(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2017년10월20일
(11) 등록번호 10-1788683
(24) 등록일자 2017년10월16일

(51) 국제특허분류(Int. Cl.)
G06F 12/08 (2016.01) *G06F 9/30* (2017.01)
G06F 9/32 (2006.01) *G06F 9/345* (2006.01)
G06F 9/38 (2006.01)
(52) CPC특허분류
G06F 12/0862 (2013.01)
G06F 9/30065 (2013.01)
(21) 출원번호 10-2015-7021641
(22) 출원일자(국제) 2014년01월18일
심사청구일자 2017년02월22일
(85) 번역문제출일자 2015년08월11일
(65) 공개번호 10-2015-0110588
(43) 공개일자 2015년10월02일
(86) 국제출원번호 PCT/US2014/012152
(87) 국제공개번호 WO 2014/113741
국제공개일자 2014년07월24일
(30) 우선권주장
13/746,000 2013년01월21일 미국(US)
(56) 선행기술조사문헌
US6775765 B1
US6260116 B1

(73) 특허권자
퀄컴 인코포레이티드
미국 92121-1714 캘리포니아주 샌 디에고 모어하우스 드라이브 5775
(72) 발명자
길버트, 매튜 엠.
미국 92121 캘리포니아주 샌 디에고 모어하우스 드라이브 5775
(74) 대리인
특허법인 남앤드남

전체 청구항 수 : 총 20 항 심사관 : 임정묵

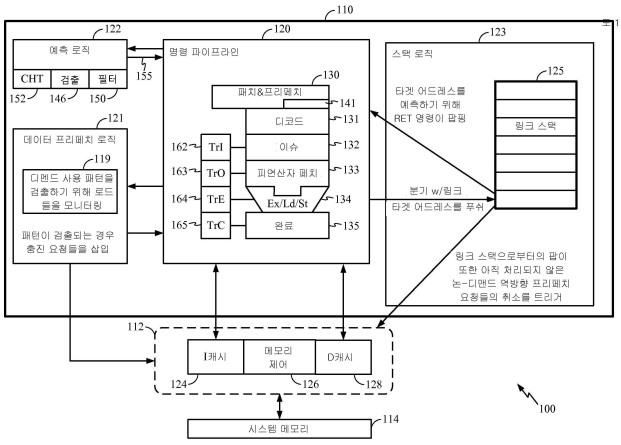
(54) 발명의 명칭 **루프에 대한 데이터 프리페치 요청들을 취소하기 위한 방법들 및 장치**

(57) 요 약

소프트웨어 루프들로부터의 탈출들을 인식하는 프리페치 로직 또는 임의의 펜딩 프리페치 요청 동작들을 취소하는 기능 복귀의 사용에 의해 캐시 공해를 감소시키기 위한 효율적인 기술이 설명된다. 프리페치 로직은, 프로그램 루프 내 메모리 액세스 명령의 반복 실행에 기초하여 데이터 액세스 스트라이드를 결정하기 위한 루프 데이터

(뒷면에 계속)

대 표 도 - 도1



어드레스 모니터를 포함한다. 데이터 프리페치 로직은 이후, 데이터 액세스 스트라이드에 따른 프리페치 명령들을 추론적으로 이슈한다. 정지 프리페치 회로는 식별된 루프 탈출에 응답하여 펜딩 프리페치 요청들을 취소하기 위해 사용된다. 프리페치 로직은 또한, 호출된 기능으로부터의 복귀를 인식하고, 호출된 기능과 연관된 임의의 펜딩 프리페치 요청 동작들을 취소할 수 있다. 프리페치 요청들이 취소되는 경우, 로드 명령들에 기초하는 것과 같은 디맨드 요청들은 취소되지 않는다. 캐시 공해를 감소시키기 위한 이 접근법은 데이터 캐시 프리페칭을 스로틀링(throttle)하기 위해 프로그램 흐름 정보를 사용한다.

(52) CPC특허분류
G06F 9/325 (2013.01)
G06F 9/3455 (2013.01)
G06F 9/383 (2013.01)
G06F 9/3832 (2013.01)

명 세 서

청구범위

청구항 1

프리페치 요청들을 취소하기 위한 방법으로서,

프로그램 루프에서의 메모리 액세스 명령들에 응답하여 캐시 프리페치 요청(cache prefetch request)들을 이슈(issue)하는 단계;

프로그램 흐름 정보의 평가에 기초하여 상기 프로그램 루프에서 루프 탈출 상황(loop exit situation)을 식별하는 단계; 및

상기 식별된 루프 탈출 상황에 응답하여 펜딩(pending) 캐시 프리페치 요청들을 취소하는 단계를 포함하는, 프리페치 요청들을 취소하기 위한 방법.

청구항 2

제 1 항에 있어서,

취소되는 상기 펜딩 캐시 프리페치 요청들은 상기 프로그램 루프에 기존의 디맨드(demand) 요청들을 포함하지 않는, 프리페치 요청들을 취소하기 위한 방법.

청구항 3

제 1 항에 있어서,

상기 루프 탈출 상황은 상기 프로그램 루프를 탈출하도록 평가하는 루프 종료 분기(loop ending branch)를 식별하는 것에 기초하는, 프리페치 요청들을 취소하기 위한 방법.

청구항 4

제 1 항에 있어서,

상기 루프 탈출 상황은 추론적인(speculative) 명령 페치 및 실행이 취소되도록 초래한 부정확한 분기 예측에 기초하는, 프리페치 요청들을 취소하기 위한 방법.

청구항 5

제 1 항에 있어서,

조건 분기 명령이 상기 루프 탈출 상황으로서 상기 프로그램 루프를 종료시키도록 결정하였음을 검출하는 단계; 및

상기 프로그램 루프에서 디맨드 프리페치 요청들을 제외한 펜딩 프리페치 요청들을 취소하는 단계를 더 포함하는, 프리페치 요청들을 취소하기 위한 방법.

청구항 6

제 1 항에 있어서,

조건 분기 명령이 상기 프로그램 루프를 종료시키도록 결정하지 않았음을 검출하는 단계를 더 포함하고, 상기 루프 탈출 상황에 대한 모니터링이 계속되는, 프리페치 요청들을 취소하기 위한 방법.

청구항 7

프리페치 요청들을 취소하기 위한 방법으로서,

프로그램 루프에서 호출되는 기능에 따라 데이터 프리페치 요청들을 이용하여 데이터를 추론적으로 프리페칭하는 단계; 및

상기 호출된 기능으로부터의 기능 탈출에 응답하여 펜딩 데이터 프리페치 요청들을 취소하는 단계를 포함하는, 프리페치 요청들을 취소하기 위한 방법.

청구항 8

제 7 항에 있어서,

상기 프로그램 루프 내의 기존 디맨드 요청들이 취소되지 않는, 프리페치 요청들을 취소하기 위한 방법.

청구항 9

제 7 항에 있어서,

상기 기능 탈출은 상기 호출된 기능으로부터의 복귀의 실행을 검출하는 것에 기초하는, 프리페치 요청들을 취소하기 위한 방법.

청구항 10

제 7 항에 있어서,

상기 기능 탈출은 상기 호출된 기능으로부터의 복귀의 추론적인 실행을 검출하는 것에 기초하는, 프리페치 요청들을 취소하기 위한 방법.

청구항 11

프리페치 요청들을 취소하기 위한 장치로서,

프로그램 루프에서 메모리 액세스 명령의 반복 실행에 기초하여 데이터 액세스 스트라이드(stride)를 결정하도록 구성된 루프 데이터 어드레스 모니터;

상기 데이터 액세스 스트라이드에 따라 상기 프로그램 루프에서의 메모리 액세스 명령들에 기초하여 프리페치 요청들을 추론적으로 이슈하도록 구성된 데이터 프리페치 로직; 및

식별된 루프 탈출에 응답하여 펜딩 프리페치 요청들을 취소하도록 구성된 정지 프리페치 회로를 포함하는, 프리페치 요청들을 취소하기 위한 장치.

청구항 12

제 11 항에 있어서,

상기 루프 데이터 어드레스 모니터는,

상기 메모리 액세스 명령의 각각의 실행을 위해 피연산자(operand) 어드레스에서의 차를 결정하기 위해 상기 메모리 액세스 명령의 반복 실행을 모니터링하도록 구성된 스트라이드 회로 – 상기 피연산자 어드레스에서의 차는 스트라이드 어드레스 값임 –; 및

다음 피연산자 어드레스를 결정하기 위해 가장 최근에 실행된 메모리 액세스 명령의 상기 피연산자 어드레스에 상기 스트라이드 어드레스 값을 가산하도록 구성된 가산 기능 회로를 포함하는, 프리페치 요청들을 취소하기 위한 장치.

청구항 13

제 11 항에 있어서,

상기 프로그램 루프 내의 기존 디맨드 요청들이 취소되지 않는, 프리페치 요청들을 취소하기 위한 장치.

청구항 14

제 11 항에 있어서,

상기 식별된 루프 탈출은, 상기 프로그램 루프를 탈출하도록 평가하는 루프 종료 분기를 식별하는 것에 기초하는, 프리페치 요청들을 취소하기 위한 장치.

청구항 15

제 11 항에 있어서,

상기 식별된 루프 탈출은, 추론적 명령 페치 및 실행을 취소하는 부정확한 분기 예측에 기초하는, 프리페치 요청들을 취소하기 위한 장치.

청구항 16

제 11 항에 있어서,

상기 정지 프리페치 회로는 추가로, 조건 분기 명령이 상기 식별된 루프 탈출로서 상기 프로그램 루프를 종료시키도록 결정하였음을 검출하고 그리고 디맨드 프리페치 요청들을 제외한 펜딩 프리페치 요청들을 취소하도록 구성되는, 프리페치 요청들을 취소하기 위한 장치.

청구항 17

제 11 항에 있어서,

상기 정지 프리페치 회로는 추가로, 조건 분기 명령이 상기 프로그램 루프를 종료시키도록 결정하지 않았음을 검출하도록 구성되고, 그리고 상기 프로그램 루프는 상기 식별된 루프 탈출이 식별될 때까지 계속되는, 프리페치 요청들을 취소하기 위한 장치.

청구항 18

제 11 항에 있어서,

상기 정지 프리페치 회로는 추가로, 약하게 예측된 루프 탈출에 기초하여 상기 프로그램 루프에서 펜딩 프리페치 요청들을 취소하지 않도록 구성되는, 프리페치 요청들을 취소하기 위한 장치.

청구항 19

컴퓨터-판독가능 프로그램 데이터 및 코드로 인코딩된 비-일시적 컴퓨터-판독가능 저장 매체로서,

상기 컴퓨터-판독가능 프로그램 데이터 및 코드는 실행되는 경우,

프로그램 루프에서의 메모리 액세스 명령들에 응답하여 캐시 프리페치 요청들을 이슈하고;

프로그램 흐름 정보에 기초하여 상기 프로그램 루프에서 루프 탈출 상황을 식별하고; 그리고

상기 식별된 루프 탈출 상황에 응답하여 펜딩 캐시 프리페치 요청들을 취소하도록

동작가능한, 컴퓨터-판독가능 프로그램 데이터 및 코드로 인코딩된 비-일시적 컴퓨터-판독가능 저장 매체.

청구항 20

프리페치 요청들을 취소하기 위한 장치로서,

프로그램 루프에서 메모리 액세스 명령의 반복 실행에 기초하여 데이터 액세스 스트라이드를 결정하기 위한 수단;

상기 데이터 액세스 스트라이드에 따라 상기 프로그램 루프에서의 메모리 액세스 명령들에 응답하여 프리페치 요청들을 추론적으로 이슈하기 위한 수단; 및

식별된 루프 탈출에 응답하여 펜딩 데이터 캐시 프리페치 요청들을 취소하기 위한 수단을 포함하는, 프리페치 요청들을 취소하기 위한 장치.

발명의 설명

기 술 분 야

[0001] [0001]본 개시물은 전반적으로 프로세싱 시스템들의 양상들에 관한 것이며 보다 구체적으로 데이터 프리페칭에 의해 발생된 캐시 공해를 감소시키기 위한 방법들 및 장치에 관한 것이다.

배 경 기 술

[0002] [0002]많은 휴대용 물건들, 이를 테면, 셀 폰들, 랩톱 컴퓨터들, 개인 휴대정보 단말기(PDA)들 등은 통신 및 멀티미디어 프로그램들과 같은 프로그램들을 실행하는 프로세싱 시스템을 활용한다. 이러한 물건들을 위한 프로세싱 시스템은 다수의 프로세서들, 명령들 및 데이터를 저장하기 위한 캐시들의 멀티-레벨들을 포함하는 복합 메모리 시스템들, 제어기들, 주변 디바이스들, 이를 테면, 통신 인터페이스들, 및 예를 들어, 하나의 칩 상에 구성되는 고정 함수 논리 블록들을 포함할 수 있다. 동시에, 휴대용 물건들은, 프로세싱 시스템에 의한 고성능 동작들을 지원하도록 종종 요구되는 배터리들의 형태로 제한된 에너지원을 갖는다. 배터리 수명을 증가시키기 위해서, 가능한 한 효율적으로 이러한 동작들을 수행하는 것이 바람직하다. 많은 개인 컴퓨터들이 또한, 전체 에너지 소비를 감소시키도록 동작하는 효율적인 설계들로 개발되고 있다.

[0003] [0003]프로그램들의 실행 시 고성능을 제공하기 위해서, 메모리 참조들의 공간 지역성의 개념에 기초되고 전반적으로 프로세서 성능을 개선시키기 위해서 사용되는 데이터 프리페칭이 사용될 수 있다. 페칭된 데이터 엘리먼트에 인접해 있거나 또는 스트라이드(stride) 어드레스 델타 또는 간접 포인터에 의해 관련되고, 추후의 액세스들에 사용될 가능성이 높은 어드레스들에서 캐시로부터 다수의 데이터 엘리먼트들을 프리페칭함으로써, 캐시 미스 레이트들이 감소될 수 있다. 캐시 설계들은 일반적으로 개별 데이터 엘리먼트 페치를 위한 데이터의 캐시 라인을 프리페칭함으로써 프리페칭의 형태를 구현한다. 하드웨어 프리페처들은 데이터의 하나 또는 그보다 많은 추가 캐시 라인들을 추론적으로 프리페칭함으로써 이것에 대해 확장할 수 있으며, 프리페치 어드레싱은 순차적인 정보, 스트라이드 또는 포인터 정보에 기초하여 형성될 수 있다. 이러한 메모리 집약적 작업부하들에 대한 하드웨어 프리페처 동작, 이를 테면, 대형 데이터의 어레이를 프로세싱하는 것은 메모리 레이턴시를 상당히 감소시킬 수 있다. 그러나, 데이터 프리페칭이 단점들이 없는 것은 아니다. 예를 들어, 데이터의 어레이를 프로세싱하기 위해 사용되는 소프트웨어 루프에서, 데이터 프리페처 회로는 루프의 최종 반복을 비롯한 루프의 추후 반복들에 사용될 데이터를 프리페칭한다. 그러나, 루프의 최종 반복을 위해 프리페칭된 데이터가 사용되지 않을 것이고 캐시에서 사용되지 않을 그 데이터를 저장함으로써 캐시 공해가 발생한다. 루프들이 펼쳐지는(unrolled) 경우 캐시 공해 문제가 복잡해진다.

발명의 내용

[0004] [0004]본원의 여러가지 양상들 중에서, 본 개시물은, 프리페칭을 위한 더욱 효율적인 방법들 및 장치들을 제공함으로써 프로세서 시스템에서 성능을 개선하고 전력 요건들을 감소시킬 수 있다는 것을 인식한다. 이를 위해서, 본 발명의 실시형태는 프리페치 요청들을 취소시키기 위한 방법을 다룬다. 프로그램 흐름 정보의 평가에 기초하여 루프 탈출 상황이 식별된다. 식별된 루프 탈출 상황에 응답하여 펜딩 캐시 프리페치 요청들이 취소된다.

[0005] [0005]다른 구현은 프리페치 요청들을 취소하는 방법을 다룬다. 데이터는 호출된 기능에 따라서 추론적으로 프리페칭된다. 펜딩 데이터 프리페치 요청들은, 호출된 기능으로부터의 기능 탈출에 응답하여 취소된다.

[0006] [0006]다른 실시형태는 프리페치 요청들을 취소하기 위한 장치를 다룬다. 루프 데이터 어드레스 모니터는 프로그램 루프의 메모리 액세스 명령의 반복 실행에 기초하여 데이터 액세스 스트라이드를 결정하도록 구성된다. 데이터 액세스 스트라이드에 따라서 프리페치 요청들을 추론적으로 이슈하도록 데이터 프리페치 로직이 구성된다. 정지 프리페치 회로는, 식별된 루프 탈출에 응답하여 펜딩 프리페치 요청들을 취소시키도록 구성된다.

[0007] [0007]다른 실시형태는 컴퓨터 판독가능 프로그램 데이터 및 코드로 인코딩된 컴퓨터 판독가능 비일시적 매체를 다룬다. 프로그램 흐름 정보의 평가에 기초하여 루프 탈출 상황이 식별된다. 펜딩 캐시 프리페치 요청들은, 식별된 루프 탈출 상황에 응답하여 취소된다.

[0008] [0008]추가 실시형태는 프리페치 요청들을 취소하기 위한 장치를 다룬다. 프로그램 루프에서 메모리 액세스 명령의 반복 실행에 기초하여 데이터 액세스 스트라이드를 결정하기 위한 수단이 활용된다. 데이터 액세스 스트라이드에 따라서 프리페치 요청들을 추론적으로 이슈하기 위한 수단이 활용된다. 식별된 루프 탈출에 응답하여 펜딩 프리페치 요청들을 취소시키기 위한 수단이 또한 활용된다.

[0009] [0009]본 발명의 다른 실시형태들은 다음의 상세한 설명으로부터 당업자들에게 자명해질 것이라는 것이 이해되며, 본 발명의 다양한 실시형태들은 예시로서 나타내어지고 설명된다. 실현될 바와 같이, 본 발명은 다른 실시형태 그리고 상이한 실시형태들이 가능하고 그의 여러가지 상세들은, 본 발명의 정신 및 범위로부터 모두 벗어나지 않고 다양한 다른 양상들로 변경이 가능하다. 그에 따라, 도면들 및 상세한 설명은 본질적으로 예시적인

것으로서 그리고 제한적이 않은 것으로서 여겨질 것이다.

도면의 간단한 설명

[0010] [0010]본 발명의 다양한 양상들이 첨부된 도면들에서 예로서 도시되며 이것으로 제한되는 것은 아니다.

[0011]도 1은 본 발명의 실시형태가 유리하게 활용될 수 있는 예시적인 프로세서 시스템을 도시한다.

[0012]도 2a는 루프 종료 분기의 검출 시 펜딩 넌-디멘드 데이터 프리페치 요청들을 취소하기 위한 프로세스를 도시한다.

[0013]도 2b는 기능 복귀의 검출 시 펜딩 넌-디멘드 데이터 프리페치 요청들을 취소하기 위한 프로세스를 도시한다.

[0014]도 3은 캐시 공해를 감소시키기 위해서 선택된 펜딩 데이터 프리페치 요청들을 취소시키도록 구성되는 프로세서 컴플렉스를 구비한 휴대용 디바이스의 특정 실시형태를 도시한다.

발명을 실시하기 위한 구체적인 내용

[0011] [0015]첨부된 도면들과 관련하여 아래에 제시되는 상세한 설명은 본 발명의 다양한 예시적인 실시형태들의 설명인 것으로 의도되고 본 발명이 실시될 수 있는 유일한 실시형태들을 나타내도록 의도되지 않는다. 상세한 설명은 본 발명의 완전한 이해를 제공할 목적으로 특정 상세들을 포함한다. 그러나, 본 발명이 이러한 특정 상세들 없이도 실시될 수 있다는 것이 당업자에게 명백할 것이다. 일부 예시들에서, 잘 알려진 구조들 및 컴포넌트들은 본 발명의 개념들을 불명료하게 하는 것을 방지하기 위해서 블록도의 형태로 나타내어 진다.

[0012] [0016]도 1은 본 발명의 실시형태가 유리하게 사용되는 예시적인 프로세서 시스템(100)을 도시한다. 프로세서 시스템(100)은 프로세서(110), 캐시 시스템(112), 시스템 메모리(114), 그리고 입력 및 출력(I/O) 시스템(116)을 포함한다. 캐시 시스템(112)은 예를 들어, 레벨 1 명령 캐시(I캐시)(124), 메모리 제어기(126), 및 레벨 1 데이터 캐시(D캐시)(128)를 포함한다. 캐시 시스템(112)은 또한 레벨 2 통합 캐시(미도시) 또는 원하는 대로 특정 구현 환경을 위한 다른 캐시 컴포넌트들을 포함할 수 있다. 시스템 메모리(114)는, I캐시(124) 또는 D캐시(128)에서 발견되지 않은 명령들과 데이터에 대한 액세스를 제공한다. 캐시 시스템(112)은 프로세서(110)에 통합될 수 있고, 또한 캐시들의 다중 레벨들을 계층 조직에 포함할 수 있다는 것을 주목한다. I/O 시스템(116)은, 프로세서(110)와 인터페이싱하는 복수의 I/O 디바이스들, 이를 테면, I/O 디바이스들(140 및 142)을 포함한다.

[0013] [0017]본 발명의 실시형태들은 조건 분기 명령들을 갖는 프로세서에 적절하게 채용될 수 있다. 프로세서(110)는, 예를 들어, 명령 파이프라인(120), 데이터 프리페치 로직(121), 예측 로직(122), 및 스택 로직 회로(123)를 포함한다. 명령 파이프라인(120)은 일련의 스테이지들, 이를 테면, 페치 및 프리페치 스테이지(130), 디코드 스테이지(131), 명령 이슈 스테이지(132), 피연산자 페치 스테이지(133), 로드(Ld) 및 저장(St) 명령들의 실행을 위한 것과 같은 실행 스테이지(134), 및 완료 스테이지(135)로 이루어진다. 당업자는, 명령 파이프라인(120) 내의 각각의 스테이지(130-135)가 프로세서의 동작 주파수 및 각각의 스테이지에서 요구되는 동작들의 복잡도(complexity)에 의존하여 다수의 추가 파이프라인 스테이지들을 포함할 수 있다는 것을 인식할 것이다. 예를 들어, 실행 스테이지(134)는, 하나 또는 그보다 많은 명령 실행 스테이지 회로들에 대응하는 하나 또는 그보다 많은 파이프라인 스테이지들, 이를 테면, 가산기, 승산기, 로직 동작들, 로드 및 저장 동작들, 시프트 및 회전 동작들, 및 복잡도가 더 높거나 더 낮은 다른 기능 회로들을 포함할 수 있다. 예를 들어, 로드 명령이 실행되는 경우, 이 로드 명령이 D캐시(128)로부터 데이터를 요청하고 요청된 데이터가 D캐시에 존재하지 않는 경우 페치 요청이 캐시 또는 시스템 메모리의 다음 레벨로 이슈된다. 이러한 페치 요청은, 명령(이 경우에는 로드 명령)의 실행에 대해 직접 응답하는 것이기 때문에 디맨드 요청인 것으로 고려된다.

[0014] [0018]프리페치 요청은, 예를 들어, 스트라이드에 기초하여 로드 어드레스들에 따라 루프 내에 하나 또는 그보다 많은 로드 명령들을 갖는 프로그램 루프의 검출과 같은 프로그램 흐름 정보에 응답하여 이루어지는 요청이다. 데이터 프리페치 로직(121)은, 프리페치 요청을 이슈하기 전에 로드 명령들의 피연산자 어드레스들의 디맨드 사용 패턴을 더욱 정확하게 식별하기 위해서, 검출된 루프의 다수의 반복들에 기초할 수 있는 이러한 프로그램 흐름 정보를 사용한다. 패턴이 검출되는 경우 충진(fill) 요청이 삽입된다. 프로세서(110)는, 프로세서 파이프라인에서 추적되는 요청과 연관된 엑스트라 플래그의 사용에 의해 디맨드 요청을 프리페치 요청으로부터 구분하도록 동작할 수 있다. 이 플래그는 또한, 각각의 아직 처리되지 않은 캐시 라인 충진이 프리페치

또는 디맨드 충진으로서 식별될 수 있는 캐시에 요청을 전파할 수 있다. 파이프라인 스테이지들 각각은 본원에 설명된 프리페치 요청 취소 방법들 및 장치로부터 벗어나지 않고 다양한 구현들을 구비할 수 있다.

[0015] [0019]프로그램에 의해 요청된 데이타가 연관된 레벨 1 D캐시(128)에 있지 않았을 경우 발생할 수 있는 지연들을 최소화하기 위해서, 페치 및 프리페치 스테이지(130)는, 검출된 프로그램 루프에서 실행되는 하나 또는 그보다 많은 메모리 액세스 명령들과 연관된 프로그램 흐름 정보를 기록한다. 프로그램 정보는, 로드 명령이 수신되었던 디코드 스테이지(131)로부터의 인디케이션을 포함할 수 있고 로드 명령에 대한 피연산자 어드레스 정보는 피연산자 페치 스테이지(133) 또는 실행 스테이지(134)와 같은, 실행 전의 파이프라인 스테이지에서 이용가능할 수 있다. 데이터 프리페치 로직(121)은, 로드 어드레스들이 패턴을 검출하는 데에 이용가능해 짐에 따라 로드 어드레스들을 모니터링한다. 이를 테면, 루프의 3회 또는 그보다 많은 반복을 통해 로드 명령들을 모니터링함으로써 컨피던스의 수용가능한 레벨에 따라 패턴이 결정된 후, 로드 명령을 루프에서 다시 접하게 되기 전에 예상 데이터에 대한 프리페치 요청이 이슈된다. 이 추론적인 프리페치 요청은, 실행 스테이지(134)에 의해 요구되는 때에 레벨 1 D캐시에서 요청된 데이터가 이용가능하다는 것을 확인한다. 이후, 로드 및 저장 실행 스테이지(134)는 메모리 계층 내 더 높은 레벨들로부터 데이터를 액세스하는 것을 대기하지 않고 레벨 1 D캐시로부터 직접 요청 데이터에 액세스할 가능성이 높다.

[0016] [0020]데이터 프리페치 로직(121)은 또한 데이터 액세스 스트라이드를 결정하기 위해 데이터 캐시 루프 데이터 어드레스 모니터를 포함할 수 있다. 이후, 데이터 프리페치 로직(121)은 데이터 액세스 스트라이드에 따라 피연산자 어드레스들 세트에 따라 프리페치 요청들을 추론적으로 이슈한다. 예를 들어, 데이터 프리페치 로직(121)은 스트라이드 값을 나타내는 로드 명령의 각각의 실행의 피연산자 어드레스 간의 차를 결정하기 위해 로드 명령의 반복 실행들을 모니터링하도록 구성되는 스트라이드 회로(119)를 포함할 수 있다. 스트라이드 회로(119)는 또한, 다음 피연산자 어드레스를 생성하기 위해서 가장 최근에 실행된 로드 명령의 피연산자 어드레스로, 결정된 스트라이드 값을 가산하도록 구성되는 가산 기능을 포함할 수 있다. 예측된 어드레스로서의 스트라이드 값과 대조적으로, 페칭된 조건 분기 명령은, 예측 로직 회로(122)에 포함된 것과 같은 분기 예측 로직을 사용하여 조건 분기가 택하여질 것인지 여부 및 분기 어드레스를 예측한다. 페칭된 비-분기 명령이 디코드 스테이지(131)로 진행하여 디코딩되고 명령 이슈 스테이지(132)에서의 실행을 위해 이슈되고, 실행 스테이지(134)에서 실행되고, 그리고 완료 스테이지(135)에서 중단된다.

[0017] [0021]예측 로직 회로(122)는 이벤트들을 모니터링하기 위한 검출 로직 회로(146), 필터(150), 및 조건 이력 테이블(152)을 포함한다. 일 구현에서, 조건 분기 명령들 중 대다수는 일반적으로, 그들의 조건들이 소프트웨어 루프의 대부분의 반복들에 대해 동일한 값으로 결정된 것으로 가정된다.

[0018] [0022]일 실시형태에서, 검출 로직 회로(146)는 도 2a에 관하여 설명된 바와 같은 소프트웨어 루프에서 사용된 조건 분기 명령들의 동적 특징들에 기초하여 동작하는 소프트웨어 루프 검출기로서 동작한다. 검출 로직 회로(146)는 또한 도 2b와 관련하여 설명된 바와 같이, 호출된 소프트웨어 기능들로부터의 탈출을 검출할 수 있다.

[0019] [0023]하나의 진입과 하나의 탈출을 갖는 소프트웨어 루프들에서, 루프 종료 분기는 일반적으로, 소프트웨어 루프에서 탈출하는 최종 반복을 제외하고 루프의 반복들 모두에 대해 소프트웨어 루프의 시작으로 역방향으로 분기하는 조건 분기 명령이다. 검출 로직 회로(146)는, 아래에서, 그리고 본 출원의 양수인에게 양도되고, 명칭이 "Suppressing Update of a Branch History Register by Loop-Ending Branches"인 미국 특허 출원 제11/066,508호(그 전체가 인용에 의해 본원에 포함됨)에서 상세하게 설명되는 바와 같이 소프트웨어 루프들의 검출을 위한 다수의 실시형태들을 가질 수 있다.

[0020] [0024]일 실시형태에 따르면, 검출 로직 회로(146)는, 조건 분기 명령 어드레스보다 적은 분기 타겟 어드레스를 갖는 조건 분기 명령들을 식별하고, 따라서 역방향 분기인 것으로 간주하며, 소프트웨어 루프의 종료를 마킹하는 것으로 가정된다. 모든 역방향 분기들이 루프 종료 분기들이 아니기 때문에, 예를 들어, 추가 모니터링 메커니즘들에 의해 고려될 필요가 있을 수 있는 어느 정도의 부정확성이 존재한다.

[0021] [0025]또한, 도 2b에 대하여 설명된 바와 같이, 기능 복귀 명령(흔히 RET로 지칭됨)이 검출될 수 있다. 일 실시형태에 따르면, 기능 복귀의 검출은 임의의 넌-디맨드 프리페치 요청들의 프리페치 취소들을 트리거링하도록 적응된다. 프리페치 요청의 취소는 또한, 루프 탈출의 검출과 같은 프로그램 흐름 정보에 응답하여 이루어진다.

[0022] [0026]다른 구현에서, 동일한 분기 명령의 반복된 실행을 인식함으로써 단순 루프들에서 루프 종료 분기가 검출될 수 있다. 최종 역방향 분기 명령을 위한 프로그램 카운터 값을 특수 목적 레지스터에 저장하고 이 저장된

값을 다음 역방향 분기 명령의 명령 어드레스와 비교함으로써, 2개의 명령 어드레스들이 일치하는 경우 루프 종료 분기가 인식될 수 있다. 코드가 소프트웨어 루프 내에 조건 분기 명령들을 포함할 수 있기 때문에, 루프 종료 분기 명령의 결정이 더욱 정교해질 수 있다. 이러한 상황에서, 다수의 특수 목적 레지스터들이, 각각의 조건 분기 명령의 명령 어드레스들을 저장하도록 하드웨어에서 실체화될 수 있다. 저장된 값들 모두를 다시 비교함으로써, 루프 종료 분기에 대한 일치가 결정될 수 있다. 통상적으로 루프 분기들은, 프로그램 카운터(PC)로부터 고정 오프셋을 갖는 조건부로 역방향을 향하는 분기들이다. 이러한 분기들의 타입들은 루프 탈출의 검출을 위한 어드레스 비교들을 필요로 하지 않을 것이다. 대신에, 일단 프로그램 루프가 조건부로 역방향을 향하는 분기에 기초하여 검출되면, 분기 예측의 결정(resolution)으로부터 루프 탈출이 결정된다. 예를 들어, 서술부가 루프로의 복귀에 대해 트루 조건인 것으로 결정되면, 예측이 오류 조건인 것으로 결정될 때 루프 탈출이 나타내어질 것이다. 펜딩 프리페치들이 존재하도록, 프로그램 루프가, 프리페치 하드웨어를 트리거하기 위해 이미 몇 번 실행됐을 것이다. 데이터 프리페치 로직(121)은, 그가 프리페칭을 시작하기 전에 패턴을 인식하도록 약간의 워밍업 디맨드 로드들을 요구한다.

[0023] [0027]또한, 루프 종료 분기는 컴파일러 또는 어셈블러에 의해 정적으로 마킹될 수 있다. 예를 들어, 일 실시형태에서, 고유 옵코드의 사용에 의해 또는 루프 종료 분기들을 위해서만 사용되는 특수 포맷 비트 필드를 셋팅함으로써, 컴파일러가 특정 타입의 분기 명령을 생성한다. 이후, 루프 종료 분기가 파이프라인 실행 동안, 이를 테면, 파이프라인에서의 디코드 스테이지 동안 용이하게 검출될 수 있다.

[0024] [0028]예측 로직 회로(122)는 필터(150), 조건 이력 테이블(CHT)(152), 및 연관된 모니터링 로직을 포함한다. 일 실시형태에서, 모니터링 프로세스는, 예측에 적합한 조건 분기 명령을 갖는 소프트웨어 루프의 하나 또는 그보다 많은 이전 실행들에서 발생했던 사전-지정된 조건 이벤트들의 상태 정보를 저장한다. 예측 로직 회로(122)의 지원 시, 필터(150)는 페칭된 조건 분기 명령이 수신되었고 CHT(152)가 인에이블되었는지 여부를 결정한다. 명령들이 파이프라인을 통해 이동함에 따라, 예를 들어, 파이프라인 스테이지들(132-135)에 의해서 추적되는 예측 정보를 제공하기 위해서 CHT(152)의 진입이 선택된다.

[0025] [0029]CHT(152) 진입은 예측 실행에 적합한 페칭된 명령을 위한 실행 이력을 기록한다. 예를 들어, 각각의 CHT 진입은, 예측 로직으로 입력되는 스테이터스(status) 비트들 및 실행 스테이터스 카운터들로부터의 카운트 값들의 조합을 적절하게 포함한다. CHT(152)는 또한, 다수의 조건 분기 명령들이 소프트웨어 루프에 존재할 수 있기 때문에, 페칭된 조건 분기 명령이 그 페칭된 명령과 연관된 CHT(152)의 진입 안으로 인덱싱하게 하는 인덱스 로직을 포함할 수 있다. 예를 들어, 소프트웨어 루프의 상부로부터 조건 분기 명령들의 수를 카운팅함으로써, 카운트는 CHT(152)에 대한 인덱스로서 사용될 수 있다. 예측 로직 회로(122)는 소프트웨어 루프들의 반복들을 카운팅하고, 실행 스테이터스 카운터들이 예를 들어, 강한 비실행 스테이터스를 나타내는 지정된 카운트 값으로 포화될 기회를 가졌었다는 것을 보장하기 위한 루프 카운터들을 포함한다. 실행 스테이터스 카운터가 포화된 경우, 다음 루프 반복 시, 연관된 페칭 조건 분기 명령의 분기 방향에 대해 예측하도록 예측 로직이 인에이블된다.

[0026] [0030]예측 로직 회로(122)는, 각각, 추적 레지스터 이슈(TrI)(162), 추적 레지스터 피연산자 페치(163), 추적 레지스터 실행(TrE)(164), 및 추적 레지스터 완료(TrC)(165)에서의 명령 이슈 스테이지(132), 피연산자 페치 스테이지(133), 실행 스테이지(134), 및 완료 스테이지(135)에서 추적된 예측 정보를 생성한다. 루프의 종료, 또는 기능 복귀를 나타내는 실패 서술부(failed predicate)를 갖는 조건 역방향 분기가, 이를 테면, 프로세서 파이프라인의 실행 스테이지(134) 동안 검출되는 경우, 펜딩 프리페치 요청들을 취소하는 신호(155)가 생성된다. 다른 실시형태에서, 분기 예측 로직에 의해 생성된 조건 분기 예측에 기초하여 펜딩 프리페치 요청들이 취소된다. 각각의 조건 분기는 일반적으로 그 조건 분기를 택할지 또는 택하지 않을지에 대해 분기 예측 로직에 의해 예측된다. 예를 들어, 이 실시예에서 프로그램 루프가 계속되는 조건 분기가 택하여졌다는 것을 예측 정보가 나타내는 경우, 명령 페처가 예측에 의해 나타내어진 프로그램 루프 상에서 명령들을 추론적으로 페칭한다. 예측 정보는 또한, 페치 & 프리페치 회로(130)에 상주할 수 있는 펜딩 프리페치 요청을 취소하는 로직 회로(141)에 결합될 수 있다. 이후, 펜딩 프리페치 요청을 취소하는 로직 회로(141)는 펜딩 프리페치 요청들이 필요하지 않다는 것을 나타내는 프로그램 흐름 정보에 기초하여 펜딩 프리페치 요청들을 추론적으로 취소할 수 있다. 예를 들어, 프로세서는 약하게 예측된 루프 탈출에 기초하여 펜딩 프리페치 요청들을 취소하지 않도록 구성될 수 있다. 하나 또는 그보다 많은 펜딩 데이터 프리페치 요청들을 취소함으로써, 프로세서(110)에서 데이터 캐시 공해가 감소되고 이러한 공해를 해결하기 위해 활용되는 전력이 감소된다. 펜딩 프리페치 요청을 취소하는 신호(155)는, 도 1에 도시된 바와 같이 프로세서 명령 파이프라인(120)에 결합되고, 프리페치 요청들의 요구를 제외하고 펜딩 중인 프리페치 요청들이 취소되게 하는 펜딩 프리페치 요청을 취소하는 로직 회로(141)에 의해 수

용된다. 또한, 데이터 캐시 내에 불필요한 데이터를 저장하지 않음으로써 프로세서 성능이 개선되는데, 이는 페칭되었었던 데이터를 퇴출시킬 수 있고 대신에 현재 누락된 것이 생성된다.

[0027] [0031]실행 스테이지(134)에 도달할 때, 루프 종료 조건 분기 명령을 위해 지정된 실행 조건이 그의 예측과는 상반되게 평가되는 경우, 잘못된 명령 경로 상의 명령들의 임의의 파이프라인 추론 실행이 예를 들어, 파이프라인을 플러쉬(flushing)함으로써 수정되고, 이러한 수정은 잘못된 명령 경로와 연관되는 펜딩 프리페치들을 취소하는 것을 포함한다. 예를 들어, 일 실시형태에서, 파이프라인에 대한 수정은 예측이 이루어졌던 스테이지에서 시작하는 파이프라인에서의 명령들을 플러쉬하는 것을 포함한다. 대안적인 실시형태에서, 파이프라인은, 루프 종료 조건 분기 명령이 최초로 페칭되었던 시작 페치 스테이지로부터 플러쉬된다. 또한, 적절한 CHT 진입은 또한 부정확한 예측 이후에 수정될 수 있다.

[0028] [0032]루프 검출기로서 작동하는 검출 회로(146)가 루프 종료 분기를 검출하도록 동작한다. 예를 들어, 루프 종료 분기는 일반적으로, 루프를 탈출하는 최종 반복을 제외한 루프의 모든 반복들에 대해 루프의 시작으로 역방향으로 분기하는 조건 분기 명령이다. 각각의 식별된 루프에 관한 정보가 필터 회로(150)로 전달되고 루프 탈출 상황 시 펜딩 프리페치 요청을 취소하는 로직 회로(141)가 각각의 식별된 루프 탈출에 응답하여 펜딩 넌-디맨드 프리페치 요청들을 취소한다.

[0029] [0033]일 실시형태에서, 필터 회로(150)는, 예를 들어, 소프트웨어 루프의 설정된 횟수의 반복, 이를 테면, 특정 루프의 3회 반복들이 발생되었다는 표시를 제공하는 루프 카운터이다. 루프의 각각의 반복의 경우, 필터는, 조건 분기 명령이 예측에 적합한지를 결정한다. 적합한 조건 분기(CB) 명령이 그 루프에 있는 경우, CB 명령을 실행하는 스테이터스가 조건 이력 테이블(CHT) 회로(152)에 기록된다. 예를 들어, 실행 스테이터스 카운터는, 적합한 CB 명령의 사전 시도되었던 실행들의 실행 이력을 기록하는 데에 사용될 수 있다. 실행 스테이터스 카운터는, 조건적으로 실행된 CB 명령을 나타내는 일 방향으로 그리고 조건적으로 실행되지 않았던 CB 명령을 나타내기 위한 반대 방향으로 업데이트된다. 예를 들어, 2개의 비트 실행 스테이터스 카운터가 사용될 수 있는데, 비실행된 스테이터스는 카운터의 감소를 유발하고 실행된 스테이터스는 카운터의 증가를 유발한다. 실행 스테이터스 카운터의 출력 상태들에는, 예를 들어, 사전 CB 명령들이 실행되었었다고 강하게 나타내어지는 것을 나타내는 "11"의 출력, 사전 CB 명령들이 실행되었었다고 약하게 나타내어지는 것을 나타내는 "10"의 출력, 사전 CB 명령들이 실행되지 않았었다고 약하게 나타내어지는 것을 나타내는 "01"의 출력, 및 사전 CB 명령들이 실행되지 않았었다고 강하게 나타내어지는 것을 나타내는 "00"의 출력이 할당된다. 실행 스테이터스 카운터 "11" 출력과 "00" 출력은 출력 값들을 포화시킬 것이다. 실행 스테이터스 카운터는 검출된 소프트웨어 루프 내 각각의 CB 명령에 대한 스테이터스와 연관되거나 이를 제공할 것이다. 그러나, 특정 구현은, 구현에 사용되는 실행 스테이터스 카운터들의 수를 제한할 수 있고 따라서 예측되는 CB 명령들의 수를 제한할 수 있다. 검출 회로(146)는 일반적으로 소프트웨어 루프로의 첫 번째 진입시에 실행 스테이터스 카운터들을 리셋한다.

[0030] [0034]대안으로, 디스에이블 예측 플래그는, 실행 스테이터스 카운터보다는 예측될 각각의 CB 명령과 연관될 수 있다. 디스에이블 예측 플래그는, 연관된 CB 명령이 실행된 것으로 사전에 결정되었다면 예측을 디스에이블하도록 능동적으로 설정된다. 실행되는 이전 CB 명령을 식별하는 것은 CB 명령에 대한 비실행 상황을 예측하기 위한 컨피던스 레벨이 수용가능한 레벨보다 낮을 것이라는 것을 의미한다.

[0031] [0035]인덱스 카운터는 또한, 어느 CB 명령이 소프트웨어 루프에서 카운팅되고 있거나 또는 평가되고 있는지를 결정하기 위해서 CHT(152)와 함께 사용될 수 있다. 예를 들어, 5개 또는 그보다 많은 CB 명령들을 갖는 루프에서, 제 1 CB 명령은 "000"의 인덱스를 가질 수 있고 제 4 적합한 조건 분기 명령은 "011"의 인덱스를 가질 수 있다. 인덱스는 대응하는 CB 명령에 대해 저장된 실행 스테이터스 카운터 값들에 액세스하기 위한 어드레스를 CHT(152)에 나타낸다.

[0032] [0036]예측 회로(122)는, 실행 스테이터스 카운터 출력 값들과 같은 특정 CB 명령에 대한 예측 정보를 수신하고, 도 1의 디코드 스테이지(131) 동안, 예를 들어, CB 명령이 일반적으로 소프트웨어 루프 시작으로 역방향으로 분기할 것이라는 것을 예측하고 루프 탈출 상황이 도달되었음을 예측하지 않는다. 일 실시형태에서, 예측 회로(122)는, CB 명령에 의해 지정된 조건이, 분기 상태 아님, 코드가 루프 탈출을 하거나 또는 미완성인 것으로 평가하는 것을 예측할 수 있다. 예측 회로(122)는 CB 명령을 추적한다. CB 명령이 루프 시작으로 역방향 분기로 예측되는 경우, 예측 정보는 이러한 스테이터스를 나타낸다. CB 명령이 역방향으로 분기하지 않는 것으로 예측되었다면, 추적 회로는 펜딩 프리페치 요청을 취소하는 신호를 생성하고 부정확한 예측이 이루어졌었는지 여부를 결정하기 위한 조건 평가가 이루어진다. 부정확한 예측이 이루어졌다면, 파이프라인이 또한 플러쉬될 수 있고, CHT(152)의 적절한 실행 스테이터스 카운터들이 업데이트되고, 일 실시형태에서, 이 특정 CB

명령이 이 시점부터 예측되지 않을 것이라는 것을 나타내기 위해 연관 CHT 진입이 마킹된다. 다른 실시형태에서, 예측 로직 회로(122)는 또한, CB 명령이 잘못 예측되었다고 결정할 때 사전 지정된 평가 기준을 변경할 수 있는데, 예를 들어, 이 시점부터 예측 기준을 더 보수적으로 잡을 수 있다.

[0033] [0037]모든 루프들이 유사한 특징들을 갖는 것은 아니라는 것을 추가로 인식된다. 특정 루프가 불량한 예측 결과들을 제공하는 경우, 그 루프는 예측을 디스에이블하기 위해 예측 로직 회로(122)에 마킹된다. 유사한 방식으로, 특정 루프가 동작 시나리오들 중 하나의 세트 하에서 양호한 예측으로 동작할 수 있고 동작 시나리오들의 상이한 세트 하에서 불량한 예측으로 동작할 수 있다. 그러한 경우, 동작 시나리오들의 인식은, 예측이 인에이블되거나, 디스에이블되거나 또는 인에이블되지만 동작 시나리오에 적합한 상이한 평가 기준을 갖게 할 수 있다.

[0034] [0038]도 2a는 루프-종료 분기의 검출 시 펜딩 넌-디맨드 데이터 프리페치 요청들을 취소하기 위한 프로세스(200)를 도시한다. 블록(202)에서, 소프트웨어 루프에 대한 프로세서 코드 실행이 모니터링된다. 결정 블록(204)에서, 소프트웨어 루프가 검출되었는지 여부에 대한 결정이 이루어진다. 예를 들어, 상술된 바와 같이, 소프트웨어 루프의 첫 번째 통과 시 소프트웨어 루프의 시작을 나타내는 위치로의 역방향 분기를 식별함으로써 소프트웨어 루프가 결정될 수 있다. 소프트웨어 루프가 식별되지 않는 경우, 프로세스(200)가 블록(202)으로 복귀한다. 소프트웨어 루프가 식별되었다면, 프로세스(200)는 블록(206)으로 진행한다. 이 시점의 코드에서, 소프트웨어 루프의 제 1 사이클이 이미 실행되었고 소프트웨어 루프의 다음 사이클이 이미 시작한 것이다.

[0035] [0039]블록(206)의 소프트웨어 루프의 다음 사이클에서, CB 명령에 대해 프로세서 코드가 모니터링된다. 결정 단계(208)에서, 예를 들어, 파이프라인 디코드 스테이지, 이를 테면 도 1의 디코드 스테이지(131) 동안, CB 명령이 검출되었는지 여부에 대한 결정이 이루어진다. CB 명령이 검출되지 않았다면, 프로세스(200)가 블록(206)으로 복귀한다. CB 명령이 검출되었다면, 프로세스(200)가 결정 블록(210)으로 진행한다. 결정 블록(210)에서, 예를 들어, 조건 서술부의 평가에 기초하여 조건 분기(CB) 명령이 루프를 종료시키도록 결정되었는지 여부에 관한 결정이 이루어진다. 검출될 수 있었던 CB 명령 평가들의 다수의 타입들이 존재한다. 예를 들어, 검출된 CB 명령의 제 1 평가는, CB 명령은 소프트웨어 루프가 종료 시점에 있지만 계속해서 루프 프로세싱을 평가하는 것으로 결정된다.. 소프트웨어 루프의 첫 번째 통과 시 소프트웨어 루프를 식별한 역방향 분기 CB 명령은, 예를 들어, 프로세서 코드로 그의 어드레스 위치에 의해 태깅된다. 또한, 소프트웨어 루프의 지정된 반복 횟수가 완료되지 않는 경우, CB 명령은 소프트웨어 루프의 시작으로 역방향으로 프로세서를 분기시킬 것을 결정한다. 검출된 CB 명령의 제 2 평가는, CB 명령이 소프트웨어 루프의 종료 시점에 있고 소프트웨어 루프를 종료시키도록 평가하는 것으로 결정될 수 있다. 검출된 CB 명령의 제 3 평가는, CB 명령이 소프트웨어 루프 내부에 있지만 택하여지거나 또는 택하여지지 않는 것으로 평가되는 경우, 프로세서 코드가 소프트웨어 루프에 남아 있는 것으로 결정될 수 있다. 또한, CB 명령의 제 4 평가는, CB 명령이 소프트웨어 루프 내부에 있지만, 택하여지거나 택하여 지지 않은 것으로 평가된 경우, 프로세서 코드가 소프트웨어 루프를 탈출한 것으로 결정될 수 있다. 제 4 평가 시, 소프트웨어 루프 내부에 있지만 역방향 분기 CB 명령의 어드레스 위치 다음의 순방향 분기로서 결정되는 CB 명령은, 소프트웨어 루프를 탈출한 것으로 간주된다.

[0036] [0040]결정 블록(210)으로 돌아가서, 검출된 CB 명령이 CB 명령의 제 1 평가 및 제 3 평가로서 소프트웨어 루프를 탈출한 것으로 결정되지 않았다면, 프로세스(200)는 블록(212)으로 진행한다. 블록(212)에서, 프로세스(200)는 정상 분기 프로세싱으로 진행하고 이후 블록(206)으로 복귀한다. 검출된 CB 명령이 CB 명령의 제 2 평가 및 제 4 평가로서 소프트웨어 루프를 탈출한 것으로 결정되었다면, 프로세스(200)가 블록(214)로 진행한다. 블록(214)에서, 프로세스(200)는, 디맨드 데이터 프리페치 요청들을 제외한 펜딩 데이터 프리페치 요청들을 취소하고, CB 명령을 프로세싱하고, 블록(202)으로 복귀하여 다음 소프트웨어 루프를 탐색하기 시작한다.

[0037] [0041]도 2b는 기능 복귀의 검출 시 펜딩 넌-디맨드 데이터 프리페치 요청들을 취소하기 위한 프로세스(250)를 도시한다. 블록(252)에서, 소프트웨어 기능 탈출을 위해 프로세서 코드 실행이 모니터링된다. 소프트웨어 기능이 추론적으로 실행될 수 있다는 것을 주목한다. 예를 들어, 소프트웨어 루프에서 함수 호출을 위해 추론적 실행이 발생할 수 있다. 소프트웨어 기능의 추론적 실행의 경우, 소프트웨어 기능 탈출, 이를 테면, RET 명령의 실행이 또한 추론적으로 실행될 수 있다. 결정 블록(254)에서, 소프트웨어 기능 탈출이 검출되었는지 여부에 관한 결정이, 예컨대, 프로세서의 실행 파이프라인에서 복귀 명령을 검출함으로써 이루어진다. 소프트웨어 기능 탈출이 검출되지 않는다면, 프로세스(250)가 블록(252)으로 복귀한다.

[0038] [0042]소프트웨어 기능 탈출이 검출되었다면, 프로세스(250)가 결정 블록(256)으로 진행한다. 결정 블록(256)에서, 이 검출된 탈출 상황이 인터럽트 루틴으로부터의 복귀인지 여부에 관한 결정이 이루어진다. 검출된 탈출

이 인터럽트 루틴으로부터의 복귀인 경우, 프로세스(250)는 블록(252)으로 복귀한다. 검출된 탈출이 인터럽트 루틴으로부터의 복귀가 아닌 경우, 프로세스(250)가 블록(258)으로 진행한다. 블록(258)에서, 프로세스(250)는 디맨드 데이터 프리페치 요청들을 제외한 펜딩 프리페치 요청들을 취소하고, 복귀 명령을 프로세싱하고, 이후 블록(252)으로 복귀하여 소프트웨어 기능 탈출을 위한 프로세서 코드의 모니터링을 계속한다.

[0039] [0043]흔히, 손으로 또는 컴파일러 최적화들을 통해, 루프의 다수 반복들이 순차적으로 실행되도록 소프트웨어 루프가 펼쳐질 것이다. 각각의 펼쳐진 반복의 이러한 순차적인 실행은 추가 프리페치 후보가 된다. 루프의 최종 반복 시, 이후, 각각의 펼쳐진 후보는 프리페칭된 데이터 캐시 공해의 문제를 컴파운딩하는 불필요한 프리페치 요청들을 생성할 수 있다. 본 발명의 실시형태는 또한, 루프의 탈출, 또는 기능으로부터의 복귀를 검출함으로써, 그리고 각각 펼쳐진 루프로부터 불필요한 프리페치 요청들 모두를 취소함으로써 루프 펼침에 적용된다.

[0040] [0044]도 3은 캐시 공해를 감소시키기 위해서, 선택된 펜딩 데이터 프리페치 요청들을 취소하도록 구성되는 프로세서 컴플렉스를 갖는 휴대용 디바이스(300)의 특정 실시형태를 도시한다. 디바이스(300)는 무선 전자 디바이스일 수 있고 소프트웨어 명령들(318)을 갖는 시스템 메모리(312)에 결합된 프로세서 컴플렉스(310)를 포함한다. 시스템 메모리(312)는 도 1의 시스템 메모리(114)를 포함할 수 있다. 프로세서 컴플렉스(310)는 프로세서(311), 레벨 1 데이터 캐시(L1 D캐시)(222), 레벨 1 명령 캐시(L1 I캐시)(326), 캐시 제어기 회로(328)를 갖는 통합 메모리 서브시스템(314), 및 예측 로직(316)을 포함할 수 있다. 프로세서(311)는 도 1의 프로세서(110)를 포함할 수 있다. 통합 메모리 서브시스템(314)은 또한 레벨 2 통합(unified) 캐시(미도시)를 포함할 수 있다. L1 I캐시(326)는 도 1의 L1 I캐시(124)를 포함할 수 있고 L1 D캐시(322)는 도 1의 L1 D캐시(128)를 포함할 수 있다.

[0041] [0045]통합 메모리 서브시스템(314)은 프로세서 컴플렉스(310)에 포함될 수 있거나 또는 프로세서 컴플렉스(310) 외부의 하나 또는 그보다 많은 별개의 디바이스들 또는 회로소자(미도시)로서 구현될 수 있다. 예시적인 실시예에서, 프로세서 컴플렉스(310)는 도 1과 도 2에서 예시되거나 또는 이와 연관된 실시형태들 중 임의의 것에 따라서 동작한다. 예를 들어, 도 3에 도시된 바와 같이, L1 I캐시(326), L1 D캐시(322), 및 캐시 제어기 회로(328)가 프로세서 컴플렉스(310) 내부에서 액세스가능하고, 프로세서(311)는 통합 메모리 서브시스템(314)의 메모리들에 또는 시스템 메모리(312)에 저장된 데이터 또는 프로그램 명령들에 액세스하도록 구성된다.

[0042] [0046]카메라 인터페이스(334)가 프로세서 컴플렉스(310)에 결합되고 또한 카메라, 이를 테면, 비디오 카메라(336)에 결합된다. 디스플레이 제어기(340)가 프로세서 컴플렉스(310)에 그리고 디스플레이 디바이스(342)에 결합된다. 코더/디코더(CODEC)(344)가 또한 프로세서 컴플렉스(310)에 결합된다. 스피커(346) 및 마이크로폰(348)은 CODEC(344)에 결합될 수 있다. 안테나(352) 및 무선 인터페이스(350)를 통해 수신된 무선 데이터가 프로세서(311)에 제공될 수 있도록, 무선 인터페이스(350)가 프로세서 컴플렉스(310)에 그리고 무선 안테나(352)에 결합될 수 있다.

[0043] [0047]프로세서(311)는 비일시적 컴퓨터-판독가능 매체, 이를 테면 시스템 메모리(312)에 저장된 소프트웨어 명령들(318)(컴퓨터, 이를 테면 프로세서(311)로 하여금 도 2의 프로그램 프로세스(200)와 같은 프로그램을 실행하게 하도록 실행가능함)을 실행하도록 구성될 수 있다. 소프트웨어 명령들(318)은, 프로세서(311)로 하여금 통합 메모리 서브시스템(314)의 메모리들과 시스템 메모리(312)에 액세스하는 명령들을 프로세싱하게 하도록 추가로 실행가능하다.

[0044] [0048]특정 실시형태에서, 프로세서 컴플렉스(310), 디스플레이 제어기(340), 시스템 메모리(312), CODEC(344), 무선 인터페이스(350), 및 카메라 인터페이스(334)는 시스템-인-패키지(system-in-package) 또는 시스템-온-칩(system-on-chip) 디바이스(304)에 포함된다. 특정 실시형태에서, 입력 디바이스(356)와 전원(358)이 시스템-온-칩 디바이스(304)에 결합된다. 또한, 특정 실시형태에서, 도 3에 도시된 바와 같이, 디스플레이 디바이스(342), 입력 디바이스(356), 스피커(346), 마이크로폰(348), 무선 안테나(352), 비디오 카메라(336), 및 전원(358)이 시스템-온-칩 디바이스(304) 외부에 있다. 그러나, 디스플레이 디바이스(342), 입력 디바이스(356), 스피커(346), 마이크로폰(348), 무선 안테나(352), 비디오 카메라(336), 및 전원(358) 각각은 시스템-온-칩 디바이스(304)의 컴포넌트, 이를 테면, 인터페이스 또는 제어기에 결합될 수 있다.

[0045] [0049]본원에 설명된 실시형태들에 따른 디바이스(300)는 다양한 전자 디바이스들, 이를 테면, 셋톱 박스, 엔터테인먼트 유닛, 내비게이션 디바이스, 통신 디바이스, 개인 휴대정보 단말기(PDA), 고정 위치 데이터 유닛, 모바일 위치 데이터 유닛, 모바일 폰, 셀룰러 폰, 컴퓨터, 휴대용 컴퓨터, 태블릿들, 모니터, 컴퓨터 모니터, 텔레비전, 튜너, 라디오, 위성 라디오, 음악 플레이어, 디지털 음악 플레이어, 휴대용 음악 플레이어, 비디오 플레이어, 디지털 비디오 플레이어, 디지털 비디오 디스크(DVD) 플레이어, 휴대용 디지털 비디오 플레이어, 데이

터 또는 컴퓨터 명령들을 저장하거나 리트리브(retrieve)하는 임의의 다른 디바이스, 또는 이들의 조합에 포함될 수 있다.

[0046] [0050]본원에 개시된 실시형태들과 관련하여 설명된 다양한 예시적인 논리 블록들, 모듈들, 회로들, 엘리먼트들 또는 컴포넌트들이 범용 프로세서, 디지털 신호 프로세서(DSP), 주문형 집적 회로(ASIC), 필드 프로그램가능 게이트 어레이(FPGA) 또는 다른 프로그래머블 로직 컴포넌트들, 이산 게이트 또는 트랜지스터 로직, 이산 하드웨어 컴포넌트들, 또는 본원에 설명된 기능들을 수행하도록 설계된 이들의 임의의 결합을 이용하여 구현되거나 또는 수행될 수 있다. 범용 프로세서는 마이크로프로세서일 수 있지만, 대안적으로, 프로세서는 임의의 종래의 프로세서, 제어기, 마이크로컨트롤러 또는 상태 머신일 수 있다. 프로세서는 또한 컴퓨팅 컴포넌트들의 결합, 예를 들어, DSP와 마이크로프로세서의 결합, 복수의 마이크로프로세서들, DSP 코어와 공조하는 하나 이상의 마이크로프로세서들, 또는 원하는 응용에 적합한 임의의 다른 이러한 구성으로서 구현될 수 있다.

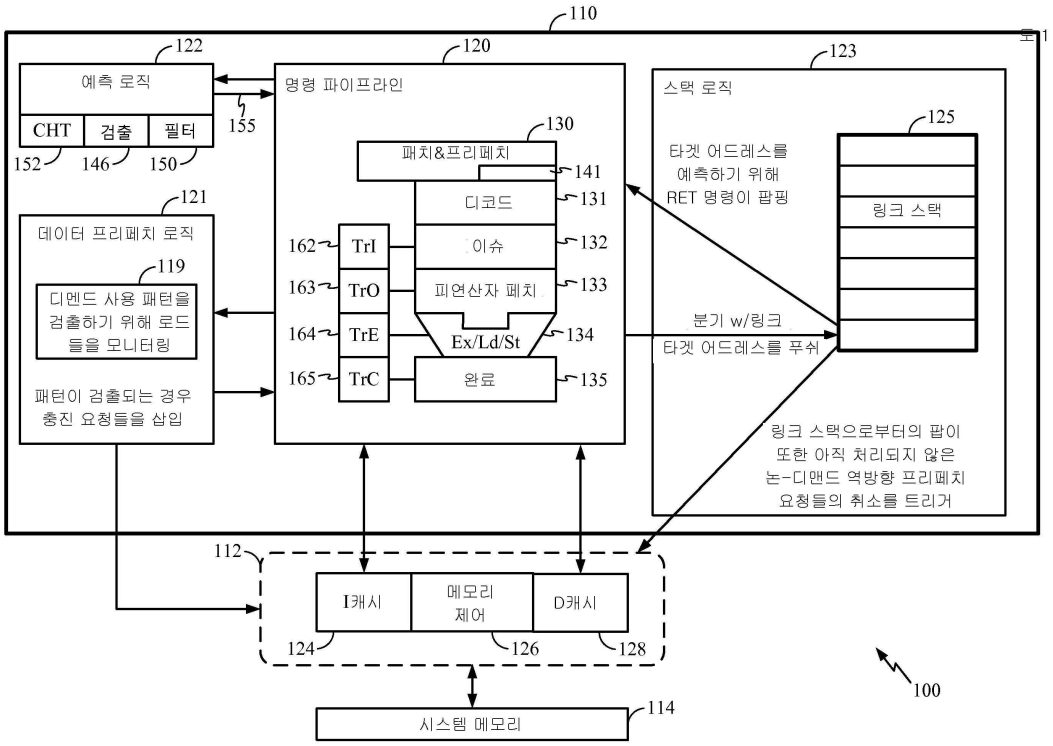
[0047] [0051]본원에 개시된 실시형태들과 관련하여 설명된 방법들은 직접 하드웨어에, 프로세서에 의해 실행된 소프트웨어 모듈에, 또는 이 둘의 조합으로 구현될 수도 있다. 소프트웨어 모듈은 RAM 메모리, 플래시 메모리, ROM 메모리, EPROM 메모리, EEPROM 메모리, 레지스터들, 하드 디스크, 착탈식 디스크, CD-ROM, 또는 본 기술분야에서 알려진 임의의 다른 형태의 비일시적 저장 매체에 상주할 수 있다. 비-일시적 저장 매체는, 프로세서가 비일시적 저장 매체로부터 정보를 판독할 수 있고 비일시적 저장 매체에 정보를 기입할 수 있도록 프로세서에 결합될 수 있다. 대안적으로, 비일시적 저장 매체는 프로세서와 일체형일 수 있다

[0048] [0052]예를 들어, 도 1의 프로세서(110) 또는 도 3의 프로세서(311)는, 예를 들어, 프로세서와 국부적으로 직접적으로 연관되는 것으로서, 이를 테면, 명령 캐시를 통해 이용가능할 수 있거나 또는 I/O 디바이스, 이를 테면, 도 1의 I/O 디바이스들(140 또는 142) 중 하나를 통해 액세스 가능한 컴퓨터 판독가능 비일시적 저장 매체에 저장된 프로그램의 제어 하에서 조건 비분기 명령들을 비롯한 명령들을 실행하도록 구성될 수 있다. I/O 디바이스는 또한, 프로세서들, 이를 테면, D캐시(128)와 국부적으로 직접 연관되거나, 또는 다른 프로세서의 메모리로부터 액세스 가능한 메모리 디바이스에 상주하는 데이터에 액세스할 수 있다. 컴퓨터 판독가능 비일시적 저장 매체는 RAM(random access memory), DRAM(dynamic random access memory), SDRAM(synchronous dynamic random access memory), 플래시 메모리, ROM(read only memory), PROM(programmable read only memory), EPROM(erasable programmable read only memory), EEPROM(electrically erasable programmable read only memory), 콤팩트 디스크(CD), DVD(digital video disk), 다른 타입들의 착탈식 디스크들, 또는 임의의 다른 적절한 비일시적 저장 매체를 포함할 수 있다.

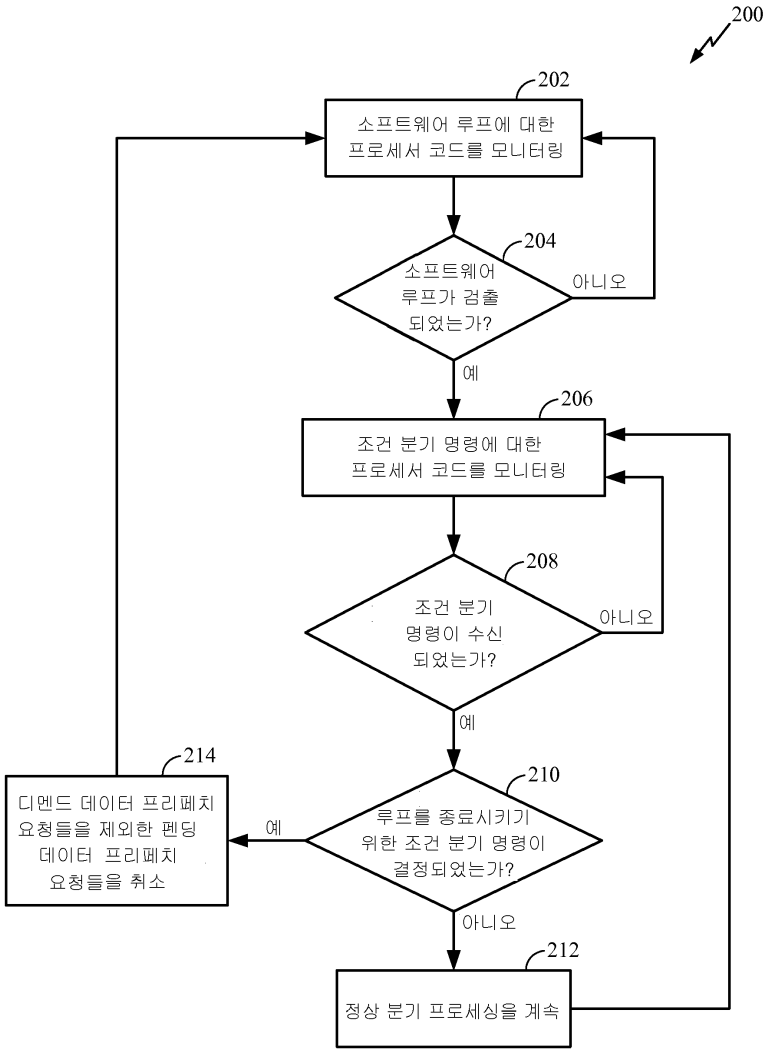
[0049] [0053]본 발명이 프로세서 시스템들에서 사용하기 위해 예시적인 실시형태들의 맥락으로 개시되었지만, 매우 다양한 구현들이 상기 설명과 아래에 뒤따라 나오는 청구항들과 일관되게 활용될 수 있다는 것이 당업자에 의해 인식될 것이다. 예를 들어, 고정 기능 구현은 또한 본 발명의 다양한 실시형태들을 활용할 수 있다.

도면

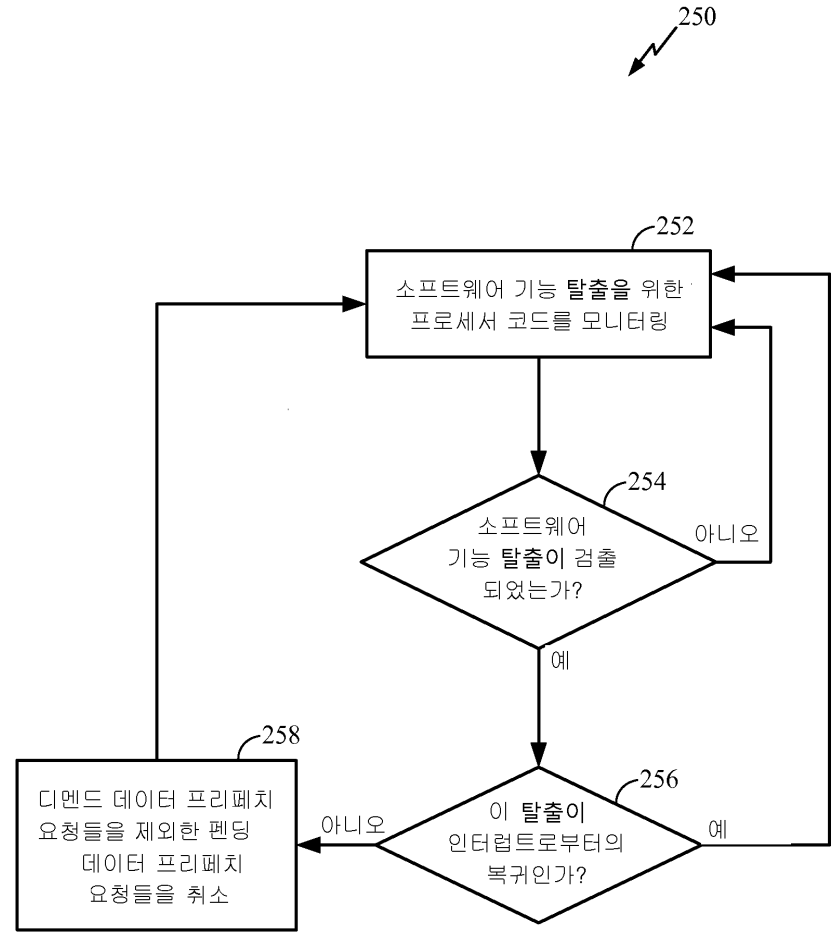
도면1


110
도1
122
예측 로직
CHT
검출
필터
152
146
150
155
120
명령 파이프라인
130
패치&프리페치
141
디코드
131
이슈
132
피연산자 페치
133
Ex/Ld/St
134
완료
135
162
TrI
163
TrO
164
TrE
165
TrC
121
데이터 프리페치 로직
119
디멘드 사용 패턴을 검출하기 위해 로드들을 모니터링
패턴이 검출되는 경우 충진 요청들을 삽입
123
스택 로직
타겟 어드레스를 예측하기 위해 RET 명령이 팝핑
125
링크 스택
분기 w/링크 타겟 어드레스를 푸쉬
링크 스택으로부터의 팝이 또한 아직 처리되지 않은 논-디맨드 역방향 프리페치 요청들의 취소를 트리거
112
I캐시
메모리 제어
D캐시
124
126
128
100
114
시스템 메모리

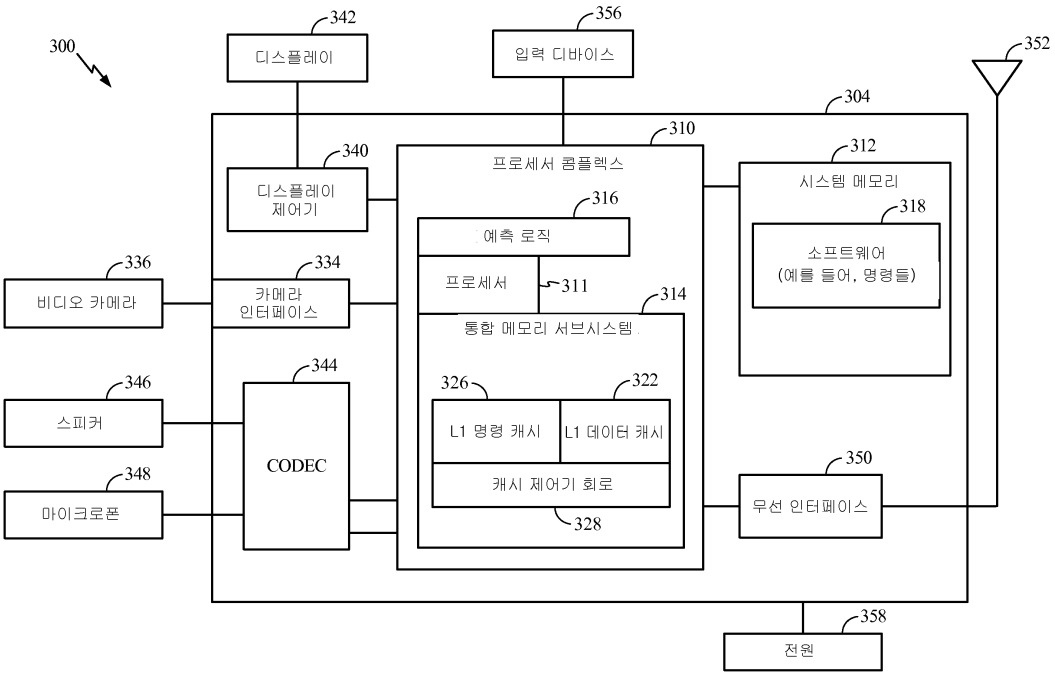
도면2a


200
202
소프트웨어 루프에 대한 프로세서 코드를 모니터링
204
소프트웨어 루프가 검출 되었는가?
아니오
예
206
조건 분기 명령에 대한 프로세서 코드를 모니터링
208
조건 분기 명령이 수신 되었는가?
아니오
예
210
루프를 종료시키기 위한 조건 분기 명령이 결정되었는가?
예
214
디멘드 데이터 프리페치 요청들을 제외한 펜딩 데이터 프리페치 요청들을 취소
아니오
212
정상 분기 프로세싱을 계속

도면2b


250
252
소프트웨어 기능 탈출을 위한 프로세서 코드를 모니터링
254
소프트웨어 기능 탈출이 검출 되었는가?
아니오
예
256
이 탈출이 인터럽트로부터의 복귀인가?
아니오
예
258
디멘드 데이터 프리페치 요청들을 제외한 펜딩 데이터 프리페치 요청들을 취소

도면3


300
342
디스플레이
356
입력 디바이스
352
304
310
프로세서 콤플렉스
340
디스플레이 제어기
312
시스템 메모리
316
예측 로직
318
소프트웨어 (예를 들어, 명령들)
336
비디오 카메라
334
카메라 인터페이스
프로세서
311
314
통합 메모리 서브시스템
344
326
322
346
스피커
L1 명령 캐시
L1 데이터 캐시
CODEC
348
마이크로폰
캐시 제어기 회로
350
무선 인터페이스
328
358
전원