라인인 경우, 상기 카피백은 수락되어서는 안 된다. 대안으로, 상기 카피백을 수락하지 않는 대신에, 상기 카피백은 수락될 수 있지만, 상기 추론적으로 무효화된 라인을 대체하는 대신에, 상기 카피백은 상기 추론적으로 무효화된 라인의 LRU 상태를 무시하여, 상기 세트에서 다른 라인을 대체할 수 있다. 따라서, 상기 어드레스 충돌 검출 하드웨어(500)는 추론적으로 무효화된 라인이 상기 라인 또는 상기 라인을 포함하는 세트의 대체 상태(예를 들면, LRU 상태, FIFO 등)에 따라 카피백에 의해 대체될 것인지 여부를 판단하는 로직을 포함할 수 있다. 카피백이 상기 추론적으로 무효화된 라인을 대체할 것이라면, 상기 어드레스 충돌 검출 하드웨어(500)는 상기 인에이블 신호를 디어서트하거나 또는 상기 카피백이 다른 라인을 대체하게 할 수 있다.

<60> 따라서, 도 2를 참조하면, 검사들이 완료되는데 여러 주기들을 기다리는 대신, 상기 캐시 제어기(419)는 제 1 요청 또는 명령에 응답하여 상기 제 1 요청 또는 명령과 관련된 검사들이 아직 완료되지 않은 경우에 라인을 추론적으로 무효화하도록 구성될 수 있다. 상기 캐시 제어기는 상기 라인이 무효하다고 표시함으로써(예를 들면, 유효로부터 무효까지의 유효성 비트를 토글링함으로써) 상기 라인을 추론적으로 무효화할 수 있다. 하나의 바람직한 실시예에 있어서, 상기 캐시 제어기는 상기 라인이 상기 라인에 저장된 데이터에 영향을 미침이 없이 무효하다고 표시함으로써, 상기 검사들 중 어느 검사라도 실패하는 경우 상기 라인은 더 쉽게 무효화되지 않을 수 있다. 상기 캐시 제어기(419)는 또한 어드레스 충돌 검출 하드웨어(500)와 같은 보호 메커니즘을 포함하여, 추론적으로 무효화된 라인들을 보호한다. 이에 따라, 상기 캐시 제어기(419)는 추론적으로 무효화한 라인과 관련하여 보호 메커니즘을 시작할 수 있다. 예를 들면, 상기 추론적으로 무효화된 라인에 대응하는 어드레스의 전체 또는 부분은 어드레스 레지스터(504)에 적재될 수 있음으로써, 도 3에 예시된 상기 보호 메커니즘을 시작할 수 있다. 상기 무효화에 대응하는 검사들이 실패한다면, 상기 캐시 제어기(419)는 상기 라인이 한번더 유효함을 표

시함으로써(예를 들어, 무효로부터 유효까지 유효성 비트를 토글링함으로써) 상기 추론적인 무효화를 번복할 수 있다. 상기 캐시 제어기(419)는 또한 상기 검사들이 완료된 후에 상기 보호 메커니즘을 차단하도록 구성될 수 있다.

<61> 이전에 언급된 바와 같이, 많은 실시예들에 있어서, 상기 캐시 제어기(419)는 LRU 또는 FIFO 대체 방식을 이용할 수 있다. 이러한 방식을 구현하기 위하여, 상기 캐시 제어기는 각 라인 또는 라인들의 그룹에 대한 대체 상태 정보를 세이브할 수 있다. LRU 대체 방식을 구현하는 일 실시예에 있어서, 이러한 대체 상태는 상기 캐시 내의 각각의 라인 또는 캐시의 일부분(예를 들면, 세트 결합 세트 내의 세트)의 관련 이용 상태를 표시할 수 있다. 대안으로, FIFO 대체 방식을 구현하는 일 실시예에 있어서, 상기 대체 상태는 각각의 라인이 상기 캐시 또는 상기 캐시의 일부분(예를 들어, 세트)에, 상기 캐시의 그 부분 내의 다른 라인들과 비교하여 얼마나 오래 있는지를 표시할 수 있다.

<62> 라인이 추론적으로 무효화될 때, 상기 캐시 또는 상기 캐시의 부분에 대응하는 대체 상태는 마치 상기 추론적인 무효화가 비-추론적인 무효화인 것처럼 업데이트될 수 있다. 그러나, 상기 추론적인 무효화가 나중에 취소된다면, 상기 업데이트된 대체 상태 정보는 상기 라인에 대한 적절한 대체 상태를 반영하지 않을 수 있다. 보상하기 위하여, 상기 캐시 제어기(419)는 상기 대체 상태를 업데이트하기 전에 라인을 추론적으로 무효화할 때 이전에 존재한 대체 상태를 세이브하여 상기 무효화를 반영하도록 구성될 수 있다. 상기 추론적인 무효화가 나중에 번복된다면, 상기 캐시 제어기(419)는 상기 세이브된 대체 상태를 복구하도록 구성될 수 있다. 따라서, 어떤 실시예들에 있어서, 상기 캐시 제어기(419)는 추론적인 무효화들을 수행할 대 상기 대체 상태를 변경 및 세이브하도록 구성될 수 있다.

<63> 이제 도 4A와 도 4B를 참조하면, 배타적인 캐시에서 추론적인 무효화들을 수행하는 방법의 일 실시예가 도시된다. 상기 기능적인 블록들이 상기 예시된 실시예에서 어떤 순서로 배열되어 있지만은, 다른 실시예들이 상기 기능적인 블럭들의 서로 다른 배라인들을 이용할 수 있기 때문에, 이러한 배열은 단지 예시적인 것일 뿐이고, 상기 방법이 임의의 특정 시간 순서를 요구함을 의미하지는 않는다. 단계(601)에서, 상기 배타적인 캐시 제어기에 의해 채움 요청이 수락된다. 어떤 실시예들에 있어서, 상기 배타적인 캐시는 L2 캐시일 수 있고, 상기 배타적인 캐시 제어기는 실제로 상기 L1 캐시와 L2 캐시 둘다를 제어할 수 있다. 상기 배타적인 캐시가 상기 요청을 수락하기 전 또는 후 어떤 시점에서, 단계(603)에서 표시된 바와 같이, 상기 요청 캐시 내의 미스 및/또는 상기 요청으로부터 생기는 상기 배타적인 캐시 내의 적중에 대한 검사들이 시작된다.

<64> 단계(605)에서 표시된 바와 같이, 단계(601)에서 수락된 상기 채움 요청이 상기 배타적인 캐시에서 적중하면, 상기 배타적인 캐시는 상기 요청 캐시에 상기 요청된 라인의 데이터를 제공한다. 편의상, 상기 요청된 라인은 본원에서 이하 "LineN"로 언급된다. 배타성을 유지하기 위하여, 단계(607)에서 표시된 바와 같이, 상기 배타적인 캐시 제어기는 LineN을 무효화한다. 그러나, 이러한 무효화는 단계(603)에서 시작된 상기 검사들이 완료되기 전에 일어나기 때문에, 상기 캐시 제어기는 상기 무효화를 추론적으로 수행한다. 상기 검사들이 실패하는 경우에 상기 무효화를 번복하기 위하여, 단계(607)에서 상기 무효화가 그 대체 상태 정보에 영향을 미치고 상기 무효화를 반영하도록 상기 대체 상태 정보를 업데이트한다면, 상기 캐시 제어기는 또한 LineN에 대응하는 이전의 대체 상태 정보(즉, LRU 정보 또는 FIFO 정보와 같은 세트 또는 캐시 내의 어떤 라인이 먼저 대체될 것인지를 판단하는 정보)를 세이브한다.

<65> 상기 추론적인 무효화의 추가 구성요소는 LineN를 보호하는 보호 메커니즘을 시작하는 상기 배타적인 캐시 제어기를 포함할 수 있다. 따라서, LineN이 무효화됨으로써, 다른 요청들을 수락하도록 상기 배타적인 캐시 제어기를 비우고, 상기 배타적인 캐시 제어기는 상기 검사들이 완료된 후까지 LineN에 잠재적으로 영향을 미치거나 또는 LineN에 따르는 요청들을 수락하는 것을 연기하거나 거절한다. 이에 따라, 단계(611)에서, 상기 배타적인 캐시에 새로운 요청이 제공되고, 단계(613)에서 도시된 바와 같이, 그 요청이 LineN에 잠재적으로 영향을 미치거나 또는 LineN에 따른다면, 상기 캐시 제어기는 상기 요청을 수락하지 않는다. 예를 들면, 상기 캐시 제어기는 LineN을 덮어쓰기하는 요청을 수락하지 않는다. 그러나, 상기 요청이 LineN과 관련되어 있지 않으면, 단계(617)에서 표시된 바와 같이, 상기 배타적인 캐시 제어기는 상기 요청을 수락하고 그에 따라 상기 요청을 처리한다. 어떤 실시예들에 있어서, 상기 배타적인 캐시 제어기는 상기 요청의 태그가 LineN의 태그와 매칭하는지를 검사함으로써 요청이 LineN에 잠재적으로 영향을 미치는지 여부를 판단할 수 있다. 일 실시예에 있어서, 카피백 요청이 제공될 때, 상기 배타적인 캐시 제어기는 또한 LineN이 현재의 대체 상태를 고려하여, 상기 카피백에 의해 대체되는지 여부를 판단하는 로직을 포함할 수 있다. LineN이 대체되면, 상기 배타적인 캐시 제어기는 상기 카피백을 수락하지 않거나 또는 상기 카피백을 수락할 수 있지만, 상기 카피백이 LineN 대신에 다른 라인을 대체하도록 할 수 있다. 단계(603)에서 시작된 상기 검사들이 실패한 경우, 단계(623)에서, 상기 배타적인

캐시 제어기는 상기 복구 메커니즘을 이용하여 상기 무효화를 번복할 수 있다. 예를 들면, 상기 배타적인 캐시 제어기는 (예를 들면, 무효로부터 유효까지 유효성 비트를 토글링함으로써) 상기 라인이 유효하다고 다시 표시하고, LineN에 대응하는 무효화 이전의 대체 상태 정보를 복구한다. 그 대신 단계(603)에서 시작된 검사들이 통과되었다면, 상기 배타적인 캐시 제어기는 상기 추론적으로 무효화된 LineN에 잠재적으로 영향을 미치는 요청들을 차단하는 것을 중단한다. 일단 상기 검사들이 통과되었다면, 상기 무효화는 비-추론적인 무효화로서 다루어질 수 있고, 이에 따라 단계(607)에서 대체 상태 정보가 세이브되면, 상기 세이브된 정보는 일단 상기 검사들이 완료되면 더이상 유지되지 않는다.

<66> 어떤 실시예에 있어서, 상기 캐시 제어기가 단계(601)에서 수락된 상기 요청을 완료하기 전에 단계(603)에서 시작된 상기 검사들 중 어떤 검사가 완료될 수 있다. 이러한 상황에서, 상기 캐시 제어기는 (단계(607)에서 도시된 상기 추론적인 무효화를 수행하는 것과 대조적으로) 상기 라인을 추론적으로 무효화하지 않는다.

<67> 단계(601)에서 수락된 상기 채움 요청이 상기 배타적인 캐시에서 미스한 경우, 단계(609)에서 표시된 바와 같이, 상기 채움 요청이 이미 더 낮은 레벨의 캐시 또는 시스템 메모리에 전송되지 않았다면, 상기 요청 캐시는 더 낮은 레벨의 캐시 또는 시스템 메모리에 채움 요청을 전송할 수 있다. 많은 실시예들에 있어서, 상기 요청 캐시는 시스템 메모리에 채움 요청을 전송함과 동시에 상기 배타적인 캐시에 채움 요청을 전송하여, 상기 채움 요청이 상기 배타적인 캐시에서 미스한 경우 상기 시스템 메모리 레이턴시를 최소화할 수 있다. 상기 채움 요청을 전송한 후, 상기 배타적인 캐시는 다른 요청들을 계속하여 수신하고 이 요청들에 응답한다.

<68> 이제 도 5를 참조하면, 캐시에서, 상기 캐시가 배타적이든지, 포괄적이든지, 또는 배타적이지도 포괄적이지도 않은지에 상관없이, 추론적인 무효화를 실시하는 방법의 일 실시예가 제공된다. 단계(701)에서, 캐시 제어기가 명령 또는 요청을 수락한다. 예를 들면, 수락된 명령은 채움 요청, 카피백, 프로브, 상태 변경 또는 무효화일 수 있다. 이러한 요청을 충족시키면, 결국 상기 수락 캐시에서의 라인을 무효화하도록 요구할 수 있다. 그러나, 상기 명령 또는 요청(예를 들면, 더 높은 레벨의 캐시에서의 미스) 또는 상기 무효화를 일으키는 이벤트(예를 들면, 배타적인 캐시에서의 적중)의 원천에서 (단계(703)에서 도시된 바와 같이) 어떤 검사들이 수행될 수 있고, 이 검사들이 실패하는 경우, 상기 요청 및/또는 상기 무효화는 취소될 수 있다. 이 검사들이 완료하는데 여러 주기들이 걸리기 때문에, 상기 캐시 제어기는 상기 라인을 추론적으로 무효화하여, 상기 캐시 제어기는 상기 검사들이 완료되는 동안에 계속하여 새로운 요청들을 수락할 수 있다. 따라서, 단계(707)에서, 상기 캐시 제어기는 상기 라인에 대한 유효성 비트를 토글링함으로써 상기 라인을 추론적으로 무효화하고, 상기 무효화가 상기 무효화된 라인에 대응하는 상기 대체 상태 정보에 영향을 미친다면, 이전의 대체 상태 정보를 세이브하고, 상기 대체 상태 정보를 업데이트하여 상기 무효화를 반영한다.

<69> 상기 캐시 제어기는 단계(703)에서 시작된 상기 검사들이 완료될 때까지 상기 추론적으로 무효화된 라인에 영향을 미치거나 이 라인에 따르는 요청들을 수락하지 않음으로써 상기 추론적으로 무효화된 라인을 보호할 수 있다. 따라서, 상기 무효화된 라인과 잠재적으로 관련된 새로운 요청이 제공되고, 상기 검사들이 완료되지 않았다면, 단계(719)에서 도시된 바와 같이, 상기 캐시 제어기는 상기 요청을 수락하지 않는다. 단계(717)에서, 상기 캐시 제어기는 상기 추론적으로 무효화된 라인과 잠재적으로 관련되지 않은 요청을 수락할 수 있다. 상기 검사들이 통과되면, 상기 무효화가 더이상 추론적이지 않기 때문에, 상기 캐시 제어기는 상기 추론적으로 무효화된 라인을 보호하는 것을 끝낼 수 있다. 상기 검사들이 실패하면, 단계(723)에서 도시된 바와 같이, 상기 캐시 제어기는 상기 라인이 유효함을 표시하기 위해 상기 무효화된 라인에 대한 유효성 비트를 토글링함으로써 상기 무효화를 번복할 수 있고, 대체 상태 정보가 세이브되었다면, 상기 세이브된 대체 상태 정보를 복구한다.

<70> 어떤 실시예들에 있어서, 상기 검사들 중 어느 검사라도 상기 캐시 제어기가 단계(701)에서 수락된 상기 요청을 완료하기 전에 실패한다면, 상기 요청은 라인이 추론적으로 무효화되기 전에 취소될 수 있다. 이러한 상황에서, 상기 캐시 제어기는 (단계(707)에서 도시된 상기 추론적인 무효화를 수행하는 것과 대조적으로) 상기 라인을 추론적으로 무효화하지 않는다.

<71> **컴퓨터 시스템**

<72> 도 6은 버스 브리지(202)를 통해 다양한 시스템 구성요소들과 연결된 프로세서(10)를 포함하는 컴퓨터 시스템(200)의 일 실시예의 블럭도를 도시한다. 컴퓨터 시스템의 다른 실시예들도 가능하며 고려될 수 있다. 상기 도시된 시스템에서, 메인 메모리(204)는 메모리 버스(206)를 통해 버스 브리지(202)와 연결되며, 그래픽 제어기(208)는 AGP 버스(210)를 통해 버스 브리지(202)와 연결된다. 여러개의 PCI 디바이스들(212A-212B)은 PCI 버스

(214)를 통해 버스 브리지(202)와 연결된다. 또한, EISA/ISA 버스(220)를 통해 하나 또는 그 이상의 EISA 또는 ISA 디바이스들(218)과의 전기적 인터페이스를 제공하기 위해 보조 버스 브리지(216)가 제공된다. 이 실시예에 있어서, 프로세서(10)는 CPU 버스(224)를 통해 버스 브리지(202)와 연결되며, 선택적 L2 캐시(228)와 연결된다. 어떤 실시예들에 있어서, 상기 프로세서는 집적된 L1 캐시(도시되지 않음)를 포함할 수도 있다.

<73> 버스 브리지(202)는 프로세서(10), 메인 메모리(204), 그래픽 제어기(208) 및 PCI 버스(214)에 장착된 디바이스들 사이에 인터페이스를 제공한다. 버스 브리지(202)와 연결된 디바이스들 중 하나로부터 동작이 수신될 때, 버스 브리지(202)는 상기 동작의 타겟(예를 들면, 특정 디바이스 또는 PCI 버스(214)의 경우, 상기 타겟은 PCI 버스(214)가 된다)을 식별한다. 버스 브리지(202)는 상기 타겟 디바이스에 상기 동작을 보낸다. 버스 브리지(202)는 일반적으로 동작을 소스 디바이스 또는 버스에 의해 이용되는 프로토콜로부터 상기 타겟 디바이스 또는 버스에 의해 이용되는 프로토콜로 변환한다.

<74> PCI 버스(214)에 ISA/EISA 버스로의 인터페이스를 제공하는 것에 더하여, 보조 버스 브리지(216)는 추가 기능을 통합할 수 있다. 컴퓨터 시스템(200)내에는, 보조 버스 브리지(216)의 외부에 있거나 또는 보조 버스 브리지(216)와 집적된 입력/출력 제어기(도시되지 않음)가 또한 포함될 수 있어, 키보드 및 마우스(222)와 다양한 직렬 및 병렬 포트들에게 동작 지원을 제공한다. 다른 실시예들에 있어서, 프로세서(10)와 버스 브리지(202) 간의 CPU 버스(224)에 외부 캐시 유닛(도시되지 않음)이 또한 연결될 수 있다. 대안으로, 상기 외부 캐시는 버스 브리지(202)에 연결될 수 있으며, 상기 외부 캐시를 위한 캐시 제어 로직은 버스 브리지(202) 안에 집적될 수 있다. L2 캐시(228)는 프로세서(10)와의 백사이드(backside) 구성으로 도시된다. L2 캐시(228)는 프로세서(10)와 분리될 수 있거나, 프로세서(10)와 함께 카트리지(예를 들면, 슬롯 1 또는 슬롯 A)에 집적되거나, 또는 프로세서(10)와 함께 반도체 기판 상에 집적될 수도 있음에 주목할 필요가 있다.

<75> 메인 메모리(204)는 응용 프로그램들이 저장되며, 프로세서(10)가 우선적으로 실행하는 메모리이다. 적절한 메인 메모리(204)는 DRAM(동적 랜덤 액세스 메모리)을 포함한다. 예를 들면, 다수의 뱅크들의 SDRAM(동기식 DRAM) 또는 램버스 DRAM(RDRAM)이 적절할 수 있다.

<76> PCI 디바이스들(212A-212B)은 네트워크 인터페이스 카드들, 비디오 가속 장치들, 오디오 카드들, 하드 또는 플로피 디스크 드라이브들 또는 드라이브 제어기들, SCSI(소형 컴퓨터 시스템 인터페이스) 어댑터들 및 전화 카드들과 같은 다양한 주변 장치들의 예시이다. 이와 유사하게, ISA 디바이스(218)는 모뎀, 사운드 카드 및 GPIB 또는 필드(field) 버스 인터페이스 카드와 같은 다양한 데이터 포착 카드등과 같은 다양한 타입의 주변 장치의 예시이다.

<77> 그래픽 제어기(208)는 디스플레이(226)에서 텍스트 및 이미지의 표현을 제어하도록 제공된다. 그래픽 제어기(208)는 메인 메모리(204)안에 그리고 메인 메모리(204)로부터 효과적으로 천이될 수 있는 3차원 데이터 구조를 표현하기 위해 이 기술분야에 일반적으로 알려져 있는 전형적인 그래픽 가속 장치를 포함할 수 있다. 따라서, 버스 브리지(202)내의 타겟 인터페이스에 대한 액세스를 요청 및 수신할 수 있음으로써 메인 메모리(204)에 대한 액세스를 획득할 수 있다는 점에서 그래픽 제어기(208)는 AGP 버스(210)의 마스터일 수 있다. 전용 그래픽 버스는 메인 메모리(204)로부터 데이터의 고속 검색을 제공해준다. 특정 동작들의 경우, 그래픽 제어기(208)는 AGP 버스(210)에 PCI 프로토콜 트랜잭션들을 발생시키도록 더 구성될 수 있다. 그러므로, 버스 브리지(202)의 상기 AGP 인터페이스는 AGP 프로토콜 트랜잭션들 뿐만 아니라 PCI 프로토콜 타겟 및 개시 트랜잭션들을 둘다 지원하는 기능을 포함할 수 있다. 디스플레이(226)는 이미지 또는 텍스트가 나타나는 임의의 전자식 디스플레이이다. 적절한 디스플레이(226)로는 음극선관("CRT"), 액정 표시기("LCD") 등이 있다.

<78> 상기 AGP, PCI 및 ISA 또는 EISA 버스들은 상기의 설명에서 예들로서 이용되었으며, 필요한 경우 임의의 버스 아키텍처들이 대체될 수 있다는 것에 주목할 필요가 있다. 또한, 컴퓨터 시스템(200)은 추가적인 프로세서들(예를 들면, 컴퓨터 시스템(200)의 선택적 구성요소로서 도시된 프로세서(10a))을 포함하는 멀티프로세싱 컴퓨터 시스템일 수 있다는 것에 주목할 필요가 있다. 프로세서(10a)는 프로세서(10)와 유사할 수 있다. 특히, 프로세서(10a)는 프로세서(10)의 동일 카피(copy)일 수 있다. 프로세서(10a)는 (도 6에 도시된 바와 같이) 독립적인 버스를 통해 버스 브리지(202)와 연결될 수 있거나, 또는 프로세서(10)와 CPU 버스(224)를 공유할 수도 있다. 또한, 프로세서(10a)는 L2 캐시(228)와 유사한 선택적 L2 캐시(228a)와 연결될 수 있다.

<79> **멀티-노드 프로세싱 시스템**

<80> 이제 도 7을 참조하면, 컴퓨터 시스템(300)의 다른 실시예가 도시된다. 다른 실시예들은 가능하며 고려될 수 있

다. 도 7의 상기 실시예에 있어서, 컴퓨터 시스템(300)은 여러개의 프로세싱 노드들(312A, 312B, 312C 및 312D)을 포함한다. 각 프로세싱 노드는 각각의 개별 프로세싱 노드(312A-312D)내에 포함된 메모리 제어기(MC)(316A-316D)를 통해 개별 메모리(314A-314D)에 연결된다. 추가적으로, 프로세싱 노드들(312A-312D)은 상기 프로세싱 노드들(312A-312D) 간에 통신을 행하는데 이용되는 인터페이스 로직을 포함한다. 예를 들면, 프로세싱 노드(312A)는 프로세싱 노드(312B)와 통신을 행하는 인터페이스 로직(318A)과, 프로세싱 노드(312C)와 통신을 행하는 인터페이스 로직(318B)과, 그리고 또다른 프로세싱 노드(도시되지 않음)와 통신을 행하는 제3의 인터페이스 로직(318C)으로 포함한다. 이와 유사하게, 프로세싱 노드(312B)는 인터페이스 로직(318D, 318E 및 318F)을 포함하고; 프로세싱 노드(312C)는 인터페이스 로직(318G, 318H 및 318I)을 포함하고; 그리고 프로세싱 노드(312D)는 인터페이스 로직(318J, 318K 및 318L)을 포함한다. 프로세싱 노드(312D)는 인터페이스 로직(318L)을 통해 다수의 입력/출력 디바이스들(예를 들면, 데이지 체인 구성(daisy chain configuration)의 디바이스들(320A-320B)과 통신을 행하기 위해 연결된다. 다른 프로세싱 노드들은 유사한 방식으로 다른 I/O 디바이스들과 통신을 행할 수 있다.

<81> 프로세싱 노드들(312A-312D)은 프로세싱 노드 간 통신을 위해 패킷 기반의 링크를 실시한다. 본 실시예에 있어서, 상기 링크는 단일 방향의 라인들의 세트들로서 실시된다(예를 들면, 라인들(324A)은 프로세싱 노드(312A)로부터 프로세싱 노드(312B)에 패킷들을 전송하고, 라인들(324B)은 프로세싱 노드(312B)로부터 프로세싱 노드(312A)에 패킷들을 전송한다). 다른 세트들의 라인들(324C-324H)은 도 7에 예시된 바와 같이, 다른 프로세싱 노드들 간에 패킷들을 전송하는데 이용된다. 일반적으로, 각 세트의 라인들(324)은 하나 또는 그 이상의 데이터 라인들, 상기 데이터 라인들에 대응하는 하나 또는 그 이상의 클럭 라인들, 그리고 전달되는 패킷의 타입을 표시하는 하나 또는 그 이상의 제어 라인들을 포함할 수 있다. 상기 링크는 프로세싱 노드들 간의 통신을 위해 캐시 일관성 방식으로 또는 프로세싱 노드와 I/O 디바이스(또는 상기 PCI 버스 또는 ISA 버스와 같은 통상의 구성 요소의 I/O 버스에 대한 버스 브리지) 간의 통신을 위해 비-일관성(non-coherent) 방식으로 동작을 행할 수 있다. 또한, 상기 링크는 도시된 바와 같이 I/O 디바이스들 간에 데이지 체인 구조를 이용하여 비-일관성 방식으로 동작을 행할 수 있다. 한 프로세싱 노드로부터 타 프로세싱 노드로 전송될 패킷이 하나 또는 그 이상의 중간 노드들을 통해 지나갈 수 있음에 주목할 필요가 있다. 예를 들면, 프로세싱 노드(312A)에 의해 프로세싱 노드(312D)에 전송된 패킷은 도 7에 도시된 바와 같이, 프로세싱 노드(312B) 또는 프로세싱 노드(312C)를 통해 지나갈 수 있다. 임의의 적절한 라우팅 알고리즘이 이용될 수 있다. 컴퓨터 시스템(300)의 다른 실시예들은 도 7에 도시된 실시예보다 더 많은 또는 더 적은 프로세싱 노드들을 포함할 수 있다.

<82> 일반적으로, 상기 패킷들은 노드들 간의 라인들(324)에서 하나 또는 그 이상의 비트 시간(bit time)들로서 전송될 수 있다. 비트 시간은 해당 클럭 라인들에서 상기 클럭 시그널의 상승 또는 하강 에지일 수 있다. 상기 패킷들은 트랜잭션들을 시작하는 명령 패킷들, 캐시 일관성을 유지하는 프로브 패킷들, 그리고 프로브들 및 명령들에 응답하는 응답 패킷들을 포함할 수 있다.

<83> 메모리 제어기 및 인터페이스 로직에 추가하여, 프로세싱 노드들(312A-312D)은 하나 또는 그 이상의 프로세서들을 포함할 수 있다. 대체로, 프로세싱 노드는 적어도 하나의 프로세서를 포함하며, 메모리 및 필요한 경우 다른 로직과 통신을 행하는 메모리 제어기를 선택적으로 포함할 수 있다. 특히, 각각의 프로세싱 노드(312A-312D)는 프로세서(10)의 하나 또는 그 이상의 카피(copy)들을 포함할 수 있다. 외부 인터페이스 유닛(18)은 상기 노드내에 상기 인터페이스 로직(318) 뿐만 아니라 상기 메모리 제어기(316)를 포함할 수 있다.

<84> 메모리들(314A-314D)은 임의의 적절한 메모리 디바이스들을 포함할 수 있다. 예를 들면, 메모리(314A-314D)는 하나 또는 그 이상의 램버스 DRAM들(RDRAM들), 동기식 DRAM들(SDRAM들), 동적 RAM 등을 포함할 수 있다. 컴퓨터 시스템(300)의 어드레스 공간(address space)은 메모리들(314A-314D) 사이에서 분리된다. 각 프로세싱 노드(312A-312D)는 어떤 메모리들(314A-314D)에 어떤 어드레스들이 매핑되는지를 결정하는데 이용되는 메모리 맵(memory map)을 포함한다. 일 실시예에 있어서, 컴퓨터 시스템(300)내의 어드레스에 대한 일관성 지점은 상기 어드레스에 대응하는 바이트들을 저장하는 메모리와 연결된 메모리 제어기(316A-316D)이다. 다시 말하면, 상기 메모리 제어기(316A-316D)는 상기 대응하는 메모리(314A-314D)에 대한 각각의 메모리 액세스가 캐시 일관성 방식으로 발생하도록 하는 역할을 한다. 메모리 제어기들(316A-316D)은 메모리들(314A-314D)에 인터페이스하는 제어 회로를 포함할 수 있다. 또한, 메모리 제어기들(316A-316D)은 메모리 요청들을 큐잉(queuing)하는 요청 큐(queue)들을 포함할 수 있다.

<85> 일반적으로, 인터페이스 로직(318A-318L)은 상기 링크로부터 패킷들을 수신하고 상기 링크에서 전송될 패킷들을 버퍼링하는 다양한 버퍼들을 포함할 수 있다. 컴퓨터 시스템(300)은 패킷들을 전송하는 임의의 적절한 흐름 제어 메커니즘을 이용할 수 있다. 예를 들면, 일 실시예에 있어서, 각 인터페이스 로직(318)은 그 인터페이스 로

직이 연결된 링크의 타단에서 상기 수신기내의 각 타입의 버퍼의 총 개수를 저장한다. 상기 인터페이스 로직은 수신 인터페이스 로직이 패킷을 저장하기 위해 비어있는 버퍼를 가지고 있지 않으면, 패킷을 전송하지 않는다. 패킷을 라우팅함으로써 수신 버퍼가 비워질 때, 상기 수신 인터페이스 로직은 상기 버퍼가 비어있음을 표시하기 위해 전송 인터페이스 로직에 메시지를 전송한다. 이러한 메커니즘은 "쿠폰-기반(coupon-based)" 시스템이라 칭해질 수 있다.

<86> I/O 디바이스들(320A-320B)은 임의의 적절한 I/O 디바이스들일 수 있다. 예를 들면, I/O 디바이스들(320A-320B)은 상기 디바이스들이 연결될 수 있는 다른 컴퓨터 시스템(예를 들면, 네트워크 인터페이스 카드들 또는 모뎀들)과 통신을 행하는 디바이스들을 포함할 수 있다. 또한, I/O 디바이스들(320A-320B)은 비디오 가속 장치들, 오디오 카드들, 하드 또는 플로피 디스크 드라이브들 또는 드라이브 제어기들, SCSI(소형 컴퓨터 시스템 인터페이스) 어댑터들 및 전화 카드, 사운드 카드, 그리고 GPIB 또는 필드 버스 인터페이스 카드들과 같은 다양한 데이터 포착 카드들을 포함할 수 있다. "I/O 디바이스"라는 용어와 "주변 장치"라는 용어는 본원에서 동일한 것으로 의도된다는 것을 주목할 필요가 있다.

<87> 다수의 변형들 및 수정들은 상기 개시물을 완전히 이해한 이 기술분야의 당업자에게 명백해질 것이다. 하기의 청구항들은 이러한 변형들 및 수정들을 모두 포함하는 것으로 해석될 수 있다.

산업상 이용 가능성

<88> 본 발명은 캐시들에 산업상 이용가능하다.

도면의 간단한 설명

<20> 도 1A는 일 실시예에 있어서 포괄적인 캐시들 간의 관계를 도시한 벤다이어그램이고;

<21> 도 1B는 일 실시예에 있어서 배타적인 캐시들 간의 관계를 도시한 벤다이어그램이고;

<22> 도 2는 캐시 서브시스템의 일 실시예의 블록도이고;

<23> 도 3은 도 2에 도시된 것과 같은 캐시 서브시스템으로 이용될 수 있는 어드레스 충돌 검출 하드웨어의 일 실시예를 도시하고;

<24> 도 4는 일 실시예에 따른 배타적인 캐시에서 무효화가 어떻게 수행되는지를 예시하는 흐름도이고;

<25> 도 4B는 도 4A에 예시된 흐름도의 계속이고;

<26> 도 5는 캐시에서, 상기 캐시가 포괄적이든 배타적이든 또는 어느 쪽도 아니든 간에 상관없이, 라인들을 추론적으로 무효화하는 방법의 일 실시예를 도시하고;

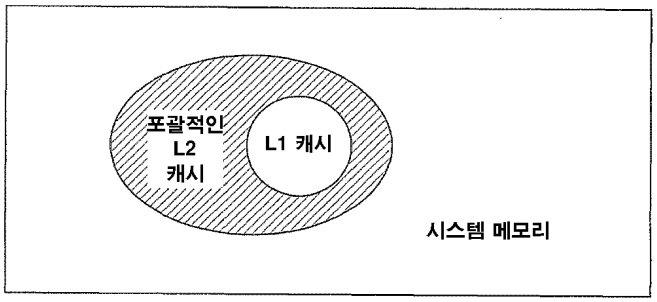
<27> 도 6은 컴퓨터 시스템의 일 실시예의 블록도이고; 그리고

<28> 도 7은 멀티-노드 프로세싱 시스템의 일 실시예를 도시한다.

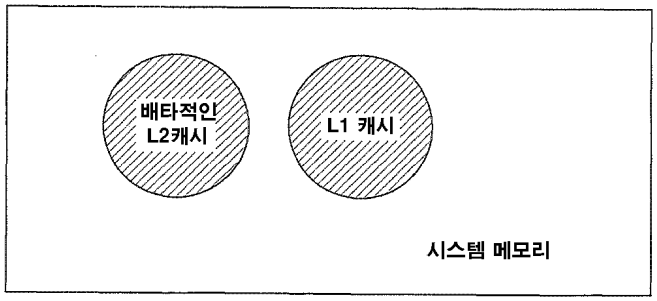
<29> 본 발명은 다양한 수정들 및 변형들을 갖지만, 본원에서는 특정 실시예들을 예시적으로 도면들에 도시하여 상세하게 설명된다. 그러나, 이러한 특정 실시예들은 본 발명을 개시된 특정 형태들로만 한정하지 않음을 이해해야 한다. 본 발명은 첨부된 청구항들에 의해 정의되는 본 발명의 정신과 범위내에 있는 모든 수정들, 등가물들 및 대안들을 포함한다.

도면

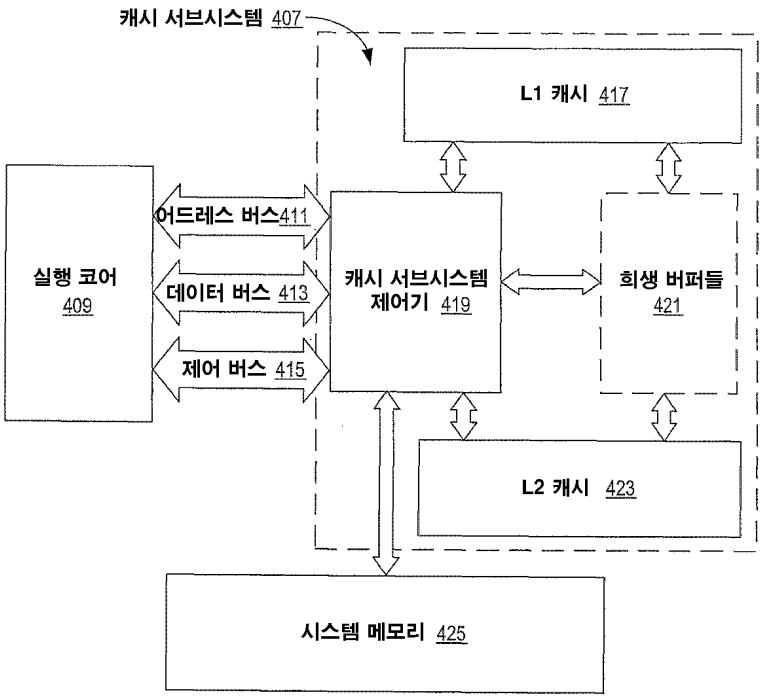
도면1A



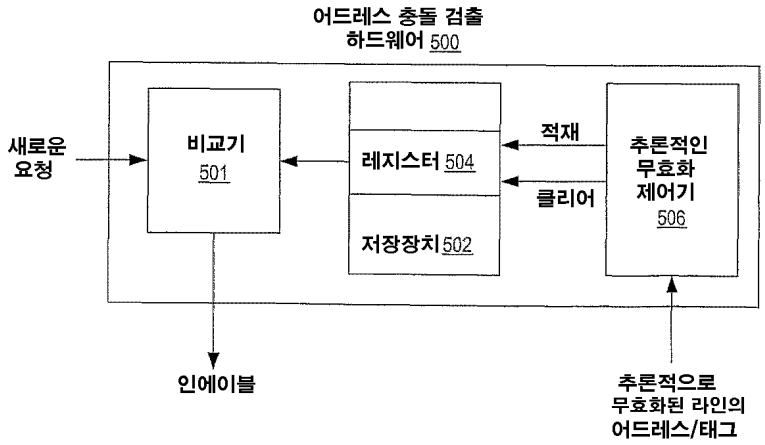
도면1B



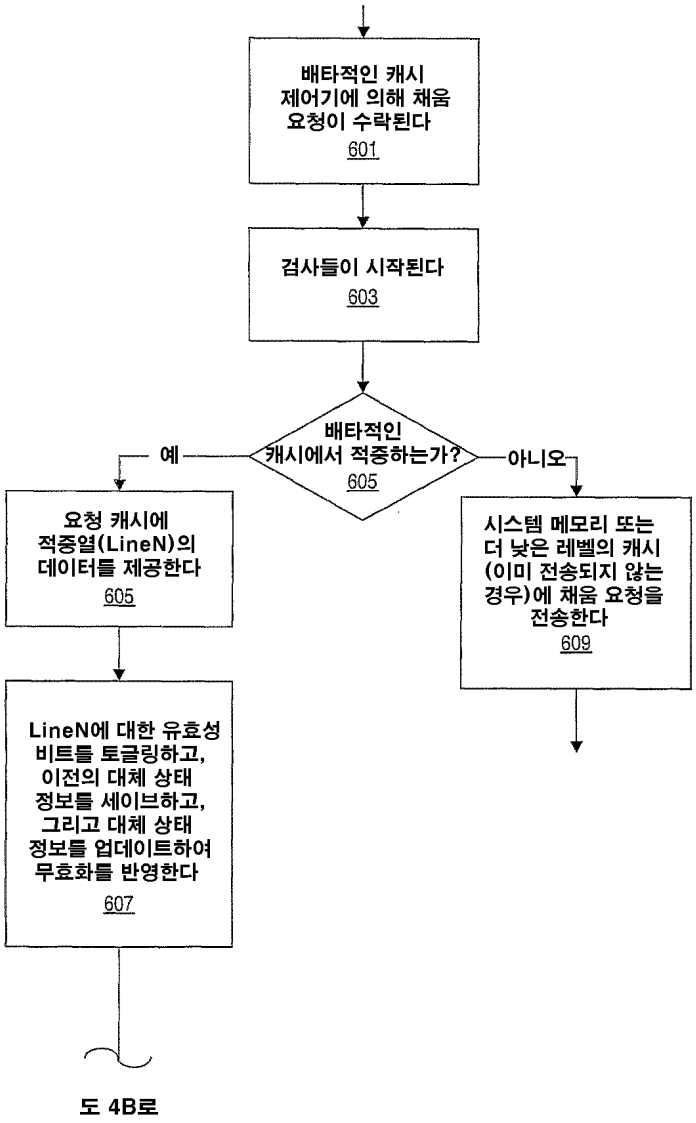
도면2



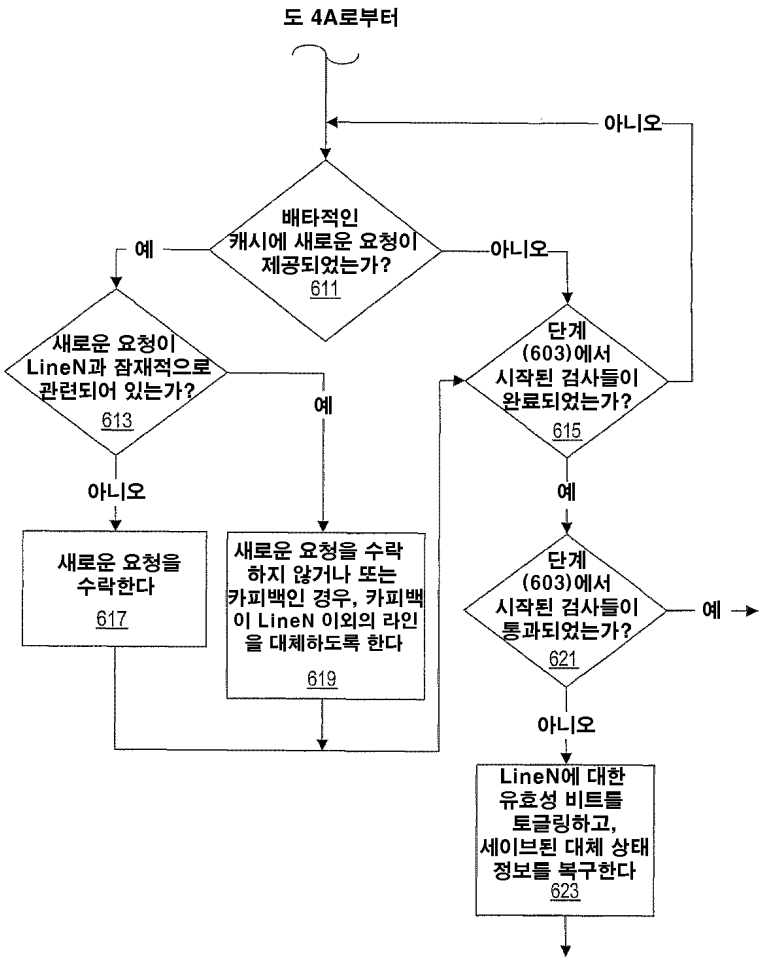
도면3


어드레스 충돌 검출
하드웨어 500
새로운
요청
비교기
501
레지스터 504
저장장치 502
적재
클리어
추론적인
무효화
제어기
506
인에이블
추론적으로
무효화된 라인의
어드레스/태그

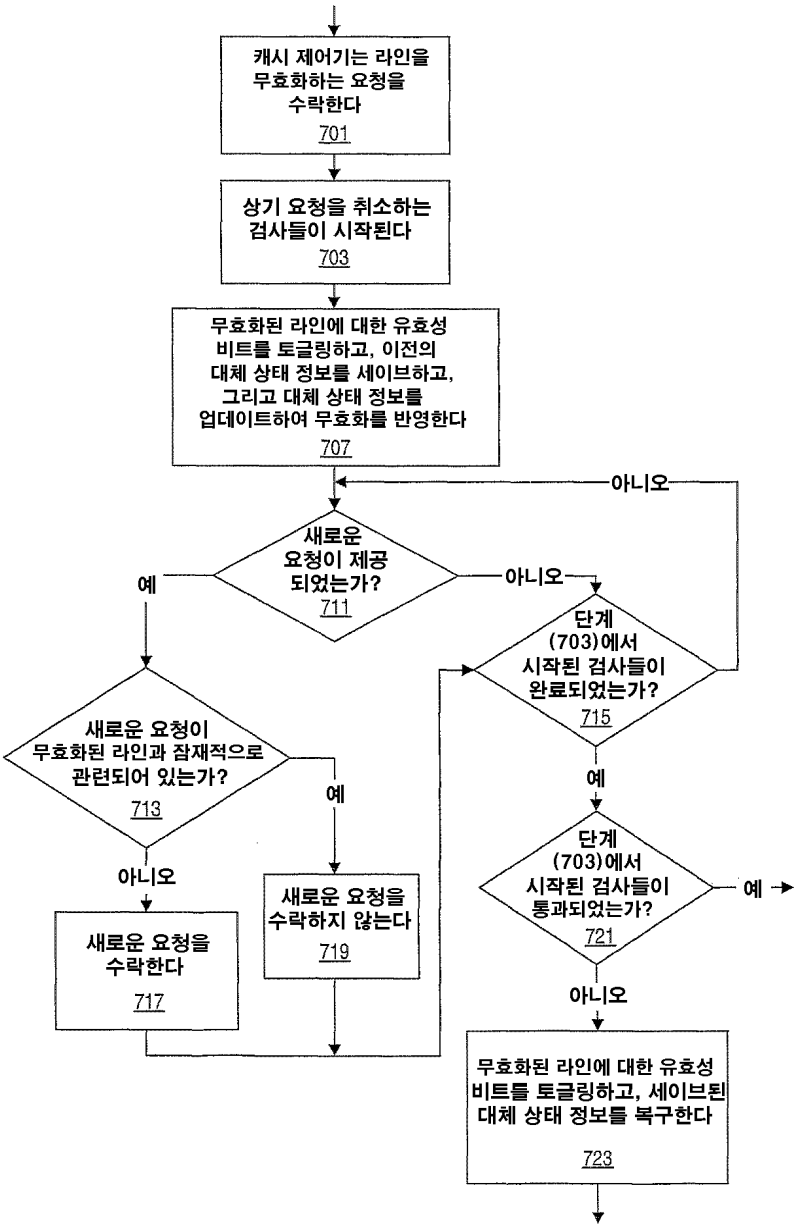
도면4A


배타적인 캐시
제어기에 의해 채움
요청이 수락된다
601
검사들이 시작된다
603
배타적인
캐시에서 적중하는가?
605
예
아니오
요청 캐시에
적중열(LineN)의
데이터를 제공한다
605
시스템 메모리 또는
더 낮은 레벨의 캐시
(이미 전송되지 않는
경우)에 채움 요청을
전송한다
609
LineN에 대한 유효성
비트를 토글링하고,
이전의 대체 상태
정보를 세이브하고,
그리고 대체 상태
정보를 업데이트하여
무효화를 반영한다
607
도 4B로

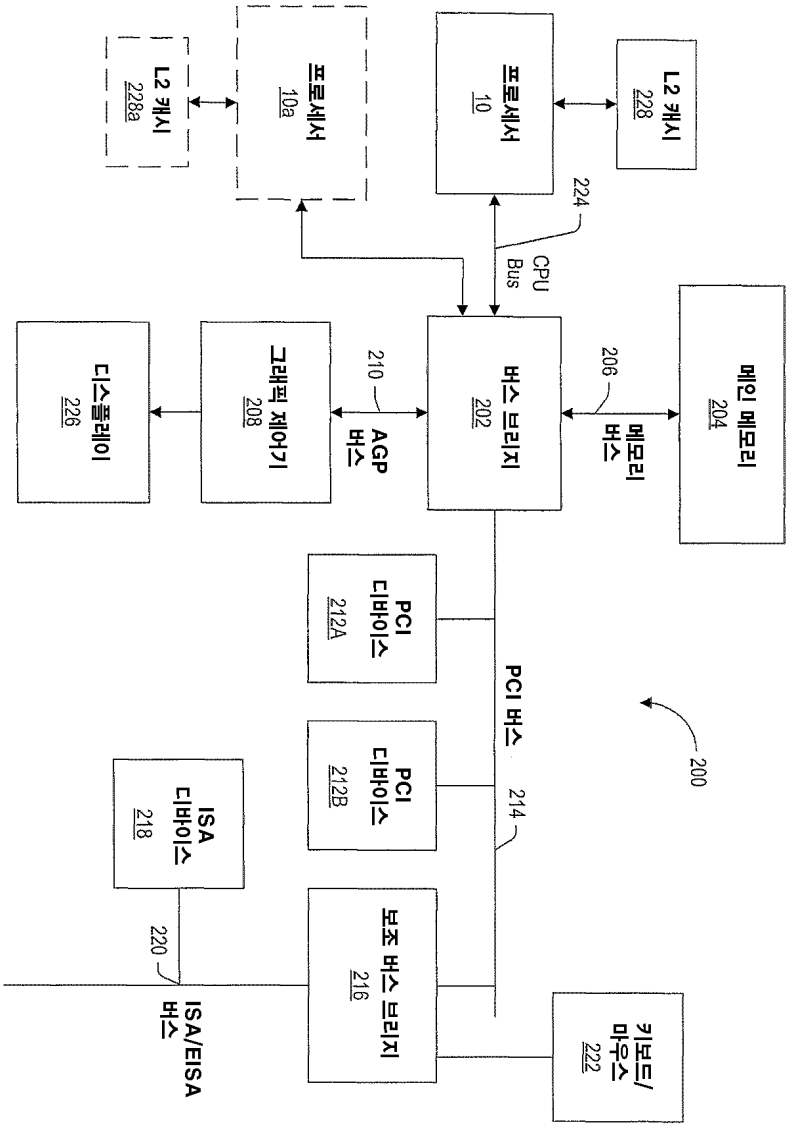
도면4B


도 4A로부터
배타적인 캐시에 새로운 요청이 제공되었는가?
611
예
아니오
아니오
새로운 요청이 LineN과 잠재적으로 관련되어 있는가?
613
예
아니오
새로운 요청을 수락한다
617
새로운 요청을 수락하지 않거나 또는 카피백인 경우, 카피백이 LineN 이외의 라인을 대체하도록 한다
619
단계 (603)에서 시작된 검사들이 완료되었는가?
615
예
단계 (603)에서 시작된 검사들이 통과되었는가?
621
예
아니오
LineN에 대한 유효성 비트를 토글링하고, 세이브된 대체 상태 정보를 복구한다
623

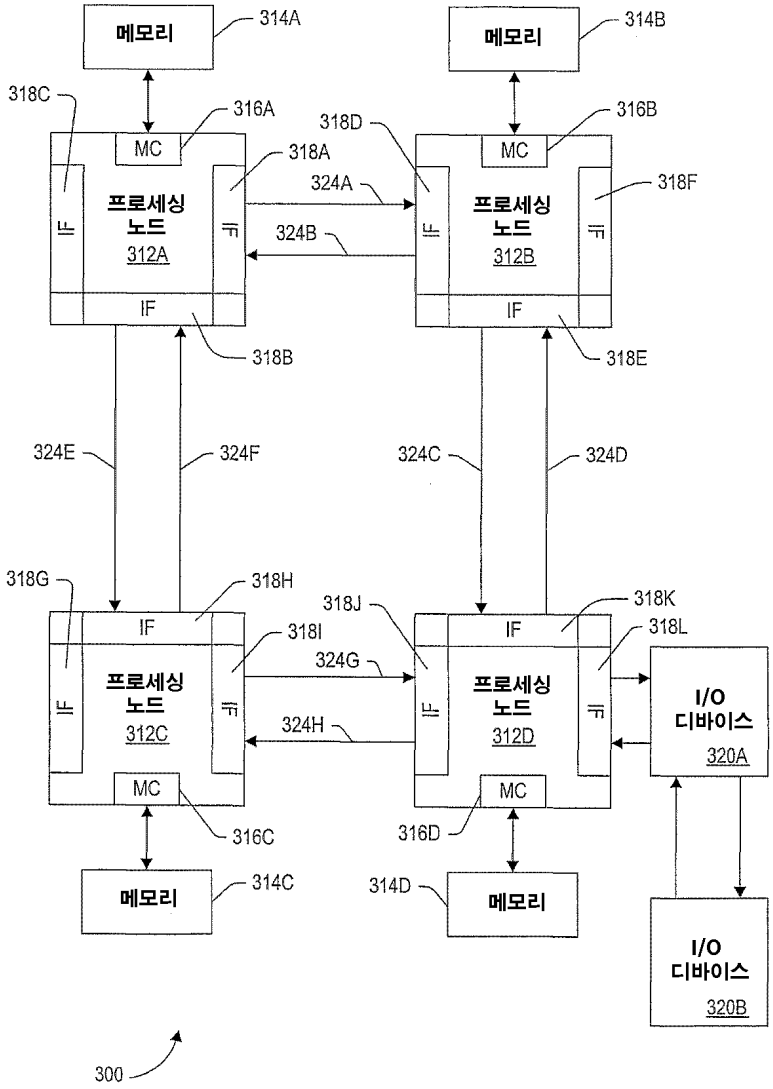
도면5


캐시 제어기는 라인을 무효화하는 요청을 수락한다
701
상기 요청을 취소하는 검사들이 시작된다
703
무효화된 라인에 대한 유효성 비트를 토글링하고, 이전의 대체 상태 정보를 세이브하고, 그리고 대체 상태 정보를 업데이트하여 무효화를 반영한다
707
새로운 요청이 제공 되었는가?
711
예
아니오
아니오
새로운 요청이 무효화된 라인과 잠재적으로 관련되어 있는가?
713
예
아니오
새로운 요청을 수락한다
717
새로운 요청을 수락하지 않는다
719
단계 (703)에서 시작된 검사들이 완료되었는가?
715
예
아니오
단계 (703)에서 시작된 검사들이 통과되었는가?
721
예
아니오
무효화된 라인에 대한 유효성 비트를 토글링하고, 세이브된 대체 상태 정보를 복구한다
723

도면6


프로세서
10
L2 캐시
228
프로세서
10a
L2 캐시
228a
224
CPU
Bus
버스 브리지
202
메인 메모리
204
206
메모리
버스
210
AGP
버스
그래픽 제어기
208
디스플레이
226
PCI
디바이스
212A
PCI
디바이스
212B
PCI 버스
214
200
ISA
디바이스
218
220
ISA/EISA
버스
보조 버스 브리지
216
키보드/
마우스
222

도면7


메모리
314A
메모리
314B
318C
316A
318D
316B
MC
MC
318A
318F
324A
프로세싱 노드 312A
프로세싱 노드 312B
324B
IF
IF
IF
IF
IF
IF
318B
318E
324E
324F
324C
324D
318G
318H
318J
318K
IF
IF
318I
318L
324G
프로세싱 노드 312C
프로세싱 노드 312D
I/O 디바이스 320A
324H
MC
MC
316C
316D
메모리
314C
314D
메모리
I/O 디바이스 320B
300