



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2021-0035226
(43) 공개일자 2021년03월31일

(51) 국제특허분류(Int. Cl.)
G06F 12/0831 (2016.01) G06F 12/0817 (2016.01)
G06F 12/0895 (2016.01)
(52) CPC특허분류
G06F 12/0833 (2013.01)
G06F 12/0817 (2013.01)
(21) 출원번호 10-2021-7004749
(22) 출원일자(국제) 2019년08월01일
심사청구일자 없음
(85) 번역문제출일자 2021년02월17일
(86) 국제출원번호 PCT/US2019/044742
(87) 국제공개번호 WO 2020/028714
국제공개일자 2020년02월06일
(30) 우선권주장
16/053,488 2018년08월02일 미국(US)

(71) 출원인
자일링크스 인코포레이티드
미합중국 95124 캘리포니아 산 호세 로직 드라이브 2100
(72) 발명자
미탈 밀린드
미국 캘리포니아주 95124 산 호세 로직 드라이브 2100
다스티다르 자이덱
미국 캘리포니아주 95124 산 호세 로직 드라이브 2100
(74) 대리인
김태홍, 김진희

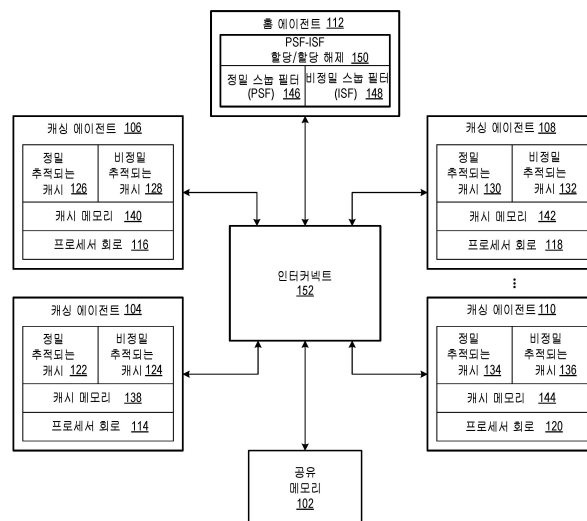
전체 청구항 수 : 총 15 항

(54) 발명의 명칭 혼성의 정밀 및 비정밀 캐시 스누브 필터링

(57) 요약

조합된 정밀 및 비정밀 스누브 필터링을 회로 및 방법이 개시된다. 메모리(102) 및 복수의 프로세서(104, 106, 108, 110)가 인터커넥트 회로부(152)에 결합된다. 복수의 캐시 회로(예컨대, 138, 122, 124)가 복수의 프로세서 회로에 각각 결합된다. 제1 스누브 필터(146)가 인터커넥트 회로부에 결합되고, 메모리의 어드레스의 제1 서브세트의 개별 캐시 라인에 의한 스누브 요청을 필터링하도록 구성된다. 제2 스누브 필터(148)가 인터커넥트 회로부에 결합되고, 메모리의 어드레스의 제2 서브세트의 캐시 라인 그룹에 의한 스누브 요청을 필터링하도록 구성된다. 각각의 그룹은 복수의 캐시 라인을 포함한다.

대표도



(52) CPC특허분류

G06F 12/0895 (2013.01)

G06F 2212/1032 (2013.01)

Y02D 10/00 (2020.08)

명세서

청구범위

청구항 1

회로 장치(circuit arrangement)에 있어서,
 인터커넥트 회로부와,
 상기 인터커넥트 회로부에 결합된 메모리와,
 상기 인터커넥트 회로부에 결합된 복수의 프로세서 회로와,
 상기 복수의 프로세서 회로에 각각 결합된 복수의 캐시 회로와,
 상기 인터커넥트 회로부에 결합되고, 상기 메모리의 어드레스의 제1 서브세트의 개별 캐시 라인에 의한 스noop 요청(snoop request)을 필터링하도록 구성되는 제1 스noop 필터(first snoop)와,
 상기 인터커넥트 회로부에 결합되고, 상기 메모리의 어드레스의 제2 서브세트의 캐시 라인 그룹에 의한 스noop 요청을 필터링하도록 구성되는 제2 스noop 필터
 를 포함하고, 각각의 그룹은 복수의 캐시 라인을 포함하는, 회로 장치.

청구항 2

제1항에 있어서, 상기 제1 스noop 필터에 그리고 상기 제2 스noop 필터에 결합된 제어 회로를 더 포함하고, 상기 제어 회로는 상기 제2 스noop 필터에 의해 필터링되는 캐시 라인 그룹을 상기 제1 스noop 필터에 의해 필터링되는 캐시 라인에 재할당하도록 구성되는, 회로 장치.

청구항 3

제2항에 있어서, 상기 제어 회로는 또한, 상기 캐시 라인 그룹에 대한 참조의 시간적 로컬성(temporal locality)에 기초하여 상기 제1 스noop 필터에 재할당할 캐시 라인 그룹을 선택하도록 구성되는, 회로 장치.

청구항 4

제2항에 있어서, 상기 제어 회로는 또한, 상기 캐시 라인 그룹 내의 캐시 라인의 단독 소유권에 기초하여 상기 제1 스noop 필터에 재할당할 캐시 라인 그룹을 선택하도록 구성되는, 회로 장치.

청구항 5

제2항에 있어서, 상기 제어 회로는 또한, 상기 캐시 라인 그룹에서 빈번하게 액세스되는 캐시 라인을 결정하는 것에 기초하여 상기 제1 스noop 필터에 재할당할 캐시 라인 그룹을 선택하도록 구성되는, 회로 장치.

청구항 6

제2항에 있어서, 상기 제어 회로는 또한,
 재할당을 위해 선택된 그룹에 속하는 캐시 라인을 결정하고,
 상기 그룹에 속하는 캐시 라인을 상기 제1 스noop 필터에 재할당하도록 구성되는, 회로 장치.

청구항 7

제2항 내지 제6항 중 어느 한 항에 있어서, 상기 제어 회로는 또한, 상기 제1 스noop 필터로부터의 제1 캐시 라인을 제2 스noop 필터에 의해 필터링되는 제1 그룹에 재할당하도록 구성되는, 회로 장치.

청구항 8

제2항 내지 제6항 중 어느 한 항에 있어서, 상기 제어 회로는 또한,

상기 제1 스눅 필터로부터의 제1 캐시 라인을 상기 제2 스눅 필터에 의해 필터링되는 제1 그룹에 재할당하고,
 상기 제1 스눅 필터에 의해 필터링되는 상기 제1 그룹에 속하는 캐시 라인을 결정하고,
 상기 제1 그룹에 속하는 캐시 라인의 캐시 라인 상태를 그룹 상태로 집계하고,
 상기 그룹 상태를 상기 제1 그룹의 태그와 연관하여 저장하고,
 상기 제1 스눅 필터로부터 상기 제1 그룹 내의 캐시 라인을 제거하도록 구성되는, 회로 장치.

청구항 9

제2항 내지 제6항 중 어느 한 항에 있어서, 상기 제어 회로는 또한,
 상기 제1 스눅 필터로부터의 제1 캐시 라인을 상기 제2 스눅 필터에 의해 필터링되는 제1 그룹에 재할당하고,
 상기 제1 스눅 필터의 디렉토리가 풀 상태(full)인 것에 응답하여 상기 제1 스눅 필터로부터의 제1 캐시 라인을
 상기 제2 스눅 필터에 의해 필터링되는 제1 그룹에 재할당하도록 구성되는, 회로 장치.

청구항 10

제2항 내지 제6항 중 어느 한 항에 있어서, 상기 제어 회로는 또한,
 상기 제1 스눅 필터로부터의 제1 캐시 라인을 상기 제2 스눅 필터에 의해 필터링되는 제1 그룹에 재할당하고,
 상기 제1 캐시 라인이 최소 사용 빈도(least recently used)인 것에 응답하여 상기 제1 스눅 필터로부터 상기
 제2 스눅 필터에 의해 필터링되는 제1 그룹에 재할당될 상기 제1 캐시 라인을 선택하도록 구성되는, 회로 장치.

청구항 11

제1항 내지 제6항 중 어느 한 항에 있어서,
 상기 제1 스눅 필터는 개별 캐시 라인의 복수의 캐시 라인 태그의 저장을 위해 구성된 제1 메모리를 포함하고,
 상기 제2 스눅 필터는 캐시 라인 그룹과 연관된 복수의 그룹 태그의 저장을 위해 구성된 제2 메모리를
 포함하고, 각각의 그룹 태그는 복수의 캐시 라인에 매핑되는, 회로 장치.

청구항 12

제2항에 있어서, 상기 제어 회로는 또한,
 상기 제2 스눅 필터에 의해 필터링되는 캐시 라인의 제1 그룹의 소유자 카운트를 결정하고,
 상기 제1 그룹의 소유자 카운트가 1과 같은 것과 상기 제1 그룹의 액세스 카운트가 제로보다 큰 것에 응답하여
 상기 제2 스눅 필터로부터의 캐시 라인 중 상기 제1 그룹의 캐시 라인을 상기 제1 스눅 필터에 재할당하도록 구
 성되는, 회로 장치.

청구항 13

방법에 있어서,
 복수의 프로세서의 캐시 회로에서 메모리의 캐시 라인을 캐싱하는 단계와,
 상기 메모리의 어드레스의 제1 서브세트의 개별 캐시 라인에 의한 스눅 요청을 제1 스눅 필터에 의해 필터링하
 는 단계와,
 상기 메모리의 어드레스의 제2 서브세트의 캐시 라인 그룹에 의한 스눅 요청을 제2 스눅 필터에 의해 필터링하
 는 단계
 를 포함하고, 각각의 그룹은 복수의 캐시 라인을 포함하는, 방법.

청구항 14

제13항에 있어서, 상기 제어 회로에 의해, 상기 제2 스눅 필터에 의해 필터링되는 캐시 라인 그룹을 상기 제1
 스눅 필터에 의해 필터링되는 캐시 라인에 재할당하는 단계를 더 포함하는, 방법.

청구항 15

제13항 또는 제14항에 있어서, 상기 제1 스누브 필터로부터의 제1 캐시 라인을 상기 제2 스누브 필터에 의해 필터링되는 제1 그룹에 제한당하는 단계를 더 포함하는, 방법.

발명의 설명

기술 분야

[0001] 본 개시내용은 캐시 스누브 필터링(cache snoop filtering)에 관한 것이다.

배경 기술

[0002] 캐시 메모리는 성능 강화를 위해 공유 메모리, 멀티프로세서 데이터 프로세싱 시스템에 채택되고 있다. 캐시 메모리에 데이터의 로컬 카피를 저장함으로써, 공유 메모리 내의 데이터에 액세스하는데 필요한 시간을 줄일 수 있다. 다수의 프로세서들 사이에서 메모리가 공유되기 때문에, 각 프로세서가 캐시 메모리 내의 최신 공유 데이터를 보고 액세스하는 것을 보장하기 위해 캐시 코히어런스 스킴(cache coherency scheme)이 필요하다.

[0003] 스누브 기반 캐시 코히어런스 프로토콜(noop-based cache coherency protocol)의 경우, 각 프로세서에 있는 캐시 제어 회로가 로컬 캐시 메모리에 캐싱된 어드레스에 대해 메모리 버스를 모니터링한다. 캐시 제어 회로가 다른 프로세서에 의한 소정의 어드레스에의 기록(write)을 검출하고 해당 어드레스가 로컬로 캐싱될 경우에, 캐시 제어 회로는 로컬 카피를 무효로 마킹할 수 있다. 캐싱되고 무효로 마킹된 어드레스에 프로세서가 액세스하려고 할 경우, 캐시 제어 회로는 최신 데이터를 인출한다.

[0004] 스누브 요청(snoop request)은 시스템 성능을 저하시킬 수 있다. 각 캐시 제어 회로가 캐시 라인 업데이트의 표시를 다른 캐시 제어 회로 전체에 브로드캐스팅한다. N개의 프로세서를 가진 시스템에서는 스누브 트래픽이 $N*(N-1)$ 에 비례할 것이다. 다수의 캐시 라인이 프로세서들 사이에서 공유될 수 없기 때문에, 스누브 트래픽의 대부분은 낭비적일 수 있다. 과도한 스누브 트래픽은 공유 메모리에의 액세스에 사용 가능한 대역폭을 감소시킨다.

[0005] 캐시 제어 회로는 불필요한 스누브 트래픽을 줄이기 위해 스누브 필터 기능을 구비하여 구현될 수 있다. 일반적으로, 스누브 필터는 어떤 어드레스가 캐싱되는지 추적하고 캐싱된 어드레스를 참조하지 않는 요청을 필터링한다. 게다가, 스누브 필터는 캐싱 가능한 어드레스의 카피를 요청한 프로세서, 또는 프로세서의 수도 추적하고, 캐싱되고 있는 어드레스에 대한 스누브를 필터링할 뿐만 아니라 추적된 어드레스를 캐싱한 프로세서에 대한 스누브도 필터링할 수 있다. 대개 캐시 필터는 두 가지 유형, 정밀 캐시 필터와 비정밀 캐시 필터에 속한다. 정밀 스누브 필터는 개별 캐시 라인의 어드레스에 의한 스누브 요청을 필터링한다. 비정밀 스누브 필터는 캐시 라인 그룹의 어드레스에 의한 스누브 요청을 필터링한다.

발명의 내용

[0006] 개시하는 회로 장치(circuit arrangement)는 인터커넥트 회로부와, 상기 인터커넥트 회로부에 결합된 메모리와, 상기 인터커넥트 회로부에 결합된 복수의 프로세서 회로를 포함한다. 복수의 캐시 회로는 복수의 프로세서 회로에 각각 결합된다. 제1 스누브 필터가 상기 인터커넥트 회로부에 결합되고, 상기 메모리의 어드레스의 제1 서브세트의 개별 캐시 라인에 의한 스누브 요청을 필터링하도록 구성된다. 제2 스누브 필터가 상기 인터커넥트 회로부에 결합되고, 상기 메모리의 어드레스의 제2 서브세트의 캐시 라인 그룹에 의한 스누브 요청을 필터링하도록 구성된다. 각각의 그룹은 복수의 캐시 라인을 포함한다.

[0007] 방법은 복수의 프로세서의 캐시 회로에서 메모리의 캐시 라인을 캐싱하는 단계를 포함한다. 상기 방법은 제1 스누브 필터가 상기 메모리의 어드레스의 제1 서브세트의 개별 캐시 라인에 의한 스누브 요청을 필터링하는 단계와, 제2 스누브 필터가 상기 메모리의 어드레스의 제2 서브세트의 캐시 라인 그룹에 의한 스누브 요청을 필터링하는 단계를 포함한다. 각각의 그룹은 복수의 캐시 라인을 포함한다.

[0008] 이하의 상세한 설명 및 청구범위를 숙고함으로써 다른 특징들을 알게 될 것이다.

도면의 간단한 설명

[0009] 본 방법 및 시스템의 다양한 양태와 효과는 이어지는 상세한 설명을 살펴보고 도면을 참조할 때에 명백해질 것이다.

- 도 1은 공유 메모리 및 공유 메모리에 액세스하는 다수의 캐싱 에이전트를 구비한 예시적인 시스템을 도시한다.
- 도 2는 정밀 스눅 필터 로직에 의해 추적되는 정밀 피추적 캐시 라인의 디렉토리 및 비정밀 스눅 필터 로직에 의해 추적되는 비정밀 피추적 캐시 라인의 디렉토리를 보여준다.
- 도 3은 혼성의 정밀 스눅 필터링 및 비정밀 스눅 필터링인 스눅 필터링 프로세스의 흐름도를 보여준다.
- 도 4는 본 개시의 방법 및 회로에 따라 구성될 수 있는 예시적인 프로그래머블 IC를 보여준다.

발명을 실시하기 위한 구체적인 내용

- [0010] 이하의 설명에서, 다수의 특정 상세내용은 여기에 공개하는 특정 예를 설명하기 위해 제공된다. 그러나, 당업자에게는, 이하에서 주어지는 모든 특정 상세 없이도 이들 예의 변형 및/또는 하나 이상의 다른 예들이 실시될 수 있음이 명백함에 틀림없다. 다른 경우에 있어서, 잘 알려진 특징들은 본 명세서에 기재하는 예시의 설명을 모호하게 하지 않도록 상세하게 설명하지 않았다. 용이한 설명을 위해, 상이한 도면들에서 동일한 엘리먼트 또는 동일한 엘리먼트의 추가 인스턴스를 지칭하는 데에 동일한 참조 번호가 사용될 수 있다.
- [0011] 정밀 및 비정밀 스눅 필터링 스킴은 확장성과 정확성 사이에 트레이드오프를 제시한다. 정밀 스눅 필터는 확장성은 떨어지지만 정확한 필터링을 제공한다. 정밀 스눅 필터는 개별 캐시 라인의 어드레스에 의한 스눅 요청을 필터링한다. 이에, 정밀 스눅 필터에 의해 발행된 스눅 요청은 통상 위조된 것이 아니며, 추적 대상 캐싱 엘리먼트만을 표적으로 한다. 정밀 스눅 필터는, 모든 캐싱 엘리먼트에 대해 추적되는 캐시의 누적 사이즈뿐만 아니라, 스눅 필터의 사이즈, 캐싱 엘리먼트의 수 사이에서 트레이드오프가 필요할 때에, 확장이 용이하지 않다.
- [0012] 비정밀 스눅 필터는 정확성은 떨어지지만 확장성을 제공한다. 비정밀 스눅 필터는 캐시 라인 세트 전체에서 캐시 라인의 총계 프레즌스(aggregate presence)를 추적하는데, 이 추적은 캐시 라인당 기준보다 더 조대한 입도에서 행해진다. 그렇기 때문에, 비정밀 스눅 필터는 정밀 스눅 필터에 비해 더 큰 확장성을 제공한다. 정밀 스눅 필터에서 프레즌스를 갖는 것으로서 마킹된 특정 캐시 라인이 추적 대상 캐싱 엘리먼트에서 존재할 수도 존재하지 않을 수도 있기 때문에, 비정밀 스눅 필터에 의해 발행된 스눅 요청은 위조될 수도 있다. 비정밀 스눅 필터는 확장이 용이하여, 더 많은 수의 추적 대상 캐싱 엘리먼트 및 추적 대상 캐싱 엘리먼트 전체에서 더 큰 누적 사이즈의 피추적 캐시 사이즈를 제공한다.
- [0013] 개시하는 회로 및 방법은 정밀 및 비정밀 양 스눅 필터의 장점을 제공하는 혼성 스눅 필터를 제공한다. 혼성 스눅 필터는 정밀 스눅 필터 및 비정밀 스눅 필터 둘 다를 채택한다. 또한, 정확성과 확장성 사이의 최적 균형을 위해 다수의 프로세싱 유닛의 캐시 상태의 미세 입도의 정밀 추적 및 조대 입도의 비정밀 추적 사이를 조절하기 위한 회로부가 제공된다. 이에 해당 회로 및 방법은 정밀 스눅 필터링만 또는 비정밀 스눅 필터링만 채택하는 접근법의 단점을 피한다.
- [0014] 혼성 스눅 필터는, 여러 프로세서들 사이에서 공유 메모리를 구비하고 프로세서와 공유 메모리 사이에 또 프로세서들 사이에 점대점 접속을 제공하는 인터커넥트 회로부를 구비한 멀티프로세서 시스템에서 특히 유용하다. 각각의 프로세서는 로컬 캐시를 구비한다. 정밀 스눅 필터는 개별 캐시 라인의 어드레스에 의한 스눅 요청을 필터링하고, 비정밀 스눅 필터는 캐시 라인 그룹을 포함하는 어드레스에 의한 스눅 요청을 필터링한다. 정밀 및 비정밀 스눅 필터링 각각의 단점들은 스눅 필터의 액세스 패턴 및 용량에 기초하여 캐시 라인의 추적을 정밀 및 비정밀 스눅 필터 사이에 재할당함으로써 감소될 수 있다.
- [0015] 도 1은 공유 메모리(102) 및 공유 메모리에 액세스하는 다수의 캐싱 에이전트(104, 106, 108, 및 110)를 구비한 예시적인 시스템을 도시한다. 각각의 캐싱 에이전트는, 공유 메모리에서 데이터에 액세스할 수 있고 공유 메모리로부터의 데이터의 카피를 로컬 캐시 메모리에 로컬로 저장할 수 있는 프로세싱 회로부를 포함한다. 예시적인 시스템에서, 캐싱 에이전트(104, 106, 108, 및 110)는 프로세서 회로(114, 116, 118, 및 120)를 각각 포함한다. 프로세서 회로의 예는 중앙 처리 장치(CPU, central processing unit), 그래픽 처리 장치(GPU, graphics processing unit), 디지털 신호 처리기(DSP, digital signal processor), 필드 프로그래머블 게이트 어레이(FPGA, field programmable gate array)로 구현된 회로, 주문형 반도체(ASIC, application specific integrated circuit)로 구현된 회로, 및 이들의 다양한 조합을 포함한다. 캐싱 에이전트 및 공유 메모리(102)는 컴포넌트들 또는 공유 버스들 사이에서 점대점 접속을 제공하는 스위치와 같은 인터커넥트 회로부에 의해 결합된다.
- [0016] 캐싱 에이전트(104, 106, 108, 및 110) 각각은 캐시 회로부를 포함한다. 캐시 회로부는 캐시 메모리 및 캐시 제어 회로부를 포함한다. 캐시 메모리는 캐시 라인의 데이터에 대한 스토리지를 제공하고, 세트 어소시에이티브이

며 캐시 라인에 대한 태그를 포함할 수 있다. 캐시 제어 회로는 변환 색인 버퍼(TLB, translation look-aside buffer)를 포함하고, 또 로컬 캐시로부터 또는 공유 메모리로부터 데이터를 리트리빙하고, 데이터를 다시 공유 메모리에 기록하고, 외부 스누프 요청을 서빙하며, 해당 스누프 요청의 결과로 캐싱된 데이터 및 캐시 상태를 관리하고, 캐시 축출 알고리즘(cache eviction algorithm)을 통해 캐시 메모리 내의 가용 스토리지를 관리하며, 캐시 내의 캐시 라인을 무효화하거나 하는 등등의 제어를 포함할 수 있다. 캐싱 에이전트(104, 106, 108, 및 110)는 캐시 제어 회로(138, 140, 142, 및 144)를 각각 포함한다.

[0017] 캐싱 에이전트 각각에서 분리된 블록으로서 정밀 추적되는 캐시 및 비정밀 추적되는 캐시가 도시되고 있지만, 캐시 메모리에 있어서 정밀 추적되는 캐시 라인은 비정밀 추적되는 캐시 라인과 교차될 수 있음을 알 것이다. 정밀 스누프 필터(146)는 정밀 추적되는 캐시 라인의 어드레스를 추적하고, 비정밀 스누프 필터(148)은 비정밀 추적되는 캐시 라인의 어드레스를 추적한다. 정밀 추적되는 캐시 라인은 개별 캐시 라인의 어드레스에 의해 추적된다. 비정밀 스누프 필터링에서는, 캐시 라인의 어드레스가 그룹에 의해 추적된다. 정밀 추적되는 각각의 캐시 라인에 대해, 정밀 스누프 필터는 해당 캐시 라인의 어드레스를 저장한다. 비정밀 추적되는 캐시 라인에 대해, 비정밀 스누프 필터는 다수의 캐시 라인의 어드레스에 매핑되는 태그를 저장한다. 정밀 스누프 필터(146)가 추적하는 공유 메모리의 어드레스의 서브세트는 캐싱 에이전트(104, 106, 108, 및 110)에서 블록(122, 126, 130, 및 134)으로서 각각 예시되고, 비정밀 스누프 필터(148)가 추적하는 공유 메모리의 어드레스의 서브세트는 캐싱 에이전트(104, 106, 108, 및 110)에서 블록(124, 128, 132, 및 136)으로서 각각 예시된다.

[0018] 제어 회로(150)와 함께, 정밀 스누프 필터(146) 및 비정밀 스누프 필터(148)는 인터커넥트(152)에 결합되는 홈 에이전트(112)에 의해 호스팅된다. 홈 에이전트는 캐싱 에이전트들 사이에서 코히어런트 트랜잭션을 서비스하는 회로이다. 홈 에이전트는 주어진 어드레스 공간을 위해 코히어런스를 유지하며, 메인 메모리에 대한 메모리 컨트롤러 회로일 필요는 없다. 소정의 트랜잭션의 플로우에서 요구되는 데이터 및 소유권 응답이 홈 에이전트에 의해 제공된다.

[0019] 정밀 스누프 필터(PSF, precise snoop filter)(146)는 개별 캐시 라인의 어드레스에 의한 스누프 요청을 필터링한다. 캐시 라인 A에 액세스할 경우, PSF는 로컬 캐시에서 캐시 라인 A를 갖는 해당 캐싱 에이전트에서의 스누프 요청 전송을 제어한다. 비정밀 스누프 필터(ISF, imprecise snoop filter)(148)는 캐시 라인 그룹에 의한 스누프 요청을 필터링한다. 캐시 라인 A에 액세스할 경우, ISF는 캐시 라인 A를 포함하는, 캐시 라인 A와 동일한 그룹에 매핑되는 캐시 라인을 갖는 해당 캐싱 에이전트에서의 스누프 요청 전송을 제어한다.

[0020] 제어 회로(150)는 PSF에 의해 추적되는 캐시 라인과 ISF에 의해 추적되는 캐시 라인 그룹 사이에서 할당 및 할당 해제 및/또는 전환(transition)을 집행한다. 캐싱 에이전트는 캐시 라인의 이력적 액세스 패턴에 따라, 가능하게는 정밀 및 비정밀 추적되는 캐시 라인의 조합을 구비하거나, 전부 정밀 추적되는 캐시 라인을 구비하거나 전부 비정밀 추적되는 캐시 라인을 구비한다. 상이한 캐싱 에이전트들은 캐싱된 어드레스의 정밀 추적된 부분과 비정밀 추적된 부분 간에 상이한 할당을 가질 수 있다.

[0021] 제어 회로(150)는 정밀 추적되는 캐시 라인을 비정밀 추적되는 그룹에 재할당할 수 있고, 비정밀 추적되는 그룹을 정밀 추적되는 캐시 라인에 재할당할 수 있다. 캐시 라인의 추적을 PSF로부터 ISF로 재할당하는 것은 신규 캐시 라인 어드레스를 위한 공간을 만들기 위해 PSF로부터의 캐시 라인의 축출에 응답하는 것일 수 있다. 축출된 캐시 라인을 포함하기 위해 신규 그룹이 ISF에 추가될 수 있다. 캐시 라인을 PSF로부터 ISF로 재할당하는 것은 하나의 캐싱 에이전트가 단독으로 소유하는 피참조 캐시 라인 또는 캐시 라인 어드레스에 대한 참조의 시간적 로컬성(temporal locality)과 같은 정책 선택에 응답하는 것일 수 있다.

[0022] 기타 정책 선택은 정적으로 할당되는 어드레스 윈도우에 대한 액세스 및 동적으로 결정되는 어드레스 핫스팟에 대한 액세스를 포함할 수 있다. 정적으로 할당되는 어드레스 윈도우는 성능에 중요한 동기화 데이터 구조를 다수의 프로세서가 공유하는 어드레스의 범위일 수 있다. 해당 윈도우 내의 데이터에 대한 액세스 또는 소유권의 동시 요청은 빈번하게 발생하고 해당 어드레스 범위 내의 캐시 라인은 PSF가 관리하기에 적합하다.

[0023] 동적으로 결정되는 어드레스 핫스팟은 캐시 라인에의 액세스의 장기 모니터링을 통해 식별될 수 있다. 단기 모니터링을 포함하는 통상의 LRU 기반 할당과 달리, 어드레스 핫스팟 검출, 및 PSF에서의 다음의 우선 할당은 빈번하게 액세스되는 어드레스의 장기적인 결정 및 특정 어드레스에 대한 빈번한 경합을 기반으로 한다.

[0024] 도 2는 정밀 스누프 필터 로직(202)에 의해 추적되는 정밀 피추적 캐시 라인의 디렉토리 및 비정밀 스누프 필터 로직(206)에 의해 추적되는 비정밀 피추적 캐시 라인의 디렉토리를 보여준다. 디렉토리(204)는 캐시 라인 태그에 의해 어드레스되는 메모리 회로로서 구현될 수 있고, 디렉토리(208)는 캐시 그룹 태그에 의해 어드레스되는 메

모리 회로로서 구현될 수 있다.

- [0025] 디렉토리(204) 내의 각각의 엔트리는 캐시 라인 태그(210), 라인 상태(212), 및 프레즌스 벡터(214)를 포함한다. 캐시 라인 태그는 PSF 로직(202)에 의해 추적되고 라인 상태 및 프레즌스 벡터로 기술되는 캐시 라인을 식별한다. 라인 상태는 피참조 캐시 라인이 공유되는지, 배타적으로 유지되는지, 유효한지 등을 표시할 수 있다. 프레즌스 벡터는 각각의 비트가 특정 캐싱 에이전트에 대응하는 비트 그룹일 수 있으며, 해당 비트의 상태는 피참조 캐시 라인이 캐싱 에이전트에 의해 캐싱되는지를 표시한다.
- [0026] 디렉토리(208) 내의 각각의 엔트리는 캐시 그룹 태그(216), 라인 상태(218), 및 프레즌스 벡터(220)를 포함한다. 캐시 그룹 태그는 ISF 로직(206)에 의해 추적되고 그룹 상태 및 프레즌스 벡터로 기술되는 캐시 라인 그룹을 식별한다. 그룹 상태는 그룹내 임의의 캐시 라인이 공유되는지, 배타적으로 유지되는지, 유효한지 등을 표시할 수 있다. 프레즌스 벡터는 각각의 비트가 특정 캐싱 에이전트에 대응하는 비트 그룹일 수 있으며, 해당 비트의 상태는 그룹내 임의의 캐시 라인이 캐싱 에이전트에 의해 캐싱되는지를 표시한다.
- [0027] 각각의 그룹 상태(218)는 액세스 카운트를 포함할 수 있다. 액세스 카운트는 캐싱 에이전트가 연관된 캐시 그룹 태그에 액세스하는 것에 응답하여 증분되고, 연관된 캐시 그룹 태그가 참조하는 캐시 그룹에 대한 추출 통지에 응답하여 감분된다. 액세스 카운트를 유지하면 캐시 그룹 태그와 연관된 액세스 카운트가 제로로 감분되는 것에 응답하여 ISF로부터의 캐시 그룹 태그의 자동 추출을 지원하게 된다. 액세스 카운트를 유지하면 동일한 캐싱 에이전트에 의한 동일한 캐시 그룹 내의 캐시 라인에의 다중 액세스를 추적할 수 있다. 액세스 카운트가 없다면, 캐시 그룹 내의 하나의 캐시 라인에 대한 하나의 캐싱 에이전트에 의한 추출 통지가 실행될 수 없는데, 캐싱 에이전트의 캐시에 존재하는 캐시 그룹 내의 다른 캐시 라인에 대한 프레즌스 벡터를 계속 유지해야 하기 때문이다.
- [0028] 도 3은 혼성의 정밀 스냅 필터링 및 비정밀 스냅 필터링인 스냅 필터링 프로세스의 흐름도를 보여준다. 제어 회로(150)는 다양한 동작들을 수행하고 PSF(146) 및 ISF(148)에서 수행되는 다른 동작들을 조정한다.
- [0029] 블록(302)에서, PSF-ISF 할당 회로는 예컨대 특정 캐시 라인 A를 참조하는 코히어런스 요청(coherence request)을 수신한다. 캐시 코히어런스 요청은 기록(write) 요청 및 판독(read) 요청을 포함한다. 캐시 코히어런스 요청이 기록 또는 판독인지(또는 기록 또는 판독의 어느 것으로도 분류될 수 없는 기타 유형)를 전달하는 것과 함께, 해당 요청은 통상 요청 유형에 기초한 추가 정보 및 다른 캐시 내의 캐시 라인의 이전 상태도 전달한다. 예를 들어, 캐시 코히어런스 요청은 캐시 라인을 기록하는 캐싱 에이전트가 배타적(그래서 다른 캐시와의 통신 없이 추후 기록을 수행하려고 함) 또는 공유 카피를 보유한다는 것을 전달할 수 있다. 다른 예는, 캐시 라인을 판독하는 캐싱 에이전트가 배타적(그래서 다른 캐시와의 통신 없이 추후 판독을 수행하려고 함) 또는 공유 카피를 수신하려고 하는 것을 전달하는 캐시 코히어런스 요청이다.
- [0030] 결정 블록(304)에서, 제어 회로(150)는 피참조 캐시 라인 A가 PSF(146) 또는 ISF(148)의 어느 쪽에 존재하는지 체크한다. PSF에서의 존재는 코히어런스 요청에서 지정된 캐시 라인의 어드레스를 PSF 내의 캐시 라인 태그에 비교함으로써 결정될 수 있다. 마찬가지로, ISF에서의 존재는 지정된 캐시 라인 어드레스의 그룹 태그를 결정된 다음, 결정된 그룹 태그가 ISF 내에 존재하는지를 결정함으로써 결정될 수 있다. 캐시 라인 어드레스를 캐시 그룹 태그에 매핑하기 위해 해시 함수가 사용될 수 있다.
- [0031] 결정 블록(306)에서는 캐시 어드레스 라인이 PSF(146) 또는 ISF(148)에서 추적되는지에 기초하여 프로세스를 총괄한다. 캐시 어드레스 라인 A가 PSF에서 추적되고 있는 것에 응답하여, 블록(308)에서 제어 회로(150)는 PSF 내의 캐시 라인 A의 상태를 코히어런스 요청과 일치시켜 업데이트하고, 필요하다면 PSF 내의 캐시 라인과 연관된 프레즌스 벡터에서 표시하는 캐싱 에이전트의 캐시 컨트롤러에 코히어런스 요청을 전송한다. 프레즌스 벡터만이 업데이트되는 경우 캐시 코히어런스 요청이 공유 그룹 상태 및 공유 판독 요청을 위해 전송될 필요는 없다.
- [0032] 캐시 라인 A가 ISF(148)에서 추적되는 캐시 라인은 그룹에 속하는 것에 응답하여, 블록(310)에서 할당 회로는 ISF 내의 그룹 태그와 연관된 프레즌스 벡터에서 표시되는 캐싱 에이전트의 캐시 컨트롤러에 코히어런스 요청을 전송한다. 그러나, 캐시 라인 A의 그룹 내에 존재하는 캐시 라인을 가진 모든 프로세서에 캐시 코히어런스 요청을 전송하는 것이 불필요한 요청 유형 및 ISF 내의 캐시 그룹 상태의 조합이 존재한다. 예를 들어, 프레즌스 벡터만이 업데이트되는 경우 캐시 코히어런스 요청이 공유 그룹 상태 및 공유 판독 요청을 위해 전송될 필요는 없다.
- [0033] 결정 블록(312)에서, 제어 회로(150)는 캐시 라인 A를 PSF(146)에 추가하기 위해 캐시 라인 A가 속한 그룹이

ISF(148)로부터 제거되어야 하는지 또는 해당 그룹이 ISF로부터 단순히 제거되어야 하는지를 결정한다. 그 결정은 애플리케이션에 따른 정책 선택에 기초할 수 있다. 예를 들어, 일부 애플리케이션은 정확성은 더 많이 확장성은 덜 필요로 할 수 있으며, 그래서 ISF에서 PSF로의 이동에 대해 덜 저항적인 정책으로 이어지게 된다. 일부 애플리케이션은 확장성은 더 많이 정확성은 덜 필요로 할 수 있으며, 그래서 ISF에서 PSF로의 이동에 더 저항적인 정책으로 이어지게 된다. ISF에서 PSF로의 캐시 라인 재할당을 트리거하는데 사용될 수 있는 간단한 예시적인 인디케이터가 캐싱 에이전트에 의한 캐시 라인의 단독 소유권을 지정하는 코히어런스 요청이다. 캐시 라인의 공유 카피를 지정하는 코히어런스 요청은 ISF에서 PSF로의 캐시 라인 재할당을 트리거하지 않는다. 마찬가지로, 제어 회로가 캐시 코히어런스 요청으로부터, 캐시 라인이 단독 소유자에게만 존재할 수 있고 또한 ISF에서 해당 소유자 카운트가 1과 매칭된다고 결정하면, 해당 그룹은 PSF에 재할당될 수 있다. 다른 예는 제어 회로가 캐시 코히어런스 요청의 유형으로부터, 요청을 프로세싱한 후의 캐시 라인의 소유자 카운트가 1이 되는 것임을 결정하는 것이다. 소유자 카운트는 프레즌스 벡터에 설정된 비트 수에 의해 표시될 수 있다.

[0034] 정책은, 그룹 내의 캐시 라인에 대한 액세스 수를 추적하고 액세스 수가 미리 규정된 기간 내에 임계 값에 도달하면(시각적 로컬성) 캐시 라인 그룹을 ISF(148)에서 PSF(146)로 재할당함으로써, 다소 제한적으로 만들어질 수 있다. 캐시 라인이 PSF에 재할당되어서는 안 된다는 결정에 응답하여, 블록(314)에서, 제어 회로는 캐시 라인 A가 속하는 캐시 라인 그룹의 그룹 상태를 업데이트한다. 상태는 캐시 코히어런스 요청의 유형에 기초한다.

[0035] 캐시 라인이 PSF(146)에 재할당되어야 한다고 또는 블록(310)에서 액세스 카운트가 제로로 감분되는 것에 따라 단순히 축출되어야 한다고 결정하는 것에 응답하여, 블록(316)에서, 할당 회로는 캐시 라인 A가 속하는 캐시 라인 그룹을 ISF(148)로부터 축출한다. 캐시 라인 그룹 축출 시에, 할당 회로는 ISF로부터 그룹 태그를 제거한다. 액세스 카운트가 제로이면, 결정 블록(318) 및 블록(330)에서 보여지는 바와 같이, 추가 액션은 필요하지 않다. 그렇지 않고, 액세스 카운트가 제로가 아니면, 결정 블록(318)은 프로세스를 블록(320)으로 진행시켜 그룹 내의 모든 캐시 라인을 PSF에 추가하기 시작한다. ISF로부터 축출된 캐시 라인 그룹의 상태가 그룹 내의 모든 캐시 라인에 적용 가능하므로, 개별 캐시 라인의 상태는 결정될 수 없으며, 이에 그룹 내의 모든 캐시 라인이 PSF에 추가될 수 있다. ISF로부터 제거된 그룹의 상태는 PSF에 추가된 캐시 라인 각각에 적용된다. 해시 함수가 캐시 라인을 그룹 태그에 매핑하는데 사용될 수 있으며, 역(inverse) 해시 함수는 재할당을 위해 전체 캐시 라인 어드레스를 생성하는데 사용될 수 있다.

[0036] ISF(148)에서 PSF(146)로의 전환 효율성은 캐시 그룹 내의 모든 캐시 라인에 대한 상태 쿼리 스놉을 프레즌스 벡터가 표시하는 소유자에게 전송함으로써 향상될 수 있다. 따라서 캐싱 에이전트에 여전히 존재하는 캐시 라인만이 PSF에서 추적된다. 또한, 캐싱 에이전트 전체에서 업데이트되는 누적 캐시 상태도 PSF에서 추적될 수 있다. 상태 쿼리 스놉의 전반적인 효과는 PSF에서 장소 및 상태의 정확성을 회복하는 것이다.

[0037] 블록(320, 322, 324, 326 및 328)의 프로세싱은 ISF(148)로부터 축출된 그룹으로부터 PSF(146)에 추가되는 각 캐시 라인에 대해 수행될 수 있다. 마찬가지로, 블록(320, 322, 324, 326 및 328)의 프로세싱은 결정 블록(304)에서 결정된 바와 같이 ISF도 PSF도 피참조 캐시 라인을 추적하지 않는 경우에 수행될 수 있다.

[0038] 결정 블록(320)에서, 제어 회로(150)는 결정 블록(304)에서 결정되는 것인 PSF(146)가 축출된 캐시 라인에 대해 또는 누락된 캐시 라인에 대해 이용 가능한 공간을 갖는지를 결정한다. 디렉토리(204)는 최대 수의 캐시 라인 태그 및 상태를 위한 저장 공간을 갖는다. 공간이 이용 가능하다면, 블록(322)에서 제어 회로는 PSF 디렉토리 내의 슬롯을 해당 캐시 라인에 할당하고 그에 따라 태그, 캐시 라인 상태, 및 프레즌스 벡터를 업데이트한다.

[0039] PSF(146)에서 캐시 라인을 위한 공간이 이용 가능하지 않으면, 블록(324)에서 제어 회로는 PSF 디렉토리에 캐시 라인을 위한 공간을 만들고 PSF에서 ISF(148)로 캐시 라인을 재할당하기 위한 축출 프로세싱의 실행을 시작한다. 블록(326)에서, 제어 회로는 최소 사용 빈도로 액세스된 캐시 라인과 같은, PSF로부터 축출할 캐시 라인을 선택하고, 축출을 위해 선택된 캐시 라인과 동일한 그룹에 속하는 PSF 내의 모든 캐시 라인을 결정한다. 동일한 그룹에 속하는 캐시 라인도 축출된다.

[0040] PSF(146)로부터 캐시 라인을 축출할 때, 캐시 라인의 태그, 상태, 및 프레즌스 정보는 ISF(148)에 추가할 정보를 보존하기 위해 블록(322 및 328)의 프로세싱 전에 판독된다. 블록(328)에서, 제어 회로는 축출을 위해 선택된 캐시 라인을 포함하는 캐시 라인 그룹을 위해 ISF에서 슬롯을 할당한다. 또한, 블록(328)에서 제어 회로는 축출을 위해 선택된 캐시 라인의 집계된 캐시 상태(그룹내 캐시 라인 상태의 논리 OR) 및 프레즌스 벡터를 ISF 내의 할당된 슬롯에 저장한다. 캐시 라인 그룹의 생성 및 축출과 연관되는 프로세싱 성능은 해시 함수를 사용하여 캐시 라인을 캐시 그룹 태그에 매핑함으로써 향상된다. 이에 PSF에서 ISF로의 캐시 라인 축출은 그룹 내의

캐시 라인들 간의 공간적 및/또는 시간적 관계를 축소시킬 수 있다. 블록(322)에서, 제어 회로는 전술한 바와 같이 캐시 라인을 PSF에 추가한다.

[0041] 도 4는 개시하는 방법 및 회로에 따라 구성될 수 있는 예시적인 프로그래머블 IC(402)를 보여준다. 프로세서 서브시스템(410)과 프로그래머블 로직 서브시스템(430)을 포함하는 프로그래머블 IC는 SOC(System On Chip)이라도 할 수 있다. 프로세서 서브시스템(410)은 사용자 프로그램의 실행을 통해 사용자 설계의 소프트웨어 부분을 구현하도록 프로그래밍될 수 있다. 프로그램은 구성 데이터 스트림의 일부로서 지정될 수도 있고 온칩 또는 오프칩 데이터 저장 디바이스로부터 리트리빙(retrieving)될 수도 있다. 프로세서 서브시스템(410)은 하나 이상의 소프트웨어 프로그램을 실행할 다양한 회로(412, 414, 416, 및 418)를 포함할 수 있다. 회로(412, 414, 416, 및 418)는 예컨대 하나 이상의 프로세서 코어, 플로팅 포인트 유닛(FPU), 인터럽트 프로세싱 유닛, 온칩 메모리, 메모리 캐시, 및/또는 캐시 코히어런트 인터커넥트를 포함할 수 있다.

[0042] 프로그래머블 IC(402)의 프로그래머블 로직 서브시스템(430)은 사용자 설계의 하드웨어 부분을 구현하도록 프로그래밍될 수 있다. 예를 들면, 프로그래밍 로직 서브시스템은 구성 데이터 스트림에서 지정된 회로의 세트를 구현하도록 프로그래밍될 수 있는 다수의 프로그래머블 리소스(432)를 포함할 수 있다. 프로그래머블 리소스(432)는 예컨대 프로그래머블 인터커넥트 회로, 프로그래머블 로직 회로, 및 구성 메모리 셀을 포함한다. 프로그래머블 로직은 예컨대, 함수 발생기, 레지스터, 산술 로직 등을 포함할 수 있는 프로그래머블 엘리먼트를 이용해서 사용자 설계 로직을 구현한다. 프로그래머블 인터커넥트 회로는 프로그래머블 인터커넥트 포인트(programmable interconnect point, "PIP")에 의해 상호접속되는 다양한 길이의 다수의 인터커넥트 라인을 포함할 수 있다.

[0043] 프로그래머블 리소스(432)는 프로그래머블 인터커넥트 회로와 프로그래머블 로직 회로가 구성되는 방법을 규정하는 구성 메모리 셀에 구성 데이터 스트림을 로딩함으로써 프로그래밍될 수 있다. 예를 들어, 구성 가능한 래치에 대한 구성 메모리 셀을 제1 값으로 설정하면 구성 가능한 래치를 싱글 에지 구동 래치(single-edge-driven)로서 동작하게 할 수 있다. 구성 메모리 셀을 제2 값으로 설정하면 구성 가능한 래치를 더블 에지 구동 래치(double-edge-driven)로서 동작하게 할 수 있다. 그런 다음 개별 메모리 셀의 집단 상태(collective state)가 프로그래머블 리소스(432)의 기능을 결정한다. 구성 데이터는 메모리로부터(예컨대, 외부 PROM으로부터) 판독되거나 외부 디바이스에 의해 프로그래머블 IC(402)에 기록될 수 있다. 일부 구현예에 있어서, 프로그래머블 로직 서브시스템(430)에 포함된 구성 컨트롤러(434)는 프로그래머블 IC의 파워온에 응답하여, 프로그래머블 IC에 결합된 비휘발성 메모리로부터 구성 데이터를 리트리빙하고 구성 데이터를 구성 메모리 셀에 로딩함으로써, 프로그래머블 리소스를 프로그래밍할 수 있다. 몇몇 다른 구현예에 있어서, 구성 데이터는 프로세서 서브시스템(410)에 의해 실행되는 시동 프로세서에 의해 구성 메모리 셀에 로딩될 수도 있다.

[0044] 프로그래머블 IC(402)는 프로세서 서브시스템(410)을 프로그래머블 로직 서브시스템(430) 내에 구현된 회로부와 상호접속시키기 위한 다양한 회로를 포함할 수 있다. 이 예에 있어서, 프로그래머블 IC(402)는 프로그래머블 로직 서브시스템(430)과 프로세서 서브시스템(410)의 다양한 데이터 포트 사이에서 데이터 신호를 라우팅할 수 있는 코어 스위치(426)를 포함한다. 코어 스위치(426)는 내부 데이터 버스 등의 프로그래머블 IC의 각종 다른 회로와 프로그래머블 로직 또는 프로세서 서브시스템(410 및 430)의 사이에서 데이터 신호를 라우팅할 수도 있다. 대안으로 또는 부가적으로, 프로세서 서브시스템(410)은 코어 스위치(426)을 바이패싱하는 프로그래밍 로직 서브시스템과 직접 접속하기 위한 인터페이스를 포함할 수도 있다. 이러한 인터페이스는 예컨대 ARM이 공표한 AMBA AXI 프로토콜 사양(AXI)을 이용해서 구현될 수 있다.

[0045] 일부 구현예에 있어서, 프로세서 서브시스템(410)과 프로그래밍 로직 서브시스템(430)은 메모리 컨트롤러(421)를 통해 온칩 메모리(422) 또는 오프칩 메모리(도시 생략)의 메모리 장소에 대해 관독 또는 기록할 수 있다. 메모리 컨트롤러(421)는, 16비트, 32비트, ECC를 갖는 16비트 등이든 아니든 메모리의 DDR(Double Data Rate)2, DDR3, LP(Low Power DDR2) 타입을 포함하나, 이에 한정되는 않는 하나 이상의 상이한 타입의 메모리 회로와 통신하도록 구현될 수 있다. 메모리 컨트롤러(421)와 통신할 수 있는 상이한 메모리 타입의 리스트는 예시를 위해서만 제공될 뿐이며, 한정적인 것으로 또는 총망라한 것으로 의도되지 않는다. 도 4에 도시하는 바와 같이, 프로그래머블 IC(402)는 메모리 관리 유닛(420)과, 서브시스템(410, 430)에 의해 사용되는 가상 메모리 어드레스를 특정 메모리 장소에 액세스하기 위해 메모리 컨트롤러(421)에 의해 이용되는 물리적 메모리 어드레스로 변환하는 TLB(translation look-aside buffer)(424)를 포함할 수 있다.

[0046] 프로그래머블 IC는 외부 회로와의 데이터 통신을 위해 입출력(I/O) 서브시스템(450)을 포함할 수 있다. I/O 서브시스템(450)은 예컨대 플래시 메모리 타입 I/O 디바이스, 고성능 I/O 디바이스, 저성능 인터페이스, 디버깅

I/O 디바이스, 및/또는 RAM I/O 디바이스를 포함하는 다양한 타입의 I/O 디바이스 또는 인터페이스를 포함할 수 있다.

[0047] I/O 서브시스템(450)은 도면부호 460A 및 460B로 나타내는 하나 이상의 플래시 메모리 인터페이스(460)를 포함할 수 있다. 예컨대, 하나 이상의 플래시 메모리 인터페이스(460)는 4비트 통신용으로 구성된 QSPI(Quad-Serial Peripheral Interface)로서 구현될 수 있다. 하나 이상의 플래시 메모리 인터페이스(460)는 병렬 8비트 NOR/SRAM 타입의 인터페이스로서 구현될 수 있다. 하나 이상의 플래시 메모리 인터페이스(460)는 8비트 및 16비트 통신용으로 구성된 NAND 인터페이스로서 구현될 수 있다. 기술하는 특정 인터페이스는 예시를 위해 제공되며 제한이 없음이 이해되어야 한다. 상이한 비트 폭을 가진 다른 인터페이스도 이용될 수 있다.

[0048] I/O 서브시스템(450)은 플래시 메모리 인터페이스(460)보다 상위 레벨의 성능을 제공하는 하나 이상의 인터페이스(462)를 포함할 수 있다. 인터페이스(462A-462C) 각각은 DMA 컨트롤러에 각각 결합될 수 있다. 예를 들어, 하나 이상의 인터페이스(462)는 USB(Universal Serial Bus) 타입의 인터페이스로서 구현될 수 있다. 하나 이상의 인터페이스(462)는 기가비트 이더넷 타입의 인터페이스로서 구현될 수 있다. 하나 이상의 인터페이스(462)는 SD(Secure Digital) 타입의 인터페이스로서 구현될 수 있다.

[0049] I/O 서브시스템(450)은 인터페이스(462)보다 하위 레벨의 성능을 제공하는 인터페이스(466A-466D) 등의 하나 이상의 인터페이스(466)를 포함할 수 있다. 예를 들어, 하나 이상의 인터페이스(466)는 GPIO(General Purpose I/O) 타입의 인터페이스로서 구현될 수 있다. 하나 이상의 인터페이스(466)는 UARTSD(Universal Asynchronous Receiver/Transmitter) 타입의 인터페이스로서 구현될 수 있다. 하나 이상의 인터페이스(466)는 SPI(Serial Peripheral Interface) 타입의 인터페이스의 형태로 구현될 수 있다. 하나 이상의 인터페이스(466)는 CAN(Controller-Area-Network) 타입의 인터페이스 및/또는 I2C(inter-integrated circuit) 타입의 인터페이스의 형태로 구현될 수 있다. 하나 이상의 인터페이스(466)는 타이머 타입의 인터페이스의 형태로 구현될 수 있다.

[0050] I/O 서브시스템(450)은 프로세서 JTAG(PJTAG) 인터페이스(468A) 및 트레이스 인터페이스(468B) 등의 하나 이상의 디버그 인터페이스(468)를 포함할 수 있다. PJTAG 인터페이스(468A)는 프로그래머블 IC(402)에 대해 외부 디버그 인터페이스를 제공할 수 있다. 트레이스 인터페이스(468B)는 디버그, 예컨대 프로세서 서브시스템(410) 또는 프로그래머블 로직 서브시스템(430)으로부터의 정보, 트레이스를 수신하기 위한 포트를 제공할 수 있다.

[0051] 이 경우에, 각각의 인터페이스(460, 462, 466, 및 468)는 멀티플렉서(470)에 결합될 수 있다. 멀티플렉서(470)는 프로그래머블 IC(402)의 외부 핀, 예컨대 프로그래머블 IC(402)가 배치되는 패키지의 볼에 직접 라우팅 또는 결합될 수 있는 복수의 출력을 제공한다. 예컨대, 프로그래머블 IC(402)의 I/O 핀은 인터페이스(460, 462, 466, 및 468) 간에 공유될 수 있다. 사용자는 구성 데이터 스트림을 통해, 인터페이스(460-468) 중 사용할 것을 선택하여 멀티플렉서(470)를 통해 프로그래머블 IC(402)의 I/O에 결합하도록 멀티플렉서(470)를 구성할 수 있다. I/O 서브시스템(450)은 인터페이스(460-468)를 프로그래머블 로직 서브시스템의 프로그래머블 로직 회로에 접속하기 위한 패브릭 멀티플렉서 I/O(FMIO) 인터페이스(도시 생략)도 포함할 수 있다. 부가적으로 또는 대안으로, 프로그래밍 로직 서브시스템(430)은 프로그래머블 로직 내에 하나 이상의 I/O 회로를 구현하도록 구성될 수 있다. 일부 구현예에 있어서, 프로그래머블 IC(402)는 전력 및/또는 안전 관리를 위한 다양한 회로를 구비한 서브시스템(440)도 포함할 수 있다. 예를 들어, 서브시스템(440)은 프로그래머블 IC(402)의 다양한 서브시스템에 전력을 공급하는데 사용되는 하나 이상의 전압 도메인을 모니터링하고 유지하도록 구성된 전력 관리 유닛(446)을 포함할 수 있다. 일부 구현예에 있어서, 전력 관리 유닛(446)은 사용중인 서브시스템에 대한 전력을 비활성화하지 않고서 전력 소비를 줄이기 위해 아이들 상태일 때 개별 서브시스템에 전력을 공급하지 않을 수 있다.

[0052] 서브시스템(440)은 정확한 동작을 보장하기 위해 서브시스템의 스테이터스를 모니터링하는 안전 회로도 포함할 수 있다. 예를 들면, 서브시스템(440)은 (예컨대, 스테이터스 레지스터(444)로 표시하는 바와 같이) 다양한 서브시스템의 스테이터스를 모니터링하도록 구성된 하나 이상의 실시간 프로세서(442)를 포함할 수 있다. 실시간 프로세서(442)는 에러 검출에 응답하여 다수의 태스크를 수행하도록 구성될 수 있다. 예를 들어, 일부 에러에 대해, 실시간 프로세서(442)는 에러 검출에 응답하여 경고(alert)를 생성할 수도 있다. 다른 예로, 실시간 프로세서(442)는 서브시스템을 리셋하여 해당 서브시스템을 정확한 동작으로 복원시키려고 시도할 수 있다. 서브시스템(440)은 다양한 서브시스템들을 상호접속시키는데 이용될 수 있는 스위치 네트워크(448)를 포함한다. 예를 들어, 스위치 네트워크(448)는 다양한 서브시스템(410, 430, 및 440)을 I/O 서브시스템(450)의 다양한 인터페이스에 접속하도록 구성될 수 있다. 일부 애플리케이션에서, 스위치 네트워크(448)는 모니터링 대상 서브시스템으로부터 실시간 프로세서(442)를 분리하는 데에도 이용될 수 있다. 이러한 분리는 실시간 프로세서(442)가 다른

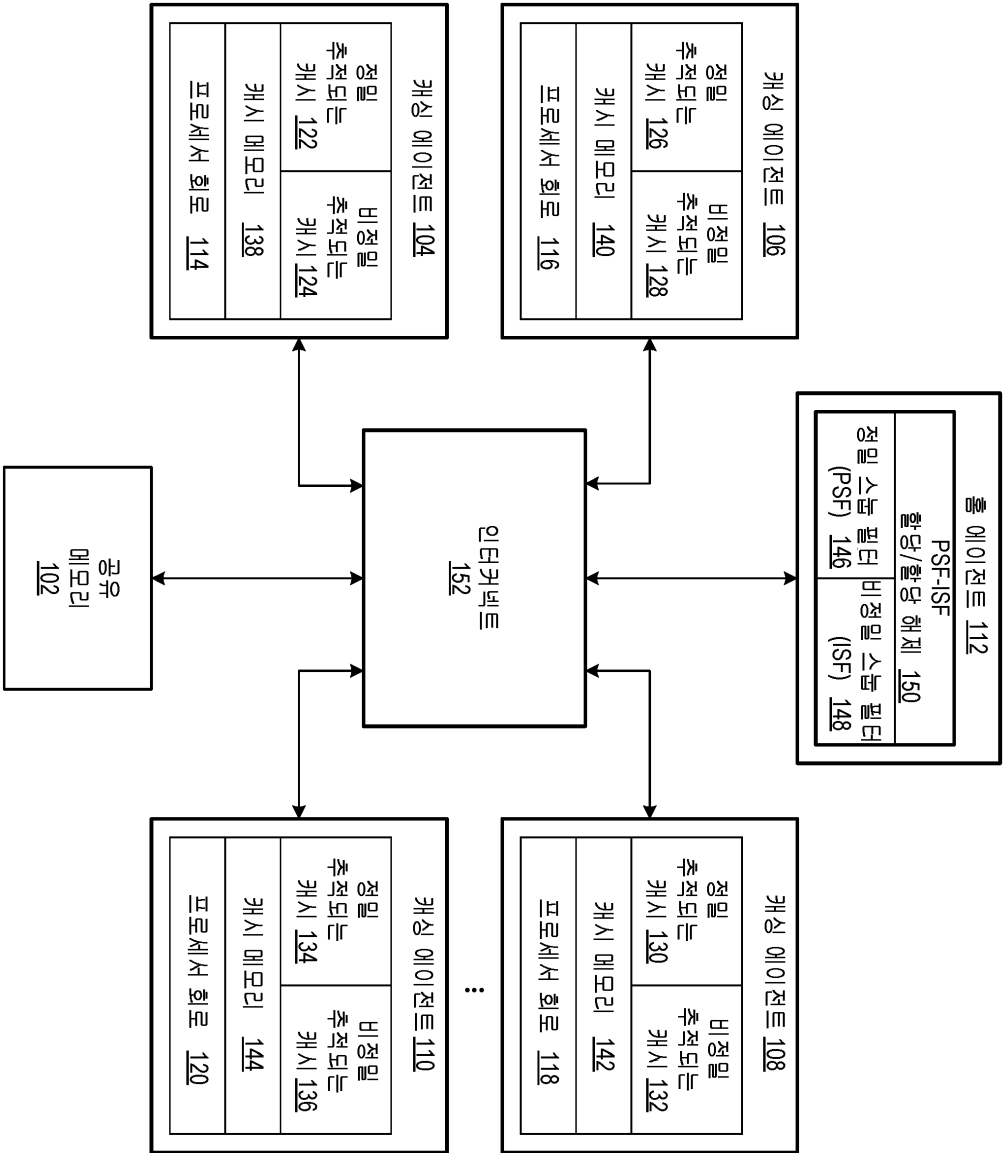
서브시스템에서 발생하는 에러로부터 영향을 받지 않게 하기 위하여 소정의 애플리케이션 표준(예컨대, IEC-61508 SIL3 또는 ISO-26262 표준)에 의해 요구될 수도 있다.

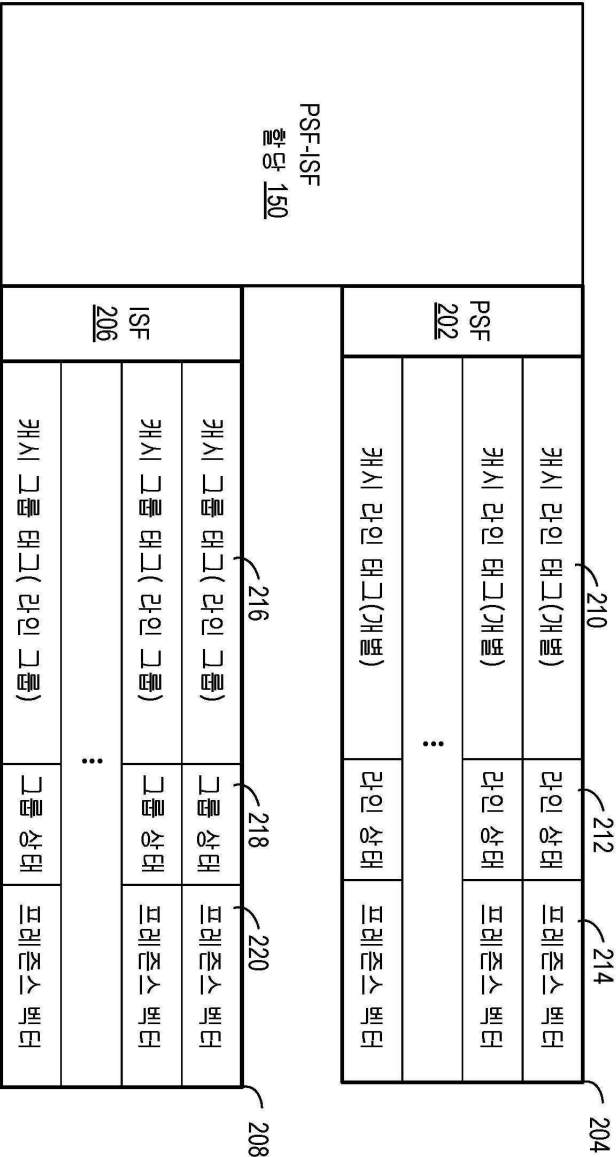
- [0053] 일례에 있어서, 회로 장치(circuit arrangement)가 제공된다. 상기 회로 장치는, 인터커넥트 회로부와, 상기 인터커넥트 회로부에 결합된 메모리와, 상기 인터커넥트 회로부에 결합된 복수의 프로세서 회로와, 복수의 프로세서 회로에 각각 결합된 복수의 캐시 회로와, 상기 인터커넥트 회로부에 결합되고, 상기 메모리의 어드레스의 제1 서브세트의 개별 캐시 라인에 의한 스noop 요청을 필터링하도록 구성되는 제1 스noop 필터(first snoop)와, 상기 인터커넥트 회로부에 결합되고, 상기 메모리의 어드레스의 제2 서브세트의 캐시 라인 그룹에 의한 스noop 요청을 필터링하도록 구성되는 제2 스noop 필터를 포함하고, 각각의 그룹은 복수의 캐시 라인을 포함한다.
- [0054] 일부 상기 회로 장치는 상기 제1 스noop 필터에 그리고 상기 제2 스noop 필터에 결합된 제어 회로를 더 포함할 수 있고, 상기 제어 회로는 상기 제2 스noop 필터에 의해 필터링되는 캐시 라인 그룹을 상기 제1 스noop 필터에 의해 필터링되는 캐시 라인에 재할당하도록 구성될 수 있다.
- [0055] 일부 상기 회로 장치에 있어서, 상기 제어 회로는 또한, 상기 캐시 라인 그룹에 대한 참조의 시간적 로컬성(temporal locality)에 기초하여 상기 제1 스noop 필터에 재할당할 캐시 라인 그룹을 선택하도록 구성될 수 있다.
- [0056] 일부 상기 회로 장치에 있어서, 상기 제어 회로는 또한, 상기 캐시 라인 그룹 내의 캐시 라인의 단독 소유권에 기초하여 상기 제1 스noop 필터에 재할당할 캐시 라인 그룹을 선택하도록 구성될 수 있다.
- [0057] 일부 상기 회로 장치에 있어서, 상기 제어 회로는 또한, 상기 캐시 라인 그룹에서 빈번하게 액세스되는 캐시 라인을 결정하는 것에 기초하여 상기 제1 스noop 필터에 재할당할 캐시 라인 그룹을 선택하도록 구성될 수 있다.
- [0058] 일부 상기 회로 장치에 있어서, 상기 제어 회로는 또한, 재할당을 위해 선택된 그룹에 속하는 캐시 라인을 결정하고, 상기 그룹에 속하는 캐시 라인을 상기 제1 스noop 필터에 재할당하도록 구성될 수 있다.
- [0059] 일부 상기 회로 장치에 있어서, 상기 제어 회로는 또한, 상기 제1 스noop 필터로부터의 제1 캐시 라인을 제2 스noop 필터에 의해 필터링되는 제1 그룹에 재할당하도록 구성될 수 있다.
- [0060] 일부 상기 회로 장치에 있어서, 상기 제어 회로는 또한 상기 제1 그룹에 속하는 상기 제1 스noop 필터에 의해 필터링되는 캐시 라인을 결정하고, 상기 제1 그룹에 속하는 캐시 라인의 캐시 라인 상태를 그룹 상태로 집계하고, 상기 그룹 상태를 상기 제1 그룹의 태그와 연관하여 저장하고, 상기 제1 스noop 필터로부터 상기 제1 그룹의 캐시 라인을 제거하도록 구성된다.
- [0061] 일부 상기 회로 장치에 있어서, 상기 제어 회로는 또한, 상기 제1 스noop 필터의 디렉토리가 풀 상태(full)인 것에 응답하여 상기 제1 스noop 필터로부터의 제1 캐시 라인을 상기 제2 스noop 필터에 의해 필터링되는 제1 그룹에 재할당하도록 구성될 수 있다.
- [0062] 일부 상기 회로 장치에 있어서, 상기 제어 회로는 또한, 상기 제1 캐시 라인이 최소 사용 빈도(least recently used)인 것에 응답하여 상기 제1 스noop 필터로부터 상기 제2 스noop 필터에 의해 필터링되는 제1 그룹에 재할당될 상기 제1 캐시 라인을 선택하도록 구성될 수 있다.
- [0063] 일부 상기 회로 장치에 있어서, 상기 제1 스noop 필터는 개별 캐시 라인의 복수의 캐시 라인 태그의 저장을 위해 구성된 제1 메모리를 포함하고, 상기 제2 스noop 필터는 캐시 라인 그룹과 연관된 복수의 그룹 태그의 저장을 위해 구성된 제2 메모리를 포함하고, 각각의 그룹 태그는 복수의 캐시 라인에 매핑된다.
- [0064] 일부 상기 회로 장치에 있어서, 상기 제어 회로는 또한, 상기 제2 스noop 필터에 의해 필터링되는 캐시 라인의 제1 그룹의 소유자 카운트를 결정하고, 상기 제1 그룹의 소유자 카운트가 1과 같은 것과 상기 제1 그룹의 액세스 카운트가 제로보다 큰 것에 응답하여 상기 제2 스noop 필터로부터의 캐시 라인 중 상기 제1 그룹의 캐시 라인을 상기 제1 스noop 필터에 재할당하도록 구성될 수 있다.
- [0065] 다른 예에서, 방법이 제공될 수 있다. 상기 방법은, 복수의 프로세서의 캐시 회로에서 메모리의 캐시 라인을 캐싱하는 단계와, 상기 메모리의 어드레스의 제1 서브세트의 개별 캐시 라인에 의한 스noop 요청을 제1 스noop 필터에 의해 필터링하는 단계와, 상기 메모리의 어드레스의 제2 서브세트의 캐시 라인 그룹에 의한 스noop 요청을 제2 스noop 필터에 의해 필터링하는 단계를 포함하고, 각각의 그룹은 복수의 캐시 라인을 포함한다.
- [0066] 일부 상기 방법은 제어 회로에 의해, 상기 제2 스noop 필터에 의해 필터링되는 캐시 라인 그룹을 상기 제1 스noop 필터에 의해 필터링되는 캐시 라인에 재할당하는 단계를 더 포함할 수 있다.

- [0067] 일부 상기 방법은, 상기 제어 회로에 의해, 상기 캐시 라인 그룹에 대한 참조의 시간적 로컬성에 기초하여 상기 제1 스눅 필터에 재할당할 캐시 라인 그룹을 선택하는 단계를 더 포함할 수 있다.
- [0068] 일부 상기 방법은, 상기 제어 회로에 의해, 상기 캐시 라인 그룹 내의 캐시 라인의 단독 소유권에 기초하여 상기 제1 스눅 필터에 재할당할 캐시 라인 그룹을 선택하는 단계를 더 포함할 수 있다.
- [0069] 일부 상기 방법은, 상기 제어 회로에 의해, 상기 캐시 라인 그룹에서 빈번하게 액세스되는 캐시 라인을 결정하는 것에 기초하여 상기 제1 스눅 필터에 재할당할 캐시 라인 그룹을 선택하는 단계를 더 포함할 수 있다.
- [0070] 일부 상기 방법은, 재할당을 위해 선택된 그룹에 속하는 캐시 라인을 결정하는 단계와, 상기 그룹에 속하는 캐시 라인을 상기 제1 스눅 필터에 재할당하는 단계를 더 포함할 수 있다.
- [0071] 일부 상기 방법은, 상기 제1 스눅 필터로부터의 제1 캐시 라인을 상기 제2 스눅 필터에 의해 필터링되는 제1 그룹에 재할당하는 단계를 더 포함할 수 있다.
- [0072] 일부 상기 방법은, 상기 제1 그룹에 속하는 상기 제1 스눅 필터에 의해 필터링되는 캐시 라인을 결정하는 단계와, 상기 제1 그룹에 속하는 캐시 라인의 캐시 라인 상태를 그룹 상태로 집계하는 단계와, 상기 그룹 상태를 상기 제1 그룹의 태그와 연관하여 저장하는 단계와, 상기 제1 스눅 필터로부터 상기 제1 그룹의 캐시 라인을 제거하는 단계를 더 포함할 수 있다.
- [0073] 양태 및 특징이 경우에 따라 개별 도면에 기술되더라도, 하나의 도면으로부터의 특징들이 다른 도면의 특징들과 조합될 수 있지만, 이 조합이 조합으로서 명시적으로 도시되거나 설명되지 않는 것이 이해될 것이다.
- [0074] 개시하는 방법 및 시스템은 캐시 코히어런스 스눅 요청을 필터링하기 위한 다양한 시스템에 적용 가능한 것으로 사료된다. 상세 설명을 고찰한 당업자에게는 다른 양태 및 특징이 분명할 것이다. 본 방법 및 시스템은 소프트웨어를 실행하도록 구성되는 하나 이상의 프로세서로서, 주문형 반도체(ASIC)로서, 또는 프로그래머블 로직 디바이스 상의 로직으로서 구현될 수 있다. 상세한 설명 및 도면은 예로만 간주되며, 본 발명의 실제 범주는 다음의 청구범위에서 지정되는 것이 의도된다.

도면

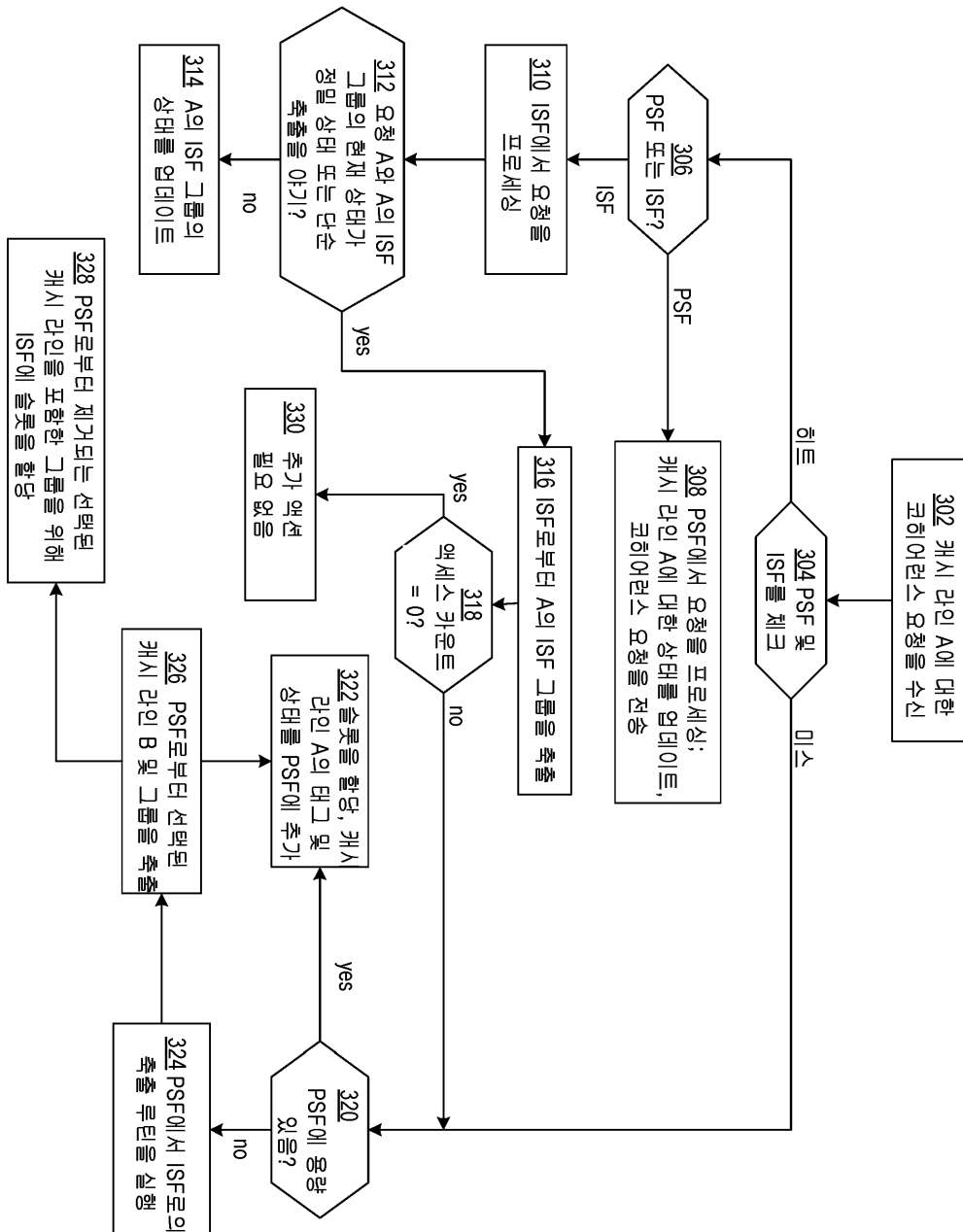
도면1





도면2

도면3



도면4

