(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2021-0019584
(43) 공개일자 2021년02월22일

(51) 국제특허분류(Int. Cl.)
G06F 9/38 (2006.01)
(52) CPC특허분류
G06F 9/3806 (2013.01)
G06F 9/3814 (2013.01)
(21) 출원번호 10-2021-7004109
(22) 출원일자(국제) 2019년07월03일
심사청구일자 없음
(85) 번역문제출일자 2021년02월09일
(86) 국제출원번호 PCT/US2019/040497
(87) 국제공개번호 WO 2020/014066
국제공개일자 2020년01월16일
(30) 우선권주장
16/030,031 2018년07월09일 미국(US)

(71) 출원인
어드밴스드 마이크로 디바이시즈, 인코포레이티드
미국 캘리포니아 95054 산타 클라라 어거스틴 드라이브 2485
(72) 발명자
클루큐어 토마스
미국 캘리포니아 95054 산타 클라라 어거스틴 드라이브 2485
자비스 앤서니
미국 캘리포니아 95054 산타 클라라 어거스틴 드라이브 2485
(74) 대리인
박장원

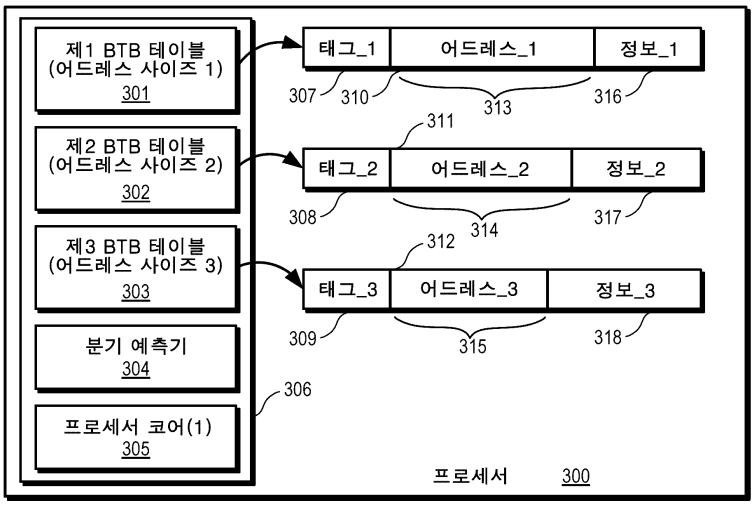
전체 청구항 수 : 총 20 항

(54) 발명의 명칭 **다중 테이블 분기 타겟 버퍼**

(57) 요 약

프로세서[120]는 분기 예측을 위한 두 개 이상의 분기 대상 버퍼(branch target buffer; BTB) 테이블들[301, 302]을 포함하며, 각 BTB 테이블은 상이한 타겟 사이즈 또는 너비의 엔트리들을 저장하거나 상이한 분기 유형의 엔트리들을 저장한다. 각 BTB 엔트리는 적어도 하나의 태그 및 타겟 어드레스를 포함한다. 소수의 타겟 어드레스 비트들만 필요로 하는 특정 분기 유형들의 경우, 각 BTB 테이블들이 더 좁아지므로 분기 명령어 유형에 따라 프로세서에서 더 많은 BTB 엔트리들이 각 BTB 테이블들로 분리될 수 있게 한다. 증가된 갯수의 이용 가능한 BTB 엔트리들이 프로세서에서 동일하거나 더 적은 공간에 저장되므로 명령어 처리 속도를 증가시킨다. 어떠한 타겟 어드레스도 저장하지 않고 이를 제공하기 위해 디코드 유닛에 의존하는 BTB 테이블들이 정의될 수 있다.

대 표 도 - 도3



(52) CPC특허분류
G06F 9/3867 (2013.01)

명 세 서

청구범위

청구항 1

프로세서[120]로서,

명령어 실행 파이프라인[100];

제1 갯수의 제1 분기 타겟 버퍼(branch target buffer; BTB) 테이블 엔트리들을 가지는 제1 BTB 테이블[301]로서, 각 제1 BTB 테이블 엔트리는 제1 태그[307] 및 제1 타겟 어드레스[313]를 포함하며, 각 제1 타겟 어드레스는 제1 너비를 가지는, 상기 제1 BTB 테이블[301];

제2 갯수의 제2 BTB 테이블 엔트리들을 가지는 제2 BTB 테이블[302]로서, 각 제2 BTB 테이블 엔트리는 제2 태그[308] 및 제2 타겟 어드레스[314]를 포함하며, 각 제2 타겟 어드레스는 상기 제1 너비와 상이한 제2 너비를 가지는, 상기 제2 BTB 테이블[302];

소정의 예측 어드레스에 대해 예측되는 타겟 어드레스를 제공하도록 구성된 분기 예측기[203]를 포함하며;

상기 프로세서는 분기 명령어의 분기 특성에 기초하여 상기 제1 BTB 테이블의 제1 BTB 테이블 엔트리 또는 상기 제2 BTB 테이블의 제2 BTB 테이블 엔트리 중 어느 하나에 상기 제1 타겟 어드레스를 포함하는 분기 기술어들(branch descriptors)을 저장하도록 구성된, 프로세서.

청구항 2

제1항에 있어서, 상기 분기 특성은 분기 타겟 어드레스 사이즈인, 프로세서.

청구항 3

제1항에 있어서, 상기 분기 특성은 분기 유형인, 프로세서.

청구항 4

제3항에 있어서, 상기 제1 BTB 테이블 엔트리들은 점프 분기 명령어, 호출 분기 명령어, 리턴 분기 명령어 및 조건 분기 명령어 중 적어도 하나에 대해 구성된, 프로세서.

청구항 5

제1항에 있어서, 상기 분기 특성은 상기 BTB 엔트리에 저장된 분기들의 갯수인, 프로세서.

청구항 6

제1항에 있어서, 상기 분기 특성은 상기 BTB 엔트리의 스레드 식별자인, 프로세서.

청구항 7

제1항에 있어서,

제1 메모리 레벨 캐시 및 제2 메모리 레벨 캐시를 더 포함하며;

상기 제1 BTB 테이블 및 상기 제2 BTB 테이블은 상기 프로세서의 동일한 메모리 레벨 캐시에 포함되는, 프로세서.

청구항 8

제1항에 있어서,

상기 제1 BTB 테이블의 상기 제1 BTB 테이블 엔트리들은 예측 조회에 대해 N-웨이 연관(N-way associative)이고;

상기 제2 BTB 테이블의 상기 제2 BTB 테이블 엔트리들은 예측 조회에 대해 M-웨이 연관이며, 여기서 M 및 N은 1 이상이고 M과 N은 상이한, 프로세서.

청구항 9

제1항에 있어서, 상기 제1 BTB 테이블 엔트리들의 제1 갯수는 상기 제2 BTB 테이블 엔트리들의 제2 갯수와 상이한, 프로세서.

청구항 10

제1항에 있어서,

상기 BTB 테이블 엔트리들 각각은 세트 내에서 최근에 가장 많이 사용된(most recently used; MRU) 상태 및 최근에 가장 적게 사용된(least recently used; LRU) 상태 중 하나로서 상기 BTB 엔트리 각각을 마킹하는 최근 사용된 상태 비트를 포함하고;

상기 프로세서는 상기 최근 사용된 상태 비트의 상태에 기초하여 교체 정책에 따라 세트 내에서 BTB 엔트리들을 축출하도록 구성된, 프로세서.

청구항 11

방법으로서,

분기 명령어에 대한 예측 실패로 인해 야기되는 리다이렉트들에 기초하여 분기 타겟 버퍼(BTB) 엔트리의 분기 유형을 결정하는 단계; 및

결정된 상기 BTB 엔트리의 분기 유형에 기초하여, 상기 BTB 엔트리를 다음:

프로세서의 제1 분기 타겟 버퍼(BTB) 테이블[301]의 제1 엔트리로서, BTB 엔트리의 제1 분기 유형에 대응하는 제1 타겟 어드레스 너비를 가지는, 상기 제1 엔트리; 및

상기 프로세서의 제2 BTB 테이블[302]의 제2 엔트리로서, 제1 분기 유형과 상이한 제2 분기 유형에 대응하는 상기 제1 타겟 어드레스 너비와 상이한 제2 타겟 어드레스 너비를 가지는, 상기 제2 엔트리

중 하나에 저장하는 단계를 포함하는, 방법.

청구항 12

제11항에 있어서, 상기 분기 유형은 분기 타겟 어드레스 사이즈인, 방법.

청구항 13

제11항에 있어서, 상기 BTB 엔트리의 상기 저장은 프로세서의 예측 트레이너 유닛에 의해 수행되는, 방법.

청구항 14

제11항에 있어서,

세트 내에서 최근에 가장 많이 사용된(MRU) 상태 및 최근에 가장 적게 사용된(LRU) 상태 중 하나로서 상기 BTB 엔트리를 마킹하는 최근 사용된 상태 비트에 기초하여 상기 제1 BTB 테이블 또는 상기 제2 BTB 테이블의 BTB 엔트리를 식별하는 단계; 및

상기 엔트리를 내부에 저장하기 전에 상기 제1 BTB 테이블 또는 상기 제2 BTB 테이블의 식별된 상기 BTB 엔트리를 축출하는 단계를 더 포함하는, 방법.

청구항 15

제11항에 있어서,

상기 제1 BTB 테이블 및 상기 제2 BTB 테이블은 상기 프로세서의 동일한 메모리 레벨 캐시에 포함되는, 방법.

청구항 16

제11항에 있어서,

상기 BTB 엔트리를 저장하기 전에, 상기 분기 명령어의 명령어 태그에 기초하여 상기 제1 BTB 테이블 및 상기 제2 BTB 테이블 중 적어도 하나를 탐색하는 단계; 및

상기 BTB 엔트리를 저장하기 전에, 상기 제1 BTB 테이블 및 상기 제2 BTB 테이블 중 상기 적어도 하나에서 BTB 엔트리가 찾아지지 않음을 식별하는 단계를 더 포함하는, 방법.

청구항 17

제11항에 있어서,

상기 BTB 엔트리의 타겟 어드레스의 사이즈가 상기 제1 BTB 테이블 또는 상기 제2 BTB 테이블 각각의 BTB 엔트리들에 대한 어드레스 사이즈를 초과할 때 상기 BTB 엔트리의 상기 타겟 어드레스의 오버플로우 비트들(overflow bits)을 오버플로우 BTB 테이블에 저장하는 단계를 더 포함하는, 방법.

청구항 18

방법으로서,

프로세서의 BTB의 복수의 분기 타겟 버퍼(BTB) 테이블들을 조회하여 현재 예측되는 블록의 종료 어드레스 및 다음 예측되는 블록의 시작 어드레스를 예측하는 단계로서, 상기 BTB는 현재 예측 어드레스에서의, 제1 BTB 테이블 및 제2 BTB 테이블을 포함하는, 상기 예측하는 단계; 및

예측된 상기 종료 어드레스 및 예측된 상기 시작 어드레스 중 적어도 하나에 기초하여 상기 프로세서에서 실행하기 위한 명령어의 일부로서 예측을 제공하는 단계를 포함하는, 방법.

청구항 19

제17항에 있어서, 상기 제1 BTB 테이블의 BTB 엔트리들은 상기 제2 BTB 테이블의 BTB 엔트리들의 타겟 어드레스 필드의 제2 너비보다 작은 제1 너비의 타겟 어드레스 필드를 가지는, 방법.

청구항 20

제17항에 있어서, 상기 현재 예측 어드레스는 프로그램 카운터를 기준으로 하는, 방법.

발명의 설명

기 술 분 야

배 경 기 술

[0001] 프로세서의 명령어 파이프라인은 명령어 스트림의 여러 명령어들이 병렬로 실행될 수 있는 다수의 파이프라인 스테이지들에서 명령어들을 처리하여 명령어 실행 처리량을 개선한다. 이러한 파이프라인들은 보통 명령어들을 페칭, 디코딩, 매핑 및 실행한 다음, 레지스터와 같은 다른 유닛에 결과들을 기록하기 위한 별도의 유닛들을 포함한다. 파이프라인의 명령어 페치 유닛은 프로세서 파이프라인의 다음 스테이지에 명령어들의 스트림을 제공한다. 명령어 페치 유닛들은 일반적으로 나머지 파이프라인에 명령어들이 지속적으로 공급되게 유지하기 위해 명령어 캐시를 사용한다.

[0002] 명령어 스트림에서의 분기 명령어는 명령어 페칭 스테이지에서 다음 명령어를 페칭하기 전에 파이프라인에서의 실행 스테이지에서 분기가 해결될 때까지 프로세서가 대기하는 경우 파이프라인 지연을 초래할 수 있다. 분기 예측기는 조건 분기가 이행될지(taken) 또는 이행되지 않을지(not taken)를 예측하려고 시도할 수 있다. 일부 구현 예들에서, 분기 예측기는 분기 명령어 자체를 디코딩하고 실행함으로써 분기 명령어가 컴퓨팅되기 전에 분기 타겟 예측을 사용하여 이행되는 조건 또는 무조건 분기의 타겟을 예측한다. 분기 타겟은 계산된 어드레스로부터의 오프셋 또는 레지스터를 통한 간접 참조에 기초할 수 있다.

[0003] 분기 타겟 버퍼(branch target buffer; BTB)는 통상적으로 예측된 분기 타겟들을 포함하는 분기 정보를 저장하는 프로세서의 단일의 작은 메모리 캐시이다. 예측에는 명령어 어드레스를 BTB에 저장되었다가 이전에 실행된

명령어 어드레스들과 비교하는 것이 수반된다. 예측이 성공하면 프로세서가 타겟 어드레스를 획득하기 위한 단계들의 실행을 건너 뛸 수 있기 때문에 예측은 일반적으로 처리 시간을 절감한다. 프로세서는 BTB에서 다음 실행 단계를 위한 어드레스를 조회하여 시간을 절감한다. 따라서, BTB가 타겟 어드레스 적중을 내는 빈도는 프로세서가 명령들을 실행할 수 있는 속도에 직접적인 영향을 미친다. 보통, 실행 속도는 BTB가 저장할 수 있는 엔트리들의 갯수와 직접적인 관련이 있다.

도면의 간단한 설명

[0004] 본 개시는 첨부 도면들을 참조하여 이해가 더 잘 되고, 이의 많은 특징들 및 이점들이 해당 기술분야의 통상의 기술자들에게 분명해질 수 있다. 상이한 도면들에서의 동일한 참조 부호들의 사용은 유사하거나 동일한 항목들을 나타낸다.

도 1은 일부 실시 예들에 따른 명령어 파이프라인 아키텍처의 블록도이다.

도 2는 일부 실시 예들에 따른 프로세싱 시스템의 블록도이다.

도 3은 일부 실시 예들에 따른 타겟 크기에 기초하여 다수의 분기 타겟 버퍼(BTB) 테이블들을 갖는 프로세서의 블록도이다.

도 4는 일부 실시 예들에 따른 분기 유형들에 기초하여 다수의 BTB 테이블들을 갖는 프로세서의 블록도이다.

도 5는 일부 실시 예들에 따른 상이한 메모리 레벨들에 다수의 BTB 테이블들을 갖는 프로세서의 블록도이다.

도 6은 일부 실시 예들에 따른 BTB 테이블에 대한 BTB 엔트리의 블록도이다.

도 7은 일부 실시 예들에 따른 BTB 테이블에 대한 BTB 엔트리의 블록도이다.

도 8은 일부 실시 예들에 따른 BTB 테이블에 대한 BTB 엔트리의 블록도이다.

도 9는 일부 실시 예들에 따라 분기 타겟 어드레스를 타겟 어드레스 사이즈에 의해 복수의 BTB 테이블들 중 하나로 저장하기 위한 방법을 도시하는 흐름도이다.

도 10은 일부 실시 예들에 따라 분기 타겟 어드레스를 분기 유형에 의해 복수의 BTB 테이블들 중 하나로 저장하기 위한 방법을 나타내는 흐름도이다.

발명을 실시하기 위한 구체적인 내용

[0005] 프로세서에 분기 타겟 버퍼(BTB) 또는 BTB 테이블을 사용하여 분기 정보를 저장하면 프로세서 속도가 크게 개선된다. 통상적으로, BTB의 엔트리들의 갯수를 늘리는 유일한 방법은 버퍼 사이즈를 늘리는 것이었다. 그러나, BTB와 같은 일부 구성요소들에 대한 로컬 스토리지를 늘리면 프로세서의 속도 및 물리적 공간 측면에서 다른 구성요소들을 희생시키기 때문에 BTB 용량 제한을 포함하여 프로세서 내의 스토리지 요소들의 갯수에는 한계가 있다. 또한, 프로세서의 아키텍처를 설계할 때 견줘 보아야 할 절충사항들 및 고려해야 할 요소들이 있다. 예를 들어, 실제로 BTB에 대한 일부 변경은 프로세서가 작동될 때 프로세서 속도 저하를 일으켰다. 다른 상황들에서는, BTB가 클수록 프로세서 작동 동안 전체 전력 소비를 늘리며, 이는 바람직하지 않다.

[0006] 통상적인 BTB의 전형적인 사이즈는 1024 엔트리이며, 각 엔트리는 예를 들어, 타겟 어드레스에 대한 20 비트를 가진다. 그러나, 모든 BTB 엔트리가 내부에 어드레스를 저장하는 데 동일한 수의 비트를 필요로 하는 것은 아니다. 예를 들어, 분기 예측기는 특정 어드레스들에 대해서는 이용 가능한 20 비트 중 5-10 비트만 필요할 수 있지만 다른 어드레스들에 대해서는 20 비트 전체가 필요할 수 있는데 이는 타겟 어드레스의 최상위 비트들의 일부가 예측 어드레스로부터 변경되지 않은 상태로 분기 정보에 인코딩될 수 있기 때문이다. BTB의 용량 및 유용성을 증가시키고 프로세서 내 BTB의 풋프린트를 거의 동일하게 유지하기 위해, 적어도 일 실시 예에서 BTB는 둘 이상의 버퍼들 또는 테이블들로 분할된다. 특정 실시 예들에 따르면, 제1 버퍼는 6, 8 또는 다른 수의 비트를 갖는 어드레스들과 같은 짧은 메모리 어드레스들에 대해 사이즈가 조정되고 지정된다. 제2 버퍼는 14, 16, 18 또는 20 비트를 갖는 어드레스들과 같이 보다 긴 메모리 어드레스들에 대해 사이즈가 조정되고 지정된다. 일부 구현 예들은 이를테면 6, 8, 10, 12, 14, 16, 18 또는 20 비트를 갖는 다양한 메모리 어드레스 길이들에 대해 사이즈가 부여되고 지정되는 제3 버퍼를 이용한다. 또한 각 BTB 엔트리는 태그 필드, 및 일부 실시 예들에서 BTB 엔트리의 하나 이상의 별개의 필드에 배치되는 다른 정보를 포함한다.

[0007] 다른 실시 예들에 따르면, 프로세서 설계자들 및 구축자들은 하나 이상의 유형의 명령어 분기에 기초하여 다수

의 BTB 테이블들을 생성 및 지정하고 이에 의해 각각의 유형들의 분기 타겟들에 대한 BTB 테이블들을 생성한다. 각 BTB 엔트리는 BTB 엔트리에 의해 추적되는 분기들을 발견하는 비용과 관련된 값을 가진다. 무조건 분기들은 프로세서 디코더에 의해 프로세서 파이프라인의 초기에 발견될 수 있지만, 조건 분기들은 실행 시간까지 발견되지 않을 수 있다. 일부 실시 예들의 프로세서들은 한 사이클에, 두 번 이상의 분기 예측들, 그리고 이에 의해 두 개의 분기 타겟 어드레스들을 내놓는다. 이러한 다수의 분기들은 단일의 BTB 엔트리에 저장된다. BTB 엔트리의 다양한 값들을 설명하기 위해, 제1 BTB는 예를 들어, 무조건 분기들에 대해 생성 및 지정되고, 제2 BTB는 예를 들어, 단일의 조건 분기에 대해 생성 및 지정되며, 제3 BTB는 예를 들어, 다수의 분기들을 기록하는 엔트리들에 대해 생성된다. 각각의 제1, 제2 및 제3 BTB 테이블들은 서로 다른 타겟 어드레스 사이즈 그리고 이에 따라 서로 다른 전체 BTB 테이블 너비를 가질 수 있으므로, 단일의 BTB가 특정 고정 너비 및 특정 고정 길이(엔트리들의 갯수)를 가졌던 이전 설계들과 비교하여 프로세서에서 BTB 테이블들의 전체 풋프린트를 줄이는 메커니즘을 제공한다. BTB 엔트리들을 두 개 이상의 BTB 테이블들로 나누거나 배치하는 것의 이러한 이점을 통해 프로세서 설계자는 BTB 테이블들 중 하나 이상의 길이를 서로 또는 단일의 기존 BTB의 기존 길이에 비해 늘리거나 줄일 수 있다. 3-BTB 시스템의 경우, 제1, 제2 및 제3 BTB 테이블들은 각각의 제1 BTB, 제2 BTB 및 제3 BTB의 엔트리에 맞지 않는 큰 타겟 어드레스들에 다수의 추가 어드레스 비트들을 제공하기 위해 각 BTB 엔트리가 가리킬 수 있는 별도의 타겟 테이블에 의해 보완될 수 있다. 운용시 일부 구현 예들에 따르면, 프로세서에서의 BTB 조회는 각 BTB 조회마다 모든 BTB 버퍼들에 걸쳐 수행되어, 단일의 기존 BTB 버퍼를 갖는 것과 유사한 성능을 제공한다.

[0008] 특정 구현 예들에서, BTB 테이블들은 단일의 메모리 유형 내에 포함되거나, 이에 반하여 다른 구현 예들에서 BTB 테이블들은 프로세서 코어와 관련하여 다수의 메모리들, 다수의 메모리 유형들 또는 다중 레벨 메모리 계층의 다수의 메모리 레벨들에 분산된다. 설명의 단순화를 위해, 여기서는 레벨 1 BTB, 레벨 2 BTB 등을 참조하며, 각각은 프로세서 코어 또는 프로세서에 대한 메모리의 레벨 또는 배치에 대응한다. 각 BTB 레벨은 단일의 BTB 테이블 또는 다수의 BTB 테이블들을 포함할 수 있다. 다른 구현 예들에서, 하나 이상의 BTB 테이블들은 단일의 메모리 캐시 또는 단일의 프로세서 코어에 대한 메모리 레벨에 구축된다. 또 다른 구현 예들에서, BTB 테이블들은 프로세서 코어들 또는 프로세서 작업들 간에 공유된다. 특정 실시 예에서, 복수의 BTB 레벨들 중 하나의 BTB 레벨은 각각의 프로세서 코어들이 그 자체의 하나 이상의 BTB 테이블을 포함하는 복수의 프로세서 코어들에 의해 공유된다. 특정 코어와 연관된 BTB 테이블들이 먼저 사용될 수 있고, BTB 엔트리들은 필요에 따라 공유 BTB 테이블로 축출될 수 있다. 프로세서가 프로세서 코어들 간에 공유되는 하나 이상의 BTB 테이블을 갖는 혜택은 탐색 속도, 전체 프로세서 속도 및 전체 프로세서 작업의 감소로 상쇄될 수 있다. 따라서, 빠른 구현을 위해 다수의 BTB 테이블들이 다중 코어 프로세서의 각 개별 프로세서 코어에 제공될 수 있다.

[0009] 운용시, 최근에 가장 적게 사용된(least recently used; LRU) 엔트리들과 같이 보다 낮은 값의 BTB 엔트리들은 먼저 제1 BTB 또는 제1 BTB 레벨로부터 다른 BTB 또는 BTB 레벨, 이를테면 메모리의 동일한 또는 다른 레벨의 BTB로 축출된다. 제2 BTB로부터 제3 BTB로의 축출도 LRU 기반으로 이루어진다. BTB 테이블들에는 다양한 유형들의 연관이 사용될 수 있다. 예로서, 제1 BTB 테이블은 4-웨이 세트 연관이다. 또 다른 BTB 테이블은 각 웨이가 웨이당 512, 1024, 2048 또는 몇몇 다른 수의 엔트리들을 저장하는 8-웨이 세트 연관 프로세서 캐시의 형태를 취한다. 일반적으로, 제1 BTB 테이블은 N-웨이 세트 연관일 수 있고, 제2 BTB 테이블은 M-웨이 세트 연관일 수 있으며 N과 M이 서로 동일하거나 상이한 정수들이다. 다른 실시 예들에서는, BTB 테이블들 중 하나 이상이 완전 연관된다. 연관의 레벨은 특정 프로세서에 대한 원하는 전력 소비 레벨, 프로세서가 작동하는 시스템, 및 특정 프로세서에 대한 예상 처리 부하 유형 및 양에 따라 선택된다.

[0010] 일부 구현 예들에서, BTB 테이블은 하위 레벨 BTB 테이블들로부터의 조건 분기들에 대해 최근에 축출된 엔트리들로 채워진 제한된 수의 엔트리들을 가지는 희생 버퍼 역할을 한다. 소정의 영역에 대해 그러한 희생 버퍼의 엔트리들의 갯수를 최대화하기 위해, 분기 예측기는 BTB 테이블에 어떠한 타겟 어드레스도 기록하지 않는다. 분기 예측기가 희생 버퍼에서 분기를 찾고 조건 분기 예측기에 의해 결정되는 바에 따라 분기가 이행된다고 예측하면, 분기 예측기는 분기 위치에 대한 정보를 디코드 유닛으로 전송하며 이는 디코드 작업이 완료되고 예측되는 위치가 실제로 조건 분기임을 확인하면 이용 가능한 타겟 어드레스로 리디렉트를 트리거한다.

[0011] 도 1은 일부 실시 예들에 따른 비순차적 명령어 실행을 구현하는 프로세서(120)의 명령어 파이프라인 아키텍처(100)의 블록도이다. 프로세서(120)의 몇 가지 요소들만이 도시되어 있다. 명령어 캐시(101)는 명령어 페치 유닛(103)에 의해 액세스된다. 데이터 캐시(102)는 로드/저장 유닛(110)에 의해 액세스된다. 명령어 캐시(101)의 명령어들은 데이터 캐시(102)로부터의 데이터를 포함하는 데이터에 작용한다. 명령어 페치 유닛(103)은 하나 이상의 분기 타겟 버퍼(BTB) 테이블(통상적으로 BT 버퍼 및 BTB라고도 함)(105)에 저장되거나 제공되는 분기 타겟

어드레스들을 생성하는 분기 예측기(104)를 포함한다. 일부 실시 예들에 따르면, 분기 타겟 어드레스들은 프로그램 카운터(113)를 기준으로 한다. BTB 테이블들(105)은 도 1에서 분기 예측기(104) 내부에 도시되어 있지만, BTB 테이블들(105)은 명령어 페치 유닛(103) 또는 분기 예측기(104)의 특정 요소들에 근접하여 프로세서(120)에 위치될 수 있거나 위치되지 않을 수 있다. 디코더(106)는 명령어 캐시(101)로부터의 명령어들을 포함하여 명령어들을 프로세서 제어 신호들로 변환한다.

[0012] 리오더 버퍼(reorder buffer)(107)는 리저베이션 스테이션들(reservation stations)(109)과 같은 프로세서의 다른 구성요소들에 의해 액세스되는 레지스터들(108)에 명령어들을 각각의 원래의 페치 순서로 저장한다. 리저베이션 스테이션들(109)은 레지스터들(108)과 같은 레지스터들의 이름을 변경하고 동적 명령어 스케줄링을 가능하게 한다. 리저베이션 스테이션들(109)은 데이터가 레지스터에 저장되고 다시 판독될 때까지 기다리는 것이 아니라, 데이터가 페치되거나 계산되자마자 프로세서가 데이터를 페칭하고 재사용할 수 있게 한다. 분기 예측기(104)에 의해 BTB 테이블들(105)에 저장된 예측 어드레스가 정확하지 않거나 명령어 스트림에서 복구 불가능한 예외가 발생하면, 리오더 버퍼(107)에서는 모든 명령어들이 클리어되고 리저베이션 스테이션들(109)은 다시 초기화된다. 리오더 버퍼(107)는 분기 타겟 어드레스 예측 실패의 롤백 제어를 위한 메커니즘을 제공한다. 리저베이션 스테이션들(109)은 로드/저장 유닛(110) 및 하나 이상의 기능 유닛(111), 이를테면 산술 논리 유닛(arithmetic logic unit, ALU), 부동 소수점 유닛(floating point unit, FPU) 및 정수 유닛(integer unit, IU)에 정보를 제공한다. 이와 함께, 명령어 페치 유닛(103), 디코더(106), 리오더 버퍼(107), 리저베이션 스테이션들(109),로드/저장 유닛(110) 및 관련 레지스터들은 명령어 실행 파이프라인의 일 실시 예이다.

[0013] 도 2는 일부 실시 예들에 따른 프로세싱 시스템(200)의 블록도이다. 프로세싱 시스템(200)은 도 1의 명령어 파이프라인 아키텍처(100) 및 BTB 테이블들(105)의 일 부분의 일례이다. 프로세싱 시스템(200)은 분기 검출기(202), 조건 분기 예측기(203), 리턴 어드레스 예측기(204) 및 분기 예측 트레이너(205)를 갖는 분기 예측기(201)를 포함한다. 프로세싱 시스템(200)은 해당 기술분야의 통상의 기술자들에 의해 이해되는 바와 같이 산술 유닛, 스케줄러, 테이블 워커 등과 같은 다른 도시되지 않은 요소들을 포함한다.

[0014] 각 현재 어드레스에 대해, 분기 예측기(201)는 현재 어드레스에서 시작하여 페칭될 바이트 블록의 종료 어드레스, 예측된 블록의 분기 유형 및 다음 예측의 시작 어드레스를 포함하는 예측 윈도우를 제공한다. 어느 분기들이 예측 윈도우의 일부인지 결정하기 위해, 분기 검출기(202)는 모든 웨이들에 걸쳐, 이를테면 BTB 테이블 1(206) 내지 BTB 테이블 N(207)으로 나타내어지는 복수의 BTB 테이블들 중 하나 이상에 걸쳐 모든 BTB 테이블들을 조회한다. 소정의 BTB 엔트리에 대한 태그 일치는 엔트리에 기록된 분기 또는 분기들이 예측 윈도우에 존재함을 나타낸다. 각 BTB 엔트리는 분기들의 위치 및 유형을 포함한다. BTB는 예측 실패한 분기들에 대해 리디렉트가 발생할 때 분기 예측 트레이너(205)에 의해 채워진다.

[0015] 도 3은 일부 실시 예들에 따른 다수의 BTB 테이블들을 갖는 프로세서의 블록도이다. 프로세서(300)는 추가 구조들 및 기능들을 포함하는 도 1의 프로세서(120) 및 BTB 테이블들(105)의 구체적인 예이다. 프로세서(300)는 제1 지정된 영역(306)에 복수의 프로세서 코어들 중 제1 프로세서 코어(305)를 포함한다. 제1 프로세서 코어(305)는 분기 예측기(304)에 의해 사용되는 제1 BTB 테이블(301), 제2 BTB 테이블(302) 및 제3 BTB 테이블(303)을 포함하는 BTB 테이블들과 상호 운용된다. 일부 실시 예들에 따르면, 분기 예측기(304)는 각 프로세스 사이클마다, 하나 이상의 분기 명령어로부터 하나 또는 두 번의 예측을 제공한다.

[0016] 각 BTB 테이블(301, 302, 303)은 엔트리가 속한 예측 어드레스를 식별하는 데 각각 사용되는 태그 필드(307, 308, 309), 하나 또는 복수의 분기의 타겟을 저장하는 데 사용되는 타겟 어드레스 필드(310, 311, 312), 및 예측 윈도우에서 분기들의 위치 및 분기 유형을 저장하는 데 사용되는 정보 필드(316, 317, 318)를 포함한다. 도 2의 분기 예측 트레이너(205)와 같은 분기 트레이너는 각 BTB 테이블들을 채우고, 트레이닝되는 각 BTB 엔트리에 대해 BTB 엔트리가 제1 BTB 테이블(301)에 기록되는지, 제2 BTB 테이블(302)에 기록되는지 또는 제3 BTB 테이블(303)에 기록되는지를 결정한다.

[0017] BTB 테이블들(301-303) 각각은 상이한 사이즈들 또는 너비들(313, 314, 315)의 타겟 어드레스 필드들(310, 311, 312)을 가진다(달리 지시되지 않는 한, 여기서 "사이즈" 및 "너비"는 호환하여 사용됨). 예를 들어, 제2 BTB 테이블(302)은 제2 어드레스 사이즈(314)의 어드레스들(311)을 가진다. 일부 실시 예들에 따르면, 제2 어드레스 사이즈(314)는 제1 어드레스 사이즈(313)보다 작다. 제2 어드레스 사이즈(314)는 제3 BTB 테이블(303)의 타겟 어드레스 필드(312)의 제3 어드레스 사이즈(315)보다 크다. 특정 실시 예들에서, 제3 BTB 테이블(303)의 제3 BTB 엔트리 세트에는 예를 들어, 제1 BTB 테이블(301)의 제1 BTB 엔트리 세트 내 BTB 엔트리들의 수에 비해 더 많은 수의 BTB 엔트리들이 있다. 이는 다른 필드들(309, 318)의 사이즈들을 일정하게 유지할 때 타겟 어드레

스 필드(312)에 대한 비트들의 수 또는 너비(315)를 감소시키면 제3 BTB 테이블(303)에 더 많은 수의 엔트리들을 허용하기 때문에 가능하다. 즉, 각 BTB 테이블(301, 302, 303)에 대한 동일한 양의 프로세서 풋프린트에 대해, 감소된 너비(315)는 제3 BTB 테이블(303)이 고정 사이즈의 지정된 영역(306) 대해 더 많은 수의 엔트리들을 가질 수 있게 한다. 다른 실시 예들에서는, 모든 타겟 어드레스들에 대해 일정한 너비의 타겟 어드레스 필드를 갖는 단일의 BTB 테이블을 갖는 이전 실시 예들에 비해 더 작은 지정된 영역(306)이 가능하다. 이에 따라, 본 예에서, 제3 BTB 테이블(303)이 프로세서(300) 및 지정된 영역(306)에서 다른 BTB 테이블들(301, 302)보다 더 작은 풋프린트를 차지할 수 있더라도, 제3 BTB 테이블(303)은 제2 BTB 테이블(302) 및 제1 BTB 테이블(301)보다 사이즈(엔트리들)가 더 큰 것으로 여겨진다.

[0018] 이를테면 제3 BTB 테이블(303)에서의 엔트리의 제3 어드레스(312)에 또는 제2 BTB 테이블(302)에서의 엔트리의 제2 어드레스(311)에 어드레스 비트들을 유지하는 데 추가 비트들이 필요하면, 추가 어드레스 테이블(도 3에 도시되지 않음)이 제1, 제2 및 제3 BTB 테이블들(301, 302, 303) 간에 공유된다. 추가 어드레스 테이블이 사용되면, 어드레스 필드의 비트들에 저장된 어드레스 값들 대신에, 복수의 어드레스 비트들이 추가 어드레스 테이블에서의 엔트리로 인덱싱된다. 즉, 추가 어드레스 테이블의 엔트리 포인터가 분기 타겟 어드레스 필드들(310, 311, 312) 중 대응하는 어드레스 필드에 대해 지정된 BTB 엔트리의 적어도 일 부분에 저장된다.

[0019] 도 4는 일부 실시 예들에 따른 분기 유형들에 기초하여 BTB 테이블들을 갖는 프로세서(400)의 블록도이다. 분기 유형들에 기초한 BTB 테이블들의 편성은 테이블들을 상이한 값의 BTB 엔트리들에 전임시키고 테이블들의 사이즈를 적절하게 조정하여 더 높은 값의 엔트리들에 더 많은 공간을 전임시킴으로써 개선된 성능을 제공한다. 프로세서(400)는 추가 구조들 및 기능들을 포함하는 도 1의 프로세서(120) 및 BTB 테이블들(105)의 구체적인 예이다. 프로세서(400)는 제1 지정된 영역(406)에 복수의 프로세서 코어들의 제1 프로세서 코어(405)를 포함한다. 제1 프로세서 코어(405)는 분기 예측기(404)에 의해 사용되는 제1 BTB 테이블(401), 제2 BTB 테이블(402) 및 제3 BTB 테이블(403)을 포함하는 BTB 테이블들과 상호 운용된다. 일부 실시 예들에 따르면, 분기 예측기(404)는 각 프로세스 사이클마다, 하나 이상의 분기 명령어로부터 하나 또는 두 번의 예측을 제공한다.

[0020] 각 BTB 테이블(401, 402, 403)은 엔트리가 속한 예측 어드레스를 식별하는 데 사용되는 태그 필드(407, 408, 409), 하나 또는 복수의 분기의 타겟을 저장하는 데 사용되는 타겟 어드레스 필드(410, 411, 412), 및 예측 윈도우에서 분기들의 위치 및 분기 유형을 저장하는 데 사용되는 정보 필드(416, 417, 418)를 포함한다. 도 2의 분기 예측 트레이너(205)와 같은 분기 트레이너는 각 BTB 테이블을 채우고, 트레이닝되는 각 BTB 엔트리에 대해 BTB 엔트리가 제1 BTB 테이블(401)에 기록되는지, 제2 BTB 테이블(402)에 기록되는지 또는 제3 BTB 테이블(403)에 기록되는지를 결정한다. 또한, BTB 테이블들(401-403) 각각은 각각의 분기 유형들에 대한 타겟 어드레스 필드들(410, 411, 412)을 가진다. BTB 테이블들(401-403) 각각은 내부에 저장된 각각의 분기 어드레스 유형 또는 유형들에 대해 전형적인 타겟 어드레스 사이즈들에 기초하여 상이한 사이즈들 또는 너비들(413, 414, 415)의 타겟 어드레스 필드들(410, 411, 412)을 가질 수 있다. BTB 테이블들(401, 402, 403) 각각은 하나 이상의 분기 유형들에 대해 지정된다. 예를 들어, 제2 BTB 테이블(402)은 제1 BTB 테이블(401)의 엔트리들과 상이한 브랜치 유형에 대한 어드레스들(411)을 저장한다. 특정 실시 예들에서, 제3 BTB 테이블(403)의 제3 BTB 엔트리 세트에는 예를 들어, 제2 BTB 테이블(402)의 BTB 엔트리 세트 내 BTB 엔트리들의 갯수에 비해 더 많은 수의 BTB 엔트리들이 있는데, 이는 BTB 엔트리로 정확하게 예측하면 절감되는 사이클 수의 측면에서 제3 BTB 테이블(403)의 엔트리들 유형이 프로세서(400)의 운용 효율성에 더 유리하기 때문이다. 내부에 저장될 분기 어드레스 유형들 및 서로에 대한 운영 효율성에 기초하여 각각의 BTB 테이블들(401-403)의 사이즈(엔트리들의 갯수)가 선택되고 BTB 테이블들(401-403)이 구축된다. 일 실시 예에서, 각 BTB 엔트리는 각각의 필드들에 적어도 하나의 태그 및 타겟 어드레스를 포함한다. 또한 각 엔트리는 스레드 ID 및 공유 표시기 또는 플래그와 같은 다른 정보도 포함할 수 있다.

[0021] 도 5는 일부 실시 예들에 따른 상이한 메모리 레벨들에 다수의 BTB 테이블들을 갖는 프로세서의 블록도이다. 프로세서(500)는 프로세서 코어(501)를 포함하며, 이는 분기 명령어들을 처리하는 분기 예측기(502)를 포함한다. 분기 예측기(502)는 레벨 1 BTB 테이블(503), 레벨 2 BTB 테이블 세트(504, 505, 506) 및 레벨 3 BTB 테이블(507)을 포함하는 다양한 구조들을 포함하고 이들과 상호 운용된다. 레벨 1 BTB 테이블(503)은 제1 레벨 메모리(508)에 있다. 레벨 2 BTB 테이블들(504, 505, 506)은 제2 레벨 메모리(509)에 있다. 레벨 3 BTB 테이블(507)은 제3 레벨 메모리(510)에 있다. 일부 실시 예들에 따르면, 레벨 1 BTB 테이블(503)의 세트가 가득 차거나 모든 BTB 엔트리 세트들이 가득 차면, BTB 엔트리는 레벨 1 BTB 테이블(503)로부터 복수의 레벨 2 BTB 테이블들 또는 레벨 2 BTB 테이블 세트(504-506) 중의 하나로 덮어 쓰이거나 축출된다. 레벨 2 BTB 테이블들(504-506) 각각은 특정 타겟 어드레스 사이즈 또는 타겟 어드레스 사이즈들의 범위 또는 다른 도면들과 관련하여 설명된 바

와 같은 특정 분기 명령어 유형에 대해 생성된다.

[0022] 레벨 2 BTB 테이블들(504, 505, 506) 중의 하나가 가득 차면, 프로세서(500)는 내부의 BTB 엔트리를 덮어 쓰거나 이로부터 BTB 엔트리를 제3 레벨 메모리(510)의 레벨 3 BTB 테이블(507)로 축출한다. 일부 실시 예들에서, BTB 가장 덜 중요한 엔트리와 같은 엔트리가 덮어 쓰인다. 다른 실시 예들에서, BTB 엔트리는 도시되지 않은 하나 이상의 BTB 테이블 중의 하나로 축출된다. 이러한 다른 BTB 테이블은 동일한 또는 상이한 메모리(이를테면 다른 메모리 레벨(508-510)의 BTB 테이블)에, 또는 프로세서의 다른 위치에, 또는 프로세서 코어(501) 외부의 메모리에, 또는 프로세서(500) 자체(이를테면 버스 또는 브리지에 의해 프로세서(500)에 결합되는 메모리)에 있을 수 있다. 예를 들어, 축출은 레벨 2 메모리(509)의 오버플로우 BTB 테이블에 대한 것이다. 일부 실시 예들에 따르면, 분기 유형 또는 분기 타겟 어드레스 사이즈에 기초하여 BTB 엔트리들의 서브 세트만(이를테면 하나의 BTB 테이블(504, 505, 506) 내 BTB 엔트리들의 서브 세트만)이 축출되게 허용된다.

[0023] 예시적인 실시 예에 따르면, 레벨 2 BTB 테이블들(504, 505, 506) 각각은 레벨 2 메모리(509)에 있다. 일부 실시 예들에서, BTB 테이블들(504, 505, 506) 각각은 4 -웨이 연관인 1,024개의 엔트리들을 포함한다. 각 BTB 엔트리는 사이즈가 68 비트(68b)이고, 최대 37 비트의 타겟 사이즈를 기록할 수 있다. 37 비트보다 큰 사이즈들의 타겟들은 간접 레벨을 통해 별도의 테이블에 기록될 수 있다. 여기서 기술되는 사이즈들은 이를테면 도 3의 BTB 테이블들(301-303), 도 4의 BTB 테이블들(401-403) 및 도 5의 BTB 테이블들(503-507)에 대해 가능한 다양한 사이즈들의 예증이 된다. 일부 실시 예들에서, BTB 테이블들은 가상 페치 어드레스의 해시를 사용하여 인덱싱 및 태깅된다.

[0024] 일부 실시 예들에 따르면, BTB 테이블들은 분기 예측 사이클당 최대 두 개의 분기들의 예측을 지원하고; 다른 실시 예들에서는, 프로세스 사이클 또는 분기 예측 사이클당 하나의 분기만이 평가된다. 바람직하게는, BTB 테이블들은 모든 분기 유형들 및 타겟 사이즈들을 지원한다. 도 5의 레벨 1 BTB 테이블(503)의 엔트리들과 같은 각 BTB 엔트리는 어느 것이 먼저 오든, 예측 어드레스에서 시작하여 정렬된 64B의 끝, 또는 제1 정적 분기의 끝, 또는 예측 윈도우에서 “마지막이 아닌”(“non-last” in a prediction window; NLIPW) 제1 동적 분기 또는 분기 쌍의 끝으로 연장되는 예측 블록에 대한 최대 두 개의 타겟들 및 동반 분기 정보를 저장한다.

[0025] 일부 실시 예들에 따르면 프로세서 설계에 따라, BTB 테이블들의 각 엔트리는 특정 필드들을 포함한다. 예시를 위해, 도 6은 일부 실시 예들에 따른 BTB 테이블에 대한 BTB 엔트리(600)의 블록도이다. 도 6에서, 다음은 BTB 테이블에 대한 68-비트 엔트리들에 대한 BTB 엔트리(600) 내의 각 위치 비트들의 필드들에 대한 설명이며, 각 BTB 엔트리(600)는 제로(0)에서 67까지 인덱싱된 68 비트를 가진다. 사이즈들 및 비트 수는 예시일 뿐이며, 여기에 제공된 지침을 사용하여 다른 사이즈들 및 비트 분포들이 구현될 수 있다. AUXPREDEN(601)은 위치 [67]의 1 비트이고 보조 예측 기들의 활성화를 나타낸다. SHRD(602)는 엔트리가 스레드들 간에 공유되는지 또는 공유 가능한지를 나타내는 위치 [66]의 1 비트이다. TID(603)는 위치 [65]의 1 비트이고 스레드 식별자(ID)를 나타낸다. TAG(604)는 위치 [64:52]에 위치된 13 비트이고 분기 엔트리에 대한 태그를 저장한다. LE(605)는 위치 [51:50]의 2 비트이고 인코드 길이 또는 "길이 인코드(length encode)"를 나타낸다. LE(605) 상태들은 이러한 값들을 갖는 다음 상태들을 포함한다: 1,1; 1,0; 및 0,1. PAIR(606)는 위치 [49]의 1 비트이고 오버플로우 어드레스 버퍼와의 쌍을 나타낸다. BRTYPE(607)은 위치 [48:44] 내에 위치된 5 비트이고 아래에서 더 상세히 설명될 분기 유형 인코딩을 나타낸다. BR1END(608)는 위치 [43:38]에 위치된 6 비트이고 [5:0]의 범위로 하나씩 증분될 수 있는 Branch1 종료 포인터를 나타낸다.

[0026] LE(605) 및 PAIR(606)의 상태들은 함께 각 (제1 레벨 또는 제2 레벨) BTB 엔트리의 위치 [37:0]의 비트들이 사용되는 방식을 나타낸다. USEOVERFLOW(USEOA)(609)는 위치 [37]의 1 비트이고 오버플로우 어드레스 버퍼의 사용을 나타낸다. 예를 들어, 위치 [48:32]에 위치된 타겟은 오버플로우 어드레스 식별자 OAIDX(610)(아래 참조)를 통해 가리켜지는 오버플로우 어드레스 버퍼 엔트리에서 오고 PAIR=0일 때 사용된다. OAIDX(610)는 위치 [36:32]에 위치된 5 비트이고 오버플로우 어드레스 버퍼의 인덱스를 나타낸다. OAIDX(610)는 PAIR=0 및 USEOA=1일 때 사용된다. BR0END(611)는 위치 [37:32]의 6 비트의 대체 할당(612)이고 [5:0]의 범위로 하나씩 증분될 수 있는 Branch0 종료 포인터를 나타낸다.

[0027] BR0TGT(613)는 인덱스 (X+1)부터 31까지의 제1 범위(615)에 걸쳐 있는 Branch0 타겟 또는 제1 분기 타겟 어드레스이며, 여기서 X는 정수이고 제1 분기 타겟 어드레스는 LE(605)의 2 비트의 상태에 따른 가변 길이를 가진다. 예를 들어, LE=1,1의 경우 BR0TGT(613)는 위치 [31:20]의 12 비트이고; LE=1,0의 경우 BR0TGT(613)는 위치 [31:16]의 16 비트이며; LE=0,1의 경우 BR0TGT(613)는 위치 [31:12]의 20 비트이다. LE=0,0일 때 BR0TGT(613)는 유효하지 않거나 사용되지 않는다.

[0028] BR1TGT(614)는 위치 인덱스 [0]부터 위치 인덱스 [X]까지의 제2 범위(616)에 걸쳐 있는 Branch1 타겟 또는 제2 분기 타겟 어드레스이며, 여기서 X는 정수이다. 예를 들어, PAIR=0 그리고 USEOA=0일 때 BR1TGT(614)는 위치 [36:0]의 37 비트이고; PAIR=0 그리고 USEOA=1일 때 BR1TGT(614)는 위치 [31:0]의 32 비트이고; PAIR=1 그리고 LE=1,1일 때 BR1TGT(614)는 위치 [19:0]의 20 비트이고(BR0TGT(613)는 상기한 바와 같이 12 비트이다); PAIR=1 그리고 LE=1,0일 때 BR1TGT(614)는 위치 [15:0]의 16 비트이며(BR0TGT(613)는 상기한 바와 같이 16 비트이다); PAIR=1 그리고 LE=0,1일 때 BR1TGT(614)는 위치 [11:0]의 12 비트이다(BR0TGT(613)는 상기한 바와 같이 20 비트이다). 이러한 식으로, BTB 테이블들의 각 레벨 1, 레벨 2 또는 레벨 3 BTB 엔트리는 일부 실시 예들에 따라 두 개의 분기 타겟 어드레스들 BR0TGT(613) 및 BR1TGT(614)를 저장할 수 있다.

[0029] 브랜치 유형 인코딩과 관련하여, 일부 실시 예들에 따르면, 5-비트 분기 유형은 Branch1의 다음과 같은 속성들을 나타낸다. 5 분기 유형 비트 중 처음 2 비트에 대해, 각 분기 명령어에 대한 분기 유형은 다음과 같이 위치 [1:0]에 지정된다: 값들이 0,0인 제1 상태는 유효하지 않음에 대응한다; 값들이 0,1인 제2 상태는 JMP(점프 유형)(jump type)에 대응한다; 값들이 1,0인 제3 상태는 CALL(호출 유형)(call type)에 대응한다; 그리고 값들이 1,1인 제4 상태는 RET(리턴 유형)(return type)에 대응한다. 5-비트 분기 유형의 위치 [2]에서, 비트는 값 0에 대한 고정 타겟 및 값 1에 대한 가변 타겟을 지정한다. 분기들은 디폴트로 고정 타겟들로 인스톨되고 분기들이 BTB 엔트리에 처음 인스톨된 것과 상이한 타겟을 얻는 경우 간접 분기들만 가변 타겟으로 변환된다.

[0030] 5-비트 분기 유형의 마지막 2 비트에 대해서는, 다음과 같은 방식에 따라 위치 [4:3]에 지향성 상태가 제공된다. 직간접 고정 분기들은 위치 [4:3]을 보고 구분된다. 0,0 값들은 DE(디코드)(decode) 리디렉트를 통해 찾아지는 무조건 디렉트(unconditional direct; UD)에 대응한다. 일부 실시 예들에 따르면, UD는 정의에 의해 정적이고 고정된다. 분기가 속한 레벨 2 BTB의 테이블을 알기 위해서는 UD에 대한 구별이 필요하다. 위치 [4:3]의 1,0 값들은 UD가 아니라, 정적인 것에 대응한다. 분기들은 디렉션 측면에서 디폴트로 정적인 것으로 인스톨되고 조건 분기들이 이행되지 않은 것으로 리디렉트되는 경우 조건 분기들만 동적 디렉션으로 변환된다.

[0031] 위치 [4:3]의 1,1 값들은 "UD가 아닌, 동적, 예측 윈도우의 마지막(Last In Prediction Window; LIPW)"상태에 대응한다. LIPW는 프로세서가 64B 캐시 라인에서 마지막 분기로 식별하는 임의의 분기에 대해 설정된다. 이러한 상태는 BTB 엔트리에서 어떠한 분기도 이행되지 않은 경우 BTB 판독 로직이 다음 페치 어드레스(fetch address; FA)를 결정하는 데 도움이 된다. LIPW가 설정된다면(그리고 어떠한 분기도 이행되지 않는 것으로 가정하면), 다음 FA는 순차적인 캐시 라인 어드레스이고, 그렇지 않으면 다음 FA는 BTB 엔트리의 마지막 분기 바로 뒤의 명령어 어드레스이다. 위치 [4:3]의 0,1 값들의 상태는 “UD가 아닌, 동적인 NLIPW”에 대응한다.

[0032] 분기 타겟 쌍은 단지 제1 분기가 이행되지 않는 것으로 예측될 수 있을 때만 값을 갖기 때문에, Branch0은 동적 고정 JMP 유형의 분기만 될 수 있다. 일 실시 예에서, 분기 타겟 쌍들은 다음 속성들을 가진다. 쌍당 하나의 분기만 가변 타겟을 가질 수 있다. 쌍들은 Call/Call, Call/Ret, Ret/Call 또는 Ret/Ret를 포함할 수 없다. 쌍들은 전술한 바와 같은 다수의 타겟 사이즈 조합들을 지원한다. 쌍을 이루는 것이 가능한지 여부와 어느 조합을 사용할지를 알기 위해, 단일 유형이 단기, 중기 및 장기 분기 예측 타겟을 구분한다. 이러한 구분은 또한 다른 BTB 테이블로부터 이를테면 오버플로우 레벨 2 BTB 테이블 또는 레벨 3 BTB 테이블로 분기 타겟 어드레스들을 희생시키거나 축출할 때 BTB 테이블들 중의 어느 BTB 테이블을 사용할지 선택하는 데 사용된다.

[0033] 다음의 설명은 예시적인 실시 예에 따라 도 4의 BTB 테이블들(401, 402, 403) 및 도 5의 BTB 테이블들(503-507)과 같은 제1 BTB 테이블들의 제1 BTB 분기 타겟 어드레스 비트들과 관련하여 더 상세한 내용을 제공한다. 일부 실시 예들에 따르면, 두 개의 분기 가치의 타겟 어드레스들을 저장하고 이를테면 분기 유형에 기초하여 두 개의 분기들에 대해 가능한 많은 경우들을 커버하기 위해 타겟 필드에 대한 32 비트가 필요하다. 필요한 타겟 사이즈는 직접 분기별로 1 비트에서 49 비트까지 다양하지만, 리턴 유형 분기들은 어떠한 타겟 비트도 필요로 하지 않는다. 가변 타겟 분기들의 경우, 마주하는 제1 타겟은 BTB에서 여기서 더 설명될 바와 같이 트레이닝되고 다른 타겟들은 간접 타겟 예측기에서 트레이닝된다.

[0034] 엔트리가 두 개의 분기 타겟 어드레스들을 저장할 때 타겟들 간에 타겟 필드가 공유되며, 각각 일반적으로 상이한 길이를 가진다. 가변 길이 분기들을 구현하는 것은 타이밍 관점에서 너무 비용이 많이 들기 때문에, 타겟들을 저장하는 데 보다 최적화된 접근법은 두 분기들에 대해 세 가지 타겟 길이 변화를 사용하고 도 4의 BTB 테이블들(401, 402, 403) 및 도 5의 BTB 테이블들(504, 505, 506)과 같은 각각의 BTB 테이블들에 점프, 호출 및 리턴 유형들과 같은 분기 유형을 사용하여 인코딩하는 것이다. 분기 타겟이 너무 커서 이의 타겟이 제2 분기의 타겟과 나란히 맞지 않는 경우들이 있다. 이러한 경우, 분기는 완전 타겟 어드레스 또는 타겟 비트들을 저장하기 위한 자체 BTB 엔트리를 필요로 한다. 다음 두 가지 경우가 있다: (1) 엔트리 스토리지당 단일 분기, 다른 분기

유형 필드 및 종료 필드 - 상술된 바와 같은 - 가 타겟 비트들로 재사용된다. 이러한 메커니즘은 37-비트 타겟 어드레스들을 지원할 수 있게 한다; 그리고 (2) 37 비트보다 긴 타겟들에서 상위 비트들은 OAIDX 필드에 의해 가리켜지는 오버플로우 어드레스 엔트리에 의해 제공된다. 제1 BTB에서 적중한 후 오버플로우 어드레스 구조를 필요로 하는 타겟 어드레스들에 대한 리디렉트들은 추가 리디렉트 페널티를 초래한다. BTB 교체 정책들에 대한 예로서, 일부 실시 예들에 따르면, 제1 BTB 테이블(401) 또는 BTB 테이블(503)은 교체 정책의 일부로서 라운드 로빈 방식을 사용한다. 제1 BTB 테이블들의 256개 세트들 각각은 2 비트 라운드-로빈(RR, round-robin) 카운터를 포함한다.

[0035] 이를테면 도 1의 명령어 파이프라인 아키텍처(100), 도 4의 프로세서(400), 또는 도 5의 프로세서(500)에서의 파이프라이닝 측면에서, BTB 테이블은 판독이 절전을 위해 억제되지 않는 한 모든 예측 플로우를 위해 판독된다. 일부 실시 예들에 따르면, 제1 BTB 판독은 BP0으로 지정된 제1 단계에서 시작된다. 후속 스테이지들은 예측 파이프라인에서의 스테이지들로서 BP1, BP2 등으로 설계된다. BTB 테이블은 BP2에서 BP0으로 리디렉트된다.

[0036] 도 4의 프로세서(400) 및 도 5의 프로세서(500)와 같은 프로세서의 일부 실시 예들에 따르면, 상술한 바와 같이 BTB 테이블들(401, 402, 403) 및 테이블들(504, 505, 506)로 구현됨에 따라 세 개의 논리 테이블들이 있다. BTB 테이블들의 각 테이블들은 도 4에서와 같은 분기 유형들의 서브 세트 또는 도 3에서와 같은 타겟 사이즈들을 지원한다. BTB 테이블들(401-403, 503-507)로 구현되는 이러한 다중 테이블 BTB들은 몇 가지 이점들을 제공한다. 예를 들어, 이러한 배열은 프로세서에서 면적 효율적이다. 각 BTB 테이블(401-403, 504-506)은 이의 분기 유형 또는 타겟 사이즈 서브 세트에 대한 스토리지 용량만 가진다. 각 BTB 테이블은 일부 실시 예들에서 많은 양의 연관성을 제공하지만, 각 BTB 테이블(401-403, 504-506)은 특정 실시 예들에서 전체 LRU를 구현할 수 있다. 높은 실제 연관성을 사용하면 구현하는 데 많은 비용이 든다. 다수의 BTB 테이블들을 사용하면 더 높은 값의 분기들(예를 들어, 쌍을 이루는 동적 분기들)을 다른 분기들과 분리한다. 다수의 BTB 테이블들(301-303, 401-403, 503-507)은 무제한 분기 타겟 사이즈를 지원하지만 복수의 BTB 테이블들(301-303, 401-403) 중 하나의 BTB 테이블만 전체 분기 타겟 어드레스 사이즈를 지원하면 된다. 다수의 BTB 테이블들의 사용은 프로세서 평면도 유연성을 향상시킨다. 예를 들어, BTB 테이블들(401-403) 중 두 개만 동적 분기들을 지원하면 된다. 이에 따라, 이러한 BTB 테이블들은 조건부 예측기에 가깝게 배치될 수 있다. 운용시 프로세서의 특정 실시 예들에 따르면, BTB 테이블들 중 두 개만이 상당한 타겟 어드레스 비트 수를 구동한다.

[0037] 다음은 예시적인 BTB 편성 및 사이즈 조정 체계에 대해 더 상세한 내용을 제공한다. 제1 BTB 테이블(401)과 같은 제1 BTB 테이블(인덱스 0)은 작은 엔트리들을 저장한다. 제2 BTB 테이블(402)과 같은 제2 BTB 테이블(인덱스 1)은 DE 리디렉트들을 통해 발견되는 분기들(예를 들어, 낮은 리디렉트 페널티를 갖는 분기들)을 저장한다. 제3 BTB 테이블(403)과 같은 제3 BTB 테이블(인덱스 2)은 다른 모든 유형들을 저장한다. 예를 들어, 분기 유형들은 다음과 같이 할당된다. 제1 BTB 테이블은 단일의 RET 분기 유형들 및 단일의 정적 고정 조건 JMP 단기 분기 유형들에 대한 어드레스들을 저장한다. 제2 BTB 테이블은 "초장기(extra-long)" 분기 유형들이 아닌 단일의 정적 고정 무조건 직접 호출 및 초장기 분기 유형들이 아닌 단일의 정적 고정 무조건 직접 점프를 저장한다. 제3 BTB 테이블은 다른 모든 분기 유형들을 저장한다. 다른 BTB 테이블 수 및 다른 유형 및 타겟 사이즈 할당은 도 3 내지 도 5와 관련하여 설명된 것들 이상으로 가능하다.

[0038] BTB 테이블들의 용량들의 일례는 총계 5,632개의 BTB 엔트리들을 갖는 구현 예에서 다음과 같다. 제1 BTB 테이블은 총계 2,048개의 BTB 엔트리들에 대해 512 세트 및 4 웨이를 가진다. 제1 BTB 엔트리들은 소계 73,728 비트에 대해 엔트리당 36 비트를 가진다. 제2 BTB 테이블은 총계 1,024개의 BTB 엔트리들에 대해 256 세트 및 4 웨이를 가진다. 제2 BTB 엔트리들은 소계 65,536 비트에 대해 엔트리당 64 비트를 가진다. 제3 BTB 테이블은 총계 2,560개의 BTB 엔트리들에 대해 512 세트 및 5 웨이를 가진다. 제3 BTB 엔트리들은 소계 184,320 비트에 대해 엔트리당 72 비트를 가진다. 모두 합해서, BTB 테이블들의 세트는 13 웨이, 5,632개의 엔트리들을 가지며, 도 4의 지정된 영역(406)과 같은 하나 이상의 지정된 영역의 요소들과 같이 323,584 비트를 필요로 한다.

[0039] 이러한 예에서, 다중 테이블 BTB 테이블들에 대해, 오버플로우 어드레스 버퍼는 도 4의 다수의 BTB 테이블들의 세트(401-403)와 도 5의 BTB 테이블들의 세트(504-506) 간에 공유된다. 제1 BTB 테이블(401, 504), 제2 BTB 테이블(402, 505) 및 제3 BTB 테이블(403, 506) 간에 분기 유형 인코딩은 상이하다. 그러나, {LE, PAIR, TYPE}에 따라 정의된 BTB 분기 유형과 {TableNumber, LE, PAIR, TYPE}에 따라 기술된 BTB 테이블들의 분기 유형 간에는 직접 매핑이 있다. 이러한 BTB 기술 (필드들)은 스왑 및 BTB 축출 동안 사용된다.

[0040] 다른 실시 예들에 따르면, 도 4의 제1 BTB 테이블(401)과 같은 제1 BTB 테이블의 BTB 엔트리들에 대한 필드 및

위치 비트들에 대해, 이러한 필드 및 위치 비트들은 다음을 포함한다: 위치 [35:34]에 2 비트를 갖는 UNUSED; 위치 [33]에 1 비트를 갖는 SHRD; 위치 [32]에 1 비트를 갖는 TID; 위치 [31:20]에 12 비트를 갖는 TAG; 위치 [19:14]에 6 비트를 갖는 BREND; 위치 [13:12]에 2 비트를 갖는 BRTYPE; 위치 [11:0]에 12 비트를 갖는 BRTGT. UNUSED 필드는 대응하는 매크로 위치에 사용되지 않은 비트들이 있는지 여부를 나타낸다. SHRD 필드는 BTB 엔트리가 스레드들 간에 공유되는지 또는 공유 가능한지 여부를 나타낸다. TID 필드는 스레드 ID이다. BREND 필드는 [5:0]의 범위로 하나씩 증분될 수 있는 분기 종료 포인터이다. BRTYPE 필드는 분기 유형 인코딩을 나타낸다. 일부 실시 예들에 따르면, 제1 BTB 테이블(401)은 RET(리턴)(return) 유형 분기들 및 적어도 하나의 유형의 JMP 분기를 포함하여 여러 유형들의 분기들에 대한 정보를 포함한다. 일부 실시 예들에 따르면, BRTYPE 필드는 다음과 같이 인코딩된다: 상태 0,X(여기서 X는 0 또는 1일 수 있다)는 유효하지 않은 상태에 대응하고, 값들이 1,0인 상태는 JUMP에 대응하며, 값들이 1,1인 상태는 RET에 대응하며 위치 [4:0]의 BRTGT는 리턴 유형에 대한 전체 분기 유형을 기록한다.

[0041] 도 7은 일부 실시 예들에 따른 제2 BTB 테이블에 대한 BTB 엔트리(700)의 블록도이다. 다음은 도 4의 제2 BTB 테이블(402) 또는 도 5의 제2 BTB 테이블(505)과 같은 제2 BTB 테이블의 BTB 엔트리들에 대한 필드들 및 위치 비트들에 대한 설명이다. 도 7의 각 BTB 엔트리(700)는 64 비트 너비로, 이는 도 6의 BTB 엔트리(600)보다 짧다. 도 7에서, BTB 엔트리(700)는 다음 필드들을 포함한다: 위치 [63]에 1 비트를 갖는 SHRD(701); 위치 [62]에 1 비트를 갖는 TID(702); 위치 [61:49]에 13 비트를 갖는 TAG(703); 위치 [48:47]에 2 비트를 갖는 LE(704); 위치 [46:45]에 2 비트를 갖는 BRTYPE(705); 위치 [44:39]에 6 비트를 갖는 BREND(706); 위치 [38:0]에 39 비트를 갖는 BRTGT(707). SHRD(701)는 BTB 엔트리가 스레드들 간에 공유되는지 또는 공유 가능한지 여부를 나타낸다. TID(702)는 스레드 ID이다. LE(704)는 길이 인코드를 나타내고 제1 BTB 테이블(401)과 같은 제1 BTB 테이블의 LE(605)와 동일한 인코딩을 사용한다. BRTYPE(705)는 분기 유형 인코딩을 나타낸다. 일부 실시 예들에 따르면, 제2 분기 유형 BTB는 오직 무조건 직접 분기들과 같은 단 하나의 분기 유형에 대한 정보를 포함한다. 이러한 필드는 JMP, CALL 및 무효 간 구분을 인코딩하기 위한 것이다. 일부 실시 예들에 따르면, 제2 분기 유형 BTB에 대한 BRTYPE은 다음과 같이 인코딩된다: 값들이 0,0인 상태는 유효하지 않은 상태에 대응하고; 값들이 0,1인 상태는 JMP(점프) 분기 유형에 대응하고; 상태 1,0은 CALL 분기 유형에 대응하며; 값들이 1,1인 상태는 다른 비합법 상태에 대응한다. 제2 BTB 엔트리(700)에 대한 BREND(706)는 [5:0]의 범위로 하나씩 증분되는 분기 종료 포인터이다. BRTGT(707)는 제1 BTB 엔트리(600)와 동일하거나 유사한 포맷을 따르는 분기 타겟이다. 오버플로우 어드레스 버퍼에서 이용 가능한 추가 어드레스 비트들을 필요로 하는 분기들은 도 8의 BTB 엔트리(800)에서와 같이 제3 분기 유형 BTB에 할당된다. 결과적으로, 일부 실시 예들에 따라 추가 어드레스 비트들에 대한 필요를 수용하기 위한 제2 분기 유형 BTB에는 1 비트의 USEOA 필드가 포함되지 않는다. 여기서 나타내어질 때, OA는 오버플로우 어드레스(overflow address) 또는 그 안의 특정 필드들, 비트들 또는 플래그들을 나타낸다.

[0042] 도 8은 일부 실시 예들에 따른 제3 BTB 테이블에 대한 제3 BTB 엔트리(800)의 블록도이다. 다음은 도 4의 제3 BTB 테이블(403) 또는 도 5의 제3 BTB 테이블(506)의 BTB 엔트리에 대한 것과 같은 제3 분기 유형 BTB 엔트리(800)의 엔트리들에 대한 필드들 및 위치 비트들에 대한 설명이다. 각 BTB 엔트리(800)는 제로(0)에서 71까지 인덱싱된 72 비트 너비이고, 다음 필드들을 포함한다: 위치 [71]에 1 비트를 갖는 UNUSED(801); 위치 [70]에 1 비트를 갖는 AUXPREDEN(802). 위치 [69]에 1 비트를 갖는 SHRD(803); 위치 [68]에 1 비트를 갖는 TID(804); 위치 [67:56]에 12 비트를 갖는 TAG(805); 위치 [55:54]에 2 비트를 갖는 LE(806); 위치 [53:52]에 2 비트를 갖는 BR1LBIAS(807); 위치 [51]에 1 비트를 갖는 PAIR(808); 위치 [50:46]에 5 비트를 갖는 BRTYPE(809); 위치 [45:40]에 6 비트를 갖는 BR1END(810); 및 위치 [39:38]에 2 비트를 갖는 BR0LBIAS(811). 필드 그룹 {BREND(812), BRTGT(814)}는 위치 [37:0]에 각각의 가변 길이 범위들(813, 815)에 걸쳐 분산된다. 예를 들어, BREND(812)는 (Y + 1)에서 37까지 위치되고, BRTGT(814)는 0에서 (Y)까지 위치되며, 여기서 Y는 정수이다. UNUSED(801)는 대응하는 매크로 위치에 사용되지 않은 비트들이 있는지 여부를 나타낸다. AUXPREDEN(802)는 루프 종료 예측기와 같은 보조 예측기들의 활성화를 나타낸다. SHRD(803)는 BTB 엔트리가 스레드들 간에 공유되는지 또는 공유 가능한지 여부를 나타낸다. TID(804)는 스레드 ID이다. LE(806)는 길이 인코드를 나타내고 제1 BTB 테이블(401)과 같은 제1 BTB 테이블의 동일한 LE 필드에 대해 그리고도 6의 LE(605)와 관련하여 상술된 것과 동일한 인코딩을 사용한다. 도 8에서, BR1LBIAS(807)는 Branch1에 대한 로컬 바이어스(local bias)가 있는지 여부를 나타낸다. PAIR(808)는 여기서의 다른 곳에서 추가로 설명된 바와 같이 사용된다. 일부 실시 예들에 따르면, 제3 분기 유형 BTB 엔트리(800)에 대한 BRTYPE(809)은 이를테면 제1 BTB 엔트리(600)에 대한 다른 BTB 테이블들에 대한 BRTYPE에 대해 설명된 바와 같이 인코딩된다. 제3 분기 유형 BTB 엔트리(800)에 대한 BR1END(810)는 [5:0]의 범위로 하나씩 증분되는 Branch1에 대한 분기 종료 포인터이다. BR0LBIAS(811)는

Branch0에 대한 로컬 바이어스가 있는지 여부를 나타낸다. BREND(812) 및 BRTGT(814)는 제1 BTB 테이블(401)과 같은 제1 BTB 테이블로부터의 USEOA, OAIDX, BR0END, BR1TGT 및 BR0TGT 필드들과 관련하여, 각각 Branch0 및 Branch1에 대한 하나 또는 두 개의 분기 타겟을 저장하는 데 사용된다. 제1, 제2 및 제3 분기 유형 BTB 테이블들(401-403) 및 각각의 BTB 엔트리들(600, 700, 800)을 참조하여 설명된 비트 수들(너비들) 및 위치들은 단지 예일 뿐이다. 프로세서 또는 시스템에 다수의 BTB 테이블들을 구현할 때 설명된 필드들 또는 다른 변수들 또는 필드들에 대한 다른 비트 수들 및 위치들이 가능하다.

[0043] 예시적인 다중 BTB 테이블 교체 정책 측면에서, 여러 가능한 실시 예들 중 하나로 다음이 제공된다. 본 실시 예에서, 제1 레벨 1 BTB 테이블은 엔트리들의 세트 중 하나 이상이 가득 찰 때까지 먼저 모든 BTB 엔트리들을 수락하며, 이러한 예에서 세트는 엔트리들의 세트-연관 그룹의 세트를 나타낸다. 다른 예에서, 제1 레벨 1 BTB 테이블은 레벨 1 BTB 테이블이 가득 찰 때까지 먼저 모든 BTB 엔트리들을 수락한다. 그 다음, 프로세서 또는 프로세서 코어가 운용될 때, BTB 엔트리들은 각각 도 3의 BTB 테이블들(301-303) 및 도 4의 BTB 테이블들(401-403)과 같이 타겟 어드레스 사이즈 또는 분기 유형에 기초하여 설계된 복수의 레벨 2 BTB 테이블들 중 하나로 축출된다. 예를 들어, 교체 정책은 도 3의 BTB 테이블들(301-303) 및 도 4의 BTB 테이블들(401-403)과 같은 각 레벨 2 BTB 테이블에 대한 전체 LRU이다. 분기 유형 BTB 교체 정책은 엔트리들을 최근에 가장 적게 사용된(least recently used, LRU) 또는 최근에 가장 많이 사용된(most recently used, MRU) 중 어느 하나로 마킹하는 것을 지원한다. 세트 중 한 웨이는 웨이가 클리어된 후 LRU로 마킹되어 새로운 BTB 엔트리가 이러한 세트에 기록되는 다음 시기에 비어 있는 엔트리가 교체되고 유효한 엔트리는 덮어 쓰여지지 않게 된다. 한 웨이는 웨이가 인스톨된 후 MRU로 마킹되어 교체를 위해 큐의 뒤쪽으로 가게 된다. 아래 표 1은 레벨 1이 레벨 1 메모리를 나타내고 레벨 2가 레벨 2 메모리를 나타내는 등의 경우들을 상세히 설명한다:

표 1

[0044]

이벤트		새로운 값		업데이트
레벨 2 BTB 적중(레벨 1 BTB 부적중)		MRU		레벨 1 BTB 희생으로 교체되는 경우 적중 엔트리를 업데이트
레벨 2 BTB 적중(레벨 1 BTB 부적중)		MRU		레벨 2 BTB 적중 위치와 상이한 경우 레벨 1 BTB 희생에 대한 레벨 2 위치를 업데이트
레벨 2 BTB 적중(레벨 1 BTB 부적중)		LRU		레벨 1 BTB 희생으로 교체되지 않는 경우 적중 엔트리를 업데이트(또한 레벨 2 BTB에서 적중 엔트리를 무효화)
레벨 1 BTB 및 레벨 2 BTB 적중		LRU		레벨 1 BTB 및 레벨 2 BTB에서 적중하는 경우 레벨 2 위치를 업데이트(또한 레벨 2 BTB에서 적중 엔트리를 무효화)
트레인 체크		MRU		축출된 레벨 1 BTB 엔트리에 대한 레벨 2 BTB 위치를 업데이트

[0045] 다음의 설명은 일부 실시 예들에 따른 파이프라인 동작과 관련하여 더 상세한 내용을 제공한다. 분기 유형 BTB는 전력 필터에 의해 취소되지 않는한 모든 예측 플로우에 대해 판독된다. 판독은 제1 위치 BP0에서 개시되고 BP4에서 BPN1로 리디렉트되며 여기서 N은 후속 사이클을 나타낸다. 레벨 1 BTB 부적중/레벨 2 BTB 적중 스왑 사례의 파이프라인 타이밍은 아래 표 2에 제시된다. 레벨 1 BTB로부터의 희생 웨이 및 레벨 2 BTB로부터의 적중 웨이 양자는 기록된다. 희생 웨이는 적중 웨이와 다른 레벨 2 BTB 테이블 및 뱅크를 가질 수 있으며, 이는 특별한 고려가 필요하다. 예를 들어, LRU 상태에 기초하여 레벨 2 희생 뱅크가 선택되고 덮어 쓰인다. 레벨 1 BTB 및 레벨 2 BTB에 대한 쓰기는 BP4에서 어써트되는 쓰기 활성화(WrEn) 신호로 일어나며 실제 쓰기는 BP5 위치에서 일어난다.

표 2

[0046]

BP0		BP1		BP2		BP3	BP4		BP5		
--		--		레벨 1 BTB 부적중			레벨 2 BTB 적중		레벨 1 BTB WrEn		Wr 대 레벨 1 BTB
--		--		레벨 1 BTB 희생			--		레벨 2 BTB WrEn		Wr 대 레벨 2 BTB

[0047] 다음의 설명은 일부 실시 예들에 따른 도 4의 프로세서(400)와 같은 프로세서의 레벨 2 BTB 물리적 편성과 관련하여 더 상세한 내용을 제공한다. 제1 레벨 2 BTB 테이블은 4 웨이를 갖는 512 세트, 엔트리당 36 비트, 세트당 2개의 매크로, 72b의 매크로 유형 및 총 8개의 매크로를 포함한다. 제2 레벨 2 BTB 테이블은 4 웨이를 갖는 256 세트, 엔트리당 64 비트, 세트당 4개의 매크로, 64b의 매크로 유형 및 총 8개의 매크로를 포함한다. 제3 레벨 2 BTB 테이블은 5 웨이를 갖는 512 세트, 엔트리당 72 비트, 세트당 5개의 매크로, 72b의 매크로 유형 및 총 20개의 매크로를 포함한다.

[0048] 인덱스의 하나 이상의 상위 비트들은 판독 활성화로 사용되어 제1 레벨 2 BTB 테이블에 대해 웨이당 매크로의 절반만 판독되고 제2 및 제3 레벨 2 BTB 테이블들에 대해 웨이당 하나의 매크로가 판독된다. 제1 및 제3 레벨 2 BTB 테이블들은 도 4의 레벨 1 BTB 테이블(401)과 같은 레벨 1 BTB 테이블보다 2배 많은 세트들을 갖기 때문에, 태그의 하위 비트는 이러한 BTB 테이블들에 대한 인덱스 최상위 비트(most significant bit, MSB)로 사용되어 BTB 테이블들은 12-비트 태그만 저장하면 된다.

[0049] 비교적 작은 클라이언트 디바이스 또는 이동 전화용 프로세서들의 실시 예들에서, 다수의 분기 유형 BTB 테이블들의 사용은 성능에 비해 높은 전력 비용을 가지므로, 전력 소비를 줄이기 위해 추가 방법들이 수행될 수 있다. 예를 들어, 정적 방법들은 일부 또는 모든 시간 동안 레벨 2 BTB 테이블들에 대한 전력 게이팅과 일부 또는 모든 시간 동안 레벨 2 BTB 테이블에 대한 클록 게이팅을 포함한다. 각 단계에서, 레벨 2 BTB 활성화 기능은 이를테면 BIOS 또는 퓨즈에 의해 설정될 수 있다. 전력 소비를 줄이기 위한 동적 방법들은 특정 레벨 2 BTB 테이블이 필요한 시기를 인식하는 추가 제어와 함께 레벨 2 BTB 테이블을 적응적으로 전력 게이팅하는 것을 포함한다. 다른 동적 방법은 애플리케이션에 의해 선호되는 전력 사용 또는 운영 체제, 펌웨어 또는 애플리케이션이 활성화하는 디바이스의 전력 설정에 따라 애플리케이션이 이의 레벨 2 BTB 테이블들의 사용을 조정하도록 운영 체제, 펌웨어 등에 의해 실행되는 애플리케이션을 조정하는 것을 포함한다.

[0050] 다음의 설명은 도 3의 제1 BTB 테이블(301), 제2 BTB 테이블(302) 및 제3 BTB 테이블(303) 및 도 4의 대응하는 유사한 테이블들과 같은 복수의 BTB 테이블들에 의한 사용을 위한 오버플로우 어드레스 버퍼와 관련하여 더 상세한 내용을 제공한다. 예측 구조체들에 모든 비트를 저장할 필요 없이 최대 49 비트의 타겟 사이즈의 분기들을 지원하기 위해, 프로세서에 오버플로우 어드레스 버퍼가 제공된다. 오버플로우 어드레스 버퍼는 분기 타겟 어드레스의 예를 들면, 최상위 또는 17 비트 세트와 같은 특정 비트 수를 저장하는 데 사용된다. 일부 실시 예들에 따르면, BTB 테이블과 같은 예측기 구조체에 의해 참조되는 최상위 17 비트는 오버플로우 어드레스 버퍼에 대한 포인터를 통해 이루어지므로 BTB 테이블의 저장 공간을 절감할 수 있다. 오버플로우 어드레스 버퍼는 많은 타겟 어드레스 비트 수를 저장해야 할 때 사용된다. 테스트에 따라, 트레이스 분석한 결과로는 소정의 트레이스에 대해, 상위 17 비트의 페치 어드레스가 명령어 세트의 다양한 분기들에 대해 제한된 수의 값들만 취하는 것으로 나타났다.

[0051] 일부 실시 예들에 따르면, 오버플로우 어드레스(OA) 버퍼는 32개의 엔트리들을 포함하며, 각 엔트리는 여기서 설명된 바와 같이 위치 [48:32]의 BTB 엔트리의 위치에 있는 가상 어드레스에 대한 단일 필드를 포함한다. 각 오버플로우 어드레스 버퍼 엔트리는 스레드들 간에 공유된다. 예측 시에, 오버플로우 어드레스 버퍼는 BTB가 제공하는 5-비트 포인터에 기초하여 어드레스의 상위 17 비트를 BTB에 제공한다. 포인터가 예측기들 중 하나에 기록되었으므로 예측 시에 사용된 오버플로우 어드레스 버퍼 엔트리가 덮어 쓰였을 수 있다. 이러한 경우, 예측은 잘못되고 리디렉트되어 예측기가 바로잡히게 한다. 서로 다른 예측기들은 예측 파이프의 서로 다른 스테이지들에서 오버플로우 어드레스 버퍼를 판독하므로, 오버플로우 어드레스 버퍼에서 다수의 동시 판독들을 필요로 한다.

[0052] 오버플로우 어드레스 버퍼는 다음과 같이 BTB의 트레이닝 동안 할당된다. 먼저 트레이닝은 분기 타겟이[48:32] 범위에 대한 현재 어드레스와 상이한지 여부를 결정하며 이 경우 오버플로우 어드레스 버퍼가 필요하다. 오버플로우 어드레스 버퍼가 필요한 경우, 오버플로우 어드레스 버퍼 엔트리가 위치 [48:32]에 원하는 타겟 어드레스를 이미 포함하고 있는지 여부를 결정하기 위해 현재 오버플로우 어드레스 버퍼 엔트리들이 위치 [48:32]의 타겟 어드레스와 비교된다. 매치 시, 매칭되는 오버플로우 어드레스 버퍼 인덱스가 예측기 BTB 테이블에 기록된다. 위치 [48:32]에 타겟과 매칭되는 오버플로우 어드레스 버퍼 엔트리가 없으면, 전체 LRU 교체 정책을 사용하여 오버플로우 어드레스 버퍼에 새로운 엔트리가 할당된다. 일부 실시 예들에 따르면, 쓰기 트레이닝은 추론적이고 일부 오버플로우 어드레스 버퍼 엔트리들은 잘못된 경로에 있는 동안 할당되었기 때문에 전혀 유용하지 않은 것들이 할당될 수 있다.

[0053] 다음의 설명은 BTB 트레이닝과 관련하여 더 상세한 내용을 제공한다. 예측이 리디렉트되면(이를테면 예측 파이

프라인의 디코드 및 실행 스테이지들에서), BTB 테이블은 프로그램 시퀀스에서 예측이 다시 발생할 때 예측 어드레스에서 예측 정확도를 향상시키도록 트레이닝 알고리즘에 의해 업데이트된다. 리디렉트가 수신되면, 어떤 종류의 트레이닝/업데이트 조치가 필요한지 결정하기 위해 예측 기록 버퍼로부터의 예측 분기 유형, 분기 타겟, EndAddr 및 다른 상태들과 함께, 리디렉트 소스로부터의 리디렉트 유형, 분기 타겟 및 EndAddr이 사용된다.

[0054] 다음의 리디렉트 사례들은 BTB 테이블 또는 BTB 테이블 세트 트레이닝을 필요로 한다. 사례 1: 예측이 이행된 분기가 예측에 사용된 BTB 엔트리 무효화에 대응하는 어떠한 분기와도 매칭되지 않았다. 사례 2: 어떠한 예측된 분기도 BTB에서 새로 찾아진 분기 트레이닝에 대응하지 않는다. 사례 3: 리디렉트된 분기 EndAddr이 어떠한 예측된 분기의 EndAddr와도 매칭되지 않고 BTB에서 새로 찾아진 분기 트레이닝에 대응한다. 사례 4: 리디렉트된 분기 EndAddr은 예측된 분기의 EndAddr과 매칭되지만 Type과 매칭되지 않고 BTB에서 분기 유형 업데이트에 대응한다. 이러한 네 번째 사례는 분기를 동적 또는 가변으로 마킹하기 위해 분기 유형이 변경되어야 하는 경우를 포함한다.

[0055] BTB 트레이닝 프로세스는 트레이닝 파이프를 통해 이루어진다. 각 스레드는 스레드에 대해 확인된 마지막 리디렉트에 대한 정보를 캡처하고 BTB 업데이트가 요청되었는지 여부를 나타내는 예측 실패 레지스터와 연관된다. 트레이닝 파이프는 요청들에 기초하여 두 스레드들 중에서 선택하고 스레드들 양자가 동시에 BTB 트레이닝을 요청할 때 발생하는 요청 충돌의 경우 라운드 로빈을 통해 중재한다. 일반적으로, 하나의 스레드에 대한 트레이닝은 다른 스레드가 이미 트레이닝 파이프에 있는 동안 시작할 수 있다. 그러나, 스레드는 다른 스레드가 현재 동일한 BTB 인덱스에 대해 트레이닝되고 있고 동일한 인덱스에서 제2 트레이닝에 의해 보일 수 있도록 BTB 테이블을 적시에 업데이트하지 않는 경우 선택될 수 없다. BTB 트레이닝은 추론에 의해 이루어지고 잘못된 경로 상에서 발생하는 리디렉트들은 BTB 업데이트드을 야기할 수 있다. 그러나, 예측 실패 레지스터가 덮어 쓰이면, 스레드에 대해 진행 중인 트레이닝은 취소된다.

[0056] 트레이닝 파이프는 트레이닝 요청이 승인된 사이클 BTN2에서 시작되고, 사이클 BTN1, 사이클 BT0, 사이클 BT1, 사이클 BT2, 사이클 BT3, 사이클 BT4가 이어지고 BTB가 쓰이는 사이클 BT5에서 완료된다. 일부 실시 예들에 따르면, 도 4의 제1 BTB 테이블(401)과 같은 제1 BTB 테이블만 새로운 또는 수정된 엔트리들로 업데이트된다. 트레이닝 파이프는 한 사이클 동안 예측 파이프를 지연시키는 트레인-체크-판독(train-check-read; TCR) 프로세스를 개시한다. TCR 프로세스의 한 가지 목적은 엔트리가 현재 레벨 1 BTB에 존재하는지 또는 트레이닝 조회 위치의 레벨 2 BTB 테이블들 중 하나에 존재하는지를 확인하는 것이다. 레벨 2 BTB 테이블에서 매치가 찾아지면, TCR 프로세스는 레벨 2 BTB와 레벨 1 BTB 간의 스왑을 트리거한 후, 트레이닝 플로우가 반복된다.

[0057] 레벨 2 BTB에서 적중하지 않는 TCR 프로세스에 따라, 다음과 같이 다양한 트레이닝 작업들이 핸들링된다. 무효화는 TCR이 적중할 때만 수행된다. 새로운 쓰기 및 업데이트의 경우, BTB 업데이트 트레이닝은 예측 실패된 분기를 BTB 엔트리 적중과 병합하거나, 새로운 BTB 엔트리를 생성하거나, 이러한 조치들 양자를 수행한다. 상이한 시나리오들은 다음과 같다. TCR 프로세스가 레벨 1 BTB에서 부적중되면, 이를테면 교체 정책에 기초하여 웨이를 선택하고 해당된다면 선택된 엔트리를 레벨 2 BTB로 축출함으로써 조회 어드레스에 새로운 BTB 엔트리가 생성된다. 일부 실시 예들에서, TCR 프로세스는 부적중을 고려하고 여기서 설명되는 바와 같이 후속 단계들을 수행할 때 세 개의 레벨들의 BTB 테이블들 중 세 개 모두의 엔트리들을 체크한다.

[0058] TCR 프로세스가 레벨 1 BTB에서 적중하면, 이를테면 BTB 테이블에서 적중이 발생하면, 조회 어드레스에 기존 BTB 엔트리가 있으며, 이는 이러한 BTB 엔트리에 이미 하나 또는 두 개의 분기가 있음을 의미한다. 트레이닝 작업이 업데이트인 경우, 예측 실패 EndAddr이 찾아진 분기들 중 하나와 매칭된다. 그렇지 않으면, 새로운 분기를 도입해야 한다. BTB에 다시 쓰여져야 하는 분기들은 최대 세 개이다. 이러한 분기들은 EndAddr에 기초하여 정렬되고 Pos0, Pos1 및 Pos2로 라벨링된다. 다음의 사례들은 다음과 같이 핸들링된다. 사례 1은 Pos0만 유효할 때 발생한다. 그 다음 Pos0 분기 트레이닝이 TCR 조회 어드레스에서 수행된다. 사례 2은 Pos0 및 Pos1만 유효할 때 발생한다. Pos0 및 Pos1 분기들이 쌍을 이룰 수 있다면, 쌍에 대한 트레이닝이 TCR 조회 어드레스에서 수행된다. 그렇지 않으면, Pos0만 TCR 조회 어드레스에서 예측 윈도우에서 “마지막이 아닌”(“non-last” in a prediction window; NLIPW) 것으로 트레이닝되고 Pos1이 예측 실패된 분기일 때 Pos1에 대한 새로운 트레이닝 플로우가 Pos0EndAddr + 1 조회 어드레스에서 개시된다. Pos1이 예측 실패된 분기가 아니라면, 트레이닝은 드롭된다. 사례 3은 Pos0, Pos1 및 Pos2가 유효할 때 발생한다. Pos0 및 Pos1 분기들이 쌍을 이룰 수 있다면, TCR 조회 어드레스에서의 쌍은 NLIPW로 트레이닝되고 Pos2가 예측 실패된 분기인 경우 Pos2에 대한 새로운 트레이닝 플로우가 Pos1EndAddr + 1 조회 어드레스에서 개시된다. Pos2가 예측 실패된 분기가 아니라면, 그것은 드롭된다. 그렇지 않으면, Pos0만 TCR 조회 어드레스에서 NLIPW로 트레이닝된다. Pos0가 예측 실패된 분기가 아니라면, 새로운 트레이닝 플로우가 예측 실패된 분기에 대한 Pos0EndAddr + 1 조회 어드레스에서 개시된다. 다시,

Pos1, Pos2 또는 Pos1과 Pos2 양자가 드롭된다.

[0059] 상술된 트레이닝 프로세스는 예측 실패된 분기 이전의 모든 분기들을 유지하고 TCR 플로우들을 이용하여 예측 실패된 분기를 인스톨하려고 시도한다. 각 리디렉트는 두 개 이하의 트레이닝 플로우를 트리거하도록 제한되므로 예측 실패된 분기가 트레이닝된다는 보장이 없다. 그러나, 엔트리들을 NLIPW로 마킹하면 다음 예측이 예측 윈도우를 여러 예측들로 분할하여 새로운 분기가 종내 트레이닝될 수 있게 한다. 각 조회 어드레스에서의 TCR 플로우는 레벨 1/레벨 2 BTB 스왑을 트리거하고 TCR 플로우가 재생되게 한다. 결과적으로, 소정의 리디렉트에 대해 최대 네 개의 TCR 플로우들이 발생할 수 있다.

[0060] 도 9는 일부 실시 예들에 따라 분기 타겟 어드레스를 타겟 어드레스 사이즈에 의해 복수의 BTB 테이블들 중 하나로 저장하기 위한 방법(900)이다. 방법(900)은 도 1에 도시된 명령어 파이프라인 아키텍처(100)의 일부 실시 예들에서, 도 2에 도시된 프로세싱 시스템(200)의 일부 실시 예들에서, 그리고 도 3의 프로세서(300)의 일부 실시 예들에서 구현된다. 예시된 실시 예에서, 프로세서 명령어들은 실행되고 있거나 도 3에 도시된 프로세서(300)의 프로세서 코어(305)와 같은 프로세싱 시스템의 하나 이상의 프로세서 코어 상에서 실행되도록 할당 또는 스케줄링된다. 여기서 논의된 바와 같이, 프로세서 명령어들은 분기 명령어를 포함한다.

[0061] 블록 901에서, BTB 트레이닝 로직은 타겟 어드레스를 분기 명령어의 예측 어드레스와 비교함으로써 명령어 타겟 어드레스 사이즈를 결정한다. 타겟 어드레스 사이즈에 기초하여, 방법은 예측된 타겟 어드레스를 저장할 복수의 사이즈 기반 BTB 테이블들 중 어느 하나를 결정 또는 선택하는 단계를 포함한다. 예를 들어, 블록 902에서, 프로세서는 타겟 사이즈가 제1 사이즈 "SIZE 1"보다 작거나 같은지 여부를 결정한다. 만약 그렇다면, 블록 903에서, 프로세서는 도 3의 제1 BTB 테이블(301)과 같은 사이즈 기반 BTB 테이블의 제1 사이즈 기반 BTB 엔트리에 BTB 엔트리를 저장한다. 그렇지 않다면, 블록(904)에서, 프로세서는 타겟 어드레스 사이즈가 3-사이즈 BTB 시스템에 대한 제2 사이즈 "SIZE 2"보다 작거나 같은지 여부를 결정한다. 만약 그렇다면, 블록 905에서, 프로세서는 분기 타겟을 제2 사이즈 기반 BTB 테이블에 저장한다. 예를 들어, 타겟 어드레스는 도 3의 제2 BTB 테이블(302)에 저장된다. 블록 904에서 분기 명령어 유형이 제2 사이즈 "SIZE 2"보다 작거나 같지 않으면, 블록 906에서 프로세서는 제3 사이즈 기반 BTB에 분기 타겟을 저장한다.

[0062] 도 10은 일부 실시 예들에 따라 분기 타겟 어드레스를 분기 유형에 의해 복수의 BTB 테이블들 중 하나로 저장하기 위한 방법을 나타내는 방법(1000)이다. 방법(1000)은 도 1에 도시된 명령어 파이프라인 아키텍처(100)의 일부 실시 예들에서, 도 2에 도시된 프로세싱 시스템(200), 도 4의 프로세서(400) 및 도 5의 프로세서(500)의 일부 실시 예들에서 구현된다. 예시된 실시 예에서, 프로세서 명령어들은 실행되고 있거나 도 4의 프로세서(400)의 프로세서 코어(405)와 같은 프로세싱 시스템의 하나 이상의 프로세서 코어 상에서 실행되도록 할당 또는 스케줄링된다. 여기서 논의된 바와 같이, 프로세서 명령어들은 분기 명령어를 포함한다.

[0063] 방법(1000)은 세 개의 BTB 테이블 시스템의 예시적인 상황에서 설명된다. 블록 1001에서, 프로세서는 조건부 명령어에 대한 BTB 엔트리 유형을 결정한다. 블록 1002에서, 프로세서는 분기 명령어 유형에 기초하여, 분기 명령어 유형이 BTB 테이블 시스템에 대한 높은 값 BTB 엔트리인지 여부를 결정한다. 만약 그렇다면, 블록 1003에서, 프로세서는 도 4의 BTB 테이블(401)과 같은 제1 BTB 테이블에 BTB 엔트리를 저장한다. 만약 그렇지 않다면, 블록 1004에서, 프로세서는 분기 명령어 유형이 중간 값 BTB 엔트리인지 여부를 결정한다. 만약 그렇다면, 블록 1005에서, 프로세서는 도 4의 BTB 테이블(402)과 같은 제2 BTB 테이블에 BTB 엔트리를 저장한다. 만약 그렇지 않다면, 블록 1006에서, 프로세서는 도 4의 BTB 테이블(403)과 같은 제3 BTB 테이블의 엔트리에 어드레스를 포함하여 BTB 엔트리를 저장한다.

[0064] 높은 값, 중간 값 등과 관련하여, 특정 유형의 조건 분기 명령어는 높은 값, 중간 값 또는 다른 유형의 BTB 엔트리일 수 있다. 예를 들어, 점프 유형이 높은 값 엔트리일 수 있고, 리턴 유형 명령어는 중간 값 엔트리일 수 있다. 일부 실시 예들에 따르면, 엔트리의 값은 엔트리를 유지하지 않는 것에 비해 또는 BTB 또는 BTB 테이블에 저장될 수 있는 다른 가능한 엔트리들에 비해 복수의 프로세서 사이클들을 절감하는 엔트리에 대응한다. 분기 명령어가 중요할 경우, 이를테면 향후 프로세스 사이클에서 유용할 경우, 프로세서는 분기 타겟을 BTB 테이블에 저장한다. 예를 들어, 타겟 어드레스는 도 4의 유형 BTB 테이블들(401-403) 중 하나에 저장된다. 명령어 유형이 유용한 분기 유형이 아닌 경우, 프로세서는 BTB 테이블 엔트리에 타겟 어드레스를 저장하지 않는다.

[0065] 여기에 개시된 바와 같이, 일부 실시 예들에서, 프로세서는: 명령어 실행 파이프라인; 제1 갯수의 제1 분기 타겟 버퍼(BTB) 테이블 엔트리들을 가지는 제1 BTB 테이블로서, 각 제1 BTB 테이블 엔트리는 제1 태그 및 제1 타겟 어드레스를 포함하며, 각 제1 타겟 어드레스는 제1 너비를 가지는, 상기 제1 BTB 테이블; 제2 갯수의 제2 BTB 테이블 엔트리들을 가지는 제2 BTB 테이블로서, 각 제2 BTB 테이블 엔트리는 제2 태그 [308] 및 제2 타겟

어드레스를 포함하며, 각 제2 타겟 어드레스는 상기 제1 너비와 상이한 제2 너비를 가지는, 상기 제2 BTB 테이블; 소정의 예측 어드레스에 대해 예측되는 타겟 어드레스를 제공하도록 구성된 분기 예측기를 포함하며; 상기 프로세서는 분기 명령어의 분기 특성에 기초하여 상기 제1 BTB 테이블의 제1 BTB 테이블 엔트리 또는 상기 제2 BTB 테이블의 제2 BTB 테이블 엔트리 중 어느 하나에 상기 제1 타겟 어드레스를 포함하는 분기 기술어들(branch descriptors)을 저장하도록 구성된다. 일 양태에서, 상기 분기 특성은 분기 타겟 어드레스 사이즈이다. 다른 양태에서, 상기 분기 특성은 분기 유형이다. 또 다른 양태에서, 상기 제1 BTB 테이블 엔트리들은 점프 분기 명령어, 호출 분기 명령어, 리턴 분기 명령어 및 조건 분기 명령어 중 적어도 하나에 대해 구성된다.

[0066] 일 양태에서, 상기 분기 특성은 상기 BTB 엔트리에 저장된 분기들의 갯수이다. 다른 양태에서, 상기 분기 특성은 상기 BTB 엔트리의 스레드 식별자이다. 또 다른 양태에서, 상기 프로세서는: 제1 메모리 레벨 캐시 및 제2 메모리 레벨 캐시를 포함하며; 상기 제1 BTB 테이블 및 상기 제2 BTB 테이블은 상기 프로세서의 동일한 메모리 레벨 캐시에 포함된다. 또 다른 양태에서, 상기 제1 BTB 테이블의 상기 제1 BTB 테이블 엔트리들은 예측 조회에 대해 N-웨이 연관(N-way associative)이고; 상기 제2 BTB 테이블의 상기 제2 BTB 테이블 엔트리들은 예측 조회에 대해 M-웨이 연관이며, 여기서 M 및 N은 1 이상이고 M과 N은 상이하다.

[0067] 다른 양태에서, 상기 제1 BTB 테이블 엔트리들의 제1 갯수는 상기 제2 BTB 테이블 엔트리들의 제2 갯수와 상이하다. 또 다른 양태에서, 상기 BTB 테이블 엔트리들 각각은 세트 내에서 최근에 가장 많이 사용된(most recently used; MRU) 상태 및 최근에 가장 적게 사용된(least recently used; LRU) 상태 중 하나로서 상기 BTB 엔트리 각각을 마킹하는 최근 사용된 상태 비트를 포함하고; 상기 프로세서는 상기 최근 사용된 상태 비트의 상태에 기초하여 교체 정책에 따라 세트 내에서 BTB 엔트리들을 축출하도록 구성된다.

[0068] 여기에 개시된 바와 같이, 일부 실시 예들에서, 방법은: 분기 명령어에 대한 예측 실패로 인해 야기되는 리다이렉트들에 기초하여 분기 타겟 버퍼(BTB) 엔트리의 분기 유형을 결정하는 단계; 및 결정된 상기 BTB 엔트리의 분기 유형에 기초하여, 상기 BTB 엔트리를 다음: 프로세서의 제1 분기 타겟 버퍼(BTB) 테이블의 제1 엔트리로서, BTB 엔트리의 제1 분기 유형에 대응하는 제1 타겟 어드레스 너비를 가지는, 상기 제1 엔트리; 및 상기 프로세서의 제2 BTB 테이블의 제2 엔트리로서, 제1 분기 유형과 상이한 제2 분기 유형에 대응하는 상기 제1 타겟 어드레스 너비와 상이한 제2 타겟 어드레스 너비를 가지는, 상기 제2 엔트리 중 하나에 저장하는 단계를 포함한다. 일 양태에서, 상기 분기 유형은 분기 타겟 어드레스 사이즈이다. 다른 양태에서, 상기 BTB 엔트리의 상기 저장은 프로세서의 예측 트레이너 유닛에 의해 수행된다.

[0069] 일 양태에서, 상기 방법은: 세트 내에서 최근에 가장 많이 사용된(MRU) 상태 및 최근에 가장 적게 사용된(LRU) 상태 중 하나로서 상기 BTB 엔트리를 마킹하는 최근 사용된 상태 비트에 기초하여 상기 제1 BTB 테이블 또는 상기 제2 BTB 테이블의 BTB 엔트리를 식별하는 단계; 및 상기 엔트리를 내부에 저장하기 전에 상기 제1 BTB 테이블 또는 상기 제2 BTB 테이블의 식별된 상기 BTB 엔트리를 축출하는 단계를 포함한다. 다른 양태에서, 상기 제1 BTB 테이블 및 상기 제2 BTB 테이블은 상기 프로세서의 동일한 메모리 레벨 캐시에 포함된다. 또 다른 양태에서, 상기 방법은: 상기 BTB 엔트리를 저장하기 전에, 상기 분기 명령어의 명령어 태그에 기초하여 상기 제1 BTB 테이블 및 상기 제2 BTB 테이블 중 적어도 하나를 탐색하는 단계; 및 상기 BTB 엔트리를 저장하기 전에, 상기 제1 BTB 테이블 및 상기 제2 BTB 테이블 중 상기 적어도 하나에서 BTB 엔트리가 찾아지지 않음을 식별하는 단계를 포함한다. 또 다른 양태에서, 상기 방법은: 상기 BTB 엔트리의 타겟 어드레스의 사이즈가 상기 제1 BTB 테이블 또는 상기 제2 BTB 테이블 각각의 BTB 엔트리들에 대한 어드레스 사이즈를 초과할 때 상기 BTB 엔트리의 상기 타겟 어드레스의 오버플로우 비트들(overflow bits)을 오버플로우 BTB 테이블에 저장하는 단계를 포함한다.

[0070] 여기에 개시된 바와 같이, 일부 실시 예들에서, 방법은: 프로세서의 BTB의 복수의 분기 타겟 버퍼(BTB) 테이블들을 조회하여 현재 예측되는 블록의 종료 어드레스 및 다음 예측되는 블록의 시작 어드레스를 예측하는 단계로서, 상기 BTB는 현재 예측 어드레스에서의, 제1 BTB 테이블 및 제2 BTB 테이블을 포함하는, 상기 예측하는 단계; 및 예측된 상기 종료 어드레스 및 예측된 상기 시작 어드레스 중 적어도 하나에 기초하여 상기 프로세서에서 실행하기 위한 명령의 일부로서 예측을 제공하는 단계를 포함한다. 일 양태에서, 상기 제1 BTB 테이블의 BTB 엔트리들은 상기 제2 BTB 테이블의 BTB 엔트리들의 타겟 어드레스 필드의 제2 너비보다 작은 제1 너비의 타겟 어드레스 필드를 가진다. 다른 양태에서, 상기 현재 예측 어드레스는 프로그램 카운터를 기준으로 한다.

[0071] 일부 실시 예들에서, 상술된 장치 및 기술들은 도 1 내지 도 7을 참조하여 상술된 시스템들, 프로세서들 및 BTB 테이블들과 같이, 하나 이상의 집적 회로(IC) 디바이스(집적 회로 패키지 또는 마이크로 칩이라고도 함)를 포함하는 시스템으로 구현된다. 이러한 IC 디바이스들의 설계 및 제조에는 전자 설계 자동화(EDA) 및 컴퓨터 이용

설계(CAD) 소프트웨어 툴들이 사용될 수 있다. 이러한 설계 툴들은 통상적으로 하나 이상의 소프트웨어 프로그램으로서 나타내어진다. 하나 이상의 소프트웨어 프로그램은 제조 시스템을 회로를 제조하도록 설계 또는 적응시키기 위해 프로세스의 적어도 일부를 수행하기 위해 컴퓨터 시스템을 조작하여 하나 이상의 IC 디바이스의 회로를 나타내는 코드에 작용하기 위해 컴퓨터 시스템에 의해 실행 가능한 코드를 포함한다. 이러한 코드는 명령어들, 데이터, 또는 명령어들 및 데이터의 조합을 포함할 수 있다. 설계 툴 또는 제조 툴을 나타내는 소프트웨어 명령들은 통상적으로 컴퓨팅 시스템에 액세스 가능한 컴퓨터 판독 가능한 저장 매체에 저장된다. 마찬가지로, IC 디바이스의 설계 또는 제조의 하나 이상의 단계를 나타내는 코드가 동일한 컴퓨터 판독 가능한 저장 매체 또는 상이한 컴퓨터 판독 가능한 저장 매체에 저장되고 그로부터 액세스될 수 있다.

[0072] 컴퓨터 판독 가능한 저장 매체는 컴퓨터 시스템에 명령어들 및/또는 데이터를 제공하기 위해 사용 중 컴퓨터 시스템에 의해 액세스 가능한 임의의 비일시적 저장 매체, 또는 비일시적 저장 매체들의 조합을 포함할 수 있다. 이러한 저장 매체들은 광 매체들(예를 들어, 컴팩트 디스크(CD), 디지털 다목적 디스크(DVD), 블루-레이 디스크), 자기 매체들(예를 들어, 플로피 디스크, 자기 테이프 또는 자기 하드 드라이브), 휘발성 메모리(예를 들어, 랜덤 액세스 메모리(RAM) 또는 캐시), 비휘발성 메모리(예를 들어, 판독 전용 메모리(ROM) 또는 플래시 메모리) 또는 미세 전자 기계 시스템(MEMS) 기반 저장 매체들을 포함할 수 있으나, 이에 제한되지 않는다. 컴퓨터 판독 가능한 저장 매체는 컴퓨팅 시스템(예를 들어, 시스템 RAM 또는 ROM)에 내장되거나, 컴퓨팅 시스템(예를 들어, 자기 하드 드라이브)에 고정적으로 부착되거나, 컴퓨팅 시스템(예를 들어, 광 디스크 또는 범용 직렬 버스(USB) 기반 플래시 메모리)에 착탈 가능하게 부착되거나, 또는 유선 또는 무선 네트워크(예를 들어, 네트워크 액세스 가능한 스토리지(NAS))를 통해 컴퓨터 시스템에 결합될 수 있다.

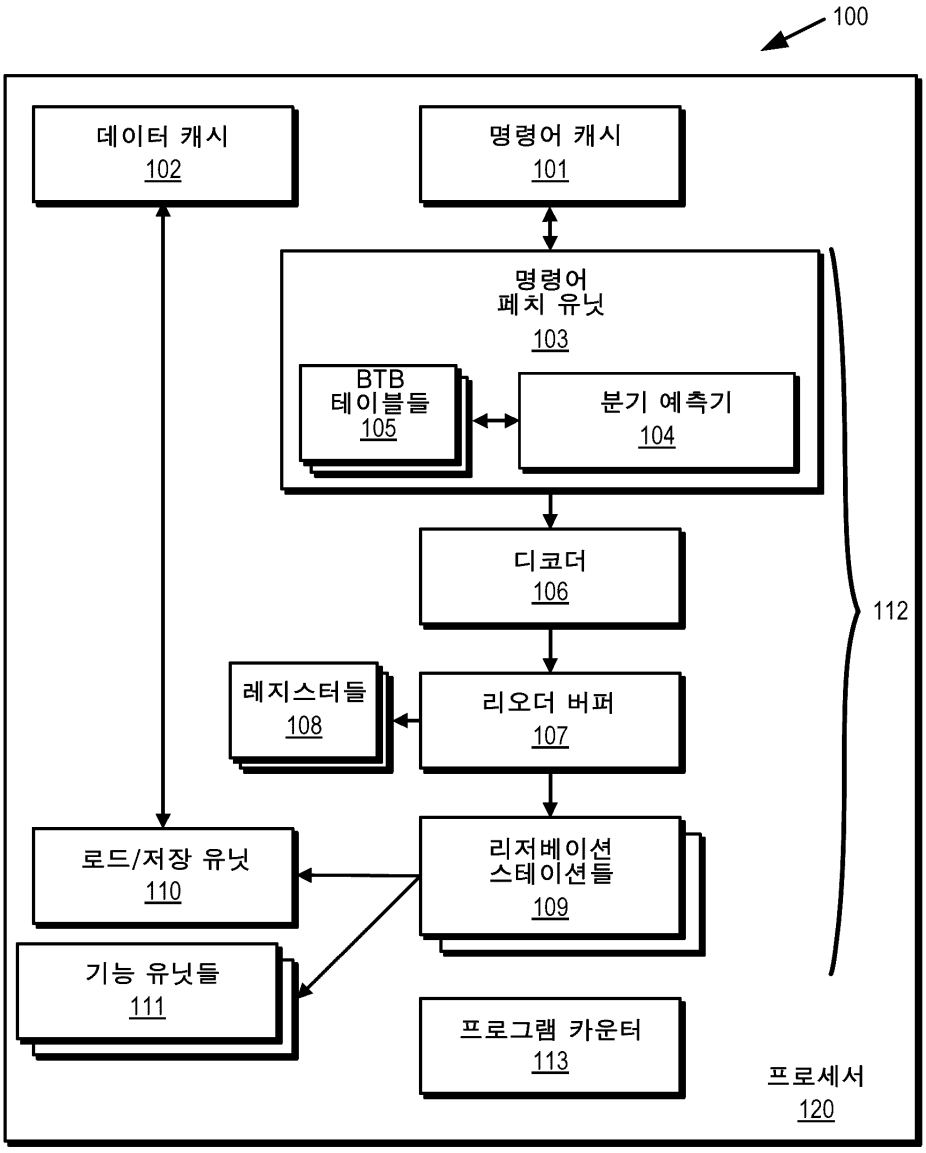
[0073] 일부 실시 예들에서, 상술된 기술들의 특정 양태들은 소프트웨어를 실행하는 프로세싱 시스템의 하나 이상의 프로세서에 의해 구현될 수 있다. 소프트웨어는 비일시적 컴퓨터 판독 가능한 저장 매체 상에 저장되거나 그 외 다르게 유형으로 구현되는 실행 가능한 명령들의 하나 이상의 세트를 포함한다. 소프트웨어는 하나 이상의 프로세서에 의해 실행될 때, 하나 이상의 프로세서를 조작하여 상술된 기술들의 하나 이상의 양태를 수행하는 명령어들 및 특정 데이터를 포함할 수 있다. 비일시적 컴퓨터 판독 가능한 저장 매체는 예를 들어, 자기 또는 광 디스크 저장 디바이스, 플래시 메모리와 같은 고체 상태 저장 디바이스들, 캐시, 랜덤 액세스 메모리(RAM) 또는 다른 비휘발성 메모리 디바이스 또는 디바이스들 등을 포함할 수 있다. 비일시적 컴퓨터 판독 가능한 저장 매체 상에 저장된 실행 가능한 명령어들은 소스 코드, 어셈블리 언어 코드, 객체 코드, 또는 하나 이상의 프로세서에 의해 해석되거나 그 외 다르게 실행 가능한 다른 명령어 포맷으로 있을 수 있다.

[0074] 일반적인 설명으로 상술된 모든 동작들 또는 요소들이 필수적인 것은 아니라는 것, 특정 동작 또는 디바이스의 일 부분이 요구되지 않을 수 있다는 것, 그리고 설명된 동작들 또는 요소들에 더하여 하나 이상의 추가 동작이 수행되거나 요소들이 포함될 수 있다. 더 나아가, 동작들이 나열된 순서가 반드시 그것들이 수행되는 순서는 아니다. 또한, 개념들은 구체적인 실시 예들을 참조하여 설명되었다. 그러나, 해당 기술분야의 통상의 기술자는 아래 청구범위에 제시된 바에 따라 본 개시의 범위에서 벗어나지 않고 다양하게 수정 및 변경될 수 있다는 것을 이해한다. 따라서, 본 명세서 및 도면들은 제한적인 의미가 아니라 예시적인 의미로 여겨져야 하고, 상기한 모든 수정은 본 개시의 범위 내에 포함되는 것으로 의도된다.

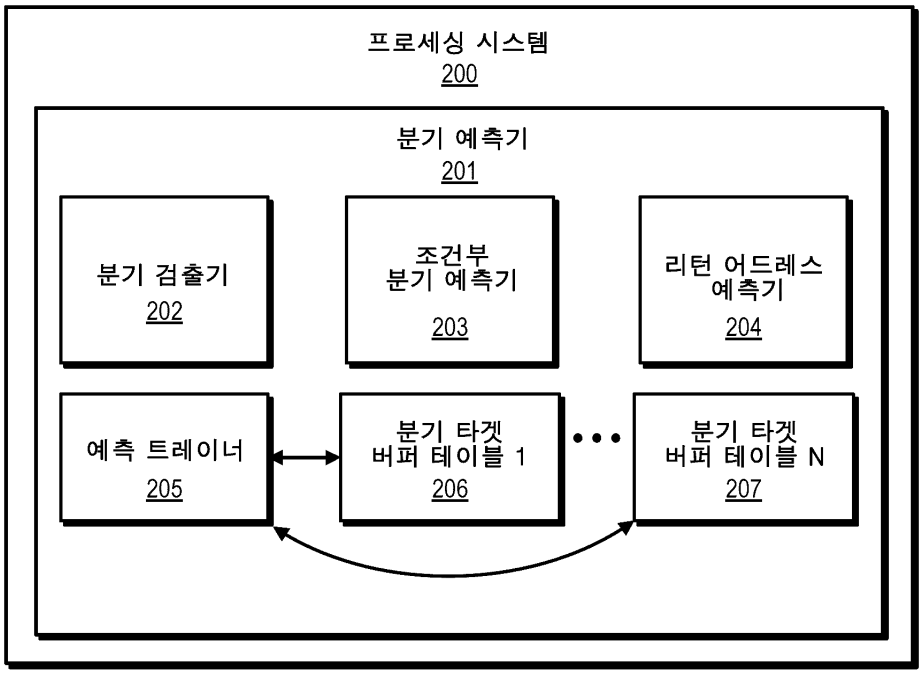
[0075] 혜택들, 다른 이점들 및 문제들에 대한 솔루션들이 구체적인 실시 예들과 관련하여 상술되었다. 그러나, 혜택들, 이점들, 문제들에 대한 솔루션들 및 임의의 혜택, 이점, 또는 솔루션이 발생시키거나 더 확연히 드러낼 수 있는 임의의 특징(들)이 임의의 또는 모든 청구항의 임계적, 필수적 또는 본질적 특징인 것으로 간주되지는 않아야 한다. 또한, 위에서 개시된 특정 실시 예들은 개시된 주제가 여기서의 교시 내용의 혜택을 받는 해당 기술분야의 통상의 기술자들에게 분명한 상이하지만 균등한 방식들로 수정 및 실시될 수 있음에 따라 단지 예시적인 것이다. 아래 청구범위에서 설명되는 것 이외에, 제시되는 여기서의 구성 또는 설계의 세부 사항들로 제한되도록 의도되지 않는다. 따라서, 위에서 개시된 특정 실시 예들은 대체 또는 수정될 수 있고 모든 그러한 변형 예들은 개시된 주제의 범위 내인 것으로 고려된다. 따라서, 여기서 추구되는 보호는 아래 청구범위에 제시된 바에 따른다.

도면

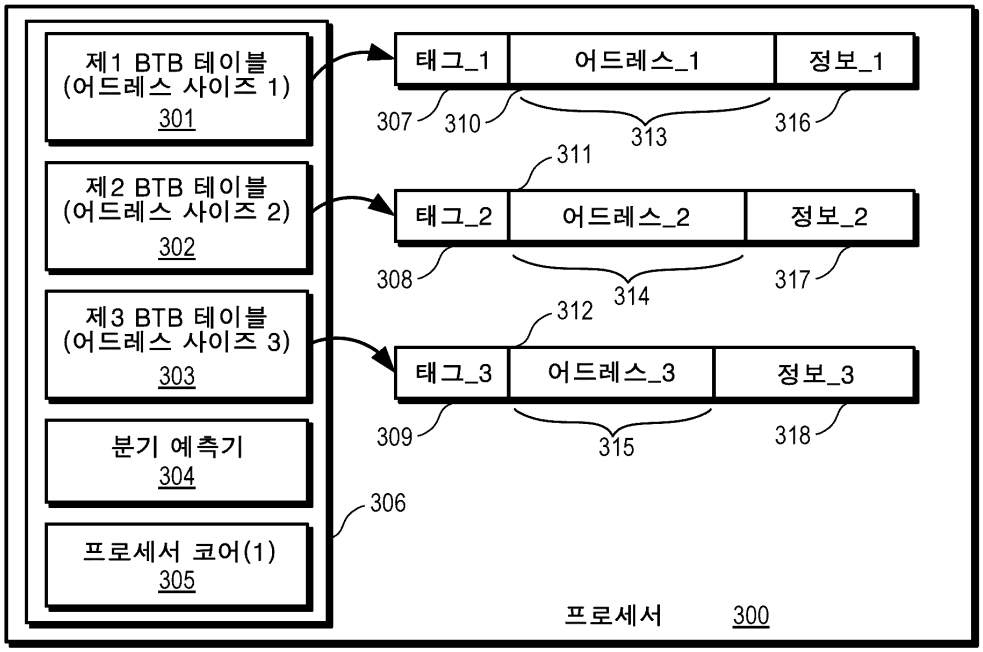
도면1


100
데이터 캐시
102
명령어 캐시
101
명령어
페치 유닛
103
BTB
테이블들
105
분기 예측기
104
디코더
106
112
레지스터들
108
리오더 버퍼
107
로드/저장 유닛
110
리저베이션
스테이션들
109
기능 유닛들
111
프로그램 카운터
113
프로세서
120

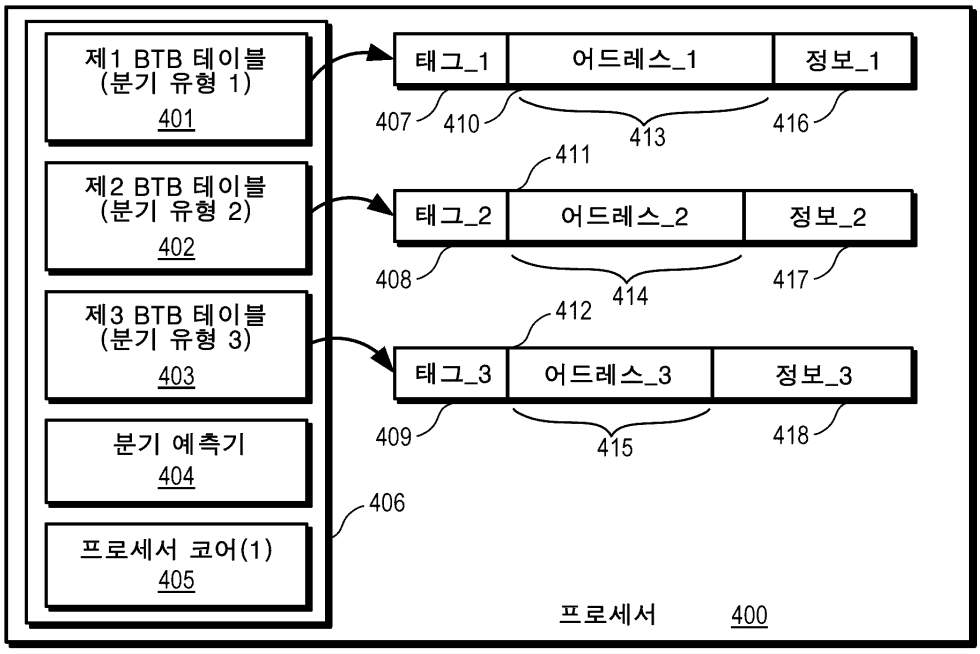
도면2


프로세싱 시스템
200
분기 예측기
201
분기 검출기
202
조건부
분기 예측기
203
리턴 어드레스
예측기
204
예측 트레이너
205
분기 타겟
버퍼 테이블 1
206
분기 타겟
버퍼 테이블 N
207

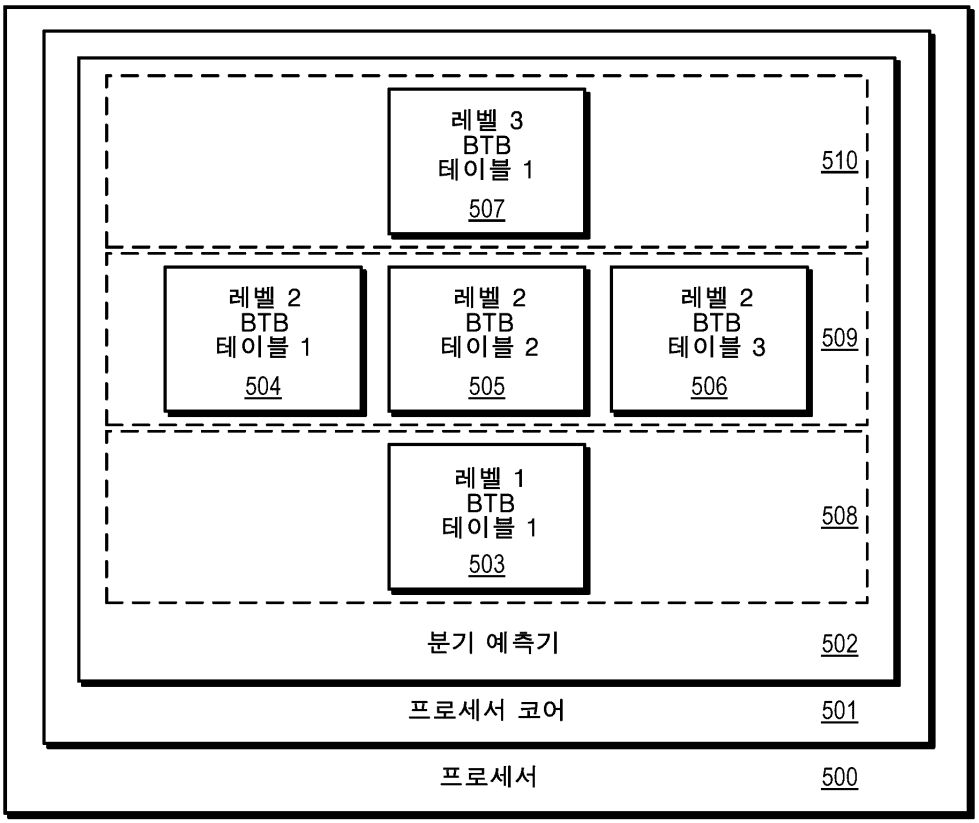
도면3


제1 BTB 테이블
(어드레스 사이즈 1)
301
제2 BTB 테이블
(어드레스 사이즈 2)
302
제3 BTB 테이블
(어드레스 사이즈 3)
303
분기 예측기
304
프로세서 코어(1)
305
306
태그_1
어드레스_1
정보_1
307
310
313
316
311
태그_2
어드레스_2
정보_2
308
314
317
312
태그_3
어드레스_3
정보_3
309
315
318
프로세서
300

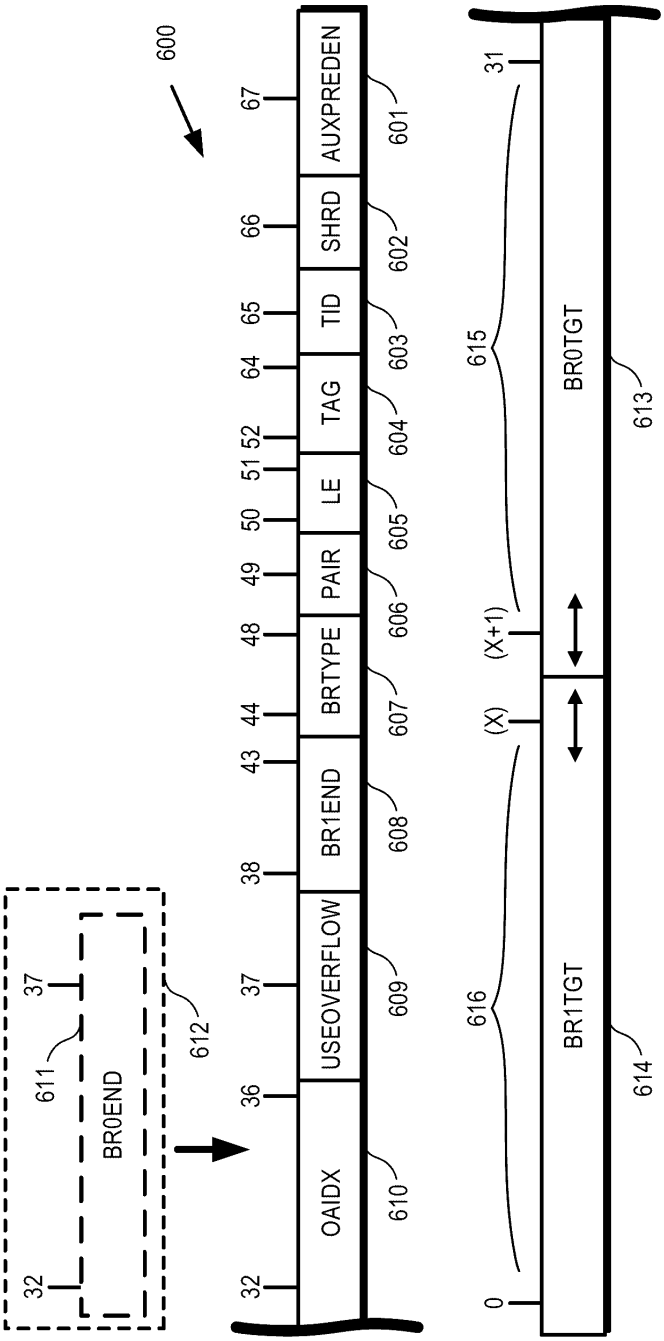
도면4


제1 BTB 테이블
(분기 유형 1)
401
제2 BTB 테이블
(분기 유형 2)
402
제3 BTB 테이블
(분기 유형 3)
403
분기 예측기
404
프로세서 코어(1)
405
406
태그_1
어드레스_1
정보_1
407
410
413
416
411
태그_2
어드레스_2
정보_2
408
414
417
412
태그_3
어드레스_3
정보_3
409
415
418
프로세서
400

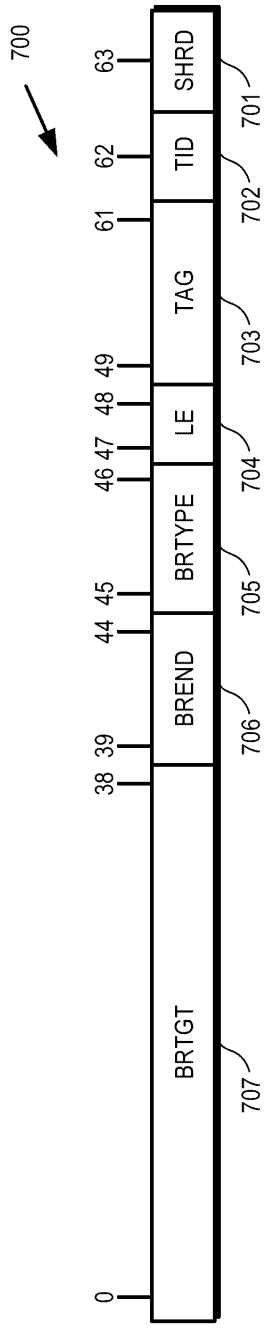
도면5


레벨 3
BTB
테이블 1
507
510
레벨 2
BTB
테이블 1
504
레벨 2
BTB
테이블 2
505
레벨 2
BTB
테이블 3
506
509
레벨 1
BTB
테이블 1
503
508
분기 예측기
502
프로세서 코어
501
프로세서
500

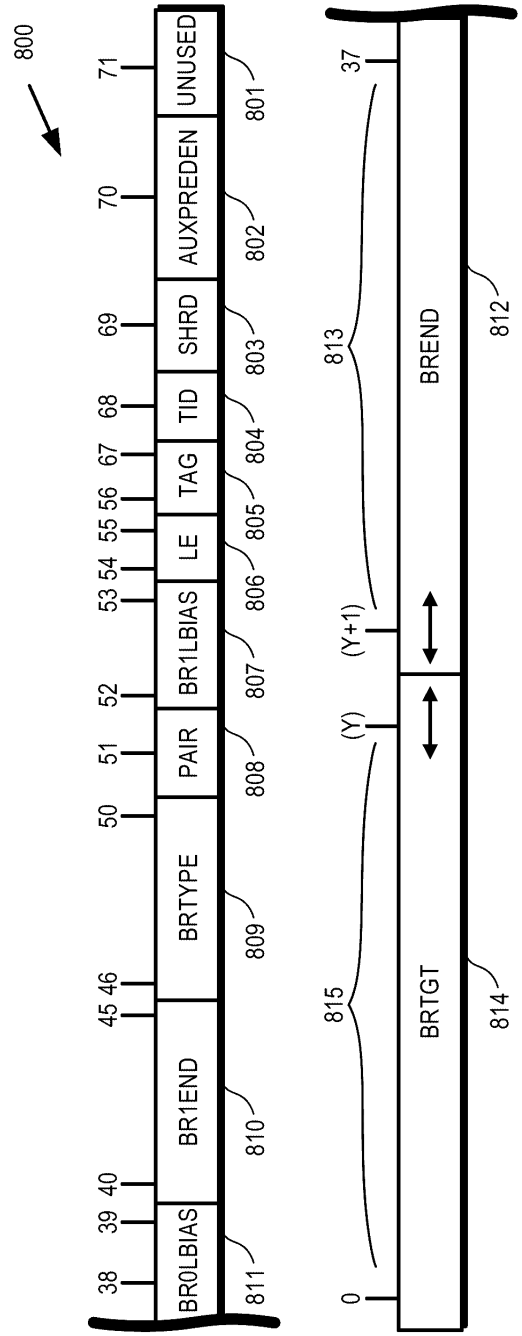
도면6


600
67
66
65
64
52
51
50
49
48
44
43
38
37
36
32
AUXPREDEN
SHRD
TID
TAG
LE
PAIR
BRTYPE
BR1END
USEOVERFLOW
OAIDX
601
602
603
604
605
606
607
608
609
610
612
611
37
32
BR0END
31
(X+1)
(X)
0
615
616
BR0TGT
BR1TGT
613
614

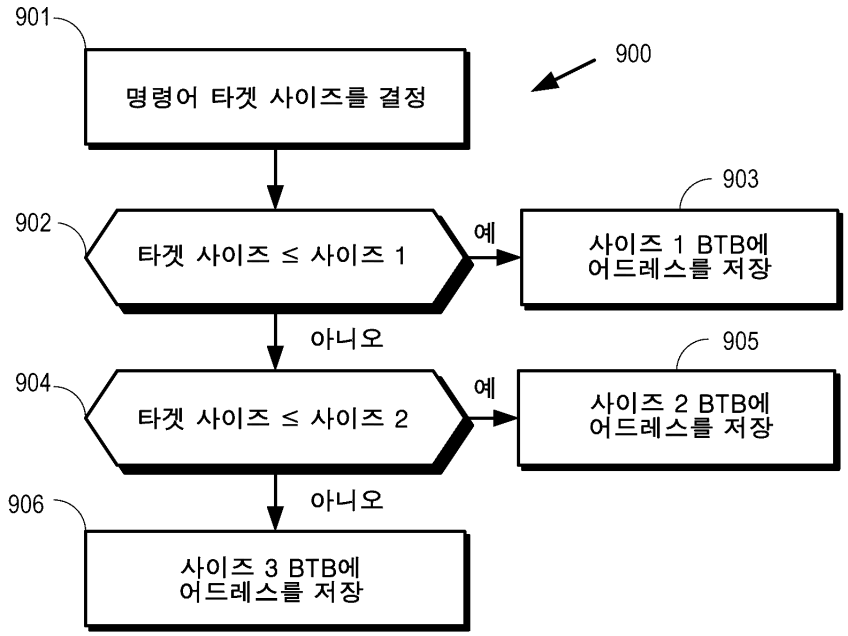
도면7


700
63
62
61
49
48
47
46
45
44
39
38
0
SHRD
TID
TAG
LE
BRTYPE
BREND
BRTGT
701
702
703
704
705
706
707

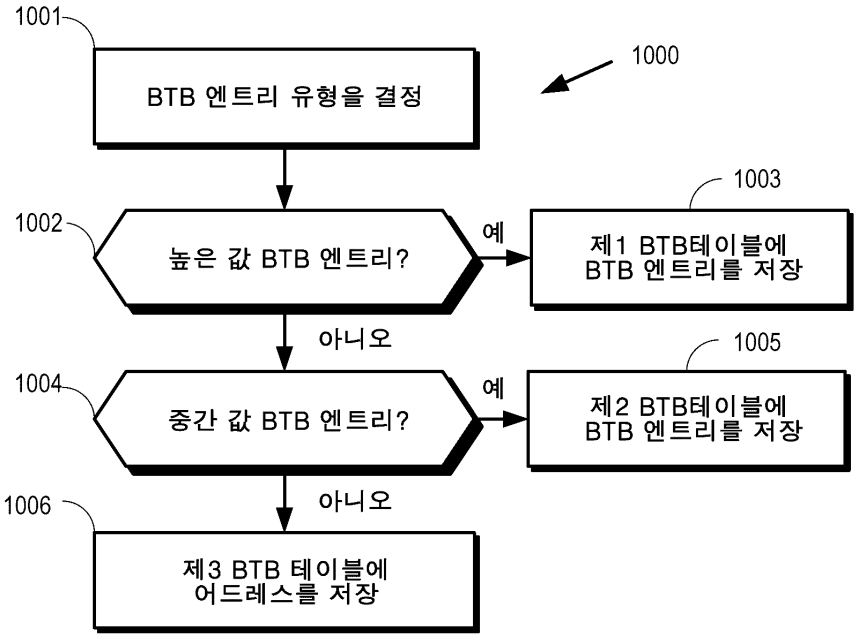
도면8


800
71
70
69
68
67
56
55
54
53
52
51
50
46
45
40
39
38
UNUSED
AUXPREDEN
SHRD
TID
TAG
LE
BR1LBIAS
PAIR
BRTYPE
BR1END
BR0LBIAS
801
802
803
804
805
806
807
808
809
810
811
37
(Y+1)
(Y)
0
BREND
BRTGT
812
813
814
815

도면9


901
명령어 타겟 사이즈를 결정
900
902
타겟 사이즈 ≤ 사이즈 1
예
903
사이즈 1 BTB에
어드레스를 저장
아니오
904
타겟 사이즈 ≤ 사이즈 2
예
905
사이즈 2 BTB에
어드레스를 저장
아니오
906
사이즈 3 BTB에
어드레스를 저장

도면10


1001
BTB 엔트리 유형을 결정
1000
1002
높은 값 BTB 엔트리?
예
1003
제1 BTB테이블에
BTB 엔트리를 저장
아니오
1004
중간 값 BTB 엔트리?
예
1005
제2 BTB테이블에
BTB 엔트리를 저장
아니오
1006
제3 BTB 테이블에
어드레스를 저장